IJCRT.ORG ISSN : 2320-2882

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)
An International Open Access, Peer-reviewed, Refereed Journal

A MULTIMODAL APPROACH TO PREDICTING STRESS LEVELS THROUGH PSYCHOLOGICAL PROFILING AND FACIAL RECOGNITION

[1]Mrs.Kalaivani V, [2]Sanjeev Kumar K S, [3]Rithesh Roshan R, [4]Raguraman K
[1]Assistant Professor, [2]Student, [3]Student, [4]Student
[1]Department of Computer Science and Engineering,
Adhiyamaan College of Engineering, Hosur, India.

Abstract: A novel multimodal stress prediction system combines psychological questioning to assess emotional states and facial recognition technology to analyze facial expressions. By integrating these approaches, the system accurately determines stress levels in users. Through a series of tailored psychological questions and real-time facial emotion recognition, the system provides a comprehensive assessment of the user's stress levels using machine learning techniques. The culmination of this process yields a precise percentage indicating the individual's stress level, enabling timely interventions and personalized stress management strategies. This innovative project aims to revolutionize stress monitoring by leveraging both psychological profiling and facial recognition technologies.

***Index Terms* -** psychological profiling, facial recognition, machine learning, stress levels.

I. INTRODUCTION

In the contemporary fast-paced world, stress has become an unavoidable facet of daily life, affecting individuals across diverse demographics. The repercussions of chronic stress are far-reaching, influencing physical health, mental well-being, and overall productivity. As society increasingly recognizes stress as a significant public health concern, there emerges a critical need for innovative approaches to assess and manage stress effectively. This research project seeks to address this imperative through a groundbreaking multimodal approach, integrating psychological profiling and facial recognition, empowered by Python, artificial intelligence (AI), and machine learning (ML) techniques. The intricacies of stress are multifaceted, influenced by individual experiences, environmental stimuli, and coping mechanisms. Conventional stress assessment methods often rely on subjective self-reporting, susceptible to biases. This project acknowledges the complexity of stress and aims to enhance prediction accuracy by considering both explicit and implicit indicators. Explicit indicators, encompassing observable behaviors and self-reported feelings, are complemented by implicit indicators like facial expressions and physiological responses. By combining these dimensions, the project strives to offer a holistic understanding of stress, enabling a personalized approach to stress management. The role of psychological profiling is pivotal, involving the analysis of behavioral patterns and responses. Through machine learning algorithms, the system learns to recognize unique psychological profiles, facilitating a more accurate assessment of stress levels. The adaptability of the system to individual experiences and coping mechanisms over time enhances precision and fosters continuous improvement.

Facial recognition serves as an implicit indicator, leveraging advanced computer vision techniques to analyze facial expressions associated with stress. Trained on a diverse dataset, the system identifies subtle cues indicative of stress levels. This adds objectivity to stress assessment, providing an independent measure of emotional states and enabling real-time monitoring. The seamless integration of Python, AI, and ML technologies forms the core of this project's implementation. Python's versatility serves as a robust foundation for developing and deploying system components, ensuring efficient data processing and integration. The AI and ML components play a pivotal role, recognizing patterns in psychological profiles and facial expressions, with continuous learning ensuring adaptation over time. The significance of this research extends to mental health and well-being, offering a personalized approach to stress prediction that empowers individuals for proactive management. Beyond personal well-being, the applications span workplace initiatives, educational settings, and public health campaigns. In workplaces, integration into employee well-being programs provides insights for stress management interventions, fostering a healthier work environment. In educational settings, addressing stress factors contributes to student well-being and academic success. At a broader level, public health campaigns can leverage insights for targeted stress reduction interventions in communities.

II. LITERATURE SURVEY

Nonverbal communication plays a significant role in daily interactions, contributing up to 93% to overall communication. Facial emotion analysis is crucial for various applications such as video surveillance, expression analysis, and detecting paralinguistic communication. In our proposed system, we provide a comprehensive overview of Facial Emotion Recognition (FER) based on traditional machine learning (ML) techniques. Despite its importance, detecting learning problems remains laborious and time-consuming, necessitating further research to simplify the process. Dyscalculia, a learning disability affecting 3 to 6% of school-aged children, is characterized by difficulties in counting, comparing numbers, and performing mathematical operations. To identify dyscalculia, a combination of tests must be administered and manually analyzed. In healthcare, artificial intelligence (AI) utilizes Random Forest algorithms to analyze complex medical data, including screening for learning problems. Our model for identifying dyscalculia utilizes machine learning techniques and inputs such as counting accuracy, time spent per question, number comparison accuracy, and arithmetic addition accuracy. The Random Forest method is employed to categorize children as dyscalculic or not based on these inputs.[1]

An application developed by integrating Flutter and machine learning aims to predict and manage stress among today's population. Its primary function is to detect and address the stress and pressure individuals experience by identifying subtle environmental stimuli that often go unnoticed. Leveraging machine capabilities, the program can make instant decisions to assist users. Researchers frequently utilize internet data to inform their decisions, and similarly, this application employs image recognition techniques and camera data to determine a person's stress levels. It captures a picture using the camera, processes it, and displays the result on the screen. Additionally, the application features a dedicated section where various stress management methods are suggested.[2]

Maintaining sound mental health is crucial for overall well-being, impacting various psychological and physiological functions. Factors such as perceived stress levels and the quality of nighttime sleep significantly contribute to a student's well-being and may be influenced by external factors over time. This study aims to explore the relationship between Perceived Stress Scale (PSS) scores and Pittsburgh Sleep Quality Index (PSQI) global scores using data from the publicly available StudentLife dataset. The objective is to classify well-being into categories of 'Good,' 'Average,' and 'Bad.' A linear regression model effectively illustrates the association between PSS and PSQI scores. Machine learning techniques, including Decision Trees (DT), Support Vector Machine (SVM), and K-nearest neighbors (K-NN), are applied to both pre-test and post-test questionnaire data. SVM demonstrates higher accuracy for pre-test data, while K-NN achieves the best accuracy for post-test data. Performance evaluation utilizes metrics such as accuracy, precision, recall, and F1 score.[3]

