ECE 445

SENIOR DESIGN LABORATORY

FINAL REPORT

A Wearable Device That Can Detect Mood

Team #32

JUNJIE REN
(junjier2@illinois.edu)
KEJUN WU
(kejunwu2@illinois.edu)
PEIDONG YANG
(peidong5@illinois.edu)
XINZHUO LI
(xinzhuo4@illinois.edu)

TA: GUO HAO

May 5, 2024

Abstract

Understanding and managing our moods are crucial for both mental well-being and physical health. Stress, depression, and anxiety can lead to health issues, impacting daily life and productivity. This project introduces a wearable device system designed to record human mood dynamics. The device captures physiological signals including heart rate variability (HRV), skin resistance (EDA), skin temperature (SKT) linked to emotional states of arousal and valence, providing an objective method for mood recognition using sensor technology. Random Forest (RF) algorithm and Extreme Gradient Boosting (XGBoost) algorithm are utilized to classify the mood. This project represents a solution in combining psychological knowledge with wearable technology to improve individual and societal well-being.

Contents

1 Introduction

1.1 Purpose

Understanding and managing our moods are significant to both our mental well-being and physical health. Mood changes, if not appropriately addressed, can lead to long-term health issues, such as headaches, asthma, heart disease, and even self-harming activities [1], which significantly impact our daily lives. Our project addresses the escalating concern of mental health issues, such as stress, anxiety, and depression, which profoundly impact individuals' health, productivity, and social well-being. Our solution is a wearable device, a wrist worn smart device similar to Yang's [2], which not only detects these mood changes through physiological signals but also offers real-time feedback and suggestions for mood management. This proactive approach aims to help people manage their moods and to mitigate the negative effects of mood fluctuations, thus enhancing overall well-being and preventing the development of more serious health conditions.

1.2 Functionality

The functionality of our wearable device encompasses a comprehensive approach to mood detection through advanced sensor technology and machine learning. By continuously measuring physiological signals - heart rate, successive heartbeat intervals, skin temperature, and skin conductance, the device captures the emotional states of users in real-time. These physiological indicators are analyzed using machine learning algorithms, enabling precise assessments of mood changes. Based on this analysis, the device offers real-time feedback and suggestions for mood management, empowering users to proactively address their emotional health issues. The device also offers benefits in various settings such as hospitals, schools, and workplaces by aiding in the early detection of mental health issues, and fostering supportive environments. Through its holistic approach, the device not only enhances individual health but also contributes to healthier communities by integrating technology with psychological insights to manage and improve mental health.

1.3 Subsystem Overview

Our project is divided into five subsystems. The block diagram 1 depicts a system designed to meet the requirements for mood detection, real-time processing and feedback, user interface and display, and user privacy and data security. The sensors collect physiological data that the system preprocesses with three steps: signal smoothing, outlier removal, and normalization to ensure quality and reliability, which is crucial for achieving the required accuracy of at least 0.40 in mood classification. The preprocessed data is then subjected to feature extraction with time-frequency analysis and fed into trained machine learning models within the Classification Subsystem to predict mood states with desired accuracy. Real-time processing is facilitated through direct connections with an Arduino, which interfaces with the sensors for continuous data measurement. User feedback is immediate via the Display Subsystem, which shows the outcome of mood predictions on a GUI, providing real-time insight and suggestions. Additionally, by storing and processing data locally, the system aligns with privacy and security standards, ensuring user data is handled confidentially and securely.

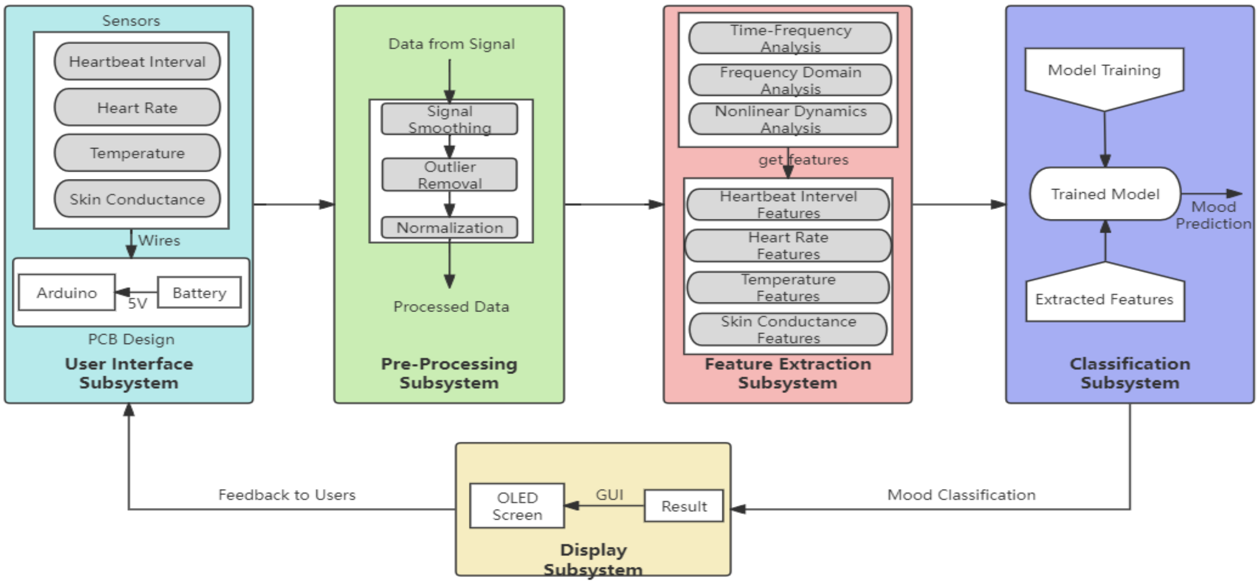


Figure 1: Block Diagram. It showcases a mood prediction system integrating sensors for data collection, preprocessing for quality, and advanced analysis for accurate classification. Features real-time Arduino interfacing, and immediate GUI feedback

2 Design

2.1 Design Procedure

2.1.1 User Interface Subsystem

In the user interface subsystem, we apply four physiological signals, including pulse rate, successive heartbeat interval (HRV), skin resistance (GSR), and skin temperature (SKT), to detect the mood.

Heart rate variability (HRV) is the index of the inconsistent gaps between each heartbeat in psychology. The classical method to detect HRV is the electrocardiogram (ECG) method. However, undergraduate students do not have access to sophisticated ECG devices. It is also complicated to analyze ECG signals in the time domain and the frequency domain. As a result, the alternative photoelectric volumetric pulse wave (PPG) method is chosen. PPG utilizes the optical signal obtained from changes in light absorption. The blood flow rate is reflected by the change in light absorption [3].

The galvanic skin response (GSR), also known as electrodermal activity (EDA) or skin conductance (SC), is a measurement of electrical conductance in the skin. According to the traditional theory, electrical parameters of the skin are not under conscious human control and they reflect sympathetic nervous system changes [4]. Strong emotions unconsciously stimulate the human's sympathetic nervous system, causing the secretion of more sweat and the change in potential response. The traditional signal processing method is chosen at the beginning to process the sensor output. However, the GSR signal containing both low and high-frequency components does not always respond in the same way to the same stimulus. Therefore, the machine learning algorithm is applied eventually to improve the accuracy of emotion recognition and identify specific emotions, such as excitement and sadness.

Skin temperature is another autonomic nervous system response beyond human control. It is mainly used for the estimation of emotion between pleasant and unpleasant emotion judgment. SKT is decreased by depression, guilt, anger, anxiety, and embarrassment [5]. Generally speaking, the temperature measurement can be divided into contact methods based on sensors and non-contact methods based on infrared thermal imagers. Compared to contact sensors, infrared measurement have the characteristics of relatively high resolution, fast response speed, and good stability. However, for undergraduate students, an advanced infrared thermal imager is unavailable. Eventually, an infrared temperature sensor is chosen to detect human skin temperature.

2.1.2 Pre-processing Subsystem

The pre-processing subsystem serves as the initial filter for the data collected from wearable sensors and the online dataset. Its main function is to clean the raw data by removing noise and irrelevant information that could negatively impact the accuracy of mood recognition. This involves techniques of signal smoothing, outlier removal, and normalization. The preprocessed data is a more reliable representation of the user's physiological state. Other methods may be used as well, but we do not implement them with the following reasons.

- **Median Filtering** is a non-linear technique used to remove noise while preserving edges. In our case, it can be less effective than other smoothing techniques because the signal noise is not impulsive.
- **Adaptive Filtering** can be used to deal with signals whose statistical properties are unknown or changing over time. However, adaptive filters can be complex to implement and require careful tuning of their parameters.

2.1.3 Feature Extraction Subsystem

The feature extraction subsystem aims at distilling key indicators from the preprocessed data that are relevant for mood recognition. It translates raw sensor data into a set of features that reflect the user's physiological patterns associated with different moods, providing informative and discriminative features. It interfaces with both the pre-processing subsystem, from which it receives the cleaned data, and the mood classification model, to which it sends the extracted features. The feature extraction subsystem focuses on Time Frequency Analysis, in which Short-Time Fourier Transform (STFT) is used.

Alternative approaches to STFT include Wavelet Transform, Discrete Wavelet Transform (DWT), and Continuous Wavelet Transform (CWT).

- **Discrete Wavelet Transform (DWT)** decomposes the signal into different frequency bands using wavelet functions, offering representation of non-stationary signals. While DWT has its advantages, STFT provides a more direct representation of signal characteristics in the time-frequency domain, making it more suitable for capturing transient features and time-varying frequency components, which are essential for mood recognition tasks.
- **Continuous Wavelet Transform (CWT)** offers variable time-frequency resolution by using wavelet functions of different scales, but STFT achieves similar adaptability by adjusting the size of the analysis window. STFT's simplicity, computational efficiency, and direct representation of signal characteristics make it a preferred choice for feature extraction in mood recognition systems.

