Weekly Work Report 10/4/2024

**This Week:**

**EEG Data Preprocessing in eeglab**

* Preprocessing for 20 subjects’ EEG data was completed this week.
* To date, 39 subjects (subject\_1 to subject\_39) have been finished preprocessing in EEGLAB.

**Label**

Label = 0: Baseline, 1: Body Movement, 2: Aha Moment, 3: Confirmation Menu

A bar of numbers with green and orange squares

Description automatically generated with medium confidence

**Classification**

**10mins – every time Aha moments v non-Aha moments**

**Positive vs Negative**

Before Aha! Moment (Physical Body Movement) vs. Aha! Moment

* Time-Frequency
  + time window size = 3s
  + 16 Channels: max, min, std, mean

RF:

Accuracy: 0.7142857142857143

F1-score: 0.712987012987013

Precision: 0.7248677248677249

Recall: 0.7142857142857143

A blue squares with white text

Description automatically generated

A graph of a curve

Description automatically generated

# Binarize the labels for ROC curve

y\_test\_bin = label\_binarize(y\_test, classes=np.unique(y))

y\_pred\_prob = knn\_classifier.predict\_proba(X\_test)

# Compute ROC curve and ROC area for each class

n\_classes = y\_test\_bin.shape[1]

fpr = dict()

tpr = dict()

roc\_auc = dict()

for i in range(n\_classes):

fpr[i], tpr[i], \_ = roc\_curve(y\_test\_bin[:, i], y\_pred\_prob[:, i])

roc\_auc[i] = auc(fpr[i], tpr[i])

# Plot ROC curve for each class

plt.figure(figsize=(8, 6))

colors = cycle(['blue', 'red', 'green', 'orange'])

for i, color in zip(range(n\_classes), colors):

plt.plot(fpr[i], tpr[i], color=color, lw=2,

label='ROC curve of class {0} (area = {1:0.2f})'

''.format(i, roc\_auc[i]))

plt.plot([0, 1], [0, 1], 'k--', lw=2)

plt.xlim([0.0, 1.0])

plt.ylim([0.0, 1.05])

plt.xlabel('False Positive Rate')

plt.ylabel('True Positive Rate')

plt.title('Receiver Operating Characteristic (ROC) Curve')

plt.legend(loc="lower right")

plt.show()

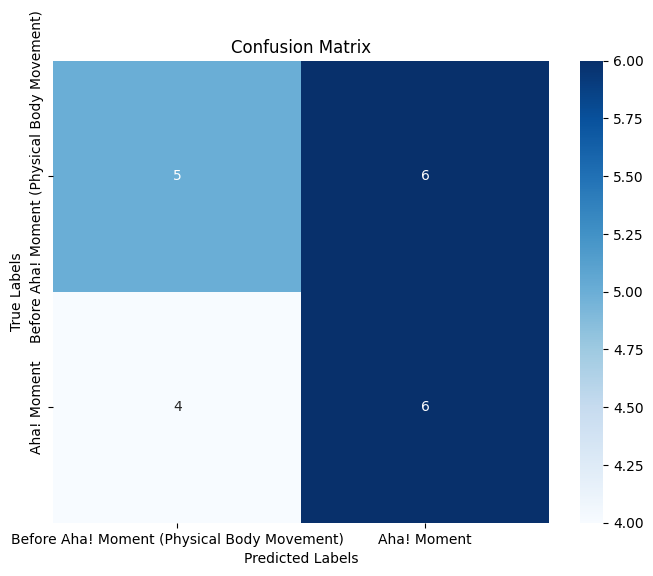
KNN

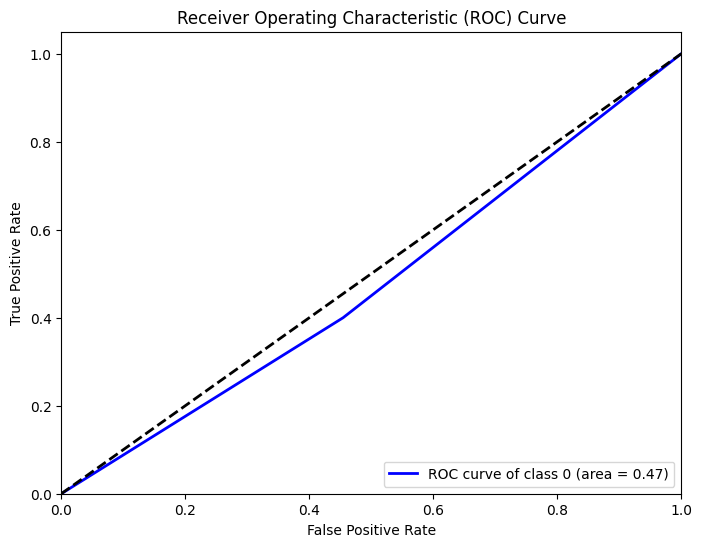
Accuracy: 0.5238095238095238

F1-score: 0.5216450216450216

Precision: 0.5291005291005291

Recall: 0.5238095238095238





Aha! Moment vs After Aha! Moment (Menu Confirmation)