The prevalence of mental health issues is anticipated to escalate in the coming years, emerging as a hidden pandemic. The global lockdown during the COVID-19 pandemic led to a significant surge in cases of depression, anxiety, and stress, highlighting the urgency of exploring the interdisciplinary field of Artificial Intelligence and Psychometry. This paper proposes a comparison of various machine learning and ensemble learning methods using a survey dataset comprising DASS-42 Psychometric Test Results and demographic information. Random Forest, Decision Tree, Support Vector Machine (SVM), AdaBoost, CatBoost, and Extreme Gradient Boosting (XGBoost) are employed to classify levels of depression, anxiety, and stress into categories of normal, mild, moderate, severe, and extremely severe. In our experiments, Support Vector

Machine outperformed other methods, achieving final F1-measures of 94%, 95%, and 91% for the prediction of depression, anxiety, and stress, respectively.[4]

Stress disorders are prevalent issues among IT professionals in today's industry. The evolving lifestyle and work cultures contribute to an increased risk of stress among employees. Despite efforts by many industries and corporations to address mental health concerns and improve workplace atmosphere, the problem remains challenging to control. In this study, we aim to utilize machine learning techniques to analyze stress patterns in working adults and identify key factors that strongly influence stress levels. To achieve this, we utilized data from the OSMI mental health survey 2017, focusing on responses from working professionals within the tech industry. After thorough data cleaning and preprocessing, various machine learning techniques were applied to train our model. The accuracy of these models was compared, with boosting demonstrating the highest accuracy among them. Using Decision Trees, we pinpointed significant features influencing stress levels, including gender, family history, and the availability of health benefits in the workplace. These findings enable industries to refine their strategies for stress reduction and foster a more comfortable workplace environment for their employees.[5]

Today, children worldwide face significant levels of stress, which can lead to serious consequences if not addressed promptly. Stress can stem from various sources such as anxiety, frustration, anger, or unease. With the advancement of computer science, particularly in healthcare, the application of Machine Learning has seen notable progress. In this context, we introduce a methodology aimed at aiding schools, parents, and students in predicting the level of stress experienced by students. We established a real-time database and surveyed over 190 school children aged 14 to 18 years old, using 26 questions to analyze their stress levels and facilitate early intervention through Machine Learning algorithms. Various Machine Learning algorithms, including Decision Tree, Logistic Regression, K Nearest Neighbors, and Random Forest, were employed, with K Nearest Neighbors achieving a maximum accuracy of 88 per cent.[6]