2.1.4 Classification Subsystem

Classification subsystem employs machine learning algorithms to interpret the features extracted from the user's data and classify them into mood states such as happy, sad, calm, or angry. It utilizes supervised learning to train models on labeled mood data. The mood classification model interfaces with the feature extraction subsystem, receiving features as input, and with the display subsystem, providing the mood predictions.

Random Forest (RF) algorithm and Extreme Gradient Boosting (XGBoost) algorithm are used in classification, and we finally balance the trade-off between these two powerful algorithms, mainly considering the classification results of valence and arousal.

Alternative approaches to mood classification include Support Vector Machines (SVM), K-Nearest Neighbors (KNN) and Neural Networks (NN). However, RF and XGBoost stand out by following advantages.

- **Ensemble Learning Advantage:** RF and XGBoost utilize ensemble learning, combining multiple decision trees to form robust models. This approach helps to handle complex, non-linear relationships between features and mood states effectively, surpassing the individual model performance of SVM, KNN and NN.
- **Robustness to Overfitting:** RF and XGBoost are robust to overfitting, thanks to techniques such as bagging and boosting. These methods aggregate predictions from multiple weak learners, mitigating overfitting and improving generalization performance. In contrast, SVM and NN may be more susceptible to overfitting, especially with the complex psychological datasets used to predict mood.
- **Computational Efficiency:** RF and XGBoost are computationally efficient and scalable, making them suitable for mood classification tasks. In contrast, NN, particularly deep learning models, often require extensive computational resources and large amounts of data for training, limiting their applicability in mood classification scenarios.

2.1.5 Display Subsystem

As a real-time mood detection wearable device system, the display subsystem is crafted to present the identified mood and relevant suggestions to the user in an intuitive and captivating manner, which is pivotal for enriching user experience. Our strategy involves showcasing mood states through text labels alongside pertinent icons or emojis, aligning with the specific mood detected by the classification subsystem. To bolster the feedback, we offer suggestions based on the identified mood. For instance, suggestions for activities like socializing or playing video games might be provided to help users relax and uplift their moods. This subsystem elevates user engagement by delivering real-time feedback aimed at enhancing or sustaining emotional well-being.

We implement Graphical User Interface (GUI) display on the computer reflecting users' mood with four kinds of emojis corresponding to the emotional states. The mood detection result from Classification Subsystem is immediately going through the Display Subsystem, presenting mood predictions on a GUI for prompt user response. The system prioritizes privacy and security by storing and processing data locally, adhering to confidentiality and security standards.

Our GUI is created by utilizing PyQt5, a powerful Python library. PyQt5 is built on top of the Qt framework, which is a popular cross-platform application development toolkit working across different operating systems. PyQt5 provides a wide range of customizable widgets, such as buttons, labels, text boxes, and sliders, allowing developers to create rich and interactive user interfaces. Developers can easily handle user interactions and events within their PyQt5 applications. There are also many alternative methods to create GUI, including WinForms and Windows Presentation Foundation, but these frequently-used methods are only limited on Windows operating system.

2.2 Design Details

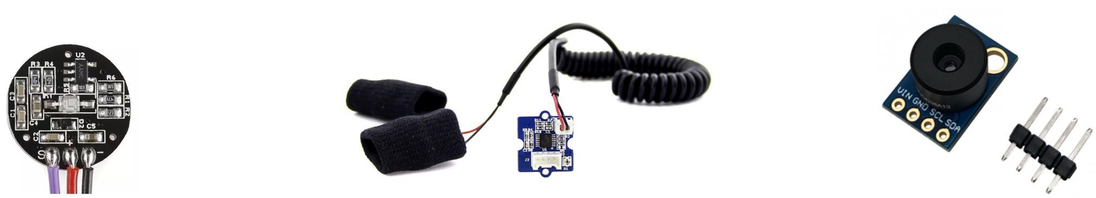
2.2.1 User Interface Subsystem

The PulseSensor is shown in (a) in Figure 2. This pulse rate sensor detects two kinds of data: the heart rate (BPM) and the successive heartbeat intervals. When the signal exceeds the threshold, the heartbeat will be registered.

The skin conductance sensor's physical graph is shown in (b) of Figure 2. Only two electrodes need to be placed on the second and third fingers of one hand, in which the resistance of the skin to a small current from an external source is measured. Notice that the equation 1 is used to transform the serial port output and calculate the final human resistance (ohm). The Serial_Port_Reading is the value display on Serial Port (between 0 to 1023). Serial_calibration is the serial port data after we use the screwdriver to adjust the resistor until the serial output is minimized.

$$\text{Human Resistance} = \frac{(1024 + 2 \times \text{Serial\_Port\_Reading}) \times 100000}{\text{Serial\_calibration} - \text{Serial\_Port\_Reading}} \tag{1}$$

To measure the skin temperature, we choose the GY906-DCC infrared temperature sensor as shown in (c) of Figure 2. The DCC represents the sensor measurement accuracy. It means medical accuracy up to 280 degrees and can infrared measure temperatures up to 10cm.



(a) (b) (c)

Figure 2: Sensors

The schematic of pulse rate sensor is showed in Figure 3. It utilizes the MCP6001 operational amplifier (op-amp), a single general-purpose op-amp, to amplify the signal from the pulse sensor and remove noise from the signal. The APDS-9008 is an ambient light photo sensor.

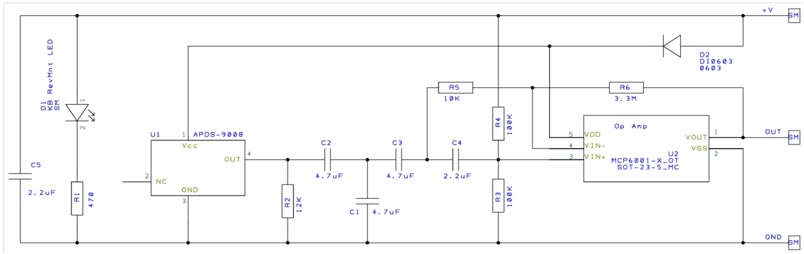


Figure 3: Pulse Rate Sensor Schematic

The detailed circuit of GSR sensor is displayed below in Figure 4. This sensor module is built around the LM324 IC.

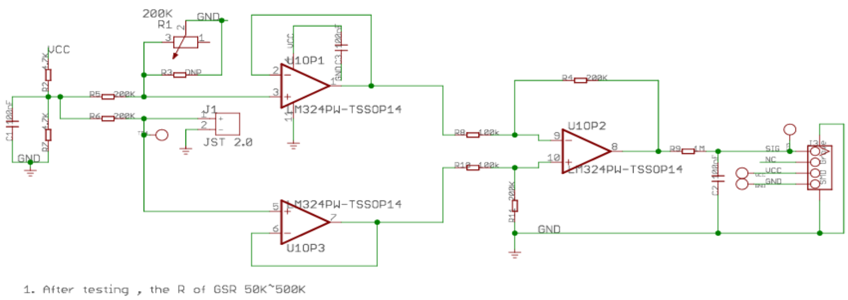


Figure 4: GSR Sensor Schematic

The skin temperature sensor contains an MLX90614 series module, which is a set of infrared temperature measurement modules. The detailed schematic of the infrared temperature sensor is displayed in Figure 5.

The Arduino Uno works as the microcontroller to synchronize all sample rates of sensors to 1Hz. All sensors are connected to the Arduino Uno via its analog input pins using wire. The wiring diagram is displayed below in Figure 6.

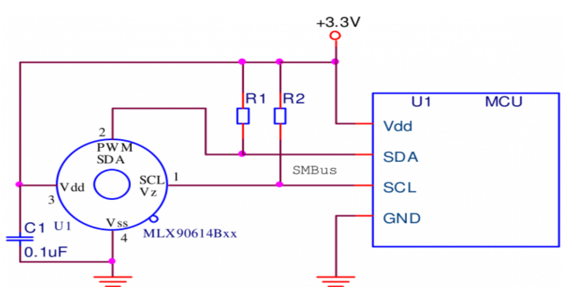


Figure 5: Infrared Temperature Sensor Schematic

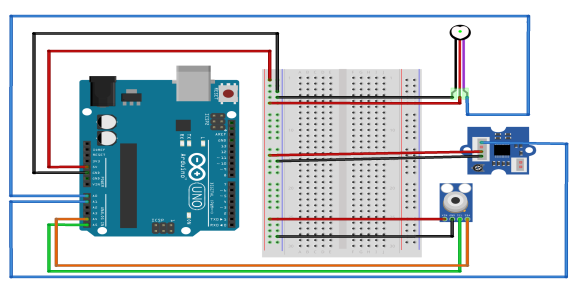


Figure 6: Sensor Arduino Wiring

2.2.2 Pre-processing Subsystem

The pre-processing subsystem contains three successive parts. They are Signal Smoothing, Outlier Removal, and Normalization.

Signal Smoothing is the first step in the data preprocessing pipeline aimed at reducing noise and fluctuations in the raw sensor data. We have applied the moving average method that averages neighboring data points. The moving average filter is a widely used technique in signal processing for its simplicity and effectiveness in attenuating high-frequency components while preserving the underlying signal trends.

