DE-41 (EE) FAIZAN, HINA, IFRAH, ZAIN

DRIVECARE – EMOTION RECOGNITION USING FACIAL EXPRESSIONS AND PHYSIOLOGICAL CHANGES

COLLEGE OF
ELECTRICAL AND MECHANICAL ENGINEERING
NATIONAL UNIVERSITY OF SCIENCES AND TECHNOLOGY
RAWALPINDI
2023

COLLEGE OF ELECTRICAL AND MECHANICAL ENGINEERING

DE-41 EE
PROJECT REPORT

DRIVECARE – EMOTION RECOGNITION USING FACIAL EXPRESSIONS AND PHYSIOLOGICAL CHANGES

Submitted to the Department of Electrical Engineering
in partial fulfillment of the requirements
for the degree of
Bachelor of Engineering
in
Electrical
2023

Submitted by

NS Zain Rehan

NS Hina Arshad

NS Ifrah Fasih

NS Muhammad Faizan Asghar

CERTIFICATE OF APPROVAL

It is to certify that the project **"DRIVECARE – EMOTION RECOGNITION USING FACIAL EXPRESSIONS & PHYSIOLOGICAL CHANGES"** was done by **NS Zain Rehan**, **NS Hina Arshad**, **NS Ifrah Fasih**, and **NS Muhammad Faizan Asghar** under the supervision of **Dr. Ahmad Rauf Subhani** and **Sobia Hayee**.

Submission: This project was submitted to the College of Electrical and Mechanical Engineering, National University of Sciences and Technology, Pakistan, as part of the requirement for the degree of Bachelor of Electrical Engineering.

Students:

1. **Zain Rehan**

NUST ID: 301822 Signature: ______________________

2. **Hina Arshad**

NUST ID: 306861 Signature: ______________________

3. **Ifrah Fasih**

NUST ID: 290156 Signature: ______________________

4. **Muhammad Faizan Asghar**

NUST ID: 321941 Signature: ______________________

Approved By:

Project Supervisor: **Dr. Ahmad Rauf Subhani**

Signature: ______________________ Date: __________________________

Project Co-Supervisor: **Sobia Hayee**

Signature: ______________________ Date: __________________________

Head of Department: **Dr. Fahad Mumtaz Malik**

Signature: ______________________ Date: __________________________

DECLARATION

We declare that no part of this project thesis has been submitted in support of an application for another degree or qualification. We have not submitted this thesis to any other university or educational institution. We are totally liable for any disciplinary action taken against us based on the nature of the proved offence, including the revocation of our degree.

Students:

1. **Zain Rehan**

NUST ID: 301822 Signature: ______________________

2. **Hina Arshad**

NUST ID: 306861 Signature: ______________________

3. **Ifrah Fasih**

NUST ID: 290156 Signature: ______________________

4. **Muhammad Faizan Asghar**

NUST ID: 321941 Signature: ______________________

ACKNOWLEDGEMENTS

First of all, we would like to thank Allah Almighty who gave us the courage to pursue our ambition for our final year project and who helped us in our struggles to achieve our goals. Secondly, we would like to thank our project supervisor Dr. Ahmad Rauf Subhani who guided us in selecting our final year project and helped us throughout our final year in achieving certain milestones for our project. Without his guidance we would not have been able to complete this project on time. Finally, we would like to thank our parents who prayed for us and who have high hopes for our future.

ABSTRACT

Our final year project, DriveCare, aims to improve road safety by helping drivers stay focused on the road and prevent accidents caused by distracted driving. To achieve this, we use facial expression recognition and physiological changes detection to determine the driver's emotional state and whether they are focused on driving. If the driver is detected as being distracted, an alarm is played to alert them to become more attentive.

Facial expression recognition is achieved by capturing digital images or videos of the driver, which are sent to a hardware system containing a Raspberry Pi module running machine learning and computer vision algorithms. The system analyzes the image or video stream to detect the driver's emotions. Similarly, a physiological parameter such as heart rate is detected through ECG signals, which can be processed to predict the driver's emotional state. When the driver is found to be distracted, an alarm is played to alert the driver to focus on driving.

SUSTAINABLE DEVELOPMENT GOALS

Goal 3 – GOOD HEALTH AND WELL-BEING:

Our project aims to improve road safety by helping drivers stay focused on the road and prevent accidents caused by distracted driving.

Figure 1. SDG Goal 3

Goal 9 – INDUSTRY, INNOVATION AND INFRASTRUCTURE

Our system detects emotions of drivers in real-time based on facial expressions to identify driver alertness. It also includes a robust classifier that can determine driver emotions and alertness using ECG signals.

Figure 2. SDG Goal 9

TABLE OF CONTENTS

Table of Contents

LIST OF FIGURES

LIST OF SYMBOLS

Acronyms

EEG Electroencephalogram
ECG Electrocardiogram
SVM Support Vector Machine
CNN Convolutional Neural Network
RNN Recurrent Neural Network
HRV Heart Rate Variability
GPU Graphical Processing Unit
CSV Comma Separated Values
PNG Portable Network Graphics
JPEG Joint Photographic Experts Group
SVG Scalable Vector Graphics
PDF Portable Document Format
GB Gigabyte
RAM Random Access Memory
RMSSD Root Mean Square of Successive Differences
MeanNN Mean of Normal-to-Normal Intervals
SDNN Standard Deviation of Normal-to-Normal Intervals
SDSD Standard Deviation of Successive Differences
CVNN Coefficient of Variation of Normal-to-Normal Intervals
CVSD Coefficient of Variation of Successive Differences
MedianNN Median of Normal-to-Normal Intervals
MadNN Median Absolute Deviation of Normal-to-Normal Intervals
MCVNN Median Coefficient of Variation of Normal-to-Normal Intervals
TINN Triangular Interpolation of Normal-to-Normal Intervals
HTI Heart Rate Turbulence Index
LF Low Frequency
HF High Frequency
VHF Very High Frequency
LFHF Ratio of Low Frequency to High Frequency Power
LFn Normalized Low Frequency Power
HFn Normalized High Frequency Power
LnHF Natural Logarithm of the Ratio of Low Frequency to High Frequency Power
SD1 Standard Deviation of the Poincaré Plot along the Line of Identity
SD2 Standard Deviation of the Poincaré Plot perpendicular to the Line of Identity
SD2SD1 Ratio of SD2 to SD1
CSI Cardiac Sympathetic Index
CVI Cardiac Vagal Index
CSI_Modified Modified Cardiac Sympathetic Index
SampEn Sample Entropy

Chapter 1 – INTRODUCTION

1.1. Emotion Recognition

Emotion recognition, also known as affective computing or facial expression recognition, is a technology that involves identifying and interpreting human emotions based on various cues, such as facial expressions, vocal intonation, gestures, and physiological signals. The goal of emotion recognition is to enable computers or machines to understand and respond to human emotions in a more natural and intuitive manner.

Emotion recognition systems typically use a combination of techniques from computer vision, machine learning, and signal processing to analyze and interpret emotional cues. Facial expression analysis is one of the common methods used, where the system analyzes facial features and movements to detect emotions like happiness, sadness, anger, fear, surprise, and disgust. This can be done by detecting specific facial muscle movements or by analyzing the overall configuration of facial features.

Other modalities, such as speech analysis, body language, and physiological signals like heart rate and skin conductance, can also be incorporated to enhance the accuracy of emotion recognition. For example, speech analysis can involve extracting features from voice recordings to identify emotions conveyed through tone, pitch, and speech patterns.

Emotion recognition technology has various applications across different fields. It can be used in human-computer interaction to create more personalized and adaptive interfaces, in market research to gauge consumer reactions, in mental health to assist with diagnosis and treatment, and in robotics to enable robots to better understand and respond to human emotions, among other uses.

1.2. Emotion Recognition Techniques To Determine Driver Alertness

Emotion recognition can be used to determine driver alertness by analyzing various cues and physiological signals associated with alertness and drowsiness. The common approaches are listed in this section.

1.2.1. Facial Expression Analysis

Emotion recognition systems can inspect the driver's facial expressions, particularly eye-related cues like eyelid movements, blink rate, and eye closure duration. Excessive eye blinking, frequent or prolonged eye closures, or drooping eyelids can indicate drowsiness or fatigue.

1.2.2. Speech Analysis

Emotion recognition systems can analyze the driver's speech patterns, voice tone, and vocal cues to detect signs of drowsiness. Changes in speech characteristics, such as slower speech rate, monotone or slurred speech, or longer pauses between words, may indicate reduced alertness.

1.2.3. Physiological Signals

Emotion recognition systems can monitor various physiological signals to assess driver alertness. This can include analyzing heart rate variability, skin conductance, and brainwave activity (using EEG). Decreased heart rate variability, reduced skin conductance, and patterns of brainwave activity associated with drowsiness can indicate a decline in alertness.

1.2.4. Driving Behavior Analysis

Emotion recognition systems can also consider driving behavior as an indicator of alertness. This may involve monitoring factors like steering patterns, lane deviation, and reaction times. Erratic or inconsistent driving behavior can suggest reduced alertness.

By combining these cues and signals, an emotion recognition system can continuously monitor a driver's state and provide alerts or warnings when signs of drowsiness or reduced alertness are detected. These alerts can help prevent accidents by prompting the driver to take necessary rest breaks or engage in activities that increase alertness, such as pulling over, stretching, or consuming caffeinated beverages. Additionally, such systems can be integrated into advanced driver assistance systems (ADAS) to provide real-time feedback and intervention to prevent potential accidents caused by driver fatigue.

1.3. Procedure For Emotion Recognition

Emotion recognition is performed using a combination of techniques from computer vision, machine learning, and signal processing. Here's an overview of the general process:

1.3.1. Data Acquisition

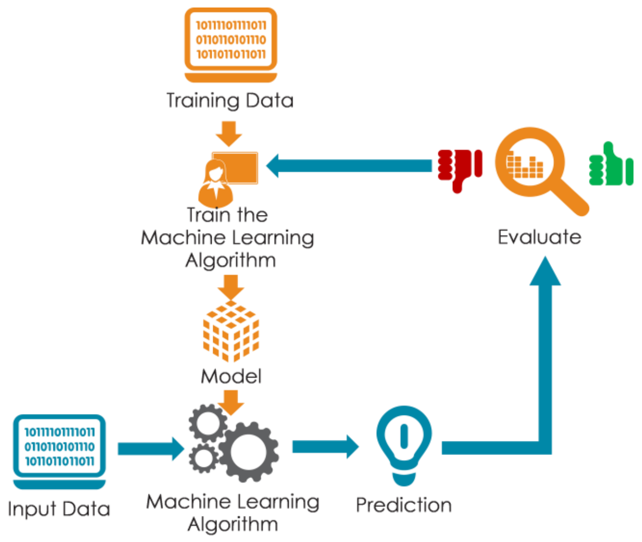
Emotion recognition systems acquire data from various sources, such as images or videos of facial expressions, audio recordings of speech, or physiological signals from sensors. This data serves as the input for subsequent analysis.

1.3.2. Feature Extraction

Relevant features are extracted from the acquired data to capture meaningful information related to emotions. For facial expression analysis, features like facial landmarks, facial muscle movements, or texture descriptors may be extracted. In speech analysis, features such as pitch, intensity, and speech rhythm are commonly used. Physiological signals like heart rate or skin conductance may undergo preprocessing and feature extraction as well.

1.3.3. Training and Model Development

Machine learning algorithms are trained on labeled datasets, where emotions are annotated or categorized for each input sample. These algorithms learn patterns and relationships between the extracted features and the corresponding emotions. Popular machine learning approaches for emotion recognition include support vector machines (SVMs), decision trees, random forests, or more recently, deep learning techniques such as convolutional neural networks (CNNs) or recurrent neural networks (RNNs).



Training Data
Train the Machine Learning Algorithm
Evaluate
Model
Input Data
Machine Learning Algorithm
Prediction

Figure 3. Machine Learning Process

1.3.4. Model Evaluation and Validation

The trained models are evaluated on separate datasets to assess their performance in accurately recognizing emotions. Various metrics like accuracy, precision, recall, or F1 score are used to evaluate the model's effectiveness.

1.3.5. Real-Time Application

Once the model is deemed satisfactory, it can be deployed for real-time emotion recognition applications. The model takes new, unseen data as input and predicts the corresponding emotion based on the learned patterns.

The evolution of emotion recognition has been influenced by advancements in technology, computing power, and the availability of large labeled datasets. Initially, emotion recognition primarily focused on analyzing facial expressions, relying on handcrafted features and traditional machine learning algorithms. However, with the rise of deep learning and the availability of large datasets like the Facial Action Coding System (FACS), researchers shifted towards using convolutional neural networks and recurrent neural networks for more robust and accurate emotion recognition.

Additionally, the field has expanded to incorporate other modalities such as speech analysis, body language, and physiological signals. Multimodal approaches that combine multiple modalities have been explored to improve accuracy and robustness in recognizing emotions.

The evolution of emotion recognition has also seen an increasing emphasis on real-time and context-aware applications. Researchers are developing systems that can adapt to individual differences, cultural variations, and dynamic environmental conditions to enhance the practical utility of emotion recognition technology.

Overall, the field of emotion recognition has evolved from rudimentary systems based on handcrafted features and traditional algorithms to more sophisticated and accurate models driven by deep learning and multimodal approaches, enabling applications in various domains such as human-computer interaction, healthcare, and driver safety.

1.4. Our Aim For This Project

The primary objective of DriveCare is to reduce traffic collisions that result from distracted driving. This can be done by preventing the driver from losing focus on the road. The project aims to develop an advanced system that can precisely identify and detect instances of distracted driving in real-time through the use of innovative methods and modern technologies like computer vision, and machine learning. To determine whether or not the driver is being distracted by emotions such as excitement, anger, surprise or fear, the system examines multiple indications including facial expressions, and ECG signals. The system takes initiatives to support safer driving behavior and reduce the risk of accidents by quickly identifying and alerting the driver to moments of distraction.