III. METHODOLOGY

Model Architecture

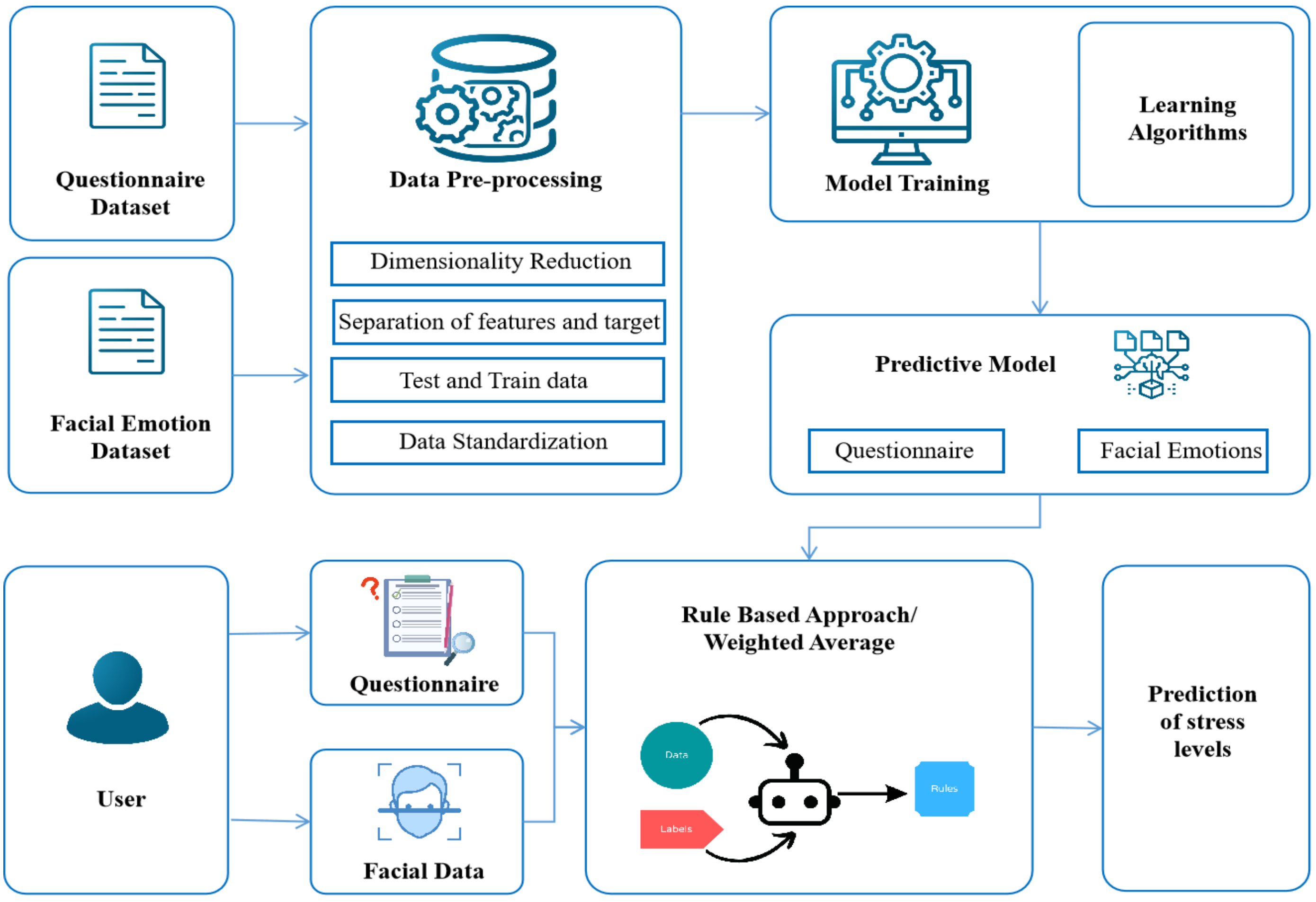


Figure 1: Architecture design of the model

The diagram shows the steps involved in the process, which can be broken down into four main modules:

1. Data collection
2. Data pre-processing
3. Feature extraction
4. Model development
5. Multi-modal fusion
6. Questionnaire design
7. Stress prediction.

Data Collection and Preprocessing

Acquire Facial Emotion Recognition (FER) data set from Kaggle, containing images labeled with facial expressions.
Obtain a data set on student mental health from Kaggle, comprising questionnaire responses providing insights into stress levels and related factors. Perform data preprocessing on the FER dataset, including image resizing, normalization, and augmentation techniques to enhance model robustness. Clean and preprocess the questionnaire data set, handling missing values, encoding categorical variables, and ensuring data consistency.

Feature Extraction and Model Development

Extract facial features using FER techniques, such as convolutional neural networks (CNNs), to capture emotional cues from facial expressions.
Derive relevant features from the questionnaire data set, considering responses related to stress levels, coping mechanisms, and lifestyle factors. Build a Facial Emotion Recognition model using deep learning architectures like CNNs, trained on the FER data set to classify facial expressions into different emotional states. Develop a stress prediction model using machine learning algorithms, leveraging features extracted from the questionnaire dataset to predict stress levels.

Questionnaire design and Multi-modal Fusion

In this stage, a questionnaire is designed to collect additional data from users about their stress levels. This data may include information about the user's current mood, recent events that may have caused stress, and other factors that can affect stress levels. Integrate the outputs of the FER model and stress prediction model to create a multimodal stress assessment system. Explore fusion techniques to combine facial emotion recognition and questionnaire-based stress prediction, enhancing the overall accuracy and reliability of stress assessment. In the stage of stress prediction, the machine learning model and the questionnaire data are used to predict the user's stress level. The model takes the user's facial data as input and outputs a predicted stress level. The questionnaire data is then used to refine the model's prediction.

The type of machine learning model used in this system is not specified, but it is likely to be a classification model, such as a support vector machine or a decision tree. The diagram shows that the system can be used to predict stress levels for multiple facial emotions. This suggests that the system is able to take into account the user's facial expression when predicting their stress level. The diagram also shows that the system can be used to collect data from multiple users. This data can be used to improve the accuracy of the model over time.

Algorithms used

Random Forest Algorithm

The Random Forest algorithm is a popular and powerful machine learning technique used for both classification and regression tasks. It belongs to the ensemble learning family, which combines multiple individual models to make more accurate predictions. Here's a detailed explanation of the Random Forest algorithm:

Random Forest is an ensemble learning method that combines the predictions of multiple decision trees to produce a single, more accurate prediction. Ensemble methods leverage the wisdom of the crowd principle, where the collective decision of multiple models often outperforms that of any single model. Decision trees are simple, tree-like structures consisting of nodes and branches, where each internal node represents a decision based on a feature, and each leaf node represents a class label or numerical value. Decision trees partition the feature space into smaller regions by recursively splitting it based on the feature that provides the most significant reduction in impurity, typically measured by metrics like Gini impurity or information gain.

Random Forest builds multiple decision trees independently and in parallel. Each decision tree is trained on a random subset of the training data, sampled with replacement (bootstrapping), and a random subset of features at each node of the tree. The randomness introduced in the sampling process helps to decorrelate the individual trees, reducing the risk of over-fitting and improving the robustness of the ensemble. For classification tasks, Random Forest employs a majority voting mechanism to determine the final predicted class. Each decision tree in the forest independently predicts the class label of a sample, and the class with the most votes across all trees is assigned as the final prediction. For regression tasks, Random Forest computes the average prediction from all individual trees to produce the final prediction.

Random Forest provides a measure of feature importance, indicating the contribution of each feature to the model's predictive performance. Feature importance is computed based on the decrease in impurity or information gain resulting from splits involving that feature across all decision trees in the forest. This information can be valuable for feature selection, dimensionality reduction, and gaining insights into the underlying data relationships. Random Forest is highly robust to overfitting, thanks to the randomness introduced during training. It can handle large datasets with high dimensionality and a mix of numerical and categorical features effectively. Random Forest requires minimal hyperparameter tuning compared to other complex models, making it relatively easy to use and deploy. It often yields high predictive accuracy and generalizes well to unseen data, making it a popular choice for various machine learning tasks.

Support Vector Machine Algorithm

Support Vector Machines (SVMs) are a powerful supervised machine learning algorithm used for classification and regression tasks. They are particularly well-suited for classification problems in which the data can be separated into distinct categories. Here's a detailed explanation of how SVM works:

SVM operates by finding the hyperplane that best separates the data into different classes. In two dimensions, this hyperplane is a line, and in higher dimensions, it's a plane or a hyperplane. The goal of SVM is to maximize the margin, which is the distance between the hyperplane and the nearest data points from each class, also known as support vectors. By maximizing the margin, SVM aims to find the hyperplane that generalizes well to unseen data. In its simplest form, SVM assumes that the data is linearly separable, meaning that it can be separated into classes by a straight line or hyperplane. SVM finds the optimal hyperplane by identifying the support vectors, which are the data points closest to the decision boundary (hyperplane). In cases where the data is not linearly separable, SVM can still be used by employing the kernel trick. The kernel trick involves mapping the input data into a higher-dimensional feature space where it becomes linearly separable. Common kernel functions include the linear kernel, polynomial kernel, radial basis function (RBF) kernel, and sigmoid kernel. These kernels allow SVM to handle complex, nonlinear relationships between features.

SVM aims to find the hyperplane that maximizes the margin, which is the distance between the hyperplane and the nearest data points (support vectors). By maximizing the margin, SVM achieves better generalization to unseen data and improves its robustness against noise and outliers. SVM uses a cost function, typically based on the hinge loss, to penalize misclassifications and determine the optimal hyperplane. The cost function includes a regularization parameter (C), which controls the trade-off between maximizing the margin and minimizing classification errors. A smaller C value results in a wider margin but may lead to more misclassifications, while a larger C value results in a narrower margin but may lead to overfitting. During training, SVM learns the parameters (weights and bias) of the hyperplane by optimizing the cost function using techniques such as gradient descent or quadratic programming. During prediction, SVM classifies new data points by determining which side of the hyperplane they fall on. Data points on one side are classified as one class, while data points on the other side are classified as the other class.

SVMs are effective in high-dimensional spaces and can handle datasets with a large number of features. They are memory efficient, as they only need to store the support vectors in memory. SVMs are versatile and can be used for both classification and regression tasks. SVMs can be sensitive to the choice of kernel function and its parameters. They are computationally intensive, particularly for large datasets. SVMs may not perform well with noisy data or datasets with overlapping classes.

Naive Bayes Classifier Algorithm

The Naive Bayes classifier is a simple yet powerful algorithm used for classification tasks in machine learning. Despite its simplicity, it often performs well in many real-world scenarios, especially when dealing with text classification and other high-dimensional data. Here's an explanation of the Naive Bayes classifier algorithm:

Naive Bayes is based on Bayes' theorem, which describes the probability of a hypothesis given the evidence. In the context of classification, it calculates the probability of a class label given the features of an instance. The "naive" aspect of Naive Bayes comes from the assumption of independence between features. It assumes that the presence of a particular feature in a class is unrelated to the presence of any other feature, given the class label. While this assumption may not hold true in reality, Naive Bayes can still perform well in practice. During the training phase, the Naive Bayes classifier learns the probability distributions of features for each class in the dataset. It calculates the prior probability of each class (P(C)) and the likelihood of each feature given each class (P(X|C)). To classify a new instance, the Naive Bayes classifier calculates the posterior probability of each class given the features of the instance using Bayes' theorem:

$$P(C|X) = (P(X|C) * P(C)) / P(X)$$

Since the denominator (P(X)) is constant for all classes, it can be ignored during classification, making the computation simpler. There are several variants of the Naive Bayes classifier, including:

Gaussian Naive Bayes: Assumes that continuous features follow a Gaussian (normal) distribution. Multinomial Naive Bayes: Suitable for text classification tasks with discrete features (e.g., word counts). Bernoulli Naive Bayes: Similar to Multinomial Naive Bayes but works with binary features (e.g., presence/absence of words). To handle cases where a particular feature does not occur in a class in the training data, Laplace smoothing (or additive smoothing) is often used. This prevents zero probabilities and ensures robustness of the model. Naive Bayes classifiers are computationally efficient and require minimal training data. They can handle high-dimensional data well and perform effectively in many real-world applications. Despite the "naive" assumption of feature independence, Naive Bayes often produces competitive results compared to more complex algorithms. The independence assumption may not hold true in some datasets, leading to suboptimal performance. Naive Bayes may struggle with rare or unseen feature combinations during classification. It can be sensitive to the quality of the input features and may perform poorly if features are highly correlated.

Logistic Regression Algorithm

Logistic Regression is a popular statistical technique used for binary classification problems, where the outcome variable is categorical and has only two possible outcomes (e.g., yes/no, true/false, 0/1). Despite its name, logistic regression is a classification algorithm rather than a regression algorithm.

Here's a detailed explanation of the Logistic Regression algorithm:

In logistic regression, the goal is to model the probability that a given input belongs to a particular class. The output of the logistic regression model, denoted as $h_w(x)$, is the probability that the input x belongs to the positive class (often denoted as class 1). The logistic regression model uses the logistic function, also known as the sigmoid function, to map the input space to the output space.

The sigmoid function is defined as:

$$\sigma(z)=1/1+e^{-z}$$

$$\text{Where } z=w^Tx+b$$

is the linear combination of input features and model parameters (weights w and bias b).

The hypothesis function $h_w(x)$ of logistic regression is defined as

$$h_w(x)=\sigma(w^Tx+b),$$

where $\sigma(z)$ is the logistic function.

This function outputs the estimated probability that the input (bf{x}) belongs to the positive class. The decision boundary of logistic regression is a hyperplane that separates the input space into regions corresponding to different class labels. The decision boundary is determined by the weights (bf{w}) and bias (b) learned during the training process. Logistic regression minimizes the log loss (also known as cross-entropy loss) to learn the optimal weights and bias.

The log loss function is defined as:

$$J(w,b)=-m1\sum i=1m[y(i)\log(hw(x(i)))+(1-y(i))\log(1-hw(x(i)))],$$

where (m) is the number of training examples, $y^{(i)}$ is the true label of the i-th example, and $h_w(x^{(i)})$ is the predicted probability.