The moving average filter operates by computing the mean of a sliding window of consecutive data points along the signal. The size of the window, often referred to as the "window length" or "kernel size," determines the degree of smoothing. As the window slides along the signal, each data point is replaced with the average of itself and its neighboring points, resulting in a smoothed signal. The moving average of a signal $x(t)$ at time t with a window size N is computed as:

$$\bar{x}(t) = \frac{1}{N} \sum_{i=t-N+1}^{t} x(i) \tag{2}$$

Where $\bar{x}(t)$ is the moving average of the signal at time t, $x(i)$ is the value of the signal at time i, N is the size of the moving window.

Figure 18 in the Appendix shows the algorithm for the moving average filter.

We use Signal-to-Noise Ratio (SNR) to quantify the ratio of signal power to noise power present in a signal. A higher SNR indicates a stronger, more dominant signal compared to the noise. The formula to calculate SNR is as follows:

$$\text{SNR} = 10\log(\frac{\text{Signal Power}}{\text{Noise Power}}) \tag{3}$$

Another measurement is the Mean Squared Error (MSE), which measures the average squared difference between the original signal and the smoothed signal. A lower MSE value is generally preferable, indicating that the filtered signal remains close to the original signal while hopefully reducing noise.

$$\text{MSE} = \frac{1}{N} \sum_{i=1}^{N} (x_i - \hat{x}_i)^2 \tag{4}$$

Design Justification:

- **Window Size:** The choice of window size determines the number of consecutive data points used to calculate the average for each point in the smoothed signal. Larger windows smooth more aggressively but can obscure finer details, while smaller windows retain more noise but better preserve signal details. In our implementation, we chose 10 as our window size. Overall, a size of 10 performs best for the performance improvement among all the choices in 3, 5, 10, 50, 100, and 500.

Outlier removal is the second step in the data preprocessing pipeline aimed at identifying and eliminating data points that deviate significantly from the majority of the dataset. We apply a modified z-score statistical method to detect outliers, the formula of the modified z-score statistical method is as follows:

$$M_i = \frac{0.6745 \cdot (X_i - \tilde{X})}{\text{MAD}} \tag{5}$$

Where M_i is the modified z-score for the ith data point, X_i is the ith data point, $\tilde{X}$ is the median of the dataset, MAD is the median absolute deviation of the dataset.

A data point is regarded as an outlier if

$$|M_i| > \text{threshold} = 5 \tag{6}$$

Design Justification:

- **Threshold:** The modified z-score threshold is often higher to adjust for the fact that MAD is typically smaller than the standard deviation. A threshold of 5 for the modified z-score is a conservative choice that helps ensure only the most extreme outliers are removed, reducing the risk of discarding valid data points. We aim to minimize false positives — instances where non-outlier data is incorrectly identified as an outlier to preserve the mood information better.

Figure 19 in the Appendix shows the algorithm for the modified Z-score method.

Normalization is the last step in the data preprocessing pipeline aimed at standardizing the range of features in the dataset. It scales the data to a predefined range or distribution, typically between 0 and 1 or with a mean of 0 and a standard deviation of 1. However, our result proves that normalization is not necessary for our tree-based models as they work by making splits in the data based on threshold values and are not influenced by the scale or distribution of the features. We still keep this step as the normalization is successful, and it can be further used for neural networks which may improve our work. We use Z-score normalization with the formula shown as follows:

$$X_{\text{normalized}} = \frac{X - \mu}{\sigma} \tag{7}$$

Where X is the original data point, μ is the mean of the dataset, σ is the standard deviation of the dataset.

Figure 20 in the Appendix shows the algorithm for the z-score normalization.

2.2.3 Feature Extraction Subsystem

Time-frequency analysis is a technique used to analyze signals in both the time and frequency domains simultaneously. It provides insights into how the frequency content of a signal changes over time, which is particularly useful for analyzing non-stationary signals such as physiological data. In this subsystem, we utilize Short-Time Fourier Transform (STFT) for time-frequency analysis.

STFT analyzes a signal's frequency content over short, overlapping time intervals, providing a time-varying representation of the signal's frequency spectrum. The formula of STFT is as follows:

$$X(m,\omega) = \sum_{n=-\infty}^{\infty} x(n) \cdot w(n-m) \cdot e^{-j\omega n} \tag{8}$$

Where $X(m,\omega)$ is the STFT of the signal at time m and frequency ω. $x(n)$ is the input signal, $w(n-m)$ is the window function centered at time m, ω is the angular frequency.

Design Justification:

- **Window Size:** The choice of window size determines the trade-off between time and frequency resolution. Larger windows offer better frequency resolution but sacrifice time resolution, whereas smaller windows prioritize time resolution at the expense of frequency resolution.

  In our implementation, the parameter *nperseg* indicating the window size is set to 256 samples, which strikes a balance between time and frequency resolution. Other window sizes, such as 512 and 1024, fail to capture temporal dynamics and frequency characteristics of physiological signals, since the time-frequency diagrams generated are blank.

Figure 21 in the Appendix shows the algorithm for Short-Time Fourier Transform.

2.2.4 Classification Subsystem

Random Forest(RF) algorithm and Extreme Gradient Boosting(XGBoost) algorithm are used to classify moods based on psychological signals measured.

Random Forest is an ensemble learning method that combines multiple decision trees to improve classification accuracy. Figure 22 in the Appendix and Figure 7 show the detailed code implementation and a vivid demonstration for Random Forest, respectively.

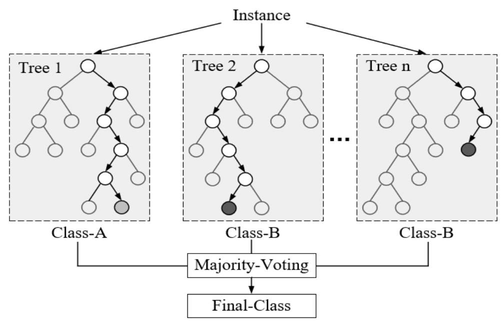


Figure 7: Random Forest Model [A powerful ensemble learning method that combines multiple decision trees to enhance classification accuracy and mitigate overfitting.]

Design Justification:

- **Number of Trees:** The number of trees (or estimators) in the Random Forest ensemble is a critical hyperparameter. Increasing the number of trees can improve model performance up to a certain point but may lead to longer training times and increased computational complexity.

  In our implementation, we have used 42 trees, which proves to be a suitable choice though cross validation we conducted on the entire labelled dataset.

- **Tree Depth:** Controlling the maximum depth of individual decision trees can help prevent overfitting. Limiting the tree depth ensures that the trees do not become too complex and helps improve the generalization performance of the model.

  In our implementation, 10 is set as the maximum depth value, which effectively prevents overfitting in K-EmoPhone dataset.

XGBoost is an efficient and scalable implementation of gradient boosting decision trees. It builds a series of decision trees sequentially, where each tree corrects the errors made by the previous ones, leading to improved overall performance. Figure 23 and Figure 8 show the detailed code implementation and a vivid demonstration for XGBoost, respectively.

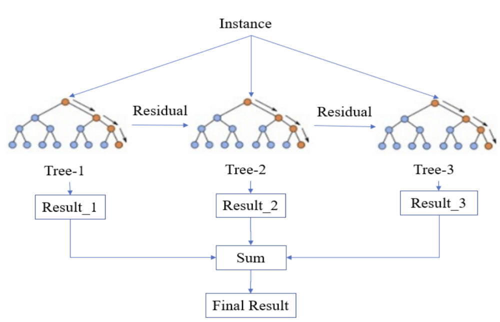


Figure 8: XGBoost Model [A highly efficient implementation of gradient boosting decision trees, sequentially building trees to correct errors and improve performance.]

Design Justification:

- **Learning Rate:** The learning rate (or shrinkage) parameter controls the contribution of each tree to the final prediction. A lower learning rate allows for more conservative updates, which may help prevent overfitting and improve model performance.

  In our implementation, learning rate is set as 0.02, which proves to be a suitable choice in terms of accuracy though cross validation.

- **Regularization Parameters:** Tuning regularization parameters can help control model complexity and prevent overfitting. Experimentation with different regularization settings is essential to find the optimal balance between bias and variance.

  In our implementation, L1 Regularization (alpha) is used, with value 0.5.

Table 1 shows the comparison of model parameters' effect on test accuracy. For the Random Forest model, the optimal number of trees is 42, which achieves the highest test accuracies of 0.626 for arousal and 0.662 for valence. The best tree depth is observed at 10, with similar accuracies. These parameters suggest that while increasing the number of trees and depth generally improves model performance, there is a threshold beyond which further increases may not yield better results and could lead to overfitting.

In the case of the XGBoost algorithm, a learning rate of 0.02 and an L1 regularization parameter of 0.5 were identified as optimal, achieving test accuracies of 0.634 and 0.659, respectively. These settings indicate that lower learning rates help in achieving more stable convergence to the optimal solution, and appropriate regularization helps in reducing model complexity and preventing overfitting.

The selection of these optimal parameters considers not only test accuracy but also factors like computational time, aligning with the design justification to balance model performance with resource efficiency, which is crucial for mood prediction applications.