Chapter 2 – BACKGROUND AND LITERATURE REVIEW

2.1. Background Of Emotion Recognition Using Facial Expressions

Emotion recognition using facial expressions is a fascinating area of research that has gained significant attention in recent years. The ability to automatically detect and interpret human emotions has numerous applications in fields such as human-computer interaction, psychology, marketing, and healthcare. This section provides a detailed background and literature review of emotion recognition using facial expressions, highlighting key research contributions and advancements.

2.1.1. Early Studies and Facial Action Coding System (FACS)

The foundation of emotion recognition using facial expressions can be traced back to the pioneering work of Paul Ekman and Wallace V. Friesen in the 1970s. They developed the Facial Action Coding System (FACS), which identified and classified facial muscle movements, known as Action Units (AUs), associated with various facial expressions. FACS provided a standardized framework for describing and analyzing facial expressions, forming the basis for subsequent research in the field.

2.1.2. Feature-Based Approaches

Early studies on emotion recognition focused on extracting handcrafted features from facial images. Researchers employed geometric features, such as the distances between facial landmarks, as well as texture-based features, such as Gabor filters or Local Binary Patterns (LBP). These features were then fed into classifiers, such as Support Vector Machines (SVMs), Hidden Markov Models (HMMs), or Artificial Neural Networks (ANNs), to recognize emotions. While these approaches achieved moderate success, they often faced challenges in dealing with variations in lighting, pose, and occlusions.

2.1.3. Deep Learning Approaches

The advent of deep learning revolutionized the field of emotion recognition. Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs) have shown remarkable performance in various computer vision tasks, including emotion recognition using facial expressions.

CNN-based architectures have been used to automatically learn discriminative features directly from raw facial images. These deep CNN models, such as VGGNet, ResNet, and Inception, have demonstrated superior performance in emotion recognition tasks, surpassing traditional handcrafted feature-based approaches. Transfer learning techniques, where pre-trained CNN models are fine-tuned on emotion-specific datasets, have also been effective in addressing data scarcity and improving performance.

RNN-based models, such as Long Short-Term Memory (LSTM) networks and Gated Recurrent Units (GRUs), have been employed to capture temporal dependencies and sequential information in facial expression sequences. These models have been successful in recognizing dynamic changes in facial expressions over time, enhancing the accuracy of emotion recognition systems.

2.1.4. Datasets and Evaluation

The availability of annotated datasets plays a crucial role in the development and evaluation of emotion recognition systems. Several benchmark datasets have been widely used in the field, such as the Cohn-Kanade dataset, the MMI dataset, the Oulu-CASIA dataset, and the Extended Cohn-Kanade dataset. These datasets provide labeled facial expression samples captured under controlled conditions, facilitating fair comparisons and benchmarking of different algorithms.

Evaluation metrics commonly used in emotion recognition include accuracy, precision, recall, F1-score, and the area under the Receiver Operating Characteristic (ROC) curve. Cross-validation and leave-one-subject-out strategies are often employed to assess the generalization performance of models.

2.1.5. Challenges

Despite significant progress, emotion recognition using facial expressions still faces several challenges. Variations in lighting conditions, pose, occlusions, and facial characteristics across individuals remain areas of concern. Efforts are being made to develop robust algorithms that are invariant to such variations and can generalize well across different populations.

Another challenge lies in recognizing complex emotions beyond basic categories. While early studies focused on basic emotions, there is growing interest in recognizing subtle and fine-grained emotions, such as contempt, pride, or confusion. The development of comprehensive emotion models and the collection of diverse and large-scale datasets are crucial in addressing this challenge.

Additionally, ethical considerations related to privacy, consent, and potential biases in automated systems need to be carefully addressed. Ensuring fairness and transparency in emotion recognition technology is essential to prevent potential misuse and unintended consequences.

2.1.6. Emotion Recognition Using Facial Expressions

Advancements in computer vision and machine learning techniques have fueled the development of automated emotion recognition systems in recent years. These systems aim to mimic human perception by analyzing facial features, such as the movement of facial muscles, facial landmarks, and overall facial expressions, to infer the emotional state of an individual. The key steps involved in emotion recognition using facial expressions are listed below.

2.1.6.1. Face Detection

The first step is to detect and locate human faces in an image or video frame. Face detection algorithms use various techniques like Viola-Jones algorithm, Haar cascades, or deep learning-based methods (e.g., convolutional neural networks) to identify facial regions.

2.1.6.2. Facial Feature Extraction

Once the face is detected, relevant facial features are extracted to capture the subtle changes in facial expressions. These features include facial landmarks (e.g., eye corners, nose tip, mouth corners), head pose, facial action units, and geometric distances between specific facial points.

2.1.6.3. Feature Representation

The extracted facial features need to be converted into a suitable representation that can be used by machine learning algorithms. Common representations include geometric features, appearance-based features (e.g., local binary patterns), or higher-level representations obtained using deep learning architectures such as convolutional neural networks (CNNs) or recurrent neural networks (RNNs).

2.1.6.4. Emotion Classification

The extracted and represented features are then used to train machine learning models, such as support vector machines (SVMs), decision trees, random forests, or deep learning models, to classify the facial expressions into specific emotional categories. These categories can be basic emotions (e.g., happy, sad, angry) or a more complex set of emotions (e.g., happiness, surprise, disgust, fear, anger, sadness, neutral).

2.1.6.5. Model Evaluation

The performance of the emotion recognition system is assessed using evaluation metrics such as accuracy, precision, recall, or F1-score. The models are typically trained and tested on labeled datasets containing facial expressions annotated with corresponding emotion labels.

2.2. Background Of Emotion Recognition Using Electrocardiography (ECG) Signals

Traditional approaches to emotion recognition primarily relied on subjective self-reporting or behavioral observations. However, these methods have limitations in terms of accuracy and objectivity. To overcome these limitations, researchers have explored physiological signals, such as electrocardiogram (ECG), as a potential modality for emotion recognition. This section provides a background and literature review of using ECG signals for emotion recognition.

2.2.1. Physiological Basis of ECG Signals

The ECG signal reflects the electrical activity of the heart, capturing the depolarization and repolarization processes. It consists of various waveform components, including the P-wave, QRS complex, and T-wave. The characteristics of these components, as well as the heart rate variability (HRV), have been found to be influenced by emotional states. This physiological basis forms the foundation for utilizing ECG signals in emotion recognition.

2.2.2. Feature Extraction

Researchers have identified a wide range of features that can be extracted from ECG signals to capture relevant information related to emotional states. These features include HRV measures, time-domain features, frequency-domain features, and non-linear features. Each feature provides unique insights into the autonomic modulation of the heart and the physiological changes associated with emotions.

2.2.3. Machine Learning Techniques

Various machine learning algorithms have been applied to ECG-based emotion recognition, including decision trees, support vector machines, artificial neural networks, and deep learning models. These algorithms leverage the extracted features from ECG signals to classify and predict emotional states accurately.

2.2.4. Multi-Modal Approaches

To improve the accuracy and robustness of emotion recognition systems, researchers have explored multi-modal approaches by combining ECG signals with other modalities such as facial expressions, speech analysis, or electroencephalogram (EEG) signals. These multimodal systems leverage the complementary information from different modalities to enhance emotion recognition performance.

2.2.5. Challenges

Collecting large-scale labeled ECG datasets with accurate and reliable emotion labels remains a challenge. Obtaining ground truth emotional annotations is subjective and often relies on self-reporting, which may introduce bias or inconsistency. Standardized protocols for data collection and annotation are essential to ensure the validity and comparability of studies.

Emotions can vary significantly across individuals, and generalizing emotion recognition models to diverse populations poses challenges. Individual differences in ECG signal characteristics, such as age, gender, and physiological conditions, need to be considered to develop robust and personalized models.

ECG signals are susceptible to various artifacts and noise, including motion artifacts, electrical interference, and baseline wander. These interferences can affect the accuracy and reliability of emotion recognition algorithms. Preprocessing techniques, such as artifact removal and noise reduction, are necessary to enhance the quality of ECG signals.

2.2.6. Emotion Recognition Using ECG Signals

Emotion recognition using ECG signals involves several steps and techniques to capture and analyze the physiological changes associated with emotional states.

2.2.6.1. Data Acquisition

ECG signals are recorded using electrodes placed on the body, typically on the chest, that detect the electrical activity of the heart. The electrodes capture the voltage fluctuations generated by the heart's electrical impulses.

2.2.6.2. Preprocessing

The acquired ECG signals undergo preprocessing to remove noise and artifacts that may interfere with accurate analysis. Common preprocessing techniques include filtering to eliminate high-frequency noise and baseline wander, as well as artifact removal methods to address motion artifacts or electrode contact issues.

2.2.6.3. Feature Extraction

Relevant features are extracted from the preprocessed ECG signals to capture the distinctive patterns associated with different emotional states. Feature extraction methods can include time-domain features (e.g., heart rate, RR intervals), frequency-domain features (e.g., power spectral density, spectral entropy), and nonlinear features (e.g., heart rate variability, fractal dimension).

2.2.6.4. Emotion Labeling

Emotion labeling involves eliciting or inducing specific emotional states in the subjects during the data recording session. This can be achieved through various methods, such as presenting emotional stimuli (images, videos, audio clips) or using specific tasks designed to evoke emotions.

2.2.6.5. Training and Classification

Once the features are extracted and labeled with corresponding emotions, machine learning or deep learning algorithms are employed to train a model for emotion classification. Various classification algorithms can be used, including support vector machines (SVM), k-nearest neighbors (KNN), random forests, or deep learning models like convolutional neural networks (CNNs) or recurrent neural networks (RNNs).

2.2.6.6. Validation and Testing

The trained model is validated and tested using separate datasets to assess its performance in recognizing emotions from ECG signals. Performance metrics such as accuracy, precision, recall, and F1 score are calculated to evaluate the model's effectiveness.

2.2.6.7. Real-Time Recognition

In some applications, real-time emotion recognition is desired. In such cases, the trained model is implemented in a system or device that can continuously acquire and process ECG signals in real-time. This allows for dynamic and immediate recognition of the user's emotional state.

Chapter 3 – METHODOLOGY

3.1. Software Components

Our project includes machine learning and computer vision approaches. These methods require different software applications for development and testing. Our models run in these programs and also have the capability to show their output in them. In this section, we will discuss the software applications, models, and software libraries we have used for our project.

3.1.1. Software Applications Used

We have used three software applications for our project namely, PyCharm, Google Colab, and Thonny IDE.

3.1.1.1. PyCharm

PyCharm is a popular Integrated Development Environment (IDE) specifically designed for Python development. It is developed by JetBrains and provides a comprehensive set of tools and features to enhance productivity and streamline the Python development workflow. PyCharm offers advanced code editing capabilities, including syntax highlighting, code completion, and intelligent code refactoring. It also supports version control systems like Git, and provides built-in tools for debugging, testing, and profiling Python code. PyCharm is available in two editions: Community (free and open-source) and Professional (commercial with additional features).

Figure 4. PyCharm Icon

3.1.1.2. Google Colab

Google Colab (short for Collaboratory) is an online platform that provides a free Jupyter notebook environment for running and executing Python code. It is a cloud-based service offered by Google and is part of the Google Cloud ecosystem. Google Colab allows users to create and share interactive notebooks that combine code, text, and visualizations. It provides access to powerful hardware resources, including GPUs and TPUs, enabling users to execute computationally intensive tasks. Colab integrates with other Google services like Google Drive, allowing users to easily import and export data. It also supports collaborative editing, making it ideal for teamwork and educational purposes.

Figure 5. Google Colab Logo

3.1.1.3. Thonny IDE

Thonny IDE is commonly used with Raspberry Pi as it provides a beginner-friendly environment for programming and learning Python on the Raspberry Pi platform. It provides a simplified interface, making it easy to write, debug, and run Python code. Thonny IDE offers features such as syntax highlighting, code completion, and a built-in debugger to assist learners in understanding and troubleshooting their code. It also includes a simple step-by-step execution mode to help beginners grasp the flow of program execution. Thonny IDE is cross-platform and supports various Python versions.

Figure 6. Thonny Logo

3.1.2. Model Used For Emotion Recognition Using Facial Expressions

The model that we have used for emotion recognition is DeepFace. This is a deep learning facial recognition algorithm developed by a group of researchers from Facebook’s artificial intelligence research division. In digital photographs, it recognises human faces. The program was trained on four million photographs shared by Facebook users and uses a nine-layer neural network with approximately 120 million connection weights. According to the Facebook Research team, the DeepFace approach achieves an accuracy of 97.35% ± 0.25% on the Labeled Faces in the Wild (LFW) data set, compared to humans who have a 97.53% accuracy rate.

The DeepFace architecture consists of four modules, namely, 2D alignment, 3D alignment, frontalization, and neural network. They are sequentially applied to a facial image to produce a 4096-dimensional feature vector that represents the face. The feature vector can then be processed further for a variety of purposes. The face may be recognised, for instance, by comparing it to a collection of feature vectors of well-known faces and selecting the one with the closest match. To guide the alignment of faces, DeepFace makes use of fiducial point detectors based on existing databases. A 2D alignment sets the stage for the facial alignment, which progresses into a 3D alignment and frontalization. Therefore, there are two steps in DeepFace's procedure. First, it adjusts an image's angles such that the subject's face is facing forward. It makes use of a 3-D representation of a face to achieve this.