The logistic regression model is trained using optimization algorithms such as gradient descent or its variants (e.g., stochastic gradient descent, mini-batch gradient descent). During training, the weights w and bias b are iteratively updated to minimize the log loss function. Regularization techniques such as L1 regularization (Lasso) and L2 regularization (Ridge) can be applied to prevent overfitting in logistic regression. Regularization adds a penalty term to the loss function, encouraging the model to learn simpler and more generalizable patterns. Once trained, the logistic regression model can be used to make predictions on new data by applying the learned weights and bias to the input features. The predicted probability y is compared to a threshold (e.g., 0.5) to determine the predicted class label. Logistic Regression is widely used in various domains, including healthcare (e.g., disease diagnosis), finance (e.g., credit scoring), and marketing (e.g., customer churn prediction), due to its simplicity, interpretability, and effectiveness for binary classification tasks.

K-Nearest Neighbour Algorithm

The K-Nearest Neighbors (KNN) algorithm is a simple yet effective supervised machine learning algorithm used for both classification and regression tasks. It is a non-parametric method, meaning it does not make any assumptions about the underlying data distribution. Instead, it relies on the principle that similar data points are close to each other in feature space.

Here's a detailed explanation of how the KNN algorithm works:

KNN operates on the principle of similarity. It assumes that similar data points should have similar labels or values. For a given unlabeled data point, KNN identifies its K nearest neighbors in the feature space based on some distance metric (e.g., Euclidean distance). The majority label (for classification) or the average value (for regression) of the K nearest neighbors is then assigned to the unlabeled data point. The parameter K represents the number of nearest neighbors to consider when making predictions. Choosing the optimal value of K is crucial and depends on the nature of the dataset and the problem at hand. A small value of K may lead to overfitting, while a large value of K may result in underfitting.

KNN relies on a distance metric to measure the similarity between data points. The most commonly used distance metric is the Euclidean distance, which calculates the straight-line distance between two points in the feature space. Other distance metrics, such as Manhattan distance or Minkowski distance, can also be used depending on the nature of the data. For classification tasks, once the K nearest neighbors are identified, the majority class among them is assigned to the unlabeled data point. For regression tasks, the average of the values of the K nearest neighbors is assigned to the unlabeled data point. KNN is an instance-based learning algorithm, meaning it does not explicitly learn a model during the training phase. Instead, during training, KNN simply stores the entire training dataset in memory.

One of the main drawbacks of KNN is its lack of scalability, especially with large datasets. Since it stores the entire training dataset in memory, the algorithm's memory and computational requirements grow with the size of the dataset. KNN does not explicitly learn a decision boundary like other classification algorithms such as logistic regression or decision trees. Instead, the decision boundary is implicit and is defined by the distribution of data points in the feature space. Simple to understand and implement, requires no training phase, works well with small datasets and non-linear data. Computationally expensive for large datasets, sensitive to the choice of K and the distance metric, requires careful preprocessing of data.

Performance and Efficiency

To evaluate the performance and efficiency of the mentioned algorithms (Random Forest, SVM, KNN, Naive Bayes, and Logistic Regression), we'll consider several factors:

Performance Metrics

- Accuracy: Measures the proportion of correctly classified instances.
- Precision: Measures the proportion of true positive predictions out of all positive predictions.
- Recall: Measures the proportion of true positive predictions out of all actual positive instances.
- F1-score: Harmonic mean of precision and recall, provides a balanced measure of performance.
- Time Complexity: Refers to the computational time required for training and testing the model.

Scalability

- Ability of the algorithm to handle large datasets efficiently.

Robustness

- Ability of the algorithm to perform well in the presence of noise or outliers in the data.

Now, let's evaluate each algorithm based on these criteria:

Random Forest

Performance : Random Forest typically performs well across various types of datasets due to its ensemble nature, which reduces overfitting and variance. It often achieves high accuracy and F1-score, especially when there are complex interactions between features. It's less sensitive to outliers compared to some other algorithms.

Efficiency : Random Forest can handle large datasets efficiently and is parallelizable, making it suitable for high-dimensional data. However, it may require more memory and computational resources compared to some other algorithms due to the ensemble of decision trees.

SVM (Support Vector Machine)

Performance : SVM is effective in high-dimensional spaces and can capture complex relationships in data. It often performs well when there is a clear margin of separation between classes. SVM tends to have good generalization ability and works well for both linear and non-linear datasets.

Efficiency : SVM can be computationally expensive, especially for large datasets, as it involves solving a quadratic optimization problem. Kernel SVM, in particular, may require significant computational resources for training. However, once trained, SVMs can make predictions efficiently.

KNN (K-Nearest Neighbors)

Performance : KNN is simple and intuitive, often performing well on datasets with clear separation between classes. However, it may struggle with high-dimensional data and noisy datasets. Choosing the optimal value of K is crucial for its performance.

Efficiency : KNN can be inefficient, especially for large datasets, as it requires storing the entire training dataset in memory and computing distances to all data points during testing. The computational complexity of KNN increases with the size of the dataset.

Naive Bayes

Performance : Naive Bayes is efficient and often performs well on text classification tasks and datasets with categorical features. However, it makes a strong assumption of feature independence, which may not hold true in practice. Despite this, Naive Bayes can provide surprisingly good performance, especially with large datasets.

Efficiency : Naive Bayes is computationally efficient, as it involves simple probabilistic calculations and does not require complex optimization procedures. It can handle large datasets with ease and is suitable for real-time applications.