Number of trees	test accuracy (arousal/valence)	Tree depth	test accuracy (arousal/valence)
30	0.601/0.629	5	0.625/0.598
42	0.626/0.662	10	0.626/0.662
54	0.637/0.671	15	0.612/0.601
66	0.656/0.650	20	0.590/0.611
Learning rate	test accuracy (arousal/valence)	L1 regularization	test accuracy (arousal/valence)
0.01	0.681/0.672	0.3	0.560/0.589
0.02	0.634/0.659	0.4	0.612/0.609
0.05	0.621/0.603	0.5	0.634/0.659
0.2	0.568/0.591	0.6	0.681/0.623

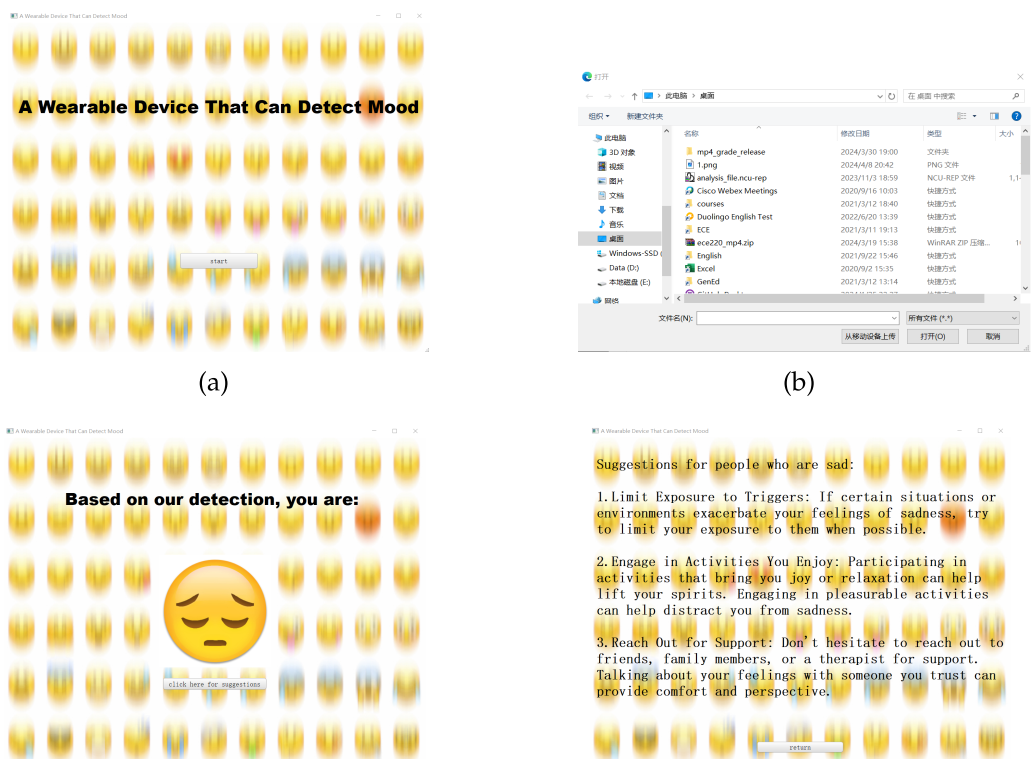
Table 1: Comparison of model parameters and test accuracies

2.2.5 Display Subsystem

This subsystem enhances user interaction by offering real-time feedback to improve or maintain users' emotional well-being. Designing a GUI that incorporates mood detection and provides appropriate suggestions requires careful consideration of visual presentation and integration with the mood detection part. Here are components of the GUI:

1) Display Components: Design areas where the detected mood and suggested activities or content will be displayed.
2) Visual Elements: Use visual cues such as icons and images to make the GUI visually appealing. Utilize emojis to represent 4 kinds of moods.
3) Integration with Mood Detection Subsystem: Implement functionality to communicate with the mood detection subsystem to retrieve the user's current mood.

Figure 9 provides a concise overview of our user interface layout. Upon launching the GUI, the initial display mirrors Figure 9(a), presenting the project title. Triggering the "start" button prompts a dialog window to appear, as illustrated in Figure 9(b), facilitating the selection of a txt file generated by the Classification Subsystem. This file encapsulates user mood data. Subsequently, our GUI parses this file, extracting keywords to discern the specific mood. In the event of insufficient mood-related data within the file, the GUI refrains from transitioning to Figure 9(c), instead issuing a message prompt: "Please open a valid result file." Upon successful parsing, the GUI proceeds to offer feedback to the user, as depicted in Figure 9(c). This feedback comprises textual content and an emoji, tailored to reflect the detected mood. For users seeking suggestions, a "click here for suggestions" button is available, leading to a transition to Figure 9(d). This page furnishes recommendations aimed at enhancing feedback and fostering emotional well-being.



(a) (b) (c) (d)

Figure 9: Layout of Graphical User Interface

3 Verification

3.1 User Interface Subsystem

Detailed requirements with reasonable tolerances of three sensors mentioned above are included in Appendix Table 7.

3.1.1 Pulse Rate Sensor

To verify it, we use a medically certified pulse detector and the smartwatch as the reference device. The user will be seated in a quiet room with a normal room temperature. We simultaneously record pulse readings from both the test sensor and the reference device when the output becomes stable. The sensor readings and the readings from the medically certified pulse detector and the smartwatch over a 1-minute period are compared. If the sensor's accuracy range is within ±2 beats per minute (bpm) and the standard deviation is less than 3, the sensor passes the accuracy test. If the sensor detects and reports stable pulse rate changes within 15 seconds, it passes the response time test. On the other hand, the successive heartbeat interval time (RRI) must correlate with the BPM and 60/RRI should be ±1 bpm compared to the BPM.

Quantitative Results

1. The reading of the medically certified pulse detector is 73.7. The reading of the smartwatch is 75.
2. The average reading of the pulse rate sensor in one minute is 72.9216, and the standard deviation is 2.2738, which meets the requirements above. The response time of stable data is 13 s, which is less than 15 s.
3. The average reading of the successive heartbeat interval is 818.5532 ms. The std is 47.7363 ms. 60/0.8185532 = 73.3 bpm, which is in the tolerance range.

3.1.2 Skin Conductance Sensor

To verify our skin conductance sensor, we connect the known resistance 200k, 400k, 600k, 800k, and 1000k ohms, to the sensor's two electrodes which are originally connected to our two fingers. These sensors generally cover the common range of human skin resistance. Compare the stable reading of our sensor to the standard known resistance. Calculate the measurement error percentage. If the error range is within the 2%, and the stable reading appear within ten seconds, the skin conductance sensor is considered to pass the accuracy test and responding time test successfully.

Quantitative Results

1. The GSR sensor reading corresponding to 200k, 400k, 600k, 800k, and 1000k ohms is 203929, 404705, 603125, 807843, 1002339. The percentage error is 1.9645%, 1.17625%, 0.520833%, 0.980375%, 0.2339%. All of the stable error ranges are less than 2%. Therefore, it passes the accuracy test.

2. The stable data comes up 1 or 2 seconds since our sample rate is 1 Hz. The GSR sensor does not have any responding time problems.

3.1.3 Skin Temperature Sensor

To verify the skin temperature sensor, we use a medically certified infrared thermometer as the standard reference device. Both it and our infrared sensor are used to test the skin temperature under an identical testing environment. The time that the reading of our infrared sensor becomes stable is recorded. Then the reading from the medically certified thermometer and our sensor will be compared. We will also repeat this comparison in the hot environmental condition (30°C) and the cold environmental condition (20°C) to observe its environmental resistance. If the skin temperature sensor has an accuracy of ±0.5°C, it passes the accuracy test. If The sensor achieves 90% of the final temperature reading within 3 seconds of application, it passes the responding time test. If the sensor maintains its accuracy within ±0.5°C across a range of environmental conditions (15°C to 35°C ambient temperature), it passes the environmental resistance test.

Quantitative Results

1. The reading of the skin temperature from the medically certified thermometer is 32.21°C. The average reading of our sensor per minute is 32.3778°C and the standard deviation inside a minute is 1.3945.
2. The reading from the sensor becomes stable within three seconds because of the method of infrared detection of our temperature sensor.
3. Repeat this experiment in different environments. The reading of the sensor remains the same, showing its environmental resistance.

3.2 Pre-processing Subsystem

3.2.1 Signal Smoothing

Detailed requirements with reasonable tolerances are included in Appendix Table 8.

We verify that signal smoothing in the pre-processing is working by measuring SNR and MSE as explained above. Improving the SNR through a smoothing algorithm indicates that the algorithm is effectively reducing noise while preserving the original signal's integrity. Decreasing the MSE suggests that the smoothing algorithm has reduced noise while maintaining the integrity and structure of the original signal, without introducing significant artifacts or alterations.

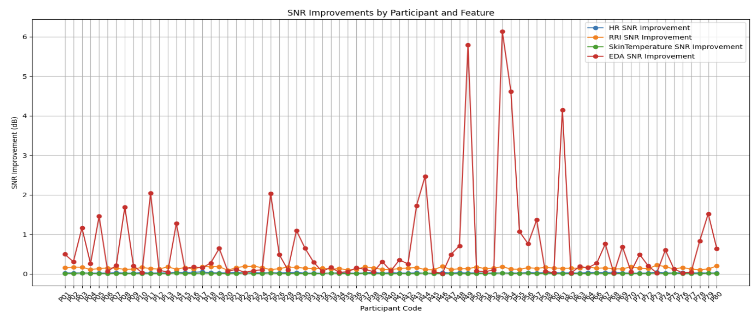


Figure 10: SNR Improvement: across different physiological features (HR, RRI, Skin Temperature, EDA) for various participants

Figure 10 indicates that the EDA data shows the most significant variation in SNR improvement, with spikes at certain participant codes, suggesting that the algorithm has a variable impact on different participants. The HR, RRI, and Skin Temperature improvements are relatively stable but minimal across participants. Some participants (like P06, P33, and P50) show exceptionally high improvements in EDA SNR, indicating specific conditions or characteristics of the data or the smoothing algorithm's performance that are particularly effective for these cases. Other features (HR, RRI, and Skin Temperature) display low variability, suggesting the smoothing does not dramatically alter the SNR for these features.