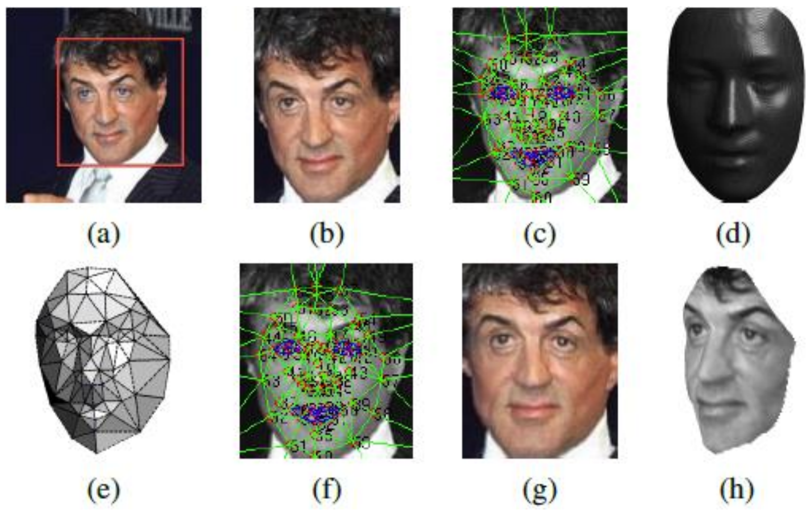


Figure 7. DeepFace Alignment Pipeline

3.1.2.1. 2D Alignment

The centre of the eyes, the tip of the nose, and the location of the mouth are among the 6 fiducial points on the identified face that the 2D alignment module finds. To assist in locating the face, these markers are projected onto a warped image. However, 2D transformation is unable to account for erroneous rotations.

3.1.2.2. 3D Alignment

DeepFace uses a general 3D model, where 2D photos are cropped as 3D counterparts, to align faces. There are 67 fiducial points in the 3D image. 67 anchor points are manually added to the image after it has been distorted in order to align it with the 67 fiducial points. Then, a 3D-to-2D camera is fitted to reduce losses. This phase is crucial since 3D identified locations on the face's contour can be erroneous.

3.1.2.3. Frontalization

The fitted camera only approximates the subject's actual face because full perspective projections are not modelled. DeepFace seeks to warp the 2D images with tiny distortions in order to decrease errors.

3.1.2.4. Neural Network

The neural network is made up of several layers that are structured as follows: convolutional layer – max pooling – convolutional layer – 3 locally connected layers – fully connected layer. The input is a 152 x 152 pixel RGB image of a face, and the output is a 4096-dimensional real vector that represents the face image's feature vector.

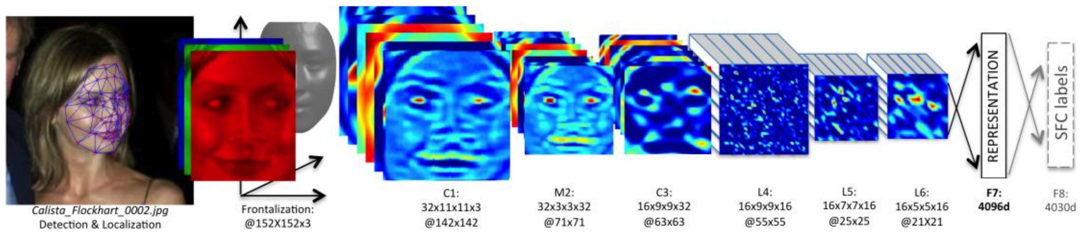


Figure 8. DeepFace Neural Network

3.1.3. Software Libraries Required For Facial Expressions Method

The software libraries required for emotion recognition using facial expressions are DeepFace, OpenCV, TensorFlow, Keras, NumPy, and Pandas. Each software library used is described in this section.

3.1.3.1. DeepFace

Figure 9. DeepFace Logo

A wide variety of facial recognition and facial attribute analysis features can be obtained by the open-source Python package named as DeepFace. Deep learning algorithms for applications like face recognition, facial expression examination, facial feature estimations, and verification of faces can be implemented.

Using the assistance of models that have been pre-trained such as VGG-Face, Google FaceNet, and OpenFace, DeepFace is a deep learning framework constructed on the foundation of prominent deep learning frameworks like Keras and TensorFlow. These algorithms are capable of obtaining multidimensional facial embedded data that represent unique features because they were trained on massive datasets.

Through the assistance of DeepFace, users are able to carry out operations such as face verification (figuring out if two faces represent a single individual), face identification (identifying a person’s face using a database of identified faces), and facial feature analysis (for predicting attributes such as age, gender, mood, and race from facial images).

A person can simply load already trained models, evaluate picture or video frames, retrieve facial features, or conduct facial recognition operations by utilizing the library's simple and intuitive API. DeepFace is worthwhile for building face analysis workflows as it includes additional utility functions for face alignment, preprocessing, and visualization.

3.1.3.2. OpenCV

Figure 10. OpenCV Logo

A broad range of tools and algorithms enabling image and video

processing, object detection and tracking, and computer vision projects can be found in the open-source computer vision and machine learning package referred to as OpenCV.
Although being developed in C++, OpenCV possesses Python interfaces for numerous different programming languages. Over 2,500 effective algorithms are offered that can be used for various applications involving computer vision. These algorithms deal with a range of subjects, which include image and video processing, object recognition, geometric transformations, picture filtering, and feature detection and extraction.

Real-time computer vision projects management represent one of the reasons why OpenCV is preferred. OpenCV includes classes and functions that make it possible to effectively extract, analyze, and present video and picture feeds from cameras or files. This makes it ideal for applications in robotics, augmented reality, driverless cars, and video surveillance.

3.1.3.3. TensorFlow

Figure 11. TensorFlow Logo

TensorFlow is an open-source machine learning framework developed by Google that is widely used for building and deploying various types of deep learning models. It provides a comprehensive ecosystem of tools, libraries, and resources to develop and train machine learning models efficiently.

At its core, TensorFlow revolves around the concept of tensors, which are multidimensional arrays representing the data used in computations. These tensors flow through a computational graph, which is a series of operations or nodes connected by edges. TensorFlow allows you to define and execute complex mathematical computations on these tensors efficiently, making it particularly suitable for deep learning tasks.

3.1.3.4. Keras

Figure 12. Keras Logo

Keras is a powerful and user friendly open-source framework that allows developers to speedily prototype and shape neural networks. With its advanced interface and flexible design, Keras removes complications of applying deep learning models, empowering operators to concentrate on the core elements of their research or application. The development procedure is sped up by the wide range of pre-built neural network layers and tools delivered by Keras. Utilizing these parts enables operators to shape complex models with little to no coding, saving both time and resources.

Furthermore, Keras links effortlessly with other renowned deep learning libraries like TensorFlow, permitting effective utilization of hardware resources, which includes GPUs. Researchers control massive datasets more competently which may improve performance and accelerate convergence.

3.1.3.5. NumPy

Figure 13. NumPy Logo

NumPy is an open-source library for numerical computing in Python. To competently perform calculations on enormous datasets, it proposes a multidimensional array object in addition to a selection of mathematical operations. Due to its capacity to effortlessly and efficiently manage arrays and matrices, NumPy is often employed in scientific and data analysis applications.

The vital part of NumPy is the ndarray (n-dimensional array) object that provides a way to save and interact with huge datasets in the most efficient way possible. It is a versatile and efficient way of conducting computational tasks on arrays that includes element-wise operations, linear algebraic operations, and statistical calculations.

3.1.3.6. Pandas

Figure 14. Pandas Logo

Renowned open-source Python package Pandas serves as a tool for data analysis and data management. It enables adaptable management and handling of structured data through the application of efficient and simple-to-use data structures, such as data frames.

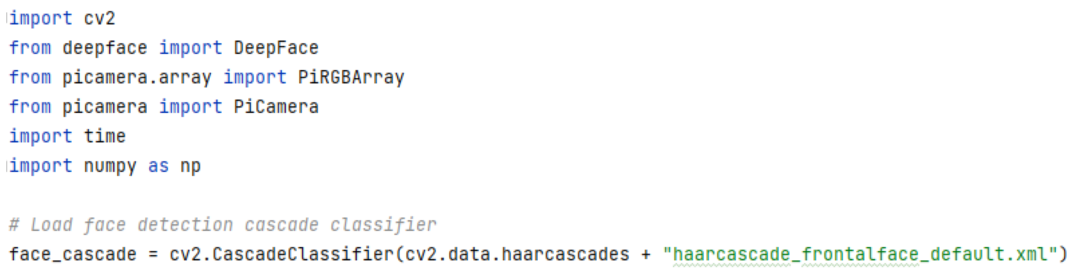
The DataFrame, a two-dimensional annotated data structure resembling to a table or spreadsheet, is the basic data structure in Pandas. With actions such as indexing, slicing, joining, merging, and reshaping allowed, DataFrames provide a powerful method to arrange, handle, and evaluate data. With Pandas, the user can perform data cleaning and preprocessing processes, use complicated data analysis approaches, and load data from a range of different format types (such as CSV, Excel, and SQL databases) to a DataFrame.

3.1.3.7. Picamera

The Picamera library is a Python library created especially for the Raspberry Pi camera module. It delivers a simple and effortless process to communicate with and regulate the camera module linked to a Raspberry Pi board. The library encourages different capture modes, which includes single image capture, constant image capture, and video recording. It provides the functionality to observe the camera feed on the Raspberry Pi's display and offers options for recording individual frames or saving images and videos to the Pi's storage.

3.1.4. Method For Driver Alertness Detection Using Facial Expressions

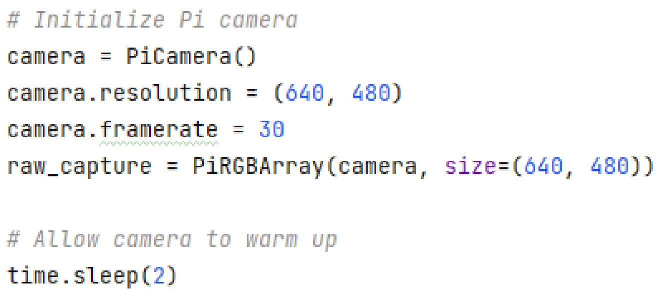
At the start of our code, the required software libraries are imported. Thereafter, the face detection haar cascade classifier is loaded. This classifier has the ability to detect faces in an image or video to determine the region of interest for the face.



```python
import cv2
from deepface import DeepFace
from picamera.array import PiRGBArray
from picamera import PiCamera
import time
import numpy as np

# Load face detection cascade classifier
face_cascade = cv2.CascadeClassifier(cv2.data.haarcascades + "haarcascade_frontalface_default.xml")
```

Figure 15. Code Snippet 1 for Facial Expressions Method

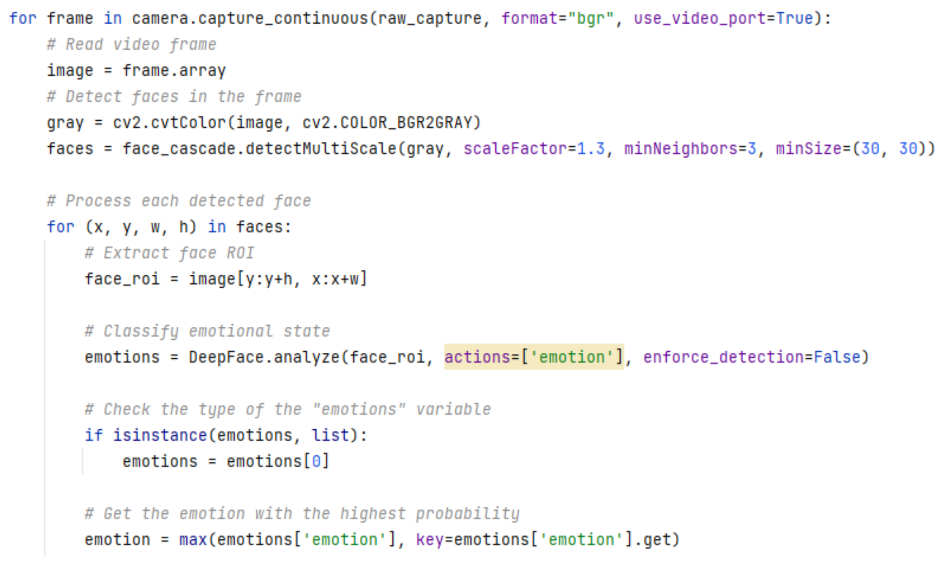
After the classifier is loaded, the Pi camera is initialized and allowed some time to warm up to adjust according to the lighting conditions before the videostream displays on the touchscreen.



```python
# Initialize Pi camera
camera = PiCamera()
camera.resolution = (640, 480)
camera.framerate = 30
raw_capture = PiRGBArray(camera, size=(640, 480))

# Allow camera to warm up
time.sleep(2)
```

Figure 16. Code Snippet 2 for Facial Expressions Method

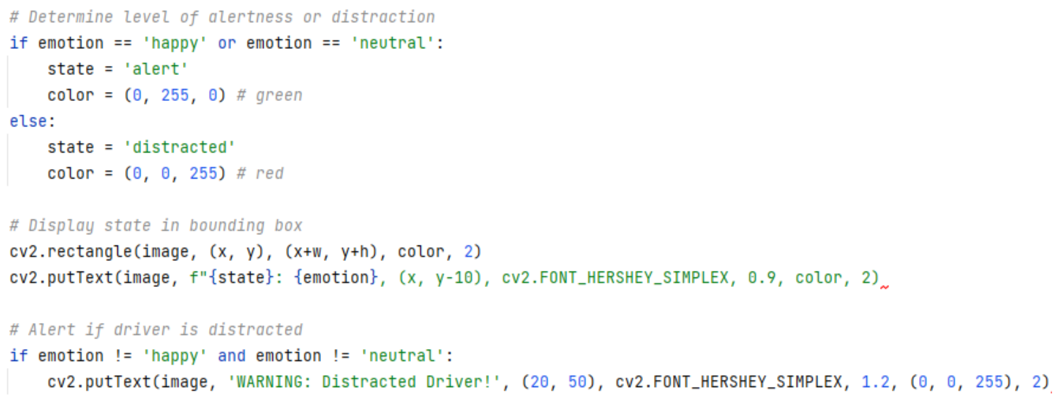
Once the camera starts displaying the video stream, it starts looking for faces in the camera frame. Any person who is present in the frame, his or her face is then detected. The detected face is cropped from the frame and then sent to the DeepFace model to analyze the detected emotion. The DeepFace model returns probabilities for the possible emotions. The emotion with the highest probability is selected.



```python
for frame in camera.capture_continuous(raw_capture, format="bgr", use_video_port=True):
    # Read video frame
    image = frame.array
    # Detect faces in the frame
    gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
    faces = face_cascade.detectMultiScale(gray, scaleFactor=1.3, minNeighbors=3, minSize=(30, 30))

    # Process each detected face
    for (x, y, w, h) in faces:
        # Extract face ROI
        face_roi = image[y:y+h, x:x+w]

        # Classify emotional state
        emotions = DeepFace.analyze(face_roi, actions=['emotion'], enforce_detection=False)

        # Check the type of the "emotions" variable
        if isinstance(emotions, list):
            emotions = emotions[0]

        # Get the emotion with the highest probability
        emotion = max(emotions['emotion'], key=emotions['emotion'].get)
```