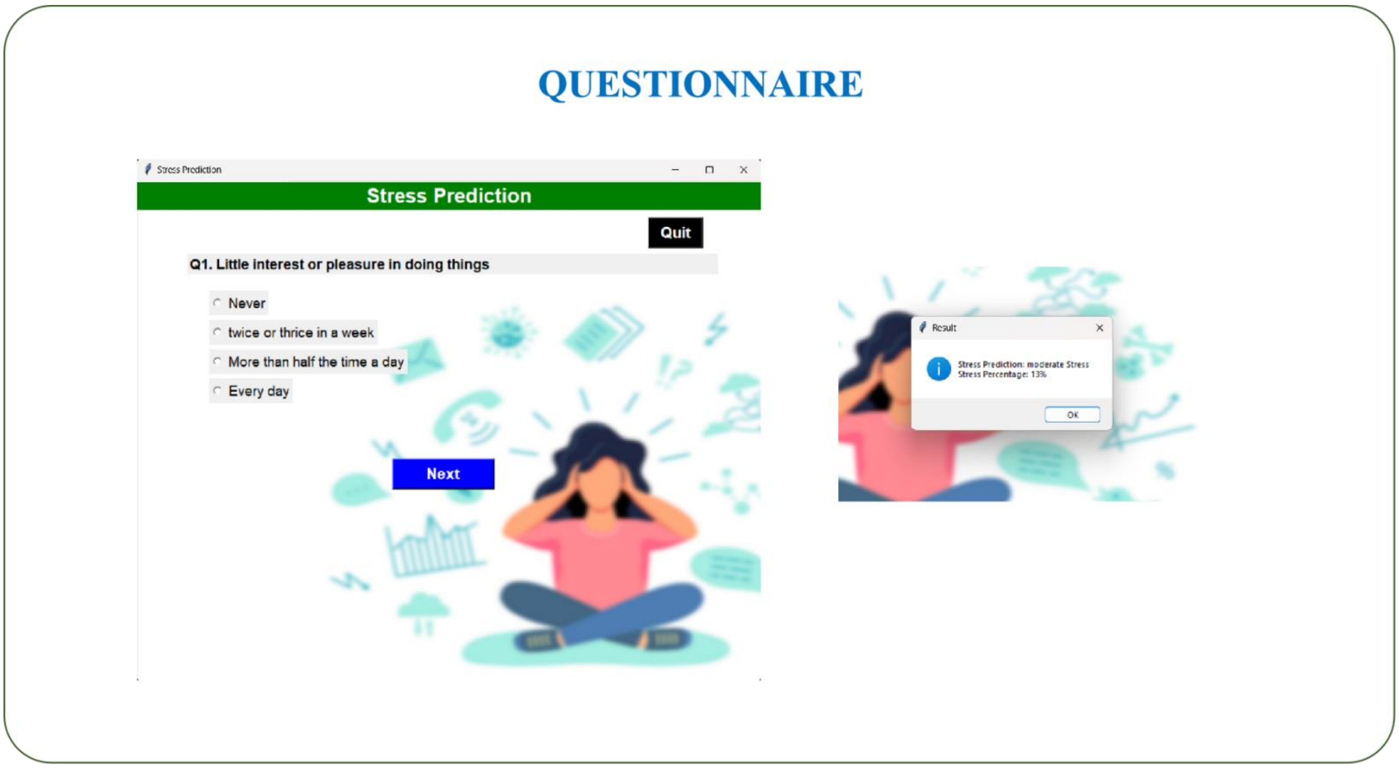
Logistic Regression

Performance : Logistic Regression is a simple yet powerful algorithm, particularly effective for binary classification tasks. It performs well when there is a linear relationship between the features and the target variable. It can also provide probabilistic interpretations of predictions.

Efficiency : Logistic Regression is computationally efficient and scalable, making it suitable for large datasets. It involves simple calculations and optimization procedures, making it less resource-intensive compared to some other algorithms.

In conclusion, while all the mentioned algorithms have their strengths and weaknesses, Random Forest tends to be a top performer in terms of both performance and efficiency across a wide range of datasets. However, the choice of algorithm ultimately depends on the specific characteristics of the dataset, the problem at hand, and computational constraints.

IV. Result / Screenshots



Q1. Little interest or pleasure in doing things

Stress Prediction: moderate Stress
Stress Percentage: 13%

Figure 2: Questionnaire Module

The provided module serves as the questionnaire component within our application, facilitating user interaction to predict their current mental state. Users engage with the application by responding to a series of 20 questions, each addressing various mental health-related aspects such as stress levels, enthusiasm, and average sleep duration. Concurrently, our application integrates a facial recognition and emotion detection module, which activates alongside the questionnaire interaction. This module employs facial expression analysis to recognize and detect the user's emotions throughout their interaction with the questionnaire.

Upon completion of all 20 questions, the user submits their responses, triggering the evaluation of their stress levels using learning algorithms. The system utilizes these algorithms to determine the most fitting assessment of the user's stress level, presenting the result as a percentage score within the interface. This holistic approach combines user-provided responses with real-time facial emotion analysis to offer a comprehensive understanding of the user's mental state.

By seamlessly integrating questionnaire-based inquiries with facial emotion detection, our application provides users with an intuitive and efficient means of assessing their mental well-being. The fusion of these modules enhances the accuracy and depth of insight into the user's emotional state, empowering them to gain valuable self-awareness and potentially seek appropriate support or intervention if needed.

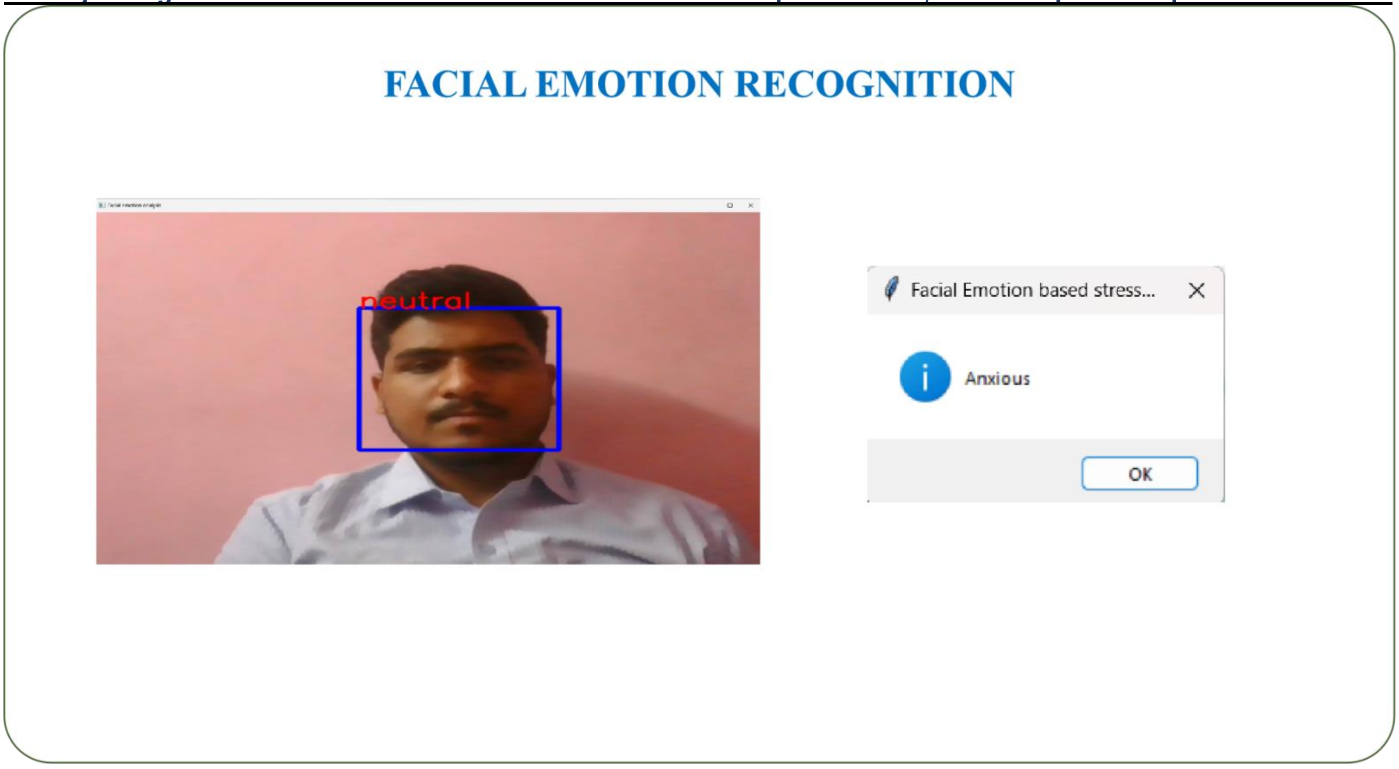


Figure 3: Facial Recognition Module

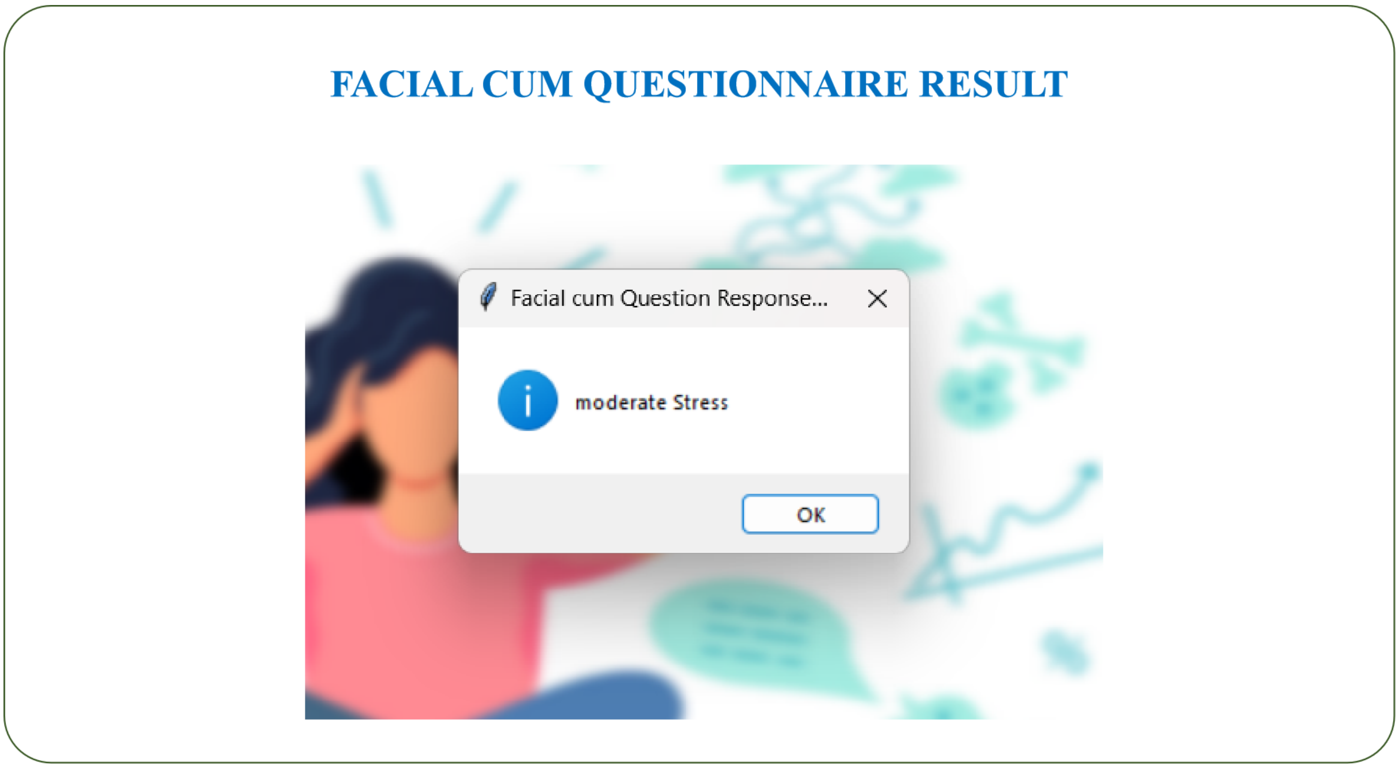

FACIAL CUM QUESTIONNAIRE RESULT
Facial cum Question Response...
moderate Stress
OK

Figure 4: Facial Cum Questionnaire Module Result

V. Conclusion

The project presents a novel approach to stress prediction by leveraging a multimodal fusion of facial emotion recognition and questionnaire-based profiling. Through the integration of advanced machine learning algorithms, including Convolutional Neural Networks (CNNs) for facial emotion recognition and ensemble methods like Random Forest for stress prediction, the system achieves robust performance in accurately assessing stress levels. By combining explicit indicators from questionnaire responses with implicit cues from facial expressions, the developed system offers a comprehensive understanding of an individual's stress state. This holistic approach not only enhances the accuracy of stress prediction but also provides personalized recommendations for stress management. Furthermore, the project demonstrates the feasibility and effectiveness of integrating diverse modalities and algorithms to tackle complex real-world problems. The deployment of the stress assessment system offers users a seamless and accessible tool for monitoring and managing their stress levels. Moving forward, ongoing refinement and optimization of the system, coupled with continuous validation and testing, will further enhance its performance and usability. Additionally, exploring additional data sources and incorporating more sophisticated modeling techniques could unlock new avenues for improving stress prediction accuracy and tailoring interventions to individual needs. The project represents a significant step towards leveraging AI-driven solutions to address mental health challenges in today's fast-paced society.

VI. Acknowledgement

I would like to express my heartfelt gratitude to all those who contributed to the successful completion of the A Multimodal Approach to Predicting Stress Levels Through Psychological Profiling and Facial Recognition Project. First and foremost, I extend my sincere thanks to our project supervisor, Mrs. V. Kalaivani for their unwavering support, guidance, and valuable insights throughout the entire development process. I am deeply appreciative of the Head of our Department, Dr. G. Fathima, faculty members and staff who provided their expertise and resources, enriching the project with their knowledge. Special thanks to my fellow teammates for their collaborative spirit, dedication, and hard work, which played a pivotal role in achieving our shared goals. I am also thankful to the academic institution for fostering an environment that encourages innovation and continuous improvement. Lastly, I extend my gratitude to friends and family for their understanding and encouragement during the project's journey. Each contribution, no matter how small, has been instrumental in making this project a success.