The graph confirms that every smoothed signal for each participant and feature shows a minimum increase in SNR of 0.001 dB compared to the original. We believe that the smoothing consistently reduces noise while preserving signal features. However, given the significant variability in EDA, further investigation might be needed to examine whether these spikes represent true signal preservation or possibly over-smoothing because of the potential risks for this feature.

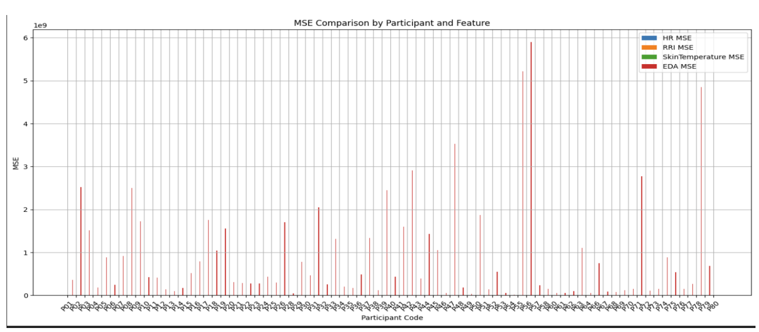


Figure 11: MSE Improvement: between the original and smoothed signals for various physiological features (HR, RRI, Skin Temperature, and EDA) across multiple participants.

In Figure 11, MSE varies significantly among participants, indicating varying levels of distortion introduced by the smoothing process across different data types and participants. High MSE values are particularly noticeable in EDA, suggesting that the smoothing may introduce significant distortion for this type of data in some participants. HR, RRI, and Skin Temperature tend to have lower MSE values, implying less distortion and potentially more effective smoothing for these signals.

Our results ensure that the MSE values for all features and participants are consistently below a predefined threshold of noise variance. Correlate these MSE findings with the SNR improvements you previously analyzed. We obtained an effective smoothing algorithm that would not only improve SNR but also maintain low MSE values to enhance signal clarity without adding substantial distortion. Some examples of these verifications are shown in Figure 12. The results indicate effective noise reduction and minimal signal distortion, as evidenced by improvements in SNR and acceptable MSE values for participants.

```
summary_results.csv > data
Participant,Feature,Original_SNR,Smoothed_SNR,SNR_Increase,MSE,Meets_SNR_Requirement,Meets_MSE_Requirement
P14,SkinTemperature,26.287490254753365,26.308768105360873,0.02127785060750753,2.36556999654771,True,True
P15,SkinTemperature,27.868003601135356,27.87138911666656,0.0033855155312032537,1.790391893675005,True,True
P16,SkinTemperature,26.773587048167037,26.779615922118545,0.006028873951507307 5,2.274620969235825 5,True,True
P17,SkinTemperature,27.145352394306205,27.162173598581063,0.01682120427485856,2.0797117347158993,True,True
P18,SkinTemperature,28.72755323111511,28.73936133294297,0.011808101827860185,1.440681468912913,True,True
P19,SkinTemperature,25.63639538782608,25.64444344172497,0.008048053898889407,2.7450113807489878,True,True
```

Figure 12: Examples of Verification: displays original and smoothed SNR, SNR increase, MSE, and compliance with predefined SNR and MSE requirements

3.2.2 Outlier Removal

Detailed requirements with reasonable tolerances are included in Appendix Table 9.

We use the modified z-score algorithm to remove wrong or irrelevant data in the datasets. We verify this by generating a detailed log of outliers detected in the dataset. The examples in the table 3.2.2 show some of the changes in the heart rate in certain participants.

	Original Mean	Cleaned Mean	Original Std	Cleaned Std
1	72.938	72.938	4.648	4.648
2	74.642	74.599	4.891	4.774
3	75.645	75.645	5.858	5.858
4	71.983	71.969	5.717	5.675
5	76.715	76.658	7.141	6.972
6	75.718	75.608	4.444	3.987

Table 2: Sample Mean and Std Before and After Removal

The dataset had a broader range of mean and standard deviation, indicating a spread affected by extreme values. After Removal, the mean and standard deviation values are reported to have aligned more closely with those of a clean dataset. This suggests a reduction in the impact of extreme values, thereby normalizing the data distribution closer to its expected statistical properties. Furthermore, we confirm that over 90% of the removed data can be classified as outlier data with several random tests, the result is much better than our target of 85%.

3.2.3 Normalization

Detailed requirements with reasonable tolerances are included in Appendix Table 10.

The mean of our normalized data among all files is 7.679208608086269e-05, which is extremely close to 0. This is expected in a properly z-score normalized dataset, where the mean should ideally be 0. The standard deviation of the normalized data is approximately 1.0 (0.9999990917266793), aligning with the target of a z-score normalization, where the standard deviation is standardized to 1. These statistics confirm that the normalization process has realized the goal of z-score normalization.

However, the data we use are not normalized, this leaves for further development. Although z-score normalization is crucial for models like neural networks due to their dependence on gradient descent and weight updates, it's generally not necessary for tree-based models which are not influenced by the scale of the features.

3.3 Feature Extraction Subsystem

Detailed requirements with reasonable tolerances are included in Appendix Table 11.

We verify the feature extraction by verifying the STFT algorithm can extract features that contain useful information in the frequency domain for model training successfully in a random signal and satisfy the standard we set.

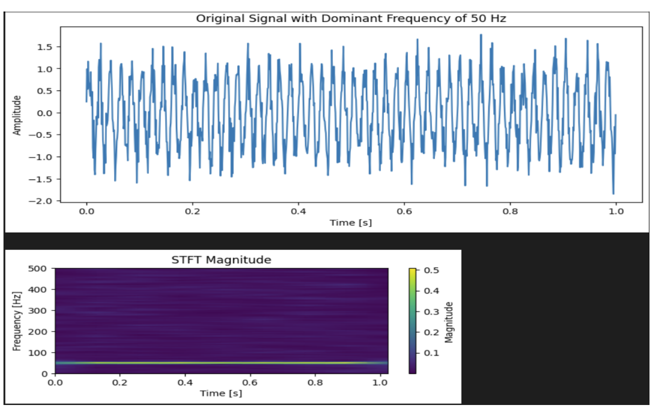


Figure 13: STFT: extracts features to generate the frequency graph below with the signal above

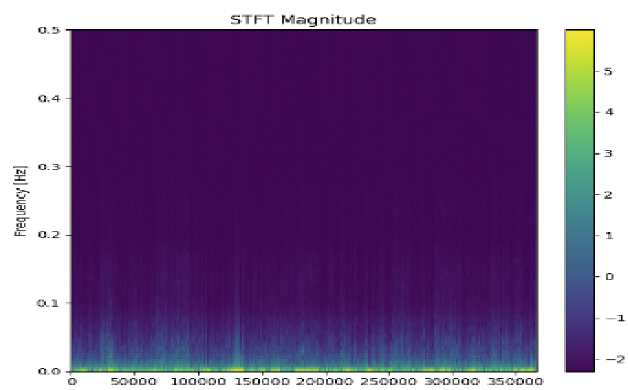


Figure 14: STFT Example: STFT result graph of the RRI for participant 15

Figure 13 uses a signal with a known dominant frequency of 50 Hz and some noise and computes its STFT to extract frequency features over time. This approach verifies that the STFT algorithm effectively reveals the frequency components of the signal across different time intervals. From the STFT results, the dominant frequency was identified as approximately 50.78125 Hz. The percentage error between the known frequency (50 Hz) and the identified frequency was computed as 1.5625%, satisfying our requirement. We have also generated the STFT graphs with our data shown in Figure 14.

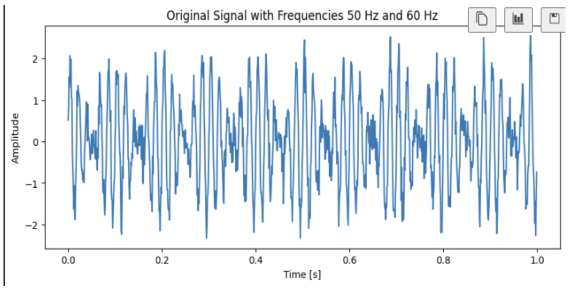


Figure 15: Original Signals of 50 Hz and 60 Hz with noise

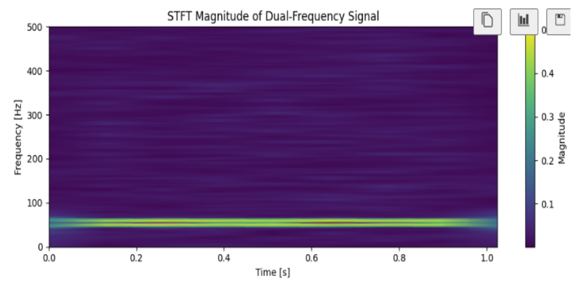


Figure 16: STFT Signals of 50 Hz and 60 Hz with noise

Then, we will generate another signal with two known dominant frequencies of 50 Hz and 60 Hz shown in Figure 15 to verify the resolution of using this method. After STFT, it is shown in Figure 16. Peaks in the average frequency spectrum magnitude were identified. The identified peaks were at approximately 50.78125 Hz and 58.59375 Hz. The distance between these peaks is about 7.8125 Hz, which is less than the 10 Hz resolution needed to distinguish the frequencies as our requirement. This indicates that the current STFT settings or method may not have sufficient resolution to meet the 10 Hz criterion for these specific closely spaced frequencies.

3.4 Classification Subsystem

Detailed requirements with reasonable tolerances are included in Appendix Table 12.