* Time-Frequency
  + time window size = 3s
  + 16 Channels: max, min, std, mean

RF

Accuracy: 0.6666666666666666

F1-score: 0.6666666666666666

Precision: 0.7467948717948717

Recall: 0.6666666666666666

A blue squares with white text

Description automatically generated

A graph of a line graph

Description automatically generated with medium confidence

KNN

Accuracy: 0.6190476190476191

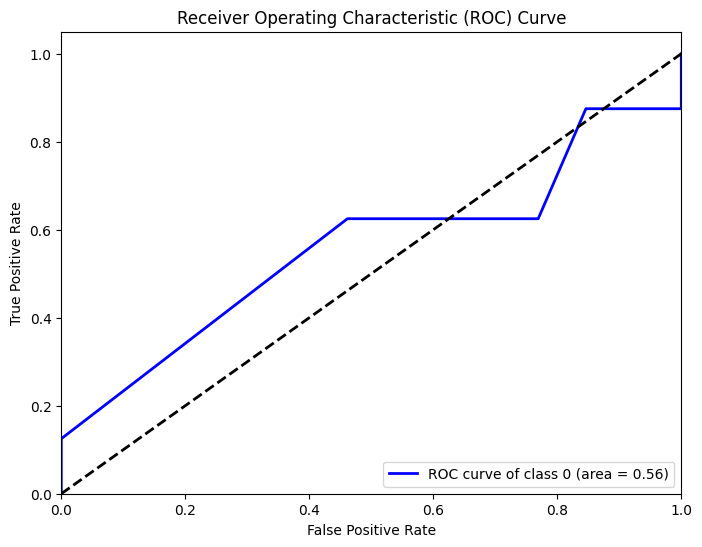
F1-score: 0.6054421768707483

Precision: 0.6031746031746031

Recall: 0.6190476190476191

A blue squares with white text

Description automatically generated



Features\_frequency\_band = ['FP1', 'FP2', 'C3', 'C4', 'P8', 'O1', 'O2', 'F7', 'F8', 'F3', 'F4', 'T7', 'T8', 'P3', 'P7', 'P4', 'Frequency']

import numpy as np

import pandas as pd

from scipy.fft import fft, fftfreq

from scipy.signal import welch

# Load preprocessed EEG data

data = pd.read\_csv('/content/drive/MyDrive/MX\_2/EEG/Subject\_20\_EEG\_Exam\_OpenBCI\_Processed.csv')

# Define EEG frequency bands

**freq\_bands** = {

'delta': (0.5, 4),

'theta': (4, 8),

'alpha': (8, 13),

'beta': (13, 30),

'gamma': (30, 100)

}

# Sampling rate (in Hz) - should be known from the data acquisition

fs = 125 # example sampling rate

# Initialize a dictionary to store frequency band power for each channel

freq\_band\_data = {band: [] for band in freq\_bands.keys()}

# Apply FFT or Welch's method to each channel

for channel in data.columns[1:-2]:

# Calculate power spectral density using Welch's method

freqs, psd = welch(data[channel], fs, nperseg=1024)

# For each frequency band, calculate the total power in that band

**for band, (low, high) in freq\_bands.items():**

**band\_power = np.sum(psd[(freqs >= low) & (freqs <= high)])**

**freq\_band\_data[band].append(band\_power)**

# Create a DataFrame from the frequency band data

freq\_band\_df = pd.DataFrame(freq\_band\_data, index=data.columns[1:-2])

# # Save the frequency band data to a CSV file

# freq\_band\_df.to\_csv('eeg\_frequency\_bands.csv')

**Others:**

* In response to Gai’s request, I developed an Aha! moment prediction model by incorporating the WMC score.
* Met with Noviya and addressed her questions.

**Next Week:**

* Conduct a literature review on labeling, brain structure, and functionality.
* Begin processing additional signals, including EDA and pupil data.
* Perform classification by adding more subjects' EEG data.
* Explore results using different models.
* Precision-Recall curve (e.g. 0.71), plot Aha vs Non-Aha.
* Two valuations, validated in negative signals, n-folds: remove 1 from n
* Other frequency bands, more features, time domain, frequency domain features

<https://github.com/universalus/EEG-States-Classification>

<https://scikit-learn.org/stable/auto_examples/model_selection/plot_roc.html#sphx-glr-auto-examples-model-selection-plot-roc-py>

10 seconds

**Final Goal:**

* Explore the dynamics of attention, stuck states, and the "Aha!" moment.