VII. References

[1]. Sujith Kumar, R,Soundar Sriram J, Sridhar C, Fathima G, “Facial Emotion Recognition Based Prediction Of Affective State Of Children With Autism Using ML”, **2023**

[2].K. Singh, N. Goel, B. Gupta and D.Bansal, “Emotion Prediction through Facial Recognition Using Machine Learning: A Survey”, **2023**

[3]. Sharisha Shanbhog M,Jeevan M, “Prediction of Student's Wellbeing from Stress and Sleep Questionnaire data using Machine Learning Approach”, **2022**

[4]. Srishti Singh, Harsh Gupta, Prashant Singh, Arun Prakash Agrawal, “Comparative Analysis of Machine Learning Models to Predict Depression, Anxiety and Stress”, **2022**

[5]. Aanchal Bisht, Shreya Vashisth, Muskan Gupta, Ena JainStress, “Prediction in Indian School Students Using Machine Learning” , **2022**

[6]. Aditya Vivek Thota,A Dharun, “Machine Learning Techniques for Stress Prediction in Working Employees”, **2021**

[7].A.Aizenstros, D. Bakker, S. Hofmann, J. Curtiss, N. Kazantzis, "Engagement with smartphone-delivered behavioral activation interventions: A study of the MoodMission smartphone application", **2021**.

[8].W. Cao et al., "The psychological impact of the COVID-19 epidemic on college students in China", **2020**.

[9]. Garima Verma and Hemraj Verma, "Model for predicting academic stress among students of technical education in India", International Journal of Psychosocial Rehabilitation, **2020**.

[10].T. Harding, V. Lopez and P. Klainin-Yobas, "Predictors of Psychological Well-Being among Higher Education Students",
Psychology, vol. 10, no. 04, pp. 578-594, **2019**.

[11]. Ravinder Ahuja and Alisha Banga, "Mental Stress Detection in University Students using Machine Learning Algorithms", International Conference on Pervasive Computing Advances and Applications-PerCAA, **2019**.

[12]. Reshma Radheshamjee Baheti and Supriya Kinariwala, "Detection and Analysis of Stress using Machine Learning Techniques", International Journal of Engineering and Advanced Technology (IJEAT), **2019**

[13].A. Sau and I. Bhakta, "Screening of anxiety and depression among the seafarers using machine learning technology", Informatics in Medicine Unlocked, **2018**.

[14].M. Srividya, S. Mohanavalli and N. Bhalaji, "Behavioral Modeling for Mental Health using Machine Learning Algorithms", Journal of medical systems, **2018**.

[15]. Jonghwa Kim,Simon Flutura Yoshiki Nakashima, "Stress Recognition in Daily Work," in Springer International Publishing, **2016**.