We employed Leave-One-Subject-Out (LOSO) Cross-Validation to estimate the general performance of our models in predicting both the affective and cognitive states of unseen users. For each participant, we partitioned our feature and label data into a testing fold containing data from that participant, and a training fold containing data from the remaining participants. Subsequently, we trained our machine-learning models on the training fold data and assessed their performance using the testing fold. With 77 participants' data retained after preprocessing, we iterated the partitioning, training, and evaluation procedures 77 times.

Subsequently, we proceeded to train the prediction models utilizing two distinct learning algorithms: Random Forest and XGBoost. These algorithms, both belonging to the family of tree-based ensemble learning methods, possess the capability to effectively manage extensive feature spaces and capture intricate non-linear relationships among features.

Leveraging this advantage, they have gained widespread adoption in predicting user behaviors and cognitive states using mobile sensor data, a scenario akin to the K-EmoPhone dataset. Additionally, we trained a baseline model that consistently predicts the majority class, facilitating comparison with our developed models

We then evaluated our prediction models using the testing fold data with performance metrics, mainly focusing on the test accuracy. The final metric was derived by averaging the metrics calculated from 77 splits.

Quantitative Results:

Table 3 presents the performances for predicting valence and arousal across different learning algorithms. Valence indicates whether emotions activate positively or negatively, while arousal represents the intensity level. Combining arousal and valence, we can obtain the corresponding mood result, whose relationship is shown in Figure 17.

Table 3: Performance Evaluation Results

	Random Forest	**XGBoost**
Valence	0.662	0.659
Arousal	0.626	0.634

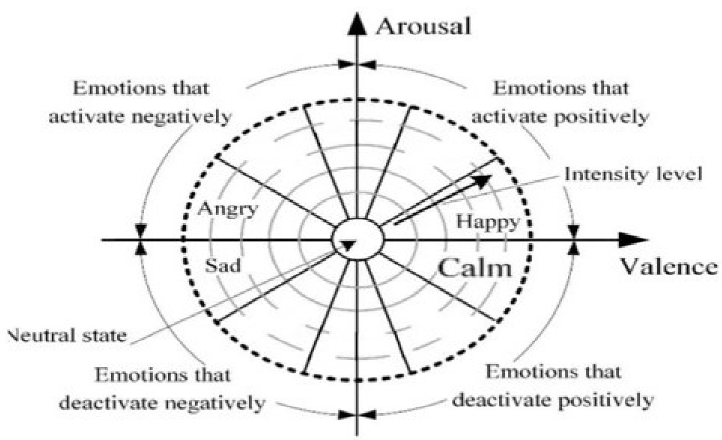


Figure 17: Relationship between Mood and Arousal & Valence

Overall, the XGBoost method gives a higher testing accuracy compared to Random Forest, demonstrating a powerful classification ability. With 0.659 and 0.634 accuracy in valence and arousal respectively, XGBoost algorithm gives the overall mood prediction accuracy:

$$0.659 \times 0.634 = 0.4178 \tag{9}$$

which exceeds our initial goal of 40% accuracy in Table 12.

3.5 Display Subsystem

For the GUI designed to provide suggestions to users based on the detected mood, we should evaluate various aspects of the interface to ensure that it remains usable, functional, and visually appealing despite potential variations or deviations. Detailed requirements are included in Appendix Table 13.

The integration of the user interface with the Mood Classification subsystem is important to the functionality of our project. Once the model generates the detected mood based on user input, it's imperative that the interface promptly delivers corresponding feedback. As previously outlined, the interface will parse a txt file generated by the Mood Classification subsystem to discern the specific mood. To validate this functionality, we conducted rigorous testing using various txt files containing different keywords corresponding to different moods. By analyzing the interface's ability to accurately display the correct page based on specific keywords, we can verify its functionality and ensure it operates as intended. This iterative testing process allows us to identify any discrepancies or errors and refine the interface accordingly, ultimately ensuring its reliability and effectiveness in providing timely feedback to users.

Another significant thing is that the interface should provide appropriate suggestions to users based on the detected mood to make positive impacts on users. Delivering pertinent suggestions to users, informed by their detected mood, entails comprehending their emotional states and providing content, activities, or interventions that are likely to have a positive impact. Careful consideration of the suggestion content is paramount, and to ensure efficacy, we have conducted a thorough review of pertinent research [6] to compile a selection of suggestions tailored to users experiencing various moods, which are detailed in the appendix C.1.

4 Costs and Schedule

4.1 Cost

4.1.1 Labor

Refer to the sample and past senior design projects, we estimate our salary to be \$40/hour, 10 hours/week for each member. The estimated salary is listed in Labor Cost Table 4.

Table 4: Labor Cost. The table shows the member name in the senior design project (Member), the hourly salary of member (Hourly Salary), the working hours in the project per person (Working Hours) and the total salary per person (Total).

Member	Hourly Salary	Working Hours	Total
Junjie Ren	\$40	$10 \times 14 = 140h$	$\$40 \times 140 \times 2.50 = \$14000$
Peidong Yang	\$40	$10 \times 14 = 140h$	$\$40 \times 140 \times 2.50 = \$14000$
Xinzhuo Li	\$40	$10 \times 14 = 140h$	$\$40 \times 140 \times 2.50 = \$14000$
Kejun Wu	\$40	$10 \times 14 = 140h$	$\$40 \times 140 \times 2.50 = \$14000$
Sum			\$56000

4.1.2 Parts

Our parts and manufacturing prototype costs are estimated to be \$39.40 in total. See Parts Cost Table 5 in detail.

Table 5: Parts Cost. The table shows the parts description (Description), the manufacturer of the parts (Manufacturer), the quantities needed in our project (Quantity), the unit cost of the parts (Cost/Unit) and the total cost of the the parts (Total Cost).

Description	Manufacturer	Quantity	Cost/Unit	Total Cost
Pulse Sensor (MDL0025)	Sichiray	1	\$5.50	\$5.50
Grove Galvanic Skin Response (GSR)	Sichiray	1	\$6.30	\$6.30
Infra Red Thermometer (GY-906-DCC)	Melexis	1	\$4	\$4
Arduino uno	Sichiray	1	\$23.60	\$23.60
Cloud Server rental fee	AUTODL	1	\$168.80	\$168.80
Sum				\$208.20

4.1.3 Total Cost

The total cost of our senior design project is:

$$\$56000 + \$208.20 = \$56208.20$$

4.2 Schedule

The weekly schedule table is given in Table 6.

Table 6: Weekly schedule table. The table shows different weeks (Date) and individual schedule for each group member in different weeks.

Date	Junjie Ren	Peidong Yang	Xinzhuo Li	Kejun Wu
3.25-3.31	Research the labeled data	Test the function of sensors	Research appropriate machine learning methods	Test the working of Arduino
4.1-4.7	Pre-processing the labeled data	Measure some physiological data by using the hardware parts	Write machine learning code	Measure some physiological data by using the hardware parts
4.8-4.14	Write and debug machine learning code	Import the measured data into computer	Write and debug machine learning code	Import the measured data into computer
4.15-4.21	Evaluate the accuracy of our trained model	use GUI in python to display possible outcomes	Improve our trained model to make it more accurate	Integrate other possible sensors to Arduino
4.22-4.28	Improve and debug machine learning model	Integrate all hardware parts as a wearable device	Improve and debug machine learning model	Merge measured data with machine learning model
4.29-5.5	Test and improve the mood classification model	Test and improve the display subsystem	Test and improve the mood classification model	Test and improve the user interface subsystem
5.6-5.12	Mock Demo	Mock Demo	Mock Demo	Mock Demo
5.13-5.19	Final Demo and Final Report	Final Demo and Final Report	Final Demo and Final Report	Final Demo and Final Report

5 Conclusions

5.1 Accomplishments

This project successfully developed a wearable device capable of accurately predicting human mood based on physiological signals. By integrating sensors to measure heart rate variability (HR), skin resistance (EDA), and skin temperature (SKT), the device captures the emotional states of users in real-time. The use of advanced machine learning algorithms, specifically Random Forest (RF) and Extreme Gradient Boosting (XGBoost), enabled the system to analyze these physiological indicators and classify mood states with high precision.

The overall mood prediction accuracy exceeded the initial goal of 40%, demonstrating the effectiveness of the system in providing a real-time, objective method for mood recognition. This achievement highlights the project's success in combining sensor technology and machine learning to offer a practical solution for monitoring and managing mood, enhancing individual well-being and mental health support.

5.2 Uncertainties

While the project achieved significant success in developing a wearable device for mood detection, several uncertainties and areas for improvement were identified. We will demonstrate some of the most obvious uncertainties in our project.

The sensors have a certain range of tolerance. For example, the pulse rate sensor showed a standard deviation of up to 2.2738 in minute-long readings. While this was within the project's expected acceptable range (±2 beats per minute), it highlights the potential for inaccuracies in detecting changes associated with sudden emotional shifts. This variability could affect the precision of mood predictions, particularly for subtle or swift changes in emotional states.

While the XGBoost model achieved an overall mood prediction accuracy of approximately 41.78%, surpassing the target of 40%, this still leaves room for improvement. The variability in the model's performance, with certain mood states, predicted more accurately than others, suggests that further refinement is needed to enhance the robustness and generalizability of the mood classification across a broader spectrum of emotional states and user demographics.

In conclusion, while the project made significant strides in mood detection using wearable technology, these uncertainties highlight the need for further research and development to refine sensor accuracy, improve algorithm consistency, and enhance feature extraction resolution.

5.3 Future Work

To enhance the efficacy and applicability of the wearable device for mood detection, several improvements and design alternatives are proposed for future work. We could explore the integration of more advanced sensors with higher precision and faster response times. To enhance the generalizability of the classification models, incorporating deep learning techniques such as Convolutional Neural Networks (CNNs) or Recurrent Neural Networks (RNNs) could be beneficial. These models can capture more complex patterns in time-series data and adapt better to individual differences. We have finished normalization of data, which can be used in these deep learning methods.

5.4 Ethical Considerations

5.4.1 Ethics

Our project addresses the widespread influence of workplace stress, anxiety, and depression, acknowledging them as pivotal challenges that undermine individual well-being. We introduce a wearable device integrated with sophisticated sensors and machine learning to achieve an innovative mood detection framework. This endeavor aligns seamlessly with the ACM code's commitment to contribute to society and human well-being, recognizing the universal stakeholder role of all individuals in the realm of computing [7].

The collection and use of personal and potentially sensitive physiological data to train our model could infringe on an individual's privacy if not handled correctly, especially in accordance with the ACM Code regarding to respect privacy and confidentiality [7]. We promise to collect only the data necessary for mood recognition to reduce the risk of privacy breaches, and we will ensure that users are fully informed about what data is collected, how it will be used, and obtain their consent. Moreover, we will implement strict access controls so that only authorized personnel can access sensitive data.

5.4.2 Safety

Ensuring safety is a top priority in our project. For electrical usage, we use the USB portal of computer as the power supply of Arduino, and the computer is powered by 220 volts of electricity. Our group fully understands and adheres to the guidelines for safe electricity usage. We will routinely check the computer and Arduino to ensure they operate in a proper environment. Moreover, we have wearable devices to measure physiological data, and sensors will be connected with Arduino. Then Arduino will be connected to the computer to import data. There is a risk of getting an electric shock when we measure the physiological data of participants. We will strictly follow the safe working practice to ensure the safety of participants. We'll make sure the wires are intact and connected well to prevent the occurrence of electrical leakage. Another safety issue is that, if the system inaccurately assesses a user's mood, it could lead to inappropriate recommendations or actions of user. Ensuring high accuracy of the mood prediction algorithms and providing users with context about the limitations of the system can minimize the risk.

References

[1] L. Malott, P. Bharti, N. Hilbert, G. Gopalakrishna, and S. Chellappan, "Detecting self-harming activities with wearable devices," in *2015 IEEE International Conference on Pervasive Computing and Communication Workshops (PerCom Workshops)*, 2015, pp. 597–602. DOI: 10.1109/PERCOMW.2015.7134105.

[2] J. Yang, Y. Li, and M. Xie, "Motionauth: Motion-based authentication for wrist worn smart devices," in *2015 IEEE International Conference on Pervasive Computing and Communication Workshops (PerCom Workshops)*, 2015, pp. 550–555. DOI: 10.1109/PERCOMW.2015.7134097.

[3] Y. Benezeth, P. Li, R. Macwan, K. Nakamura, R. Gomez, and F. Yang, "Remote heart rate variability for emotional state monitoring," in *2018 IEEE EMBS international conference on biomedical & health informatics (BHI)*, IEEE, 2018, pp. 153–156.

[4] G. Udovičić, J. Đerek, M. Russo, and M. Sikora, "Wearable emotion recognition system based on gsr and ppg signals," in *Proceedings of the 2nd international workshop on multimedia for personal health and health care*, 2017, pp. 53–59.

[5] P. Vos, P. De Cock, V. Munde, K. Petry, W. Van Den Noortgate, and B. Maes, "The tell-tale: What do heart rate; skin temperature and skin conductance reveal about emotions of people with severe and profound intellectual disabilities?" *Research in developmental disabilities*, vol. 33, no. 4, pp. 1117–1127, 2012.

[6] J. Heiy and J. Cheavens, "Back to basics: A naturalistic assessment of the experience and regulation of emotion," *Emotion (Washington, D.C.)*, vol. 14, Jul. 2014. DOI: 10.1037/a0037231.

[7] ACM. ""ACM Code of Ethics and Professional Conduct"." (2018), [Online]. Available: https://www.acm.org/code-of-ethics (visited on 05/06/2024).

Appendix A Requirement and Verifications

A.1 User Interface Subsystem

The User Interface subsystem consists of sensors and Arduino. Its responsibility is to obtain physiological data from our users and transmit them to the computer for further processing. The requirements and verifications of the user interface sensor system are listed in Table 7.

Table 7: R&V table of User Interface

Requirements	**Verifications**
The user interface sensor subsystem is capable of obtaining at least four types of physiological data simultaneously using wire.	All types of data could be seen simultaneously on the computer's screen instead of using the sensor to test one by one.
1.The sensor's accuracy range is within ±2 beats per minute (bpm). The standard deviation is less than 3. 2.The sensor must detect and report stable pulse rate changes within 15 seconds. 3.The successive heartbeat interval time must be detected simultaneously with the heartbeat. The interval time (RRI) must correlate with the BPM and 60/RRI should be ±1 bpm compared to the BPM.	1.Use a medically certified pulse detector and the smartwatch as the reference device. The user will be seated in a quiet room with a normal room temperature. 2.Simultaneously record pulse readings from both the test sensor and the reference device when the output becomes stable. 3.Compare the sensor readings against the readings from the medically certified pulse detector and the smartwatch over a 1-minute period. Calculate the mean and standard deviation of the sensor's readings.
1.The GSR sensor must have an accuracy of ±2% 2.The sensor's response time to changes in conductance should be no more than 3 seconds from the time of physiological change to sensor output stabilization.	1.Connect the known resistance (200k, 400k, 600k, 800k, 1000kohms) to the sensor's electrodes. 2.Check whether the reading of the sensor is consistent to the known resistance.

1.The skin temperature sensor must have an accuracy of ±0.5°C 2.The sensor must achieve 90% of the final temperature reading within 3 seconds of application. 3.The sensor must maintain its accuracy within ±0.5°C across a range of environmental conditions (15°C to 35°C ambient temperature).	1.Use a medically certified infrared thermometer and our infrared sensor to test the skin temperature. Keep the testing place identical. 2.Record the time that the reading of our infrared sensor become stable. Compare the reading from the medically certified thermometer and our sensor. 3.Repeat this comparison in the hot environmental condition (30°C) and the cold environmental condition (20°C).

A.2 Pre-processing Subsystem

The data pre-processing subsystem consists of 3 successive parts. They are Signal Smoothing, Outlier Removal and Normalization.

1) Signal Smoothing is a technique used to remove high-frequency noise from a signal while preserving its underlying trends or features. See Table 8 for its Requirements and Verifications.

Table 8: R&V table of Signal Smoothing

Requirements	**Verifications**
Smoothed signals should exhibit a significant reduction in noise compared to the original signal, with a minimum increase in SNR of 0.001 dB.	1.Apply a random noisy signal from K-EmoPhone dataset to the smoothing algorithm. 2.Using the software SciPy form Python language, measure the signal-to-noise ratio (SNR) of the original and smoothed signals by power spectral density method. 3.See if the smoothed signal exhibits a minimum increase in SNR of 0.001 dB compared to the original signal. 4.Repeat the above steps and ensure that the tested dataset covers at least 1/3 of the standard dataset, verifying the smoothed signals exhibit consistent noise reduction and preservation of signal features.

The mean squared error (MSE) between the original and smoothed signals should be below the threshold of noise variance ,indicating minimal distortion.	1.Calculate the mean squared error (MSE) between the original and smoothed signals using Math Solver, a powerful software calculator. 2.See if the MSE between the original and smoothed signals is below the threshold of the noise variance, indicating acceptable distortion. 3.Repeat the above steps for all signals tested with SNR, verifying the smoothing algorithm does not introduce significant distortion.

2) Outlier removal is a crucial step in data preprocessing aimed at identifying and eliminating data points that deviate significantly from the rest of the dataset.See Table 9 for its Requirements and Verifications.

Table 9: R&V table of Outlier Removal

Requirements	**Verifications**
The outlier removal process should effectively identify data points that deviate significantly from the rest of the dataset.	1.Apply the outlier detection algorithm to a test dataset containing known outliers. Compare the identified outliers with the ground truth. 2.See if identified outliers match the known outliers in the test dataset with an accuracy degree higher than 85%.
The outlier removal process should effectively remove data points that deviate significantly from the rest of the dataset.	1.Remove the identified outliers from the test dataset. Calculate the statistical properties (e.g., mean, standard deviation) of the dataset before and after outlier removal. 2.After outlier removal, see if the statistical properties of the dataset align more closely with the underlying distribution of the data. Specifically, see if the mean and standard deviation are closer to the values calculated from a clean dataset.

3) Normalization is used to rescale the features of a dataset to a standard range, typically between 0 and 1 or with a mean of 0 and a standard deviation of 1. See Table 10 for its Requirements and Verifications.

Table 10: R&V table of Normalization

Requirements	**Verifications**
All features in the dataset should be uniformly scaled to prevent dominance by features with larger scales.	1.Apply the normalization algorithm to the dataset and examine the range of normalized values for each feature 2.Check if all normalized feature values fall within the specified range (e.g., 0 to 1 for min-max normalization, mean of 0 and standard deviation of 1 for z-score normalization). 3.Calculate the minimum and maximum values of each normalized feature. If all normalized values fall within the specified range, the requirement for uniform scaling is met.
The normalization process should not introduce bias or distortions in the data.	1.Calculate the mean and standard deviation of the normalized dataset. 2.Check if the mean of the normalized dataset is close to 0, and the standard deviation is close to 1 for z-score normalization. For min-max normalization, check if the mean is within a small tolerance of the midpoint of the range, and the standard deviation is approximately equal to half the range. 3.If the mean is close to 0 and the standard deviation is close to 1 (or within specified tolerances), the requirement for bias and distortion is met.