Figure 17. Code Snippet 3 for Facial Expressions Method

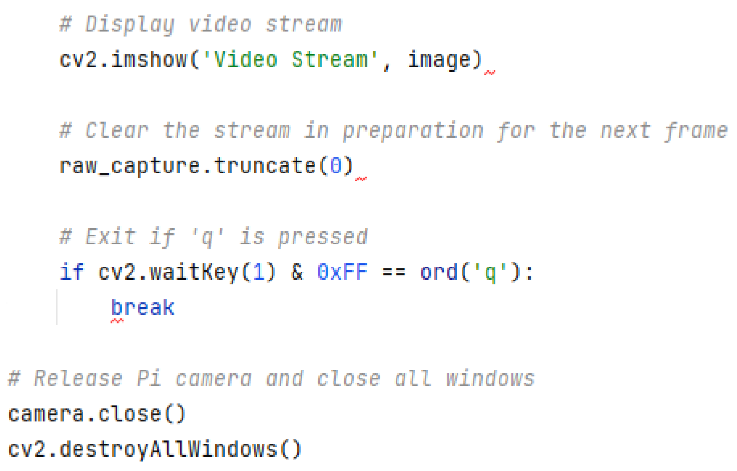
Possible emotions that can be detected include: happy, angry, fear, sad, neutral, surprise, and disgust. If the detected emotion is ‘happy’ or ‘neutral’, the state of the driver is alert, otherwise the driver’s state is distracted. The emotion and driver state are annotated onto the image.



```python
# Determine level of alertness or distraction
if emotion == 'happy' or emotion == 'neutral':
    state = 'alert'
    color = (0, 255, 0) # green
else:
    state = 'distracted'
    color = (0, 0, 255) # red

# Display state in bounding box
cv2.rectangle(image, (x, y), (x+w, y+h), color, 2)
cv2.putText(image, f"{state}: {emotion}", (x, y-10), cv2.FONT_HERSHEY_SIMPLEX, 0.9, color, 2)

# Alert if driver is distracted
if emotion != 'happy' and emotion != 'neutral':
    cv2.putText(image, 'WARNING: Distracted Driver!', (20, 50), cv2.FONT_HERSHEY_SIMPLEX, 1.2, (0, 0, 255), 2)
```

Figure 18. Code Snippet 4 for Facial Expressions Method

The annotated text is then displayed directly on the video stream around the user’s face to allow the user to view the results. This process continues until the user presses the ‘q’ key on the keyboard.



```
    # Display video stream
    cv2.imshow('Video Stream', image)

    # Clear the stream in preparation for the next frame
    raw_capture.truncate(0)

    # Exit if 'q' is pressed
    if cv2.waitKey(1) & 0xFF == ord('q'):
        break

# Release Pi camera and close all windows
camera.close()
cv2.destroyAllWindows()
```

Figure 19. Code Snippet 5 for Facial Expressions Method

3.1.5. Dataset Used for Emotion Recognition Using ECG Signals

We have utilized the DREAMER dataset for building an accurate emotion recognition algorithm using ECG signals. The DREAMER dataset is a comprehensive database designed for studying and advancing emotion recognition using EEG (Electroencephalogram) and ECG (Electrocardiogram) signals. It serves as a valuable resource for researchers and developers working in the field of affective computing, human-computer interaction, and emotion recognition systems.

3.1.5.1. Overview Of Dataset

The DREAMER database was developed as a collaborative effort between several research institutions and universities. It was created with the aim of providing a standardized and reliable dataset for emotion recognition studies. The database consists of physiological signals captured from human participants during various emotional stimuli, inducing a range of emotional states.

3.1.5.2. Data Collection

To build the DREAMER database, EEG and ECG signals were recorded from 23 participants, including both males and females. The participants were subjected to audiovisual stimuli, including video clips from movies and music videos, designed to elicit different emotional responses. They were asked to watch these stimuli while their physiological signals were being recorded.

3.1.5.3. EEG Signals

Electroencephalography (EEG) signals were recorded using high-density electrode arrays placed on the participants' scalps. These electrodes captured the electrical activity of the brain, providing insights into the neural correlates of emotion. The EEG signals were sampled at a frequency of 256 Hz, providing a high temporal resolution.

3.1.5.4. ECG Signals

Electrocardiography (ECG) signals were recorded simultaneously with the EEG signals. ECG measures the electrical activity of the heart, which can reflect changes in emotional states. The ECG signals were sampled at a frequency of 256 Hz as well.

3.1.5.5. Emotion Induction

The emotional stimuli used in DREAMER were carefully selected to induce a wide range of emotions. The database includes video clips from multiple genres, such as comedy, drama, horror, and action. Additionally, a variety of music videos were incorporated, covering different musical genres. This diversity allows researchers to study emotional responses in various contexts.

3.1.5.6. Annotation and Metadata

DREAMER provides extensive annotations and metadata for each recording session. The emotional responses of participants were assessed using the Self-Assessment Manikin (SAM) method, where participants rate their emotional valence (positive/negative) and arousal (low/high) levels. These ratings help in quantifying the emotional states experienced during the experiment.

3.1.5.7. Availability and Usage

DREAMER is freely available to researchers, making it a widely used resource in the field. The database can be accessed and downloaded from the official DREAMER website or other authorized platforms. Researchers can utilize the dataset for training and evaluating machine learning algorithms, developing emotion recognition models, and exploring the relationship between emotions and physiological signals.

3.1.5.8. Importance and Impact

The DREAMER database plays a crucial role in advancing emotion recognition research. By providing a standardized dataset, it enables researchers to compare and benchmark different algorithms and methodologies. It also encourages collaboration and knowledge sharing within the scientific community, fostering innovation in emotion recognition technologies.

3.1.6. Dimensions of Emotional Experience

The three dimensions of valence, dominance, and arousal are frequently employed to define and quantify emotional experiences. They offer a structure for comprehending and classifying various emotional states.

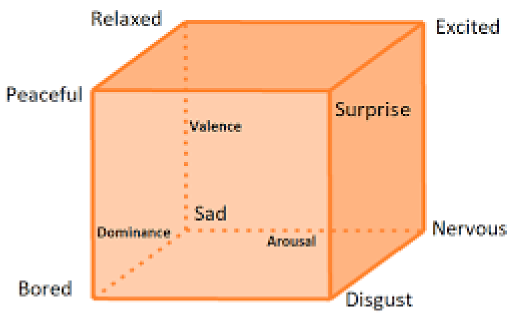


Figure 20. 3D Model of Valence, Arousal and Dominance

3.1.6.1. Valence

The subjective nature of an emotional experience is referred to as valence. It shows how pleasant or terrible an emotion is to a certain extent. Emotions can be extremely positive (like pleasure and happiness) or extremely negative (like despair and fear). Valence is frequently visualised as a spectrum, with positive feelings at one end, negative emotions at the other, and neutral emotions in the centre.

3.1.6.2. Dominance

The degree of power or influence someone feels over their emotional state or the circumstances they are in is referred to as dominance. It shows the person's perceived authority, control, or power over their emotions or the surrounding environment. High dominance expresses a sense of being overpowered or weak, whereas Low dominance suggests a sense of control or being in command.

3.1.6.3. Arousal

Arousal denotes the degree of intensity of an emotional experience. It relates to emotional-related physiological and psychological activity. Low arousal is characterised by tranquilly, relaxation, or a lack of stimulation, while high arousal is characterised by an elevated level of attentiveness, excitement, or activation. A continuum of arousal, spanning from low to high intensity, can also be imagined.

3.1.8. Software Libraries Required for ECG Signals Method

The software libraries required for emotion recognition using ECG signals are SciPy, NeuroKit2, scikit-learn, and Plotly. Each software library used is described in this section.

3.1.8.1. SciPy

Figure 21. SciPy Logo

SciPy is a Python library that includes a variety of functions for scientific computing and signal processing. It provides a wide range of numerical routines and algorithms that were required in the development of the emotion recognition model.

SciPy’s signal module includes processes for filtering, Fourier analysis, and other signal processing operations. This aided us in the preprocessing of ECG signals, allowing for noise reduction and the extraction of essential features.

The statistics module in SciPy includes a number of statistical functions that aid in data analysis and model evaluation. It enabled the generation of assessment measures such as Mean Squared Error (MSE) and provided statistical insights into the model's performance.

3.1.8.2. NeuroKit2

Figure 22. NeuroKit2 Logo

NeuroKit2 is a Python package built primarily for the processing of physiological signals, such as ECG readings. It includes a wide range of tools for signal processing, feature extraction, and analysis.

NeuroKit2 has preprocessing and analysis functions for ECG signals allowing for the extraction of key variables such as heart rate variability (HRV) and morphological properties. These characteristics were critical in capturing the emotional patterns found in the ECG signals.

NeuroKit2 provides simple methods for extracting time-domain and frequency-domain physiological properties from ECG signals. These characteristics contributed to the emotion detection procedure by providing vital insights into the autonomic modulation of the heart.

3.1.8.3. scikit-learn

Figure 23. scikit-learn Logo

scikit-learn is a popular Python machine-learning package that provides a complete collection of tools for a variety of tasks such as classification, regression, and model evaluation. It was crucial in training and testing our emotion recognition model.

This package includes multiple machine learning algorithms, including XGBoost and Random Forest Regressor. These algorithms were used in the model training process to learn intricate correlations between retrieved data and emotion labels.

scikit-learn provides metrics and functions for evaluating the performance of machine learning models. Metrics such as Mean

Squared Error (MSE) and accuracy were used to assess the model's predictive accuracy and accurate emotional recognition ability.

3.1.8.4. Plotly

Figure 24. Plotly Logo

Plotly is a Python module for building aesthetically appealing and interactive data visualisations. Plotly provides high-quality visualisations that are suited for publication or presentation. The library supports exporting visualisations in a variety of formats, including PNG, JPEG, SVG, and PDF.

Plotly provides a variety of plot types, including line plots, scatter plots, bar charts, histograms, and 3D surface plots. It includes interactive elements such as zooming, panning, and tooltips to improve the user experience. Colours, typefaces, marks, and other visual aspects can all be customised to reflect personal design preferences.

Plotly has an easy-to-use interface with intuitive functionalities and syntax, making it suitable for both novice and experienced users. It provides extensive documentation and examples to help users create visualisations. Plotly works on a variety of platforms, including Windows, macOS, and Linux. It is compatible with a variety of Python environments, including Jupyter Notebook, Spyder, and Python scripts.

3.1.8.5. Other Libraries Used

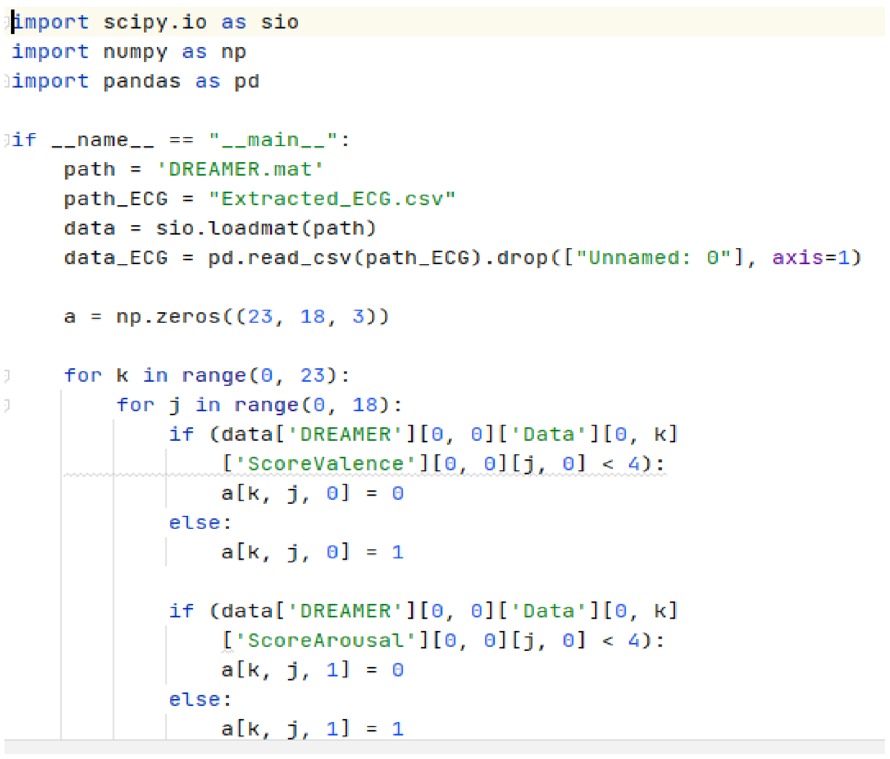
Some libraries are used that are common to both the facial expressions method and the ECG signals method. They have already been described and are only mentioned here for completeness. These libraries include Pandas, NumPy, Keras, and TensorFlow.

