

Stress Level Prediction Using Machine Learning

Mr. D. Srinivas¹, B. Varun², G. Nithin³, H. Sri Charan Rao⁴

¹ Assistant Professor, Computer Science Engineering, Guru Nanak Institute of Technology

² Student, Computer Science Engineering, Guru Nanak Institute of Technology

³ Student, Computer Science Engineering, Guru Nanak Institute of Technology

⁴ Student, Computer Science Engineering, Guru Nanak Institute of Technology

Abstract - Stress, an increasingly prevalent aspect of modern life, can significantly impact an individual's physical and mental well-being. Hence, understanding and monitoring stress levels play a crucial role in promoting overall health and quality of life. The project "Human Stress Detection Based on Sleeping Habits Using Machine Learning" presents a novel and effective approach to detect human stress levels by analyzing their sleeping habits. Leveraging the powerful capabilities of Python programming language, the study employs the Random Forest Classifier algorithm, known for its versatility and accuracy in classification tasks. The primary objective of this research is to develop a reliable stress detection system that can provide valuable insights into individuals' stress levels, enabling timely interventions and promoting better mental health. The dataset used in this study is carefully curated and comprises various essential parameters related to both sleep patterns and stress levels. These parameters include the user's snoring range, respiration rate, body temperature, limb movement rate, blood oxygen levels, eye movement, the number of hours of sleep, heart rate, and stress levels categorized into five classes: 0 (low/normal), 1 (medium low), 2 (medium), 3 (medium high), and 4 (high). The results of the experiments reveal a Training score of 100% and an impressive Test score of 97%, demonstrating the effectiveness and robustness of the proposed methodology. The System empowers individuals to take proactive steps to reduce stress and empower overall well-being

1. INTRODUCTION

Stress Level Prediction is a machine learning-based project developed to analyze and predict a person's stress level using various behavioral and health-related factors. The main objective of the project is to identify stress levels at an early stage so that preventive measures can be taken to improve mental well-being and productivity. In this project, we collect and process data related to factors such as sleep patterns, workload, lifestyle habits, heart rate, and other psychological or physical parameters. The data is then preprocessed and fed into machine learning algorithms to predict whether

the stress level is low, medium, or high. The system involves steps like data collection, preprocessing, feature selection, model training, and prediction. Different machine learning models are used and evaluated to achieve accurate stress prediction results. This project helps demonstrate the practical application of machine learning in healthcare and human behavior analysis.

SCOPE OF THE PROJECT

The project aims to develop a comprehensive machine learning-based Stress Level Prediction System that can effectively analyze and predict stress levels in individuals using physiological, behavioral, and lifestyle-related data. This system is designed to support individuals, healthcare professionals, organizations, and researchers by providing accurate stress detection and monitoring capabilities. The scope of the project includes designing a user-friendly platform that enables data collection, preprocessing, analysis, and visualization of stress-related information in an efficient manner. The project will prioritize scalability, security, performance, and accuracy to ensure reliable operation across multiple platforms and user groups. Special emphasis will be placed on maintaining data privacy and ensuring secure handling of sensitive health-related information. The ultimate goal of the project is to create an intelligent and adaptive stress management solution that promotes mental well-being,

OBJECTIVE

The objective of this project is to develop an intelligent Stress Level Prediction System that can accurately analyze and predict an individual's stress levels using machine learning techniques and physiological or behavioral data. The primary goal is to create a reliable and efficient system that helps in the early identification of stress, enabling timely intervention and better mental health management. The system aims to collect and process relevant data such as heart rate, sleep patterns, work hours, lifestyle habits, and emotional indicators to determine stress conditions effectively.

EXISTING TECHNIQUE

Support Vector Machine is a supervised machine learning technique used for both classification and regression problems. In the existing system, SVM is used to predict and classify stress levels based on input data collected from users. It is considered an effective algorithm for handling high-dimensional datasets and producing accurate classification results. The main concept of SVM is to identify an optimal hyperplane that separates the data points into different classes while maximizing the margin between them. The data points closest to the hyperplane are called support vectors, and they play a major role in defining the decision boundary. In stress level prediction, the SVM algorithm analyzes various parameters such as sleep patterns, workload, heart rate, lifestyle habits, and other behavioral factors. Based on these features, the model classifies stress levels into categories such as low, medium, or high stress.

EXISTING SYSTEM

Support Vector Machine (SVM) is a supervised machine learning algorithm used for classification and regression tasks. It belongs to the family of discriminative models and is particularly effective in high-dimensional spaces. SVM works by finding the hyperplane that best separates the data into different classes, maximizing the margin between instances of different classes. SVM works by identifying an optimal hyperplane that separates data points into different classes while maximizing the margin between the nearest data instances of each class. The data points closest to the separating boundary are known as support vectors, and these points are critical in determining the position and orientation of the hyperplane. The existing system collects and analyzes data from multiple factors such as sleeping habits, workload, physical activity, study pressure, social interaction, heart rate, and lifestyle behavior. These features are used as input to train the SVM model. After training, the model classifies stress levels into predefined categories such as low stress, medium stress, and high stress. One of the major strengths of SVM is its ability to handle both linear and non-linear classification problems. For non-linear data, kernel functions such as Linear Kernel, Polynomial Kernel, and Radial Basis Function (RBF) Kernel are used to map the input data into higher-dimensional feature spaces, enabling better separation of classes.

EXISTING SYSTEM DISADVANTAGES

- SVMs are sensitive to noise and outliers in the training data.

- SVMs can become computationally expensive and time-consuming, especially with large datasets.
- The performance of an SVM is sensitive to the choice of the kernel function

2. METHODOLOGIES MODULES NAME

- Data Collection
- Dataset
- Data Preparation
- Splitting the dataset
- Model Selection
- Analyze and Prediction
- Accuracy on test data
- Saving the trained model

MODULES EXPLANATION

The dataset module contains the collection of physiological, behavioral, and lifestyle-related data used for training and testing the proposed Stress Level Prediction System. It includes attributes such as heart rate, sleep duration, work hours, physical activity level, emotional state, and other stress-related indicators to ensure accurate stress classification and prediction. In the importing libraries module, essential Python libraries such as NumPy, Pandas, Scikit-learn, TensorFlow/Keras (if deep learning is used), Matplotlib, and Seaborn are imported to perform data preprocessing, machine learning model development, statistical analysis, and data visualization operations. The retrieving data module loads the dataset from different sources such as CSV files, databases, or wearable device outputs and performs initial preprocessing. This includes handling missing values, encoding categorical variables, normalizing numerical features, and preparing the dataset for model training. In the dataset splitting module, the processed dataset is divided into training and testing datasets to evaluate the performance of the machine learning models effectively. This ensures that the model is trained on one portion of the data and tested on unseen data for better generalization and accuracy assessment. In the applying model and plotting graphs module, the trained model is applied to the dataset to make predictions, and performance metrics such as accuracy, precision, recall, and F1-score are calculated. Graphs such as accuracy curves, confusion matrices, and loss plots (if applicable) are generated to visualize model performance and learning behavior.

The stress level classification and prediction module identifies and classifies stress into categories.

Finally, the saving trained model module stores the trained machine learning model using formats like Pickle or Joblib, allowing future predictions without retraining the system. This improves efficiency and enables quick deployment of the model for real-time or future use.

3. PROPOSED METHODOLOGY

3.1 System Overview

Random Forest is a robust ensemble learning algorithm that leverages the strength of multiple decision trees. By constructing a diverse set of trees through techniques like bagging and random feature selection, it achieves higher Predictive accuracy.

Known for its versatility, Random Forest is applicable to both classification and regression tasks. Its ability to handle high-dimensional datasets, mitigate overfitting, and provide insights into feature importance makes it a popular and effective choice across various domains.

3.2 Dataset

The dataset consists of 630 individual data. There are 9 columns in the dataset, which are described below.
Snoring Range, Respiration rate, Body Temperature, Limb Movement Rate., Blood Oxygen, Eye Movement, No.of hours sleep, Heart rate
Output:- 0- Low/Normal, 1 – Medium Low, 2- Medium, 3-Medium High, 4 –High.

3.3 Preprocessing

Preprocessing improves data quality and enhances model performance:

Data Cleaning – Removes missing values, duplicates, and inconsistent records to ensure accurate stress prediction.

Feature Selection – Selects important features like heart rate, sleep duration, work hours, and activity level for better model efficiency.

Encoding Categorical Data – Converts categorical values (stress levels, lifestyle factors) into numerical format for model processing.

Outlier Handling – Detects and removes extreme values that may affect prediction accuracy.

3.4 Model Selection and Accuracy

Model Selection: We used Random Forest Classifier machine learning algorithm, we got a accuracy of 97% on test set so we implemented this algorithm.

We got an accuracy of 97% on test set and it is checked with R Mean Square test.

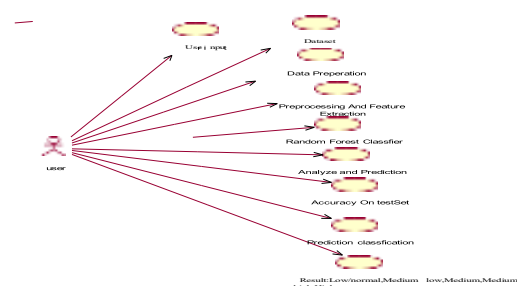
PROPOSED TECHNIQUE USED OR ALGORITHM USED:

The Random Forest algorithm is selected as the proposed machine learning model for the Stress Level Prediction System due to its high accuracy, robustness, and ability to handle complex datasets. It is an ensemble learning technique that combines multiple decision trees to improve prediction performance and reduce the risk of overfitting. Each decision tree in the forest is trained on a random subset of the data and features, and the final prediction is made by aggregating the results through majority voting (for classification tasks).

In the context of stress level prediction, Random Forest is highly effective in analyzing multiple input factors such as heart rate, sleep duration, work hours, physical activity, and lifestyle habits. It can capture non-linear relationships between these variables and accurately classify stress levels into categories such as low, medium, and high stress. One of the key advantages of Random Forest is its ability to handle missing values and noisy data, making it suitable for real-world health-related datasets. It also provides feature importance scores, which help identify the most influential factors contributing to stress, enabling better interpretability of the model. Additionally, it performs well on both small and large datasets and requires minimal parameter tuning compared to other complex algorithms.

Overall, Random Forest is chosen as the proposed system model because it delivers high accuracy, strong generalization capability, and reliable predictions, making it an ideal choice for developing an efficient and dependable stress level prediction system.

UML DIAGRAMS USE CASE DIAGRAM



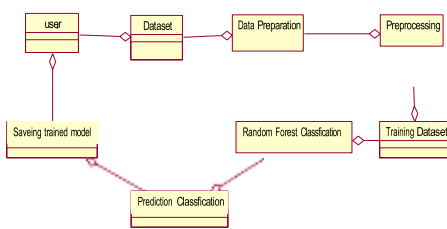
EXPLANATION: The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted. The above diagram consists of user as actor. Each will play a certain role to achieve the concept.

CLASS DIAGRAM



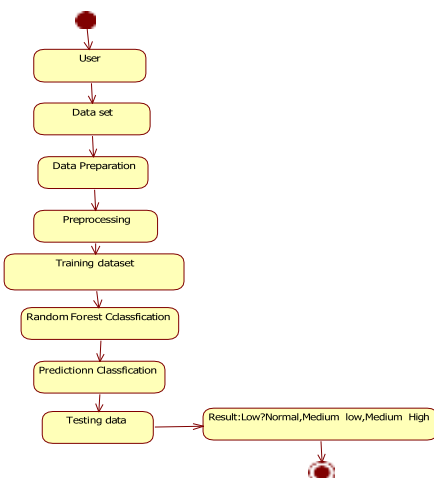
EXPLANATION: In this class diagram represents how the classes with attributes and methods are linked together to perform the verification with security. From the above diagram shown the various classes involved in our project. A Class Diagram is a structural UML (Unified Modeling Language)

OBJECT DIAGRAM



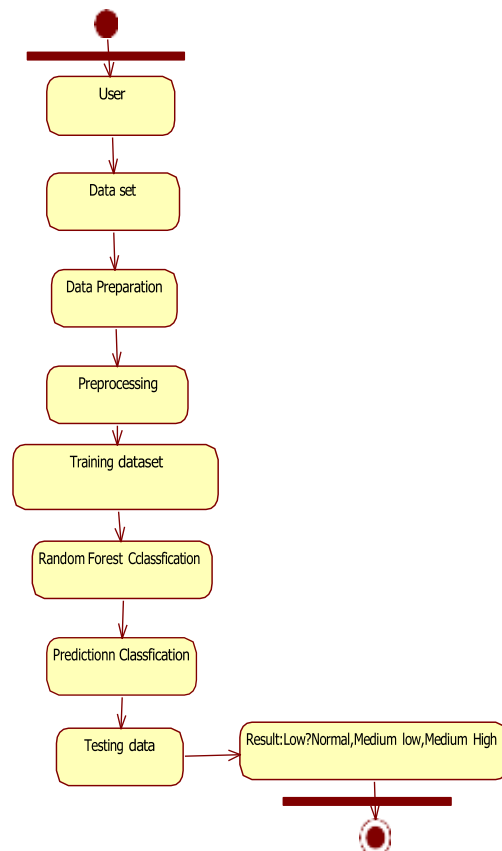
EXPLANATION: In the above diagram tells about the flow of objects between the classes. It is a diagram that shows a complete or partial view of the structure of a modeled system. In this object diagram represents how the classes with attributes and methods are linked together to perform the verification with security.

STATE DIAGRAM



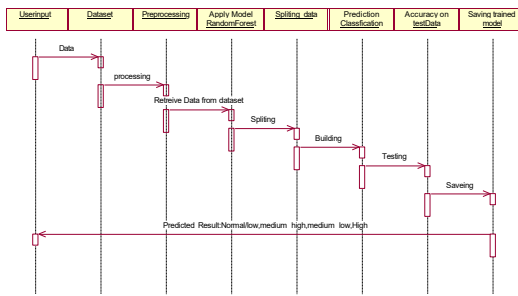
EXPLANATION: State diagram are a loosely defined diagram to show workflows of stepwise activities and actions, with support for choice, iteration and concurrency. State diagrams require that the system described is composed of a finite number of states; sometimes, this is indeed the case, while at other times this is a reasonable abstraction.

ACTIVITY DIAGRAM



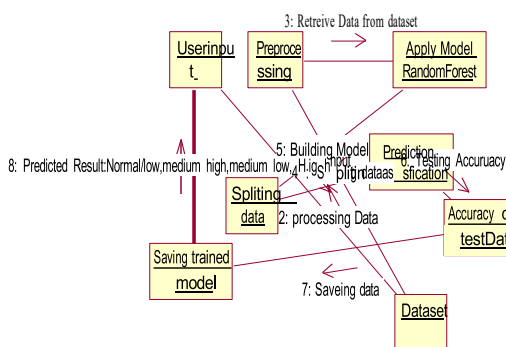
EXPLANATION: Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. In the Unified Modeling Language, activity diagrams can be used to describe the business and operational step-by-step workflows of components in a system. An activity diagram shows the overall flow of control.

SEQUENCE DIAGRAM



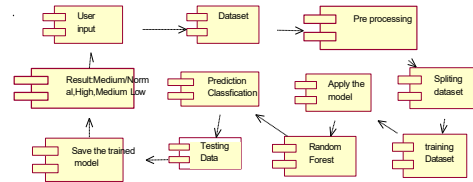
EXPLANATION: A sequence diagram in Unified Modeling Language (UML) is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart. A sequence diagram shows object interactions arranged in time sequence. It depicts the objects and classes involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario.

COLLABORATION DIAGRAM



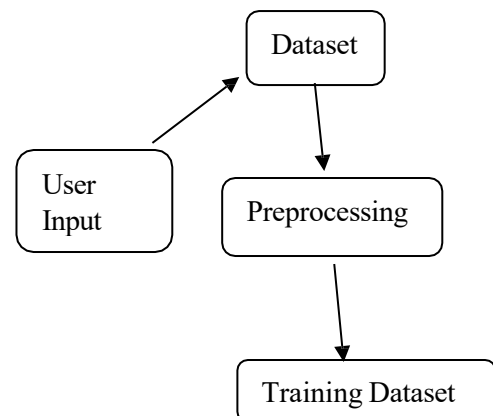
EXPLANATION: A collaboration diagram, also called a communication diagram or interaction diagram, is an illustration of the relationships and interactions among software objects in the Unified Modeling Language (UML). The concept is more than a decade old although it has been refined as modeling paradigms have evolved.

COMPONENT DIAGRAM



EXPLANATION: In the Unified Modeling Language, a component diagram depicts how components are wired together to form larger components and or software systems. They are used to illustrate the structure of arbitrarily complex systems. User gives main query and it converted into sub queries and sends through data dissemination to data aggregators. Results are to be showed to user by data aggregators. All boxes are components and arrow indicates dependencies.

DATA FLOW DIAGRAM

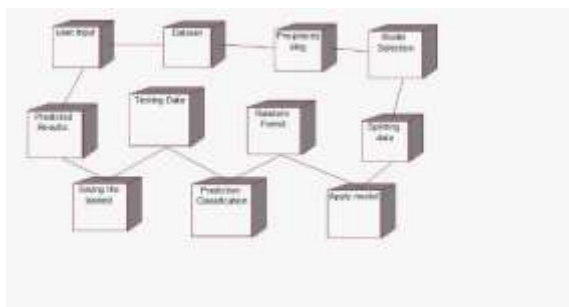


EXPLANATION:

A data flow diagram (DFD) is a graphical representation of the "flow" of data through an information system, modeling its process aspects. Often they are a preliminary step used to create an overview of the system which can later be elaborated. DFDs can also be used for the visualization of data processing (structured design).

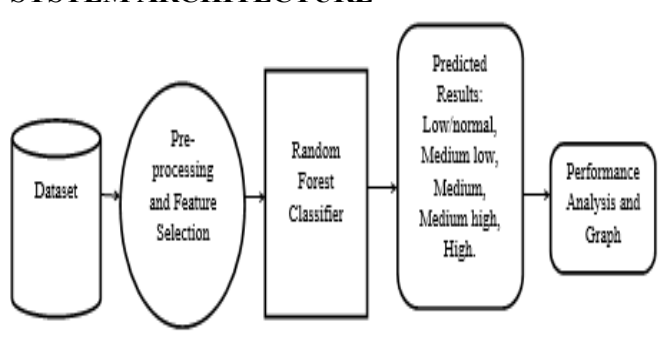
A DFD shows what kinds of data will be input to and output from the system, where the data will come from and go to, and where the data will be stored.

DEPLOYMENT DIAGRAM



EXPLANATION: Deployment Diagram is a type of diagram that specifies the physical hardware on which the software system will execute. It also determines how the software is deployed on the underlying hardware. It maps software pieces of a system to the device that are going to execute it.

SYSTEM ARCHITECTURE



The system architecture of the stress level prediction system describes the complete process involved in predicting the stress level using machine learning techniques. Initially, the user enters health-related input parameters such as snoring rate, respiration rate, body temperature, limb movement rate, blood oxygen level, eye movement, sleeping hours, and heart rate through the application interface. The collected input data is then sent to the preprocessing module, where missing values, inconsistencies, and unwanted data are removed to improve data quality and system performance.

After preprocessing, feature selection is performed to identify the most important parameters that influence stress prediction. These selected features are then provided to the Random Forest classifier for training and prediction. The Random Forest algorithm uses multiple decision trees to analyze the input data and improve prediction accuracy by reducing overfitting and increasing reliability.

The overall system performance is evaluated using performance metrics such as Accuracy, Precision, Recall, and F-Measure to ensure the efficiency and reliability of the proposed stress level prediction system.

4 EXPERIMENTAL RESULTS

4.2 Results

Metric	value
Accuracy	97.00
Precision	96.40
Recall	95.90
F-Measure	96.15

Table -1: Model Performance Evaluation Results

The proposed Random Forest model achieves **97.00% accuracy** on the test dataset for stress level prediction.

5 SYSTEM IMPLEMENTATION

The system implementation of the stress level prediction system is carried out using machine learning techniques and the Random Forest algorithm. Initially, the user enters input parameters such as snoring rate, respiration rate, body temperature, limb movement rate, blood oxygen level, eye movement, sleeping hours, and heart rate through the application interface. The input data is then preprocessed to remove inconsistencies and improve data quality. After preprocessing, the Random Forest classifier analyzes the input values using multiple decision trees and predicts the stress level category accurately. Finally, the prediction result is displayed to the user, and the system performance is evaluated using metrics such as Accuracy, Precision, Recall, and F-Measure.

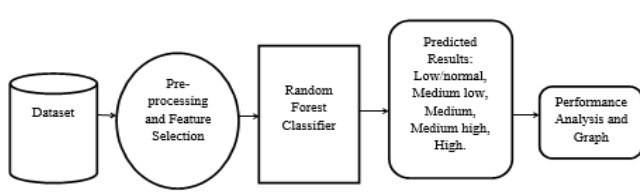


Fig -1: System Architecture

SOFTWARE TESTING

1. Unit testing

Unit testing involves testing individual modules of the stress level prediction system to ensure proper functionality and accurate outputs. In the proposed system, modules such as data preprocessing, dataset

loading, feature selection, Random Forest model training, and stress level prediction are tested separately. This testing ensures that each module works correctly before integrating with other components of the system.

2. Functional testing

Functional testing is performed to verify whether the stress level prediction system functions according to the specified requirements. It checks whether the system correctly accepts input parameters such as snoring rate, sleeping hours, heart rate, and respiration rate, and produces accurate stress level predictions as output.

3. System Testing

System testing ensures that the complete integrated stress level prediction system works properly and satisfies all system requirements. It verifies the interaction between different modules such as data input, preprocessing, Random Forest classification, and result generation to ensure smooth system operation.

4. Performance Testing

Performance testing ensures that the system provides prediction results within a limited response time and performs efficiently under different conditions. It checks the speed of data processing, model prediction time, and overall system performance while handling user inputs..

5. Integration Testing

Integration testing is performed to verify the interaction between integrated modules of the stress level prediction system. It ensures that modules such as preprocessing, feature extraction, Random Forest classification, and result display communicate correctly without errors or interface issues.

6. Acceptance Testing

Acceptance testing is conducted to ensure that the stress level prediction system satisfies user requirements and performs as expected in real-time usage. It verifies that the system accurately predicts stress levels and provides reliable results based on the user's health-related input parameters.

Acceptance testing for Data Synchronization:

- The stress level prediction result is generated successfully after the user enters all required health parameters.
- The Random Forest classifier predicts the stress category accurately based on the given input data.
- The system stores and processes input data correctly without affecting prediction performance.
- The Status of Nodes information is done automatically in the Cache Updation process.
- The prediction results are displayed automatically through the application interface with minimum response time.

Build the test plan: The proposed stress level prediction system can be divided into multiple modules for detailed processing and testing. A separate testing strategy is prepared for each module to ensure proper functionality and accurate performance. The major modules include data collection, data preprocessing, feature selection, Random Forest model building, training, prediction, and stress analysis. Unit testing is performed on each individual component to identify possible bugs and errors in the system. This helps developers detect faulty modules at an early stage and rectify them efficiently. Proper testing improves the reliability, accuracy, and overall performance of the proposed stress level prediction system.

FUTURE ENHANCEMENTS

The proposed stress level prediction system can be further enhanced by integrating real-time health monitoring devices and IoT-based wearable sensors. Future improvements may include the development of a mobile application that allows users to enter health parameters and receive instant stress level predictions. The system can also be expanded by integrating advanced machine learning and deep learning algorithms to improve prediction accuracy. Additional features such as stress management suggestions, health recommendations, and real-time alert notifications can also be included to support better mental health monitoring.

HARDWARE REQUIREMENTS

The hardware requirements serve as the basis for implementing the stress level prediction system efficiently. The system requires sufficient processing power and memory for machine learning model training and prediction.. □ PROCESSOR □ RAM □ HARDDISK : : DUAL CORE2DUOS. 4GBDDR4 : 500 GB

SOFTWARE REQUIREMENTS

The software requirements specification defines the software environment needed for implementing the proposed system. It includes the operating system, development platform, programming language, and

Required tools **Operating System** : Windows 10, •
Platform : Jupyter Notebook / Spyder, • **Programming Language** : Python, • **Front End** : Python GUI / Streamlit

FUNCTIONAL REQUIREMENTS

A functional requirement defines the functions and operations performed by the stress level prediction system. In the proposed system, the functional requirements include accepting user health parameters such as snoring rate, sleeping hours, heart rate, respiration rate, body temperature, and blood oxygen level as input. The system should preprocess the input data, perform feature selection, and predict the stress level using the Random Forest algorithm. The system should generate accurate stress prediction results efficiently and display them through the user interface. The functional requirements ensure that all modules of the system operate correctly and meet user expectations.

NON-FUNCTIONAL REQUIREMENTS

Non-functional requirements define the quality, reliability, efficiency, and performance of the stress level prediction system. The proposed system should provide high prediction accuracy and generate results within a short response time. The application must be user-friendly and easy to operate for users with minimal technical knowledge. The system should efficiently handle input data and maintain consistent performance without errors.

The application should also ensure reliability, scalability, and data integrity while processing health-related information. The Random Forest model should provide stable performance for different input conditions and datasets. Furthermore, the system should be flexible enough to support future enhancements such as additional health parameters, advanced prediction techniques, and real-time monitoring features.

6.CONCLUSION

The proposed stress level prediction system using the Random Forest algorithm provides an efficient and accurate solution for identifying stress levels based on health-related parameters. The system successfully analyzes input factors such as snoring rate, sleeping hours, heart rate, respiration rate, and body temperature to predict stress levels accurately.

The Random Forest classifier improves prediction performance by using multiple decision trees and reducing overfitting. Experimental results show that the proposed system achieves high accuracy, precision, recall, and F-Measure, making it reliable for stress level

analysis techniques. The system is simple, user-friendly, and capable of providing quick prediction results with better reliability and efficiency. The preprocessing and feature selection techniques help improve the overall model performance and prediction accuracy. The proposed system can assist users in monitoring their stress conditions at an early stage and support better health management. Furthermore, the system can be enhanced in the future by integrating real-time health monitoring devices, mobile applications, and integrating real-time health monitoring devices, mobile applications, and cloud-based analytics to enable continuous tracking of stress levels. This would allow users to receive instant alerts and personalized recommendations based on their physiological and behavioral data.

ACKNOWLEDGEMENT

We express our sincere gratitude to our project guide Mr. D. Srinivas, for his valuable guidance, encouragement, and continuous support throughout the development of this project. His suggestions and technical knowledge helped us successfully complete the project work we would also like to thank the Head of the Department and all the faculty members of the Computer Science Engineering Department for providing the necessary facilities and support during the project development process. We extend our heartfelt thanks to our parents and friends for their motivation, encouragement, and support throughout the completion of this project. Finally, we express our gratitude to everyone who directly or indirectly contributed to the successful completion of this work.

REFERENCES

- [1] N. Schneiderman, G. Ironson, and S. D. Siegel, "Stress and health: Psychological, behavioral, and biological determinants," *Annu. Rev. Clin. Psychol.*, vol. 1, pp. 607–628, 2005, doi: 10.1146/annurev.clinpsy.1.102803.144141.
- [2] A. N. Vgontzas, S. Pejovic, and M. Karataraki, "Sleep, Sleep Disorders, and Stress," *Encycl. Stress*, pp. 506–514, 2007, doi: 10.1016/B978-012373947-6.00349-4.
- [3] G. Fink, "Stress : Concepts , Cognition , Emotion , and Behavior : Handbook of Stress Stress : Concepts , Definition , and History George Fink Florey Institute of Neuroscience and Mental Health," no. October, 2017, [Online]. Available: https://www.researchgate.net/profile/George-Fink/publication/317026245_Stress_Concepts_Cognition_Emotion_and_Behavior_Handbook_of_Stress/links/59d17f1b0f7e9b4fd7fa28b3/Stress-Concepts-Cognition-Emotion-and-Behavior-andbook-of-Stress.pdf

- [4] Q. Bukhsh, A. Shahzad, and M. Nisa, "A study of learning stress and stress management strategies of the students of postgraduate level: A case study of Islamia university of Bahawalpur, Pakistan," *Procedia - Soc. Behav. Sci.*, vol. 30, pp. 182–186, 2011, doi: 10.1016/j.sbspro.2011.10.036.
- [5] A. M. Shahsavarani, E. A. M. Abadi, and M. H. Kalkhoran, "Stress Assessment and Development of a Primary Care of Psychology Service," *Int. J. Med. Rev.*, vol. 2, no. 2, pp. 230–241, 2014.
- [6] A. Khademi, Y. El-Manzalawy, L. Master, O. M. Buxton, and V. G. Honavar, "Personalized sleep parameters estimation from actigraphy: A machine learning approach," *Nat. Sