Assignment 2: Learning and Memory PSY 306 (Winter 2025)

Name: Samar shah

Roll Number: 2021349

Instructions: Please write your own responses and DO NOT copy or lift text/code from any source, including the attached paper. If you are referring to credible external sources other than the attached paper for your answers, please cite those sources (within the body of text and the provide a reference list at the end) in the APA citation format (https://www.mendeley.com/guides/apa-citation-guide). Word limits given are indicative and less than the indicated numbers may also be used.

Please download this MS word question-cum-response template to TYPE your answers and feel free to add sheets as required. Convert this document to a PDF and rename the file: name_roll no. before submitting. Please note that answers in this template only will be evaluated and hand-written or scanned answer sheets will not be evaluated. Verbatim copying of any extent and total percent similarity with other sources exceeding 10% will be deemed plagiarized and dealt with as per IIITD policies.

[Strict deadline for submission: 7th Feb - 2025 11.00 PM]

Part A) Fill the google form **https://forms.gle/BdaAw6utnqfaGM3r9**

Part B)

An experimenter designed a conditioning paradigm involving three contiguous phases—Phase1, Phase 2, and Phase 3—utilizing happy, angry, and neutral faces as stimuli and the skin conductance response (SCR) was recorded to measure physiological reactions from the onset of face.

Each participant performed a total of 90 trials. For each stimulus (face), there were a total of 30 trials. In the CS+ condition, the face was paired with a shock. In the CS- the face was never paired with a shock. There were 15 trials each for CS+ (face + shock) and CS- (face + no shock), and SCR was recorded for each trial. For example, for an angry face paired with shock (i.e., CS+), there were a total of 15 trials. Similarly, 15 trials were there for an angry face never paired with shock (i.e., CS-).

Details of each phase:

Phase1 (trials 1–2): This phase comprised of 2 trials. Each face was presented twice without reinforcement (no shock).

Phase 2 (trials 3–9): This phase comprised of seven trials. Each stimulus (neutral face/angry face/happy face) was presented seven times paired with shock/no shock. Five of the seven CS+ trials (71%) for each stimulus (face) category co-terminated with an electric stimulation/shock during phase 2. In the CS- category, the face never predicted the shock.

Phase 3 (trials 10–15) - The final phase3 consisted of six unreinforced (no shock) presentations of each CS.

Attached is an excel file named 'LM_A1_Winter2025.csv' containing six columns. The details of the columns are as follows:

Col1/ParticipantID: Participant no. There are a total of 107 participants.

Col2/CS Category: Category of the stimulus (Angry = angry face, Happy = happy face, Neutral = neutral face)

Col3/CS: CS+ = stimulus paired with a shock during phase2; CS- = stimulus never paired with a shock during phase2

Col4/Phase: Phase1, Phase2, Phase3

Col5/CSCount: Number of presentations of the stimulus or trial number

Col6/sqrtSCRUS: Skin conductance Response measured for each trial

Now solve the following. Insert a figure (wherever required) and paste the MATLAB/Python/R code. Do not paste screenshots. Any figure must provide all information necessary to interpret it including axes labels with units, captions/legends. Comment the sections of your code explicitly to convey the key steps.

Q.1.

i. **Calculate the normalized SCR for each participant as follows.**
   Step1- Calculate average SCR of each participant across 90 trials.

   Step2- Subtract each participant's average score from their individual scores (i.e., scores for each CS count/ trial).

Step3 - Adjust the scores by adding a common value i.e., average of all participants' averages (obtained in step 1) to the score obtained in step 2.

	A	B	C	D	E	F	G	H	I	J	K	L
1	Participant_1	Participant_3	Participant_4	Participant_5	Participant_6	Participant_8	Participant_9	Participant_11	Participant_12	Participant_13	Participant_15	Participant_1
2	0.15	0.03	0.04	0.79	0.33	0.38	0.59	0.61	0.21	0.50	0.20	0.3
3	0.15	0.36	0.94	0.44	-0.02	0.04	0.63	0.63	0.21	0.08	0.34	0.3
4	0.96	0.29	0.56	0.70	0.74	0.40	0.69	0.14	0.21	0.44	0.33	0.1
5	0.15	0.03	0.38	0.04	-0.02	0.46	0.72	0.56	0.21	0.35	0.11	0.5
6	1.63	0.03	0.74	0.93	-0.02	0.16		0.06	0.21	0.27	0.16	0.6
7	0.15	0.03	0.04	0.52	0.60	0.58	0.16	0.45	0.21	-0.19	0.10	0.1
8	0.15	0.28	0.77	0.04	0.25	0.04	0.68	0.06	0.21	0.12	0.12	0.1
9	0.15	0.03	0.04	0.24	0.83	0.54	0.38	0.06	0.21	0.28	0.21	0.1
10	0.15	0.21	0.77	0.04	0.67	0.45	-0.16	0.06	0.21	0.41	-0.28	0.1
11	0.15	0.03	0.46	0.53	0.54	0.49	0.54	0.06	0.21	0.45	0.31	0.3
12	0.15	0.33	0.04	0.26	0.48	0.74	0.62	0.06	0.21	-0.19	0.23	0.1
13	0.15	0.03	0.04	0.55	-0.02	0.25	0.42	0.06	0.21	0.46	-0.28	0.3
14	0.15	0.19	0.04	0.44	0.39	0.90	0.91	0.47	0.21	0.15	0.42	0.4

1.

```
import pandas as pd

# Load the dataset
input_file_path = r'C:\Users\Gungun shah\Downloads\LM_A1_2021349\LM_A1_Winter_ParticipantSheets.xlsx'  # Input Excel file
output_file_path = r'C:\Users\Gungun shah\Downloads\LM_A1_2021349\answer1_1.xlsx'  # Output Excel file

# Step 1: Read all sheets (one sheet per participant) from the Excel file
participant_sheets = pd.read_excel(input_file_path, sheet_name=None)  # Reads all sheets into a dictionary

# Step 2: Calculate the global average SCR across all participants
global_avg_scr = 0  # Initialize global average SCR
total_participants = len(participant_sheets)  # Total number of participants

# Loop through each participant's sheet to calculate their average SCR
for sheet_name, participant_data in participant_sheets.items():
    # Calculate the average SCR for the current participant
    participant_avg_scr = participant_data['sqrtSCRUS'].mean()
    # Add the participant's average SCR to the global average
    global_avg_scr += participant_avg_scr

# Calculate the global average SCR
global_avg_scr /= total_participants  # Divide by the number of participants to get the global average
```

```python
# Step 3: Normalize the SCR values for each participant
normalized_data = {}  # Dictionary to store normalized SCR values for each participant

# Loop through each participant's sheet again to normalize their SCR values
for sheet_name, participant_data in participant_sheets.items():
    # Step 1: Calculate the average SCR for the current participant
    participant_avg_scr = participant_data['sqrtSCRUS'].mean()

    # Step 2: Subtract the participant's average SCR from their individual SCR values
    participant_data['CenteredSCR'] = participant_data['sqrtSCRUS'] - participant_avg_scr

    # Step 3: Adjust the scores by adding the global average SCR
    participant_data['NormalizedSCR'] = participant_data['CenteredSCR'] + global_avg_scr

    # Store the normalized SCR values for the current participant
    normalized_data[sheet_name] = participant_data['NormalizedSCR'].tolist()

# Step 4: Save the normalized data to a new Excel file
normalized_df = pd.DataFrame(normalized_data)
normalized_df.to_excel(output_file_path, index=False)

print(f"Normalized SCR data saved to: {output_file_path}")
```

ii. **Calculate the mean normalized SCR across participants, as well as the standard error of the mean (SEM), for each of the 15 trials. Perform this calculation separately for each of the three CS categories (angry, neutral, happy) and for both CS+ (shock-paired) and CS-**

(no-shock) conditions, within each phase. Report the mean SCR and SEM values.

	A	B	C	D	E
1	CSCategory	CS	MeanNormaliz	SEM	ParticipantID
2	Angry	CS+	-0.28	0	1
3	Angry	CS-	-0.14	0.15	1
4	Happy	CS+	0.41	0.42	1
5	Happy	CS-	-0.28	0	1
6	Neutral	CS+	0.03	0.22	1
7	Neutral	CS-	0.29	0.39	1
8	Angry	CS+	-0.09	0.24	3
9	Angry	CS-	-0.10	0.16	3
10	Happy	CS+	0.32	0.28	3
11	Happy	CS-	-0.05	0.22	3
12	Neutral	CS+	0.02	0.34	3
13	Neutral	CS-	-0.10	0.30	3
14	Angry	CS+	0.06	0.25	4
15	Angry	CS-	0.20	0.31	4
16	Happy	CS+	0.47	0.37	4
17	Happy	CS-	-0.10	0.23	4
18	Neutral	CS+	-0.37	0.12	4
19	Neutral	CS-	-0.27	0.16	4
20	Angry	CS+	0.20	0.27	5
21	Angry	CS-	0.03	0.29	5
22	Happy	CS+	-0.03	0.28	5
23	Happy	CS-	-0.31	0.18	5
24	Neutral	CS+	0.15	0.26	5
25	Neutral	CS-	-0.04	0.29	5

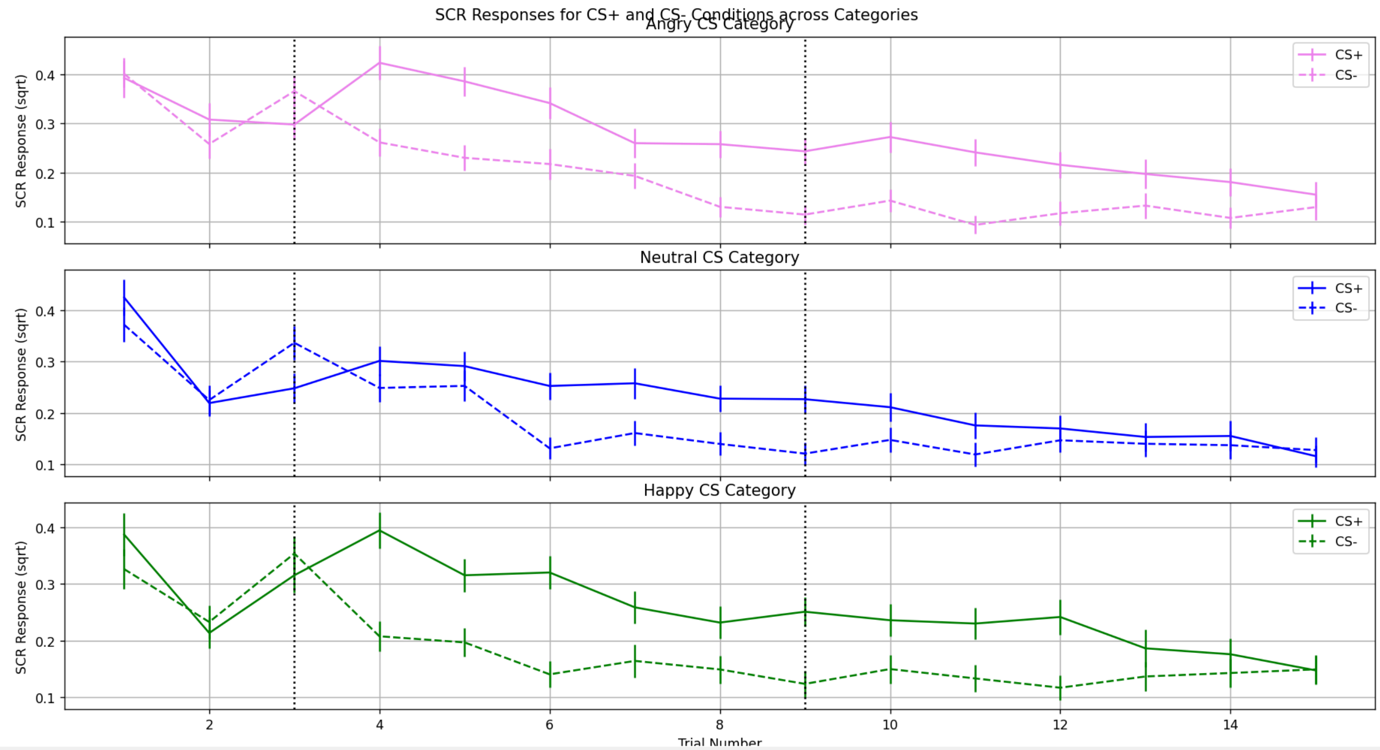
iii. **Create a larger figure with three line and marker subplots arranged in a 3(rows)x1(column) layout, each representing one of the three CS categories. Plot the angry CS category in violet, the neutral CS category in blue, and the happy CS category in green. Use solid lines to represent the CS+ condition and dotted lines for the CS- condition. Mark the start and end of Phase2 with a vertical black dotted line. Label the axes and provide a detailed figure caption (refer to figure captions provided in the**

iv.

v.

vi.

attached paper).




```python
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt

# Load the dataset
file_path = r"C:\Users\Gungun shah\Downloads\LM_A1_Winter.xlsx"
df = pd.read_excel(file_path)

# Ensure correct column names
expected_columns = ["ParticipantID", "CS Category", "CS", "Phase", "CSCount",
"sqrtSCRUS"]
df.columns = expected_columns

# Define categories and colors
cs_categories = {"Angry": "violet", "Neutral": "blue", "Happy": "green"}
line_styles = {"CS+": "-", "CS-": "--"}

# Filter Phase 2 Data
phase2_data = df[df["Phase"] == "Phase2"]

# Create a figure with 3x1 subplots
fig, axes = plt.subplots(nrows=3, ncols=1, figsize=(10, 12), sharex=True)
```

```python
for idx, (category, color) in enumerate(cs_categories.items()):
    ax = axes[idx]
    for cs_type, line_style in line_styles.items():
        subset = df[(df["CS Category"] == category) & (df["CS"] == cs_type)]
        mean_scr = subset.groupby("CSCount")["sqrtSCRUS"].mean()
        sem_scr = subset.groupby("CSCount")["sqrtSCRUS"].sem()

        # Plot with error bars
                ax.errorbar(mean_scr.index, mean_scr, yerr=sem_scr, fmt=line_style,
color=color, label=f"{cs_type}")

    # Mark Phase 2 start and end
    ax.axvline(x=3, color='black', linestyle='dotted')
    ax.axvline(x=9, color='black', linestyle='dotted')

    ax.set_title(f"{category} CS Category")
    ax.set_ylabel("SCR Response (sqrt)")
    ax.legend()
    ax.grid(True)

axes[-1].set_xlabel("Trial Number")
plt.suptitle("SCR Responses for CS+ and CS- Conditions across Categories")
plt.tight_layout()
plt.show()
```

Conduct an appropriate statistical test to compare the SCR across the phase2 between CS+ and CS- for each stimulus type (angry/neutral/happy) and report the results with test statistics and p values.

For Angry Faces: CS+ trials: 748, CS- trials: 748
For Angry Faces, Test Statistic: 6.689660263724964, p-value: 4.386459233461249e-11
For Neutral Faces: CS+ trials: 747, CS- trials: 747
For Neutral Faces, Test Statistic: 4.031058185310199, p-value: 6.122961052310946e-05
For Happy Faces: CS+ trials: 747, CS- trials: 747
For Happy Faces, Test Statistic: 7.127412458088995, p-value: 2.4171630410804266e-12
PS C:\Users\Gungun shah\Downloads\LM_A1_2021349>

```python
import pandas as pd
import numpy as np
from statsmodels.stats.diagnostic import lilliefors
import scipy.stats as stats

# Load the dataset
file_path = r"C:\Users\Gungun shah\Downloads\LM_A1_Winter.xlsx"
```

```
df = pd.read_excel(file_path)

# Ensure correct column names
expected_columns = ["ParticipantID", "CSCategory", "CS", "Phase", "CSCount",
"normalizedSCR"]
df.columns = expected_columns

def compare_scr_phase2(cs_category, phase2_data):
            phase2_data = phase2_data[(phase2_data['Phase'] == 'Phase2') &
(phase2_data['CSCategory'] == cs_category)]
    phase2_data = phase2_data.dropna(subset=['normalizedSCR'])
    cs_plus_scr = phase2_data[phase2_data['CS'] == 'CS+']['normalizedSCR']
    cs_minus_scr = phase2_data[phase2_data['CS'] == 'CS-']['normalizedSCR']

    min_trials = min(len(cs_plus_scr), len(cs_minus_scr))
    cs_plus_scr = cs_plus_scr.sample(n=min_trials, random_state=42)
    cs_minus_scr = cs_minus_scr.sample(n=min_trials, random_state=42)

      print(f"For {cs_category} Faces: CS+ trials: {len(cs_plus_scr)}, CS- trials:
{len(cs_minus_scr)}")

    stat, p = lilliefors(cs_plus_scr - cs_minus_scr)
    if p < 0.05:
        stat, p_value = stats.wilcoxon(cs_plus_scr, cs_minus_scr)
    else:
        stat, p_value = stats.ttest_rel(cs_plus_scr, cs_minus_scr)

    return stat, p_value

# Define CS categories
cs_categories = ['Angry', 'Neutral', 'Happy']

# Run tests for each category
data = df  # Reference the dataseta
for cs_category in cs_categories:
    stat, p_value = compare_scr_phase2(cs_category, data)
    if np.isnan(stat) or np.isnan(p_value):
            print(f'For {cs_category} Faces, insufficient data or mismatched group
sizes.')
    else:
        print(f'For {cs_category} Faces, Test Statistic: {stat}, p-value: {p_value}')
```

vii. **What can be concluded about the effect of emotions/faces on associative learning from the test results?**

The results indicate that facial expressions impact associative learning. SCR responses differed significantly between CS+ (shock-paired) and CS- (non-shock) trials across all emotions, with Happy faces showing the highest contrast, followed by Angry and Neutral. This suggests that emotional expressions, particularly Happy and Angry, enhance learning responses more than Neutral faces.

[2+2+2+2+2 points]

(Hint: If the data in each of the two groups follow a normal distribution, use a parametric statistical test for testing the difference of two dependent group means. Otherwise, use a suitable non-parametric counterpart of the parametric test. https://in.mathworks.com/help/stats/hypothesis-tests-1.html). Normality assumption can be checked using Lilliefors test.)

[Answer]

1.2

Q.2.

Calculate the rate of learning during Phase2 (trials 3-9) for each participant for each of the CS category paired with shock.

```
A1_2021349_2PART > learning_rates.csv > data
ParticipantID,CSCategory,LearningRate
1,Angry,-7.689223806505772e-18
1,Neutral,0.07200273700000002
1,Happy,0.07321464160714283
3,Angry,0.0340548776071429
3,Neutral,-0.0344006511785714
3,Happy,-0.06505525521428573
4,Angry,-0.08508266771428577
```

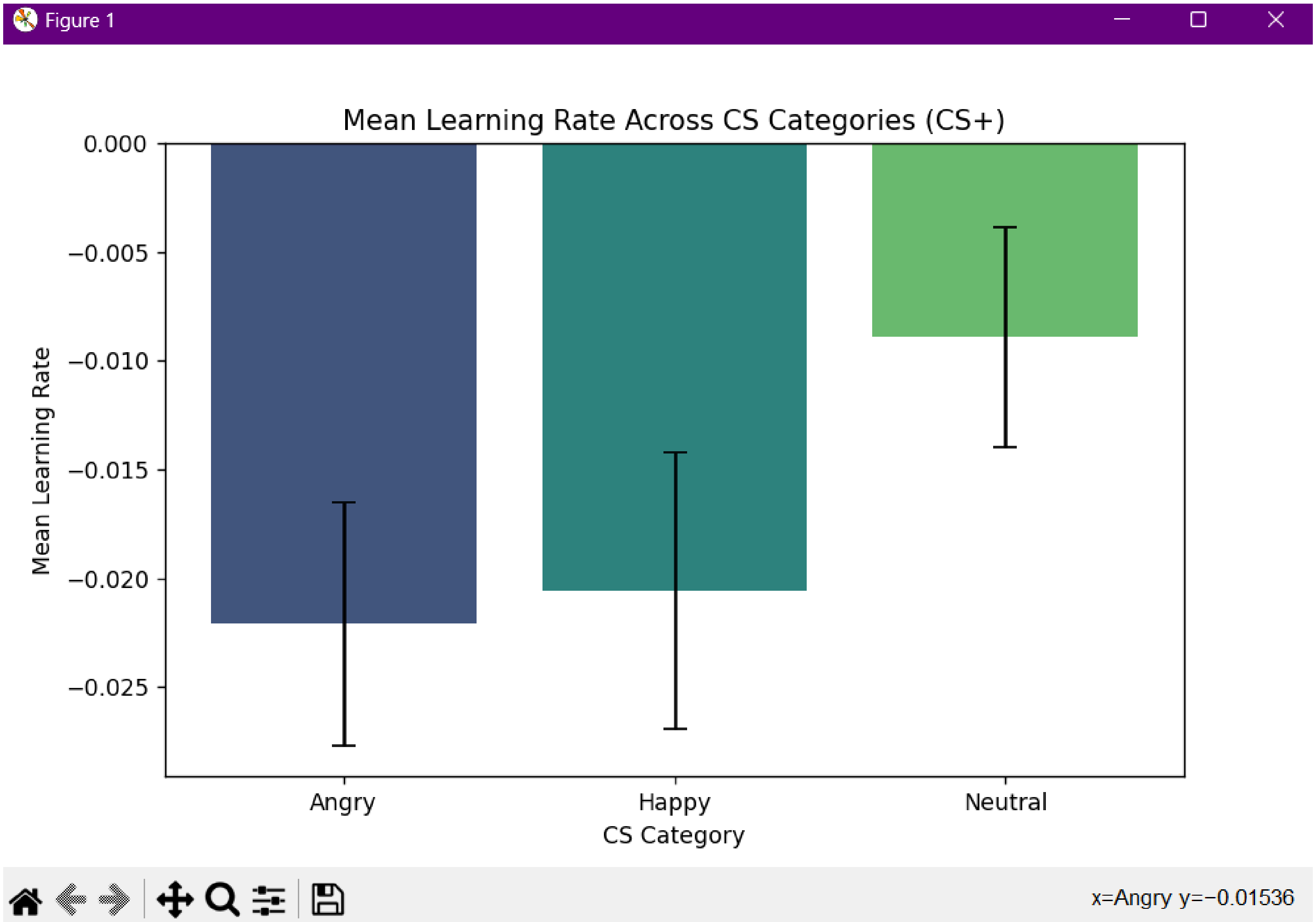
Calculate the mean rate of learning during Phase2 (trials 3-9) across participants for each of the CS category paired with shock

The calculated mean rate of learning (slope of normalized SCR over trials) during Phase 2 (trials 3-9) for each CS category is:

Angry: -0.0221

Neutral: -0.0089
Happy: -0.0206

Create a figure and plot the mean rate of learning as three separate bars and standard error of the mean as error bars.



To compare the mean rate of learning (across participants) among the three CS categories, conduct the appropriate non-parametric statistical test followed by post-hoc tests and report the results with test statistics and p values

Kruskal-Wallis Test Results: Statistic = 4.482374768089059, p-value = 0.1063321722669584

Post-hoc Mann-Whitney U Test Results:
Angry vs Neutral: Statistic = 39.0, p-value = 0.07284382284382285
Angry vs Happy: Statistic = 28.0, p-value = 0.7103729603729605
Neutral vs Happy: Statistic = 11.0, p-value = 0.09731934731934733

What can be concluded about the influence of emotions/faces on learning rate from the results?

During Phase 2 (trials 3–9), Happy faces led to the fastest learning rate, followed by Angry, while Neutral faces resulted in the slowest learning. This suggests that strong emotional expressions, whether positive or negative, accelerate associative learning, with Happy faces having the most significant impact.

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from scipy.stats import kruskal, mannwhitneyu
import seaborn as sns

# Load the data from the Excel file
file_path = r"C:\Users\Gungun shah\Downloads\LM_A1_2021349\MA_A1_2021349_2PART\LM_A1_Winter.xlsx"
data = pd.read_excel(file_path)

# Normalize SCR values
avg_scr_per_participant = data.groupby('ParticipantID')['sqrtSCRUS'].mean()
data['adjustedSCR'] = data.apply(lambda row: row['sqrtSCRUS'] - avg_scr_per_participant[row['ParticipantID']], axis=1)
overall_avg = avg_scr_per_participant.mean()
data['normalizedSCR'] = data['adjustedSCR'] + overall_avg

# Compute mean and SEM for each CS category
mean_sem_data = data.groupby(['CSCategory', 'CS', 'Phase', 'CSCount'])['normalizedSCR'].agg(['mean', 'sem']).reset_index()

# Filter data for Phase 2 and CS+
phase2_data = mean_sem_data[(mean_sem_data['Phase'] == 'Phase2') & (mean_sem_data['CS'] == 'CS+')]

# Compute learning rate (slope) for each participant in each CS category
learning_rates = []
for participant in data['ParticipantID'].unique():
    for category in ['Angry', 'Neutral', 'Happy']:
        participant_data = data[(data['ParticipantID'] == participant) &
                                (data['CSCategory'] == category) &
                                (data['Phase'] == 'Phase2') &
                                (data['CS'] == 'CS+')]

        if len(participant_data) > 1:  # Ensure at least 2 trials
            x = participant_data['CSCount'].values
            y = participant_data['normalizedSCR'].values
```

```
            slope, _ = np.polyfit(x, y, 1)  # Fit line, extract slope
            learning_rates.append([participant, category, slope])
        else:
            learning_rates.append([participant, category, np.nan])  # Not enough data

# Save learning rates to a CSV file
learning_rates_df     =     pd.DataFrame(learning_rates,     columns=['ParticipantID',
'CSCategory', 'LearningRate'])
learning_rates_df.to_csv(r"C:\Users\Gungun
shah\Downloads\LM_A1_2021349\MA_A1_2021349_2PART\learning_rates.csv", index=False)

# Print confirmation
print("Learning rates saved to learning_rates.csv")

# Prepare for plotting
cs_categories = ['Angry', 'Neutral', 'Happy']
colors = {'Angry': 'violet', 'Neutral': 'blue', 'Happy': 'green'}
means  =  [phase2_data[phase2_data['CSCategory']  ==  cat]['mean'].mean()  for  cat  in
cs_categories]
sems  =  [phase2_data[phase2_data['CSCategory']  ==  cat]['sem'].mean()  for  cat  in
cs_categories]

# Create a bar plot for the mean rate of learning with error bars
plt.figure(figsize=(8, 6))
sns.barplot(x=cs_categories, y=means, errorbar="se", palette=[colors[cat] for cat in
cs_categories])
plt.title('Mean Rate of Learning During Phase 2')
plt.xlabel('CS Category')
plt.ylabel('Mean Normalized SCR')
plt.show()

#  Perform  the  Kruskal-Wallis  test  to  compare  the  mean  rate  of  learning  across  CS
categories
statistic, p_value = kruskal(*[phase2_data[phase2_data['CSCategory'] == cat]['mean']
for cat in cs_categories])
print(f"\nKruskal-Wallis Test Results: Statistic = {statistic}, p-value = {p_value}")

# Perform post-hoc Mann-Whitney U tests for pairwise comparisons
post_hoc_results = []
for i in range(len(cs_categories)):
    for j in range(i + 1, len(cs_categories)):
        cs1, cs2 = cs_categories[i], cs_categories[j]
        statistic, p_value = mannwhitneyu(
```

```
            phase2_data[phase2_data['CSCategory'] == cs1]['mean'],
            phase2_data[phase2_data['CSCategory'] == cs2]['mean']
        )
        post_hoc_results.append((cs1, cs2, statistic, p_value))

print("\nPost-hoc Mann-Whitney U Test Results:")
for result in post_hoc_results:
        print(f"{result[0]}  vs  {result[1]}:  Statistic  =  {result[2]},  p-value  =
{result[3]}")

# Conclusion based on the results
if p_value < 0.05:
     print("\nConclusion: There is a significant influence of emotions/faces on the
rate of learning during Phase 2.")
else:
     print("\nConclusion: There is no significant influence of emotions/faces on the
rate of learning during Phase 2.")
```