A.3 Feature Extraction Subsystem

The feature extraction subsystem consists of 3 essential parts. They are Time-Frequency Analysis, Frequency Domain Analysis, and Nonlinear Dynamics Analysis.

Since the above 3 strategies share the same purpose of extracting features and they complement with each other, their methods for requirements and verifications could be identical, as shown in the Table 11.

Table 11: R&V table of feature extraction

Requirements	Verifications
Accuracy: The feature analysis should accurately identify the dominant frequency components of the signal with an error margin of less than 15%	1.Generate synthetic signals with known dominant frequency components, from the K-EmoPhone dataset. 2.Apply the feature analysis method to the synthetic signals and extract the dominant frequency components. 3.Compare and calculate the percentage error between the identified dominant frequencies and the known frequencies, and see if the error margin is less than 15% .
Resolution: The feature analysis should have a frequency resolution of at least 10 Hz to distinguish between closely spaced frequency components.	1.Generate synthetic signals with closely spaced frequency components, from the K-EmoPhone dataset. 2.Apply the feature analysis method to the synthetic signals and examine the frequency spectrum. Measure the distance between adjacent frequency peaks to determine the frequency resolution. 3.Verify that the feature analysis method can resolve adjacent frequency components with a separation of at least 10 Hz.

A.4 Mood Classification Model Subsystem

The mood classification model subsystem consists of 2 essential supervised learning algorithms. They are Random Forest and Extreme Gradient Boosting (XGBoost).

Since the above 2 algorithms share the same purpose of classifying mood and they complement with each other, their requirements and verifications methods could be identical, as shown in the Table 12.

Table 12: R&V table of Mood Classification

Requirements	Verifications
The mood classification model should achieve a classification accuracy of at least 40% when predicting mood states based on the given psychological data.	1.Using supervised learning algorithm like Random Forest and XGBoost, train the classification model based on the labelled data K-EmoPhone dataset. 2.Use the trained model to predict mood states for the samples in the testing dataset. Compare the predicted mood states with the ground truth labels to assess classification accuracy. 3.Calculate the classification accuracy as the percentage of correctly predicted mood states out of the total number of samples in the testing dataset. See if it is higher than the desired 40% threshold.

A.5 Display Subsystem

We will utilize Graphical User Interface (GUI) in Python to display the detected mood and corresponding suggestions to users. Designing a display via a GUI involves various considerations to ensure usability, functionality, and aesthetic appeal. Providing appropriate suggestions to users based on the detected mood involves understanding the user's emotional state and offering content, activities, or interventions that are likely to resonate positively with them. The requirements and verifications of the display subsystem are listed in Table 13.

Table 13: R&V table of Display Subsystem

Requirements	Verification
The interface must integrate with Mood Classification Model. After the model generates the detected mood based on our input, the interface should give real-time feedback.	1.Once the mood output is stored in a txt file, the GUI can open and read it, and then update the relevant GUI components to reflect the detected mood. This could involve changing colors or icons displaying four mood classifications with corresponding emojis.
The interface should provide appropriate suggestions to users based on the detected mood to make positive impacts on users.	1.Consult professional psychology references for professional advice related to each detected mood. 2.Offer a variety of suggestions to cater to different preferences and interests. Include options for relaxation, entertainment, socializing, and self-care to accommodate diverse user needs.

The interface has aesthetic design elements including color schemes, typography, icons, and visual hierarchy to enhance usability and appeal.	1.Design the layout of GUI, considering the placement and arrangement of components to ensure clarity and usability. Use principles of visual hierarchy and proximity to organize components logically. 2.Choose appropriate colors, emojis, and other visual elements to enhance the aesthetic appeal and usability of the GUI. Consider using different colors or visual cues to represent different mood categories. 3.Define how users will interact with the GUI components to input or select the mood information. Ensure intuitive interaction patterns and provide appropriate feedback to users.

Appendix B Pseudocode of Algorithms

B.1 Moving Average

Algorithm 1 Moving Average Filter

```
function MovingAverageFilter(data, window_size)
    smoothed_data ← []
    for i from 0 to len(data) − window_size do
        window_sum ← 0
        for j from i to i + window_size − 1 do
            window_sum ← window_sum + data[j]
        end for
        smoothed_data.append(window_sum/window_size)
    end for
    return smoothed_data
end function
```

Figure 18: Moving Average Filter Algorithm, which is utilized in signal processing to smooth out fluctuations in data by averaging neighboring data points within a specified window size

B.2 Modified Z-score

Algorithm 2 Modified Z-score Method for Outlier Removal

```
function ModifiedZScore(data, threshold)
    median ← medianofdata
    mad ← medianabsolutedeviationofdata
    for i from 0 to len(data) − 1 do
        modified_z_score ← 0.6745·(data[i]−median) / mad
        if modified_z_score > threshold then
            Remove data[i] from data
        end if
    end for
    return data
end function
```

Figure 19: Modified Z-score Method for Outlier Removal, which is used to identify and eliminate outliers from a dataset by measuring deviations from the median in terms of median absolute deviation (MAD)

B.3 Normalization

Algorithm 3 Z-score Normalization

function ZSCORENORMALIZATION($data$)
  $mean \leftarrow meanofdata$
  $std\_dev \leftarrow standarddeviationofdata$
  for i from 0 to $len(data) - 1$ **do**
    $data[i] \leftarrow \frac{data[i] - mean}{std\_dev}$
  end for
  return $data$
end function

Figure 20: Z-score Normalization. It standardizes the values of a dataset by subtracting the mean and dividing by the standard deviation

B.4 Short-Time Fourier Transform

Algorithm 4 Short-Time Fourier Transform (STFT)

function STFT($data, window\_size, overlap$)
  $stft\_data \leftarrow []$
  for i from 0 to $len(data) - window\_size$ step $overlap$ **do**
    $windowed\_data \leftarrow data[i : i + window\_size]$
    $stft \leftarrow FourierTransform(windowed\_data)$
    $stft\_data.append(stft)$
  end for
  return $stft\_data$
end function

Figure 21: Short-Time Fourier Transform (STFT), which transforms input signal data into its frequency-domain representation by dividing it into short segments and applying the Fourier Transform to each segment separately

B.5 Random Forest

Algorithm 5 Random Forest Training

function RandomForestTrain($Data, T, m$)
  for $t = 1$ **to** T **do**
    $Subset \leftarrow$ RandomlySample($Data$)
    $SelectedFeatures \leftarrow$ RandomlySelectFeatures($Subset, m$)
    $Tree \leftarrow$ BuildDecisionTree($Subset, SelectedFeatures$)
    Add $Tree$ **to** $Forest$
  end for
  return $Forest$
end function

Figure 22: Random Forest Algorithm implementation[A powerful ensemble learning method that combines multiple decision trees to enhance classification accuracy and mitigate overfitting]

B.6 Extreme Gradient Boosting

Algorithm 6 XGBoost Training

function XGBoostTrain($Data, T, \eta$)
  Initialize $Model_0$ with a constant value
  for $t = 1$ **to** T **do**
    Compute the negative gradient of the loss function: $\nabla L(y_i, \hat{y}_i^{(t-1)})$
    Fit a weak learner to the negative gradients: $h_t = \arg\min_h \sum_{i=1}^{n} L(y_i, \hat{y}_i^{(t-1)} + h(x_i))$
    Update the model: $Model_t = Model_{t-1} + \eta \cdot h_t$
  end for
  return $Model_T$
end function

Figure 23: Extreme Gradient Boosting Algorithm implementation[A highly efficient implementation of gradient boosting decision trees, sequentially building trees to correct errors and improve performance]

Appendix C Supplementary Materials

C.1 Suggestions for Users

Suggestions for people who are happy:

1) Share Your Happiness: Celebrate your successes and achievements with friends and loved ones, and share kind words and gestures to uplift those around you.

2) Engage in Acts of Kindness: Doing something kind for others can boost your own happiness levels. Giving back to others can enhance your sense of well-being.

3) Reflect on Your Happiness: Take time to reflect on what makes you happy and why. Understanding the sources of your happiness can help you cultivate more of it in your life.

Suggestions for people who are calm:

1) Maintain Balance: Strive to maintain a healthy balance of your life. Avoid overcommitting yourself and prioritize self-care to prevent feelings of stress.

2) Practice Mindfulness with Joy: Incorporate mindfulness practices that emphasize joy and happiness. Pay attention to moments of joy and savor them fully.

3) Laugh Often: Seek out opportunities to laugh. Watch a funny movie, read a humorous book, or spend time with people who make you laugh. Laughter can bring a sense of happiness.

Suggestions for people who are angry:

1) Take Deep Breaths: Deep breathing can help to calm the body's physiological response to anger. Practice deep, slow breaths to relax and regain composure.

2) Seek Solitude: If you feel overwhelmed by anger, remove yourself from the situation temporarily. Engage in another calming activity until you feel more composed.

3) Laugh Often: Seek Support: Talk to a trusted friend, family member, or therapist about your feelings of anger. They can provide support and advice on how to cope with challenging emotions.

Suggestions for people who are sad:

1) Limit Exposure to Triggers: If certain situations or environments exacerbate your feelings of sadness, try to limit your exposure to them when possible.

2) Engage in Activities You Enjoy: Participating in activities that bring you joy or relaxation can help lift your spirits. Engaging in pleasurable activities can help distract you from sadness.

3) Don't hesitate to reach out to friends, family members, or a therapist for support. Talking about your feelings with someone you trust can provide comfort and perspective.