3.1.9. Method For Driver Alertness Detection Using ECG Signals

The model for driver alertness detection using ECG signals could only be built on our laptop by running the different algorithms and finding which algorithm gives the least mean square error.

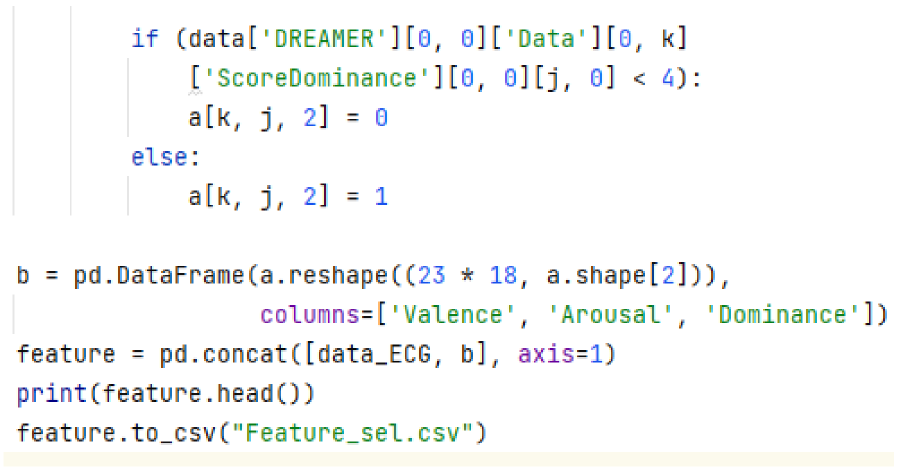
3.1.9.1. Feature Extraction

The original ECG signals from the dataset were used directly for feature extraction. As ECG signals are less susceptible to interferences due to their higher voltage amplitudes, no preprocessing was required.



```
import scipy.io as sio
import numpy as np
import pandas as pd

if __name__ == "__main__":
    path = 'DREAMER.mat'
    path_ECG = "Extracted_ECG.csv"
    data = sio.loadmat(path)
    data_ECG = pd.read_csv(path_ECG).drop(["Unnamed: 0"], axis=1)

    a = np.zeros((23, 18, 3))

    for k in range(0, 23):
        for j in range(0, 18):
            if (data['DREAMER'][0, 0]['Data'][0, k]
                ['ScoreValence'][0, 0][j, 0] < 4):
                a[k, j, 0] = 0
            else:
                a[k, j, 0] = 1

            if (data['DREAMER'][0, 0]['Data'][0, k]
                ['ScoreArousal'][0, 0][j, 0] < 4):
                a[k, j, 1] = 0
            else:
                a[k, j, 1] = 1
```

Figure 25. Code Snippet 1 for ECG Analysis Method



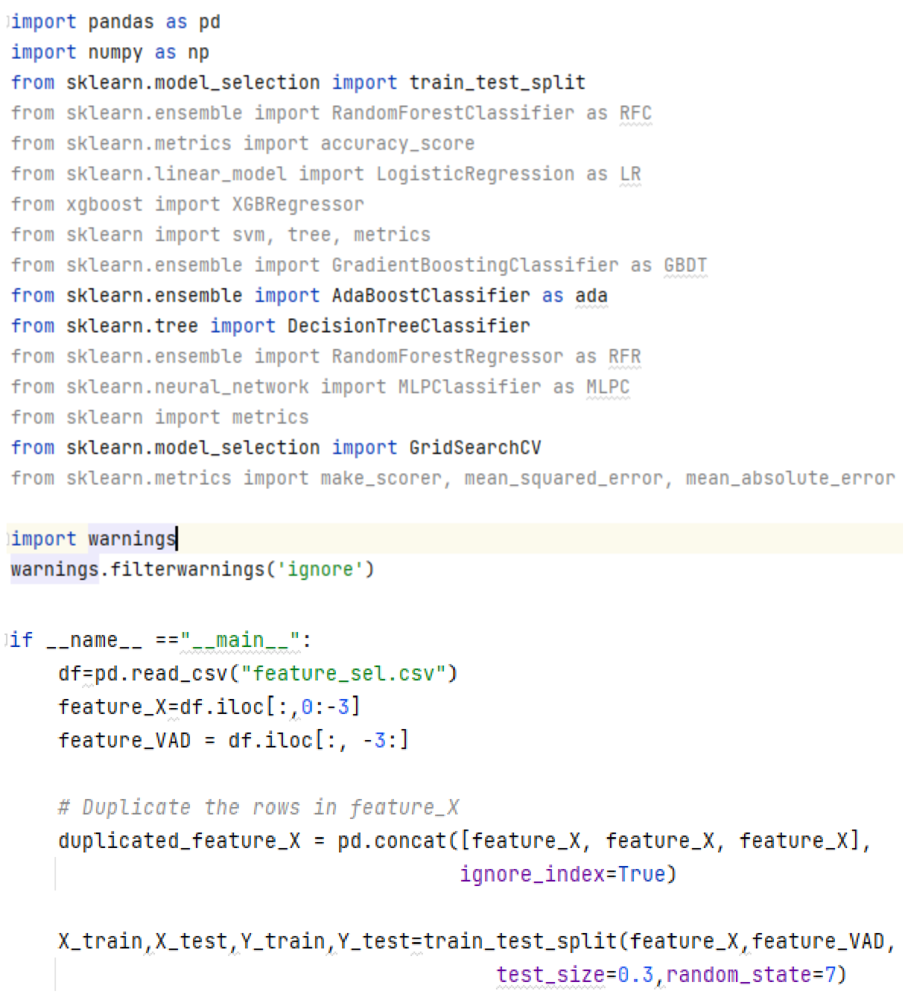
```
            if (data['DREAMER'][0, 0]['Data'][0, k]
                ['ScoreDominance'][0, 0][j, 0] < 4):
                a[k, j, 2] = 0
            else:
                a[k, j, 2] = 1

b = pd.DataFrame(a.reshape((23 * 18, a.shape[2])),
                 columns=['Valence', 'Arousal', 'Dominance'])
feature = pd.concat([data_ECG, b], axis=1)
print(feature.head())
feature.to_csv("Feature_sel.csv")
```

Figure 26. Code Snippet 2 for ECG Analysis Method

Relevant features were extracted from the raw ECG signals to capture the emotional characteristics. The selected features include rate_mean, RMSSD, MeanNN, SDNN, SDSD, CVNN, CVSD, MedianNN, MadNN, MCVNN, TINN, HTI, LF, HF, VHF, LFHF, LFn, HFn, LnHF, SD1, SD2, SD2SD1, CSI, CVI, CSI_Modified, and SampEn. These features were chosen based on their potential to represent emotional states within ECG signals.

3.1.9.2. Model Training and Hyperparameter Optimization

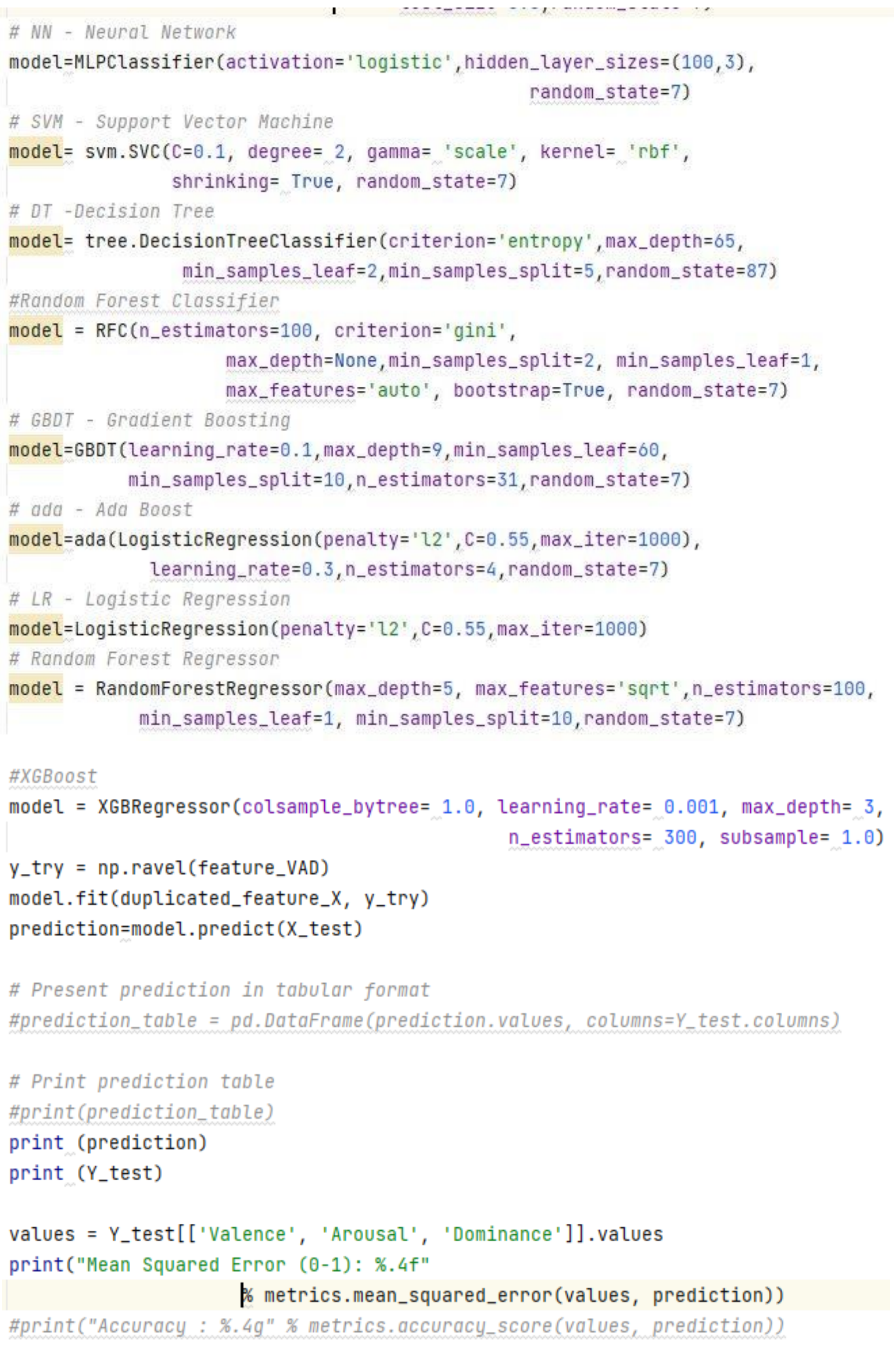
Several machine learning models were trained to predict the valence, arousal, and dominance values associated with each ECG signal. The models used in this study included Random Forest Regressor, Gradient Boosting, Neural Network, Support Vector Machines (SVM), XGBoost Regressor, AdaBoost, and Decision Tree.



```
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier as RFC
from sklearn.metrics import accuracy_score
from sklearn.linear_model import LogisticRegression as LR
from xgboost import XGBRegressor
from sklearn import svm, tree, metrics
from sklearn.ensemble import GradientBoostingClassifier as GBDT
from sklearn.ensemble import AdaBoostClassifier as ada
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestRegressor as RFR
from sklearn.neural_network import MLPClassifier as MLPC
from sklearn import metrics
from sklearn.model_selection import GridSearchCV
from sklearn.metrics import make_scorer, mean_squared_error, mean_absolute_error

import warnings
warnings.filterwarnings('ignore')

if __name__ =="__main__":
    df=pd.read_csv("feature_sel.csv")
    feature_X=df.iloc[:,0:-3]
    feature_VAD = df.iloc[:, -3:]

    # Duplicate the rows in feature_X
    duplicated_feature_X = pd.concat([feature_X, feature_X, feature_X],
                                     ignore_index=True)

    X_train,X_test,Y_train,Y_test=train_test_split(feature_X,feature_VAD,
                                                   test_size=0.3,random_state=7)
```

Figure 27. Code Snippet 3 for ECG Analysis Method

To optimize the models, a grid search approach was employed. A wide range of hyperparameter combinations were explored for each model to identify the best combination. Hyperparameters such as max_depth, max_features, min_samples_leaf, min_samples_split, n_estimators, learning_rate, activation, alpha, hidden_layer_sizes, solver, C, degree, gamma, kernel, shrinking, colsample_bytree, subsample, algorithm, and random_state were considered. Grid search allowed for the identification of the best combination of hyperparameters that yielded the best performance in terms of mean squared error (MSE).



```
# NN - Neural Network
model=MLPClassifier(activation='logistic',hidden_layer_sizes=(100,3),
                                                random_state=7)
# SVM - Support Vector Machine
model= svm.SVC(C=0.1, degree= 2, gamma= 'scale', kernel= 'rbf',
               shrinking= True, random_state=7)
# DT -Decision Tree
model= tree.DecisionTreeClassifier(criterion='entropy',max_depth=65,
                min_samples_leaf=2,min_samples_split=5,random_state=87)
#Random Forest Classifier
model = RFC(n_estimators=100, criterion='gini',
                    max_depth=None,min_samples_split=2, min_samples_leaf=1,
                    max_features='auto', bootstrap=True, random_state=7)
# GBDT - Gradient Boosting
model=GBDT(learning_rate=0.1,max_depth=9,min_samples_leaf=60,
           min_samples_split=10,n_estimators=31,random_state=7)
# ada - Ada Boost
model=ada(LogisticRegression(penalty='l2',C=0.55,max_iter=1000),
             learning_rate=0.3,n_estimators=4,random_state=7)
# LR - Logistic Regression
model=LogisticRegression(penalty='l2',C=0.55,max_iter=1000)
# Random Forest Regressor
model = RandomForestRegressor(max_depth=5, max_features='sqrt',n_estimators=100,
            min_samples_leaf=1, min_samples_split=10,random_state=7)

#XGBoost
model = XGBRegressor(colsample_bytree= 1.0, learning_rate= 0.001, max_depth= 3,
                                              n_estimators= 300, subsample= 1.0)
y_try = np.ravel(feature_VAD)
model.fit(duplicated_feature_X, y_try)
prediction=model.predict(X_test)

# Present prediction in tabular format
#prediction_table = pd.DataFrame(prediction.values, columns=Y_test.columns)

# Print prediction table
#print(prediction_table)
print (prediction)
print (Y_test)

values = Y_test[['Valence', 'Arousal', 'Dominance']].values
print("Mean Squared Error (0-1): %.4f"
                     % metrics.mean_squared_error(values, prediction))
#print("Accuracy : %.4g" % metrics.accuracy_score(values, prediction))
```

Figure 28. Code Snippet 4 for ECG Analysis Method

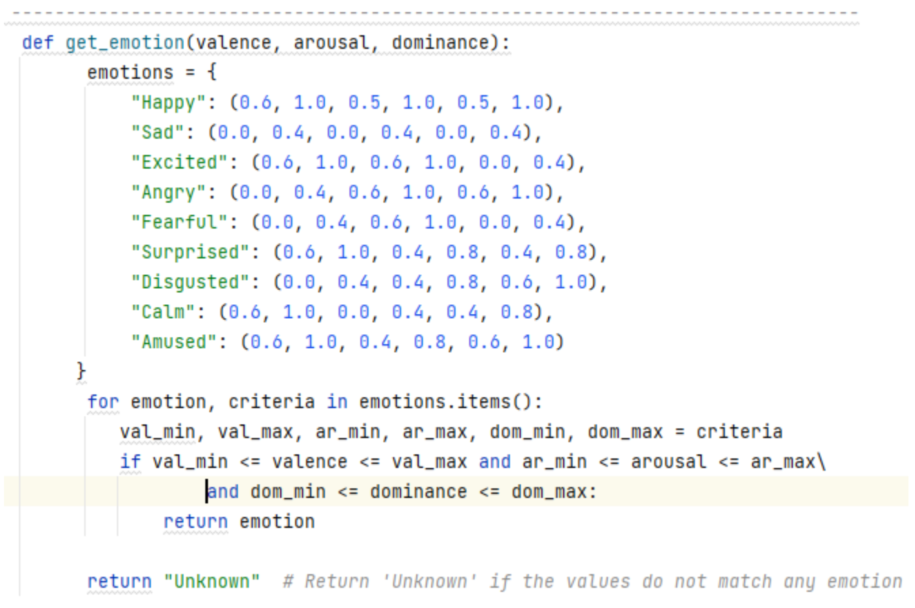
3.1.9.3. Model Evaluation and Selection

The trained models were evaluated using appropriate metrics to assess their performance in predicting the valence, arousal, and dominance values. For regression models the MSE was computed while accuracy was used as the evaluation metric for classifiers. The performance of each model was compared based on their MSE and accuracy values.

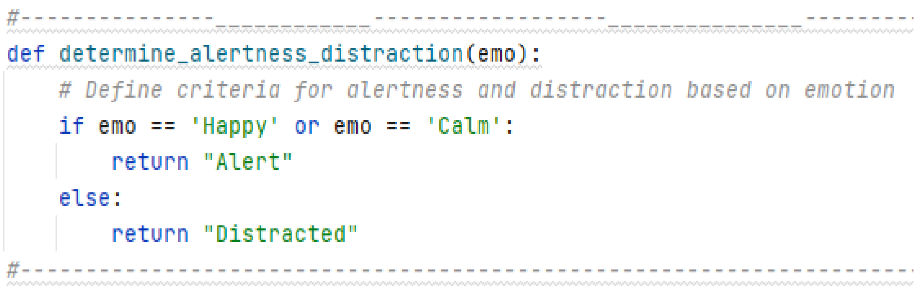
Based on the evaluation results, the model with the lowest MSE and highest accuracy was selected as the best performing model for emotion recognition using ECG analysis.

3.1.9.4. Emotion Mapping and Alertness Criteria

Using the predicted valence, arousal, and dominance values, the recognized emotions were mapped onto the valence-arousal-dominance (VAD) 3D plane. This mapping allowed for a visual representation of the emotional states associated with the ECG signals. The emotions derived from the VAD values were used to classify the state of a person being either alert or distracted.



```python
def get_emotion(valence, arousal, dominance):
        emotions = {
            "Happy": (0.6, 1.0, 0.5, 1.0, 0.5, 1.0),
            "Sad": (0.0, 0.4, 0.0, 0.4, 0.0, 0.4),
            "Excited": (0.6, 1.0, 0.6, 1.0, 0.0, 0.4),
            "Angry": (0.0, 0.4, 0.6, 1.0, 0.6, 1.0),
            "Fearful": (0.0, 0.4, 0.6, 1.0, 0.0, 0.4),
            "Surprised": (0.6, 1.0, 0.4, 0.8, 0.4, 0.8),
            "Disgusted": (0.0, 0.4, 0.4, 0.8, 0.6, 1.0),
            "Calm": (0.6, 1.0, 0.0, 0.4, 0.4, 0.8),
            "Amused": (0.6, 1.0, 0.4, 0.8, 0.6, 1.0)
        }
        for emotion, criteria in emotions.items():
            val_min, val_max, ar_min, ar_max, dom_min, dom_max = criteria
            if val_min <= valence <= val_max and ar_min <= arousal <= ar_max\
                    and dom_min <= dominance <= dom_max:
                return emotion

        return "Unknown"  # Return 'Unknown' if the values do not match any emotion
```



```python
#---------------------------------------------------------------------
def determine_alertness_distraction(emo):
    # Define criteria for alertness and distraction based on emotion
    if emo == 'Happy' or emo == 'Calm':
        return "Alert"
    else:
        return "Distracted"
#---------------------------------------------------------------------
```

Figure 29. Code Snippet 5 for ECG Analysis Method

3.1.7. Regression Algorithms Used for Emotion Recognition Using ECG Signals

Different regression algorithms have been tested on the dataset to determine the model that gives the best performance. Seven algorithms have been used that include random forest regressor, gradient boosting, neural network, support vector machine, XGBoost regressor, AdaBoost, and decision tree. The mean square error for each algorithm was determined.

3.1.7.1. Random Forest Regressor

The power of decision trees and ensemble learning are combined in the well-known machine learning method known as Random Forest. During the training phase, a large number of decision trees are built. A random subset of the training data is used to build each decision tree, and a random subset of features are taken into account for splitting at each node. The random forest makes a final determination during prediction by combining the forecasts of all individual trees. This ensemble strategy aids in lowering overfitting and enhancing generalization efficiency. High-dimensional data handling, handling missing values, and providing measurements of feature relevance are all strengths of Random Forest. Due to its dependability, adaptability, and capacity for handling complicated datasets, it is frequently employed for classification and regression problems.

3.1.7.2. Gradient Boosting

An effective machine learning approach called gradient boosting combines the forecasting skills of several weak learners to produce a robust prediction model. It functions by iteratively adding weak models to the ensemble, often decision trees, while concentrating on the errors generated by the prior models. The algorithm calculates the gradient of the loss function with respect to the expected values in each iteration and modifies the parameters of the new model to minimize this gradient. The method continuously increases its forecast accuracy by periodically refreshing the model to remove errors. Combining the predictions of each weak model in the ensemble yields the final forecast.

3.1.7.3. Neural Network

A computational model called a neural network is modeled after the structure and operation of the human brain. It is made up of linked neurons arranged in layers. Each neuron receives inputs, runs a calculation, and then outputs the results. Through a process known as training, neural networks are made to discover and detect patterns or relationships in data. Based on the supplied input data and desired outputs, the network modifies the strengths of connections (known as weights) between neurons during training. The network may learn complicated representations and make predictions or classifications on unobserved input thanks to this optimization process.

3.1.7.4. Support Vector Machines

The supervised machine learning algorithm support vector machines (SVM) is frequently employed for classification and regression tasks. Finding the hyperplane(s) that maximize the margin, or the distance between the hyperplane and the closest data points of various classes, is the objective of SVM. SVM is able to generalize to unknown data well thanks to margin maximization. By utilizing a kernel function, which implicitly translates the input to a higher-dimensional space, making it separable, SVM is able to handle both linearly separable and non-linearly separable data. SVM is renowned for its capacity to manage high-dimensional data, efficiently handle outliers, and generalize even with a small number of training instances.

3.1.7.5. XGBoost Regressor

A robust machine learning algorithm that is a member of the ensemble learning family is the XGBoost (Extreme Gradient Boosting) regressor. It produces high-performance regression models by combining the characteristics of the gradient boosting and decision tree algorithms. By adding new models that correct the faults of the older ones, XGBoost develops an ensemble of weak predictive models—typically decision trees—and iteratively trains them to minimize the loss function. It continuously improves the model's predictions by focusing more on samples that were incorrectly categorized throughout the training process. In order to manage model complexity, reduce overfitting, and improve generalization, XGBoost uses a regularization strategy. It also makes use of a gradient-based optimization technique, which increases its computing effectiveness and capacity for handling huge datasets.

3.1.7.6. AdaBoost

A well-known machine learning algorithm called AdaBoost (Adaptive Boosting) combines the predictions of several weak classifiers to produce a strong classifier. It operates by repeatedly training weak classifiers on various weighted iterations of the training set. The method emphasizes the challenging examples by giving bigger weights to the instances that were incorrectly identified in the previous iteration. Simple models, such as decision stumps (small decision trees with a single decision node), are frequently used as the weak classifiers. Weighted voting is used to merge the weak classifiers after all iterations with the weights being dependent on how well each classifier performs individually.

3.1.7.7. Decision Tree

A decision tree is a machine learning approach for classification and regression tasks. It has a structure like a flowchart, with each internal node signifying a test on a specific feature, each branch signifying the test's result, and each leaf node signifying a class label or anticipated value. Recursively splitting the data based on the features is how the decision tree algorithm learns, with each step seeking to decrease impurity or maximize information gain. During training, the algorithm selects the most useful characteristic to divide the data into branches that capture various decision-making processes. By moving up the tree from the root node to a leaf node and then following the necessary branches based on the test outcomes, the resulting tree can be utilized to generate predictions.

3.2. Hardware Components

Our hardware components consist of a Raspberry Pi 3 Model B, Raspberry Pi Camera Rev 1.3 and a 7-inch Raspberry Pi Touchscreen.

3.2.1. Raspberry Pi Module

The Raspberry Pi 3 Model B is a single-board computer developed by the Raspberry Pi Foundation. It is a lightweight and portable system that can be used for a wide range of projects. It has a quad-core, 64-bit, 1.2 GHz ARM Cortex-A53 processor, which offers better performance than earlier models. The Model 3 B contains 1 GB of RAM, making it possible to run numerous applications and provide multitasking capability. Although it lacks built-in memory, it does contain a microSD card slot for storage. The operating system can be installed and data can be kept on a microSD card.

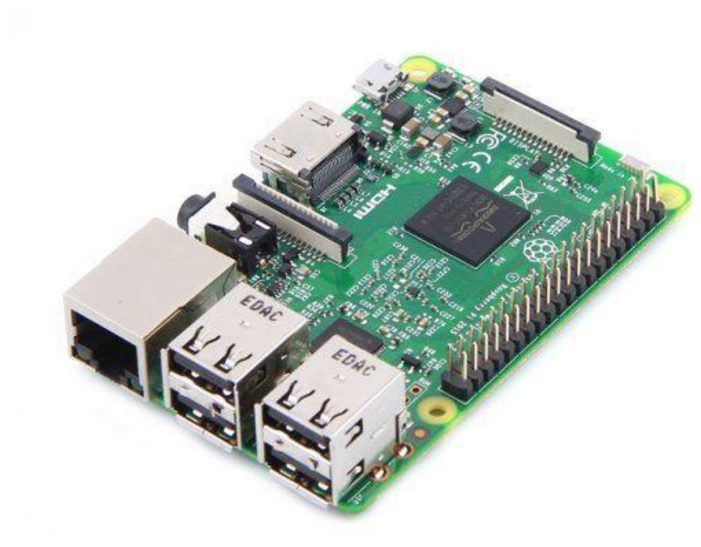


Figure 30. Raspberry Pi 3 Model B

For connectivity purposes, there are wired and wireless connectivity modes supported by the board. It has four USB 2.0 connections, an Ethernet port for wired internet access, as well as integrated Bluetooth 4.2 and Wi-Fi 802.11n for wireless communication. The Model 3 B supports video output up to 1080p and has an HDMI connector for connecting to screens. Additionally, it has a 3.5 mm audio connector for speakers or headphones. The board also contains general purpose input/output (GPIO) pins. A 40-pin GPIO header is available, enabling interaction with a variety of sensors, actuators, and other external devices. A 5 V power supply is needed to power the board, which can be done using a micro-USB connector. The Raspberry Pi Model 3 B is compatible with a variety of operating systems, including Linux-based variants like Raspbian and Ubuntu and other specialised OS alternatives.

3.2.2. Raspberry Pi Camera

The Raspberry Pi Camera Module Rev. 1.3 is a small camera made especially for Raspberry Pi boards. An OmniVision OV5647 image sensor is used by the camera module. It has a 5-megapixel sensor that can record both still photos and video. The camera module has a 2592 x 1944 pixel maximum still image resolution. It offers a range of video capture resolutions, including VGA (640 x 480) at 90 frames per second, 720p at 60 frames per second, and 1080p at 30 frames per second (fps). The module comes with a fixed-focus lens. It features a wide-angle field of vision of about 53 degrees and a focal length of about 3.6 mm. A flat ribbon cable is used to link the camera module to the Raspberry Pi board. It connects to the Raspberry Pi board’s special CSI (Camera Serial Interface) port.

Figure 31. Raspberry Pi Camera Rev 1.3

The camera module and Raspberry Pi board communicate with one another via the CSI-2 (Camera Serial Interface 2) protocol. High-speed and effective data communication between the camera module and the board is made possible by this interface. The camera module is equipped with a number of features, such as automated white balance, automatic exposure control, and automatic image quality control. During video recording, it also offers still image capture in addition to video stabilisation. The official Raspberry Pi camera software, which includes camera drivers and utilities, supports the camera module. Additionally, it effortlessly interfaces with well-known software platforms and libraries, including the Picamera Python library. The Raspberry Pi Camera Rev. 1.3 is used in a variety of projects related to computer vision, robotics, surveillance systems, photography, and video recording.

3.2.3. Raspberry Pi Touchscreen

The Raspberry Pi 7-inch Touchscreen is a display component made especially for Raspberry Pi boards. The touchscreen has a 7-inch diagonal TFT LCD display with an 800 x 480 pixel resolution that provides a clear and bright visual output. The capacitive touch panel used in the display allows for multi-touch input. It supports up to 10-point touch, making swipe and pinch-to-zoom operations possible. The touchscreen can be connected to the Raspberry Pi board via a ribbon wire that fits into the board’s dedicated display connector

(DSI). This link offers both power and data transmission between the board and the display. The touchscreen module has mounting holes and an integrated stand that make it simple to affix it to the Raspberry Pi board. It can be utilised as an independent display or integrated into unique projects or enclosures.

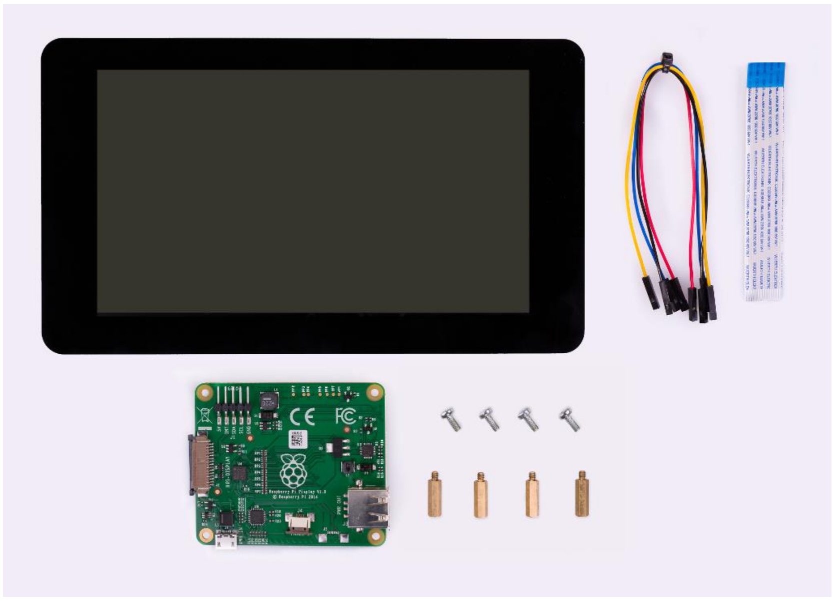


Figure 32. Raspberry Pi 7” Touchscreen Display Kit

A separate power source is needed for the touchscreen module. It can be powered by an external power source or by a micro-USB cable attached to the Raspberry Pi board. The Raspberry Pi 7-inch touchscreen is compatible with Raspberry Pi boards of all generations, including the 2, 3, and 4 models. It is supported by a number of third-party programs and libraries in addition to the original Raspberry Pi operating system. Additional capabilities offered by the touchscreen module include an on-screen keyboard, adjustable backlight brightness, and an automatic power-saving mode when not in use. Additionally, an on-screen display (OSD) menu for changing display options is supported. There are numerous uses for the touchscreen. For media centres, smart home control panels, gaming consoles, industrial automation, and other devices, it functions as an interactive display. Projects that call for user involvement benefit greatly from its small size and touch capabilities.

3.2.4. Complete Hardware Setup

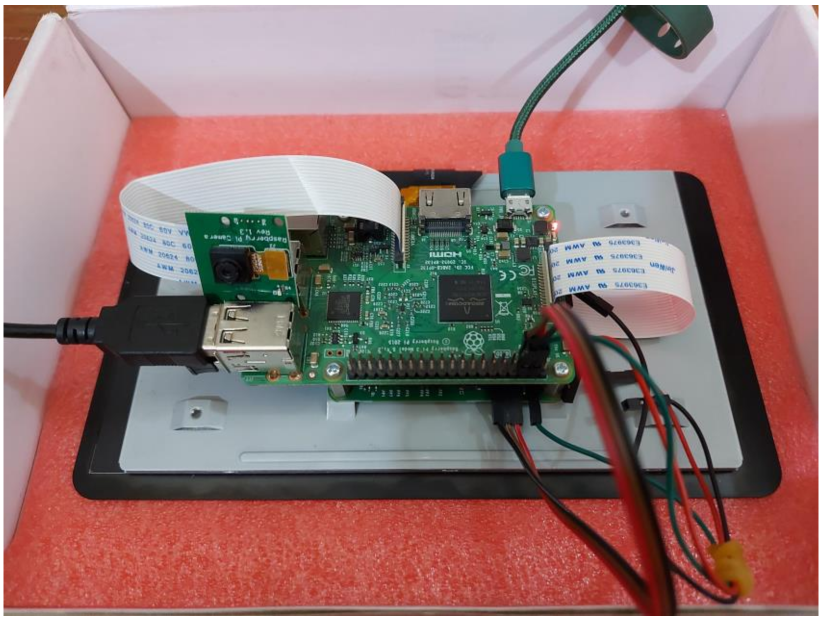


Figure 33. Raspberry Pi Hardware Setup

For the complete hardware setup, we have connected the Raspberry Pi board with the Raspberry Pi touchscreen. The Raspberry Pi camera is attached to the board using the CSI port. The board and touchscreen are powered using the micro USB port. For power transmission to the touchscreen, two GPIO pins are utilized. The black wire is connected to the GND pin and the green wire is connected to the 5V pin on the touchscreen board. For communication between the touchscreen and the Raspberry Pi module, further two pins are used. The brown wire is connected to the SCL (Serial Clock Line) pin and the red wire is connected to the SDA (Serial Data Line) pin on the touchscreen board. Additionally, a mouse and keyboard have been connected to the Raspberry Pi using the USB ports provided on the board.

The operating system that we have used for the Raspberry Pi is Raspberry Pi OS 32-bit Buster. The 32-bit version of Raspberry Pi OS, based on Debian Buster, is designed to run on the ARM architecture used by the Raspberry Pi boards. Raspberry Pi OS 32-bit Buster includes the PIXEL desktop environment. PIXEL (Pi Improved Xwindows Environment, Lightweight) provides a user-friendly graphical interface for easy navigation and interaction with the system.

For operating system installation, we required a laptop and a microSD card connected to the laptop. We first downloaded Raspberry Pi Imager on our laptop from the Raspberry Pi’s official website. This application contained different operating systems that were available to flash onto the microSD card. We selected our required OS and flashed it onto our microSD card. The microSD card was then inserted into the Raspberry Pi board’s microSD card slot and the board was powered on. After the Raspberry Pi booted up, we were able to configure the settings for our Raspberry Pi module. Our project code was written on our laptop. We copied it onto a USB drive and connected it to the Raspberry Pi’s USB port. The code was transferred onto the Raspberry Pi’s microSD card.

The facial and emotion algorithm runs on Thonny IDE that is installed on the Raspberry Pi. The Raspberry Pi camera takes the picture or video input of the person in front of it. This input is given to the DeepFace model that analyzes the picture or video to determine the possible emotion. Based on the emotion detected, the driver alert state is checked. The detected emotion and alert state are displayed on the Raspberry Pi touchscreen which allows the user to observe the results.

Chapter 4 – RESULTS

4.1. Results for Driver Emotion Recognition Using Facial Expressions

We tested our implementation for still images as well as real-time videos. It works equally well for both types of inputs. This section displays some of the results of our model for driver emotion recognition using facial expressions.

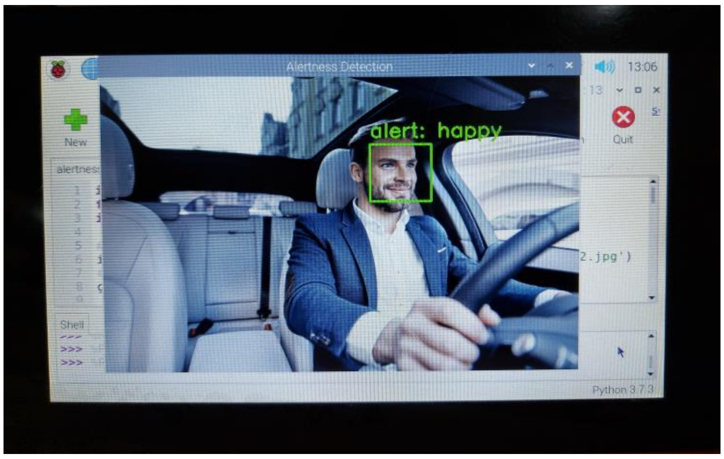


Figure 34. Detection of Driver Alertness in Still Image

In Figure 34 it can be observed that our model displays the emotion of the driver as happy and state as alert for a still image.

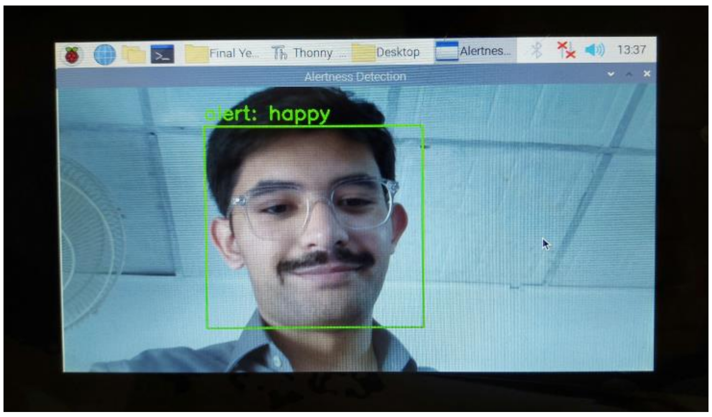


Figure 35. Alert and Happy Detected on Real-time Video

In Figure 35 it can be seen that in a real-time video, the emotion is detected as happy and state as alert.

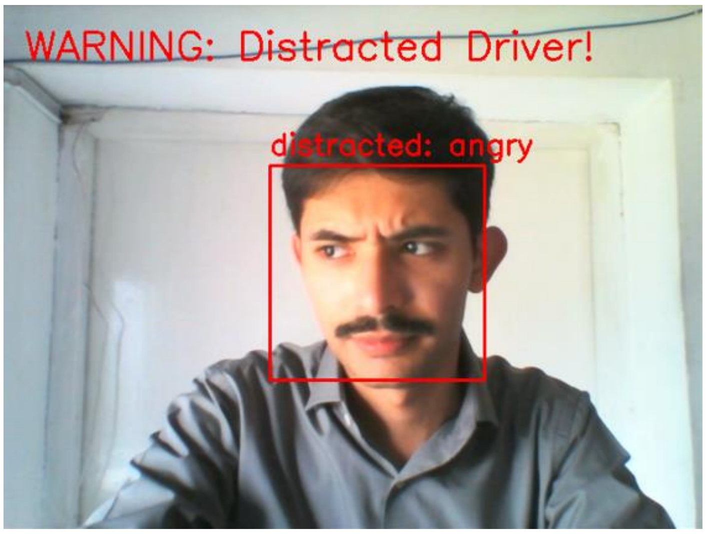


Figure 36. Distracted and Angry Detected on Real-time Video

The image in Figure 36 is from a real-time video. The emotion is detected as angry and state as distracted.

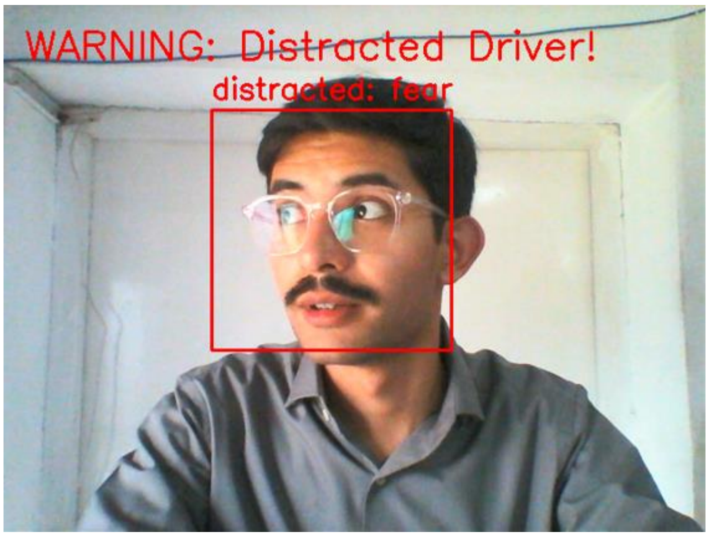


Figure 37. Distracted and Fear Detected on Real-time Video

The image in Figure 37 is from a real-time video. The emotion is detected as fear and state as distracted.

4.2. Results for Driver Emotion Recognition Using ECG Signals

4.2.1. MSE Results for Models

Random Forest Regressor: Mean Squared Error (MSE) is 25.3

Gradient Boosting Mean Squared Error (MSE) is 56.28

Neural Network: Mean Squared Error (MSE) is 43.2

SVM: Mean Squared Error (MSE) is 56.8

XGBoost Regressor: Mean Squared Error (MSE) is 24.94

AdaBoost: Mean Squared Error (MSE) is 37.2

Decision Tree: Mean Squared Error (MSE) is 44.27

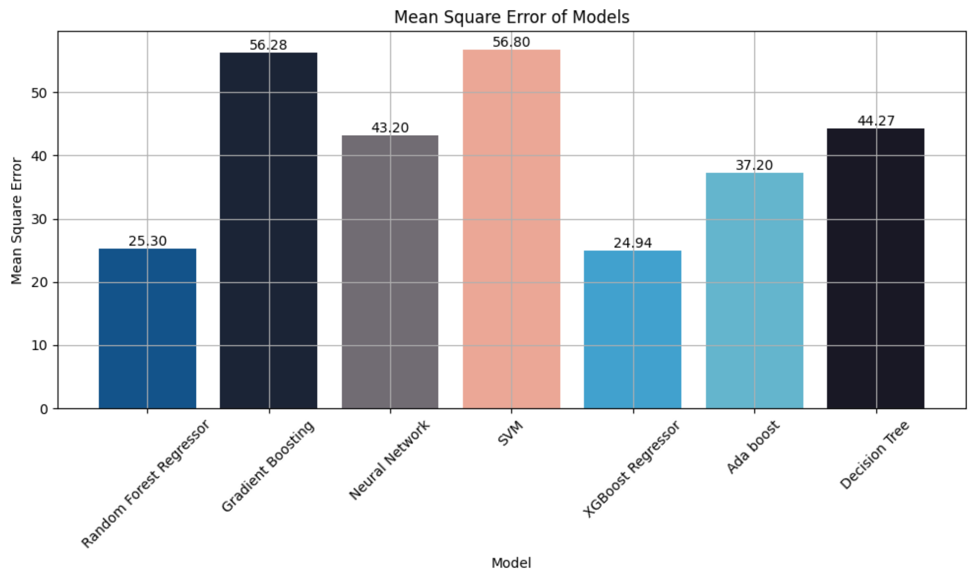


Figure 38. MSE Results of Different Models

The XGBoost Regressor achieved the lowest MSE of 24.94. XGBoost is known for its gradient boosting technique, which can handle complex relationships in the data effectively and make more accurate predictions.

The Random Forest Regressor also performed well with an MSE of 25.3, indicating good performance compared to other models. This model may have effectively captured the underlying patterns in the data.

The Neural Network model achieved an MSE of 43.2, which is relatively higher than the Random Forest and XGBoost models. This may indicate that the neural network architecture or hyperparameters need further tuning to improve its performance.

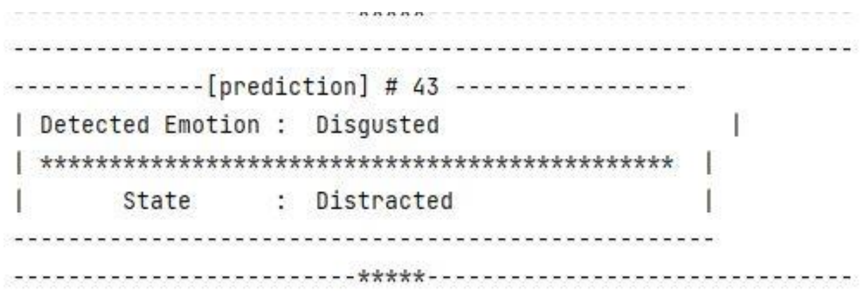
The AdaBoost model had an MSE of 37.2, indicating moderate performance. Adjusting the learning rate or exploring different boosting algorithms could potentially enhance its predictions.

The Decision Tree model achieved an MSE of 44.27, which is higher than the Random Forest and XGBoost models. This suggests that ensemble methods like Random Forest and XGBoost outperform a single decision tree in capturing the complexity of the data.

The Gradient Boosting and SVM models obtained higher MSE values of 56.28 and 56.8, respectively. This indicates that these models may not have captured the underlying patterns in the data as effectively as the Random Forest and XGBoost models.

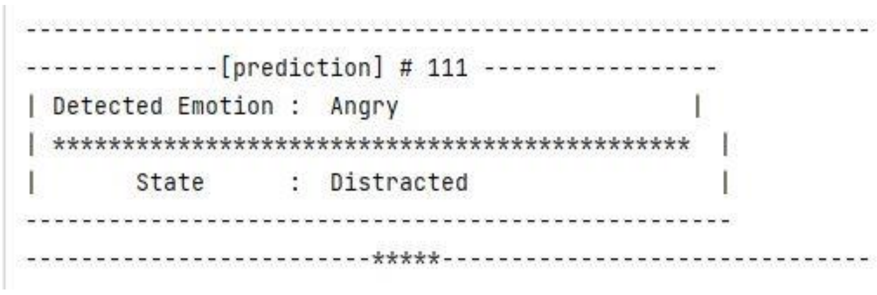
4.2.2. Model Predictions

The following are some of the predictions the model gives on the examples of the dataset. Most of the time the model detects distraction.



```
-------------------------------------------------------------
--------------[prediction] # 43 -----------------
| Detected Emotion :  Disgusted                  |
| *********************************************  |
|       State      :  Distracted                 |
---------------------------------------------------
------------------------*****------------------------------
```

Figure 39. ECG Model Prediction 1



```
---------------------------------------------------------------
--------------[prediction] # 111 -----------------
| Detected Emotion :  Angry                      |
| **********************************************  |
|       State      :  Distracted                  |
---------------------------------------------------
-------------------------*****-------------------------------
```

Figure 40. ECG Model Prediction 2

4.2.3. Confusion Matrix For Decision Tree Classifier

The confusion matrix was plotted for the decision tree classifier. By using the confusion matrix values for Valence, Arousal, and Dominance, we can analyze the performance of the model and draw insights regarding its accuracy and potential areas for improvement.

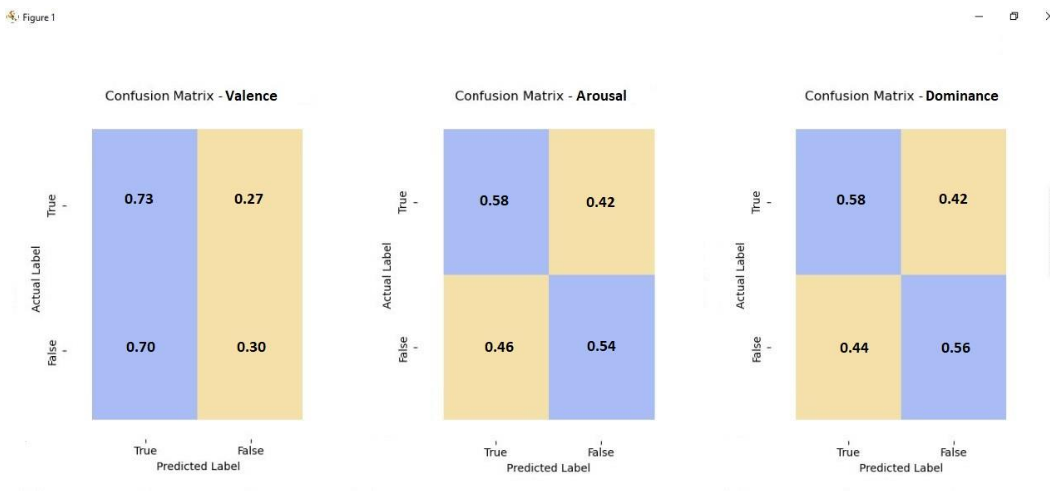


Figure 41. Confusion Matrix For Decision Tree Classifier

For the valence predictions, the model shows a relatively higher accuracy in predicting positive valence (0.73) compared to negative valence (0.30). However, it has a higher tendency to misclassify negative Valence, as indicated by the higher FN rate (0.70). To improve the model's performance, addressing the false negatives and reducing the misclassification of negative valence would be beneficial.

For the arousal predictions, the model shows a relatively balanced accuracy for both positive arousal (0.58) and negative arousal (0.54). However, there is still room for improvement in reducing the false positives (0.42) and false

negatives (0.46) to enhance the model's performance.
For the dominance predictions, the model exhibits a similar pattern in predicting positive dominance (0.58) and negative dominance (0.55), indicating a relatively balanced accuracy. However, there is still room for improvement in reducing the false positives (0.42) and false negatives (0.44) to enhance the model's overall performance.

In general, the model performs reasonably well in predicting positive emotions across all three dimensions (valence, arousal, and dominance). However, it faces challenges in accurately predicting negative emotions, as evident from the higher false negative rates. This could be due to imbalances in the dataset or the model's sensitivity to certain features.

Chapter 5 – CONCLUSION AND FUTURE WORK

5.1. Conclusion

In this project, we explored the detection of emotions for a driver using two separate models—one for facial expressions and another for ECG signals. The emotion detection model for facial expressions was pre-trained. The emotion detection model using ECG signals was trained and evaluated independently to identify the emotional states accurately. Alertness criteria were set up for determining the state of alertness of a driver. The results obtained from each model were promising, demonstrating the potential of utilizing facial expressions and ECG signals as reliable indicators of emotions.

The facial expression model successfully analyzed facial features and patterns to recognize various emotional states. By leveraging computer vision techniques and deep learning algorithms, it achieved high accuracy in emotion classification based on facial cues. This model can be particularly useful in scenarios where visual data is readily available, such as video analysis or real-time emotion recognition applications.

On the other hand, the ECG-based emotion detection model focused on analyzing the electrical activity of the heart to infer emotional states. By extracting relevant features from the ECG signals and employing machine learning algorithms, this model demonstrated its capability to discern different emotional responses accurately. This approach can be valuable in situations where direct access to facial expressions is limited or not feasible, such as remote monitoring or healthcare settings.

5.2. Future Work Suggestions

While the results obtained from the individual models are encouraging, there are several areas that could be explored in future work to enhance the overall performance and utility of the emotion detection system.

5.2.1. Combining the Models

One possible direction for future work is to investigate the potential benefits of combining the facial expression and ECG-based models. By leveraging the complementary nature of these modalities, a fusion approach may yield improved emotion recognition results. This could involve designing a fusion architecture that integrates the outputs of both models or exploring multimodal deep learning techniques to jointly analyze facial expressions and ECG signals.

5.2.2. Dataset Diversity and Size

To further enhance the robustness and generalizability of the emotion detection system, it is essential to expand the training dataset used for ECG signals in terms of diversity and size. Including data from different demographics, cultures, and contexts would help reduce bias and increase the model's ability to recognize emotions across a broader population. Additionally, augmenting the dataset with synthetic data or using transfer learning approaches from related tasks could mitigate data scarcity issues.

5.2.3. Real-Time Implementation

While the models developed in this project provide accurate emotion detection, their real-time implementation is an important consideration. Future work could focus on optimizing the models for faster inference and ensuring their efficiency in real-time applications. Techniques such as model compression, quantization, or hardware acceleration could be explored to enable the deployment of the emotion detection system on resource-constrained devices or in time-critical scenarios.

5.2.4. Ethical and Privacy Considerations

As emotion detection technologies become more prevalent, it is crucial to address ethical concerns and privacy implications. Future work should pay attention to ensuring transparency, fairness, and accountability in the design and deployment of these models. Ethical frameworks and guidelines should be developed to govern the collection, storage, and usage of sensitive emotional data, emphasizing user consent, data anonymization, and secure storage practices.

REFERENCES

[1] Y. Taigman, M. Yang, M. Ranzato and L. Wolf, "DeepFace: Closing the Gap to Human-Level Performance in Face Verification," 2014 IEEE Conference on Computer Vision and Pattern Recognition, Columbus, OH, USA, 2014, pp. 1701-1708, doi: 10.1109/CVPR.2014.220.

[2] T. Ahonen, A. Hadid, and M. Pietik¨ainen. Face description with local binary patterns: Application to face recognition. PAMI, 2006. 3, 5

[3] A. Krizhevsky, I. Sutskever, and G. Hinton. ImageNet classification with deep convolutional neural networks. In ANIPS, 2012. 1, 2, 3, 4

[4] Y. Sun, X. Wang, and X. Tang. Deep convolutional network cascade for facial point detection. In CVPR, 2013. 2

[5] D. Yi, Z. Lei, and S. Z. Li. Towards pose robust face recognition. In CVPR, 2013. 2

[6] (2023) Raspberry Pi Touchscreen Setup. [Online].

Available: https://www.instructables.com/Raspberry-Pi-Touchscreen-Setup/

[7] Katsigiannis, S., & Ramzan, N. (2018). DREAMER: A Database for Emotion Recognition Through EEG and ECG Signals From Wireless Low-cost Off-the-Shelf Devices. IEEE Journal of Biomedical and Health Informatics, 22, 98-107.

[8] Sun, B., & Lin, Z. (2022). Emotion Recognition using Machine Learning and ECG signals.

[9] Nita, S., Bitam, S., Heidet, M., & Mellouk, A. (2022). A new data augmentation convolutional neural network for human emotion recognition based on ECG signals. Biomed. Signal Process. Control., 75, 103580.

[10] Mao, A., Du, Z., Lu, D., & Luo, J. (2022). Attention emotion recognition via ECG signals. Quant. Biol., 10, 276.

[11] Xu, Y., Liu, G., Hao, M., Wen, W., & Huang, X. (2010). Analysis of affective ECG signals toward emotion recognition. Journal of Electronics (China), 27, 8-14.

[12] Sharma, M., & Mathew, R.R. (2020). Emotion Recognition Using Physiological Signals. Proceeding of the International Conference on Computer Networks, Big Data and IoT (ICCBI - 2019).

[13] Romeo, L., Cavallo, A., Pepa, L., Berthouze, N., & Pontil, M. (2022). Multiple Instance Learning for Emotion Recognition Using Physiological Signals. IEEE Transactions on Affective Computing, 13, 389-407.

[14] Shu, L., Xie, J., Yang, M., Li, Z., Li, Z., Liao, D., Xu, X., & Yang, X. (2018). A Review of Emotion Recognition Using Physiological Signals. Sensors (Basel, Switzerland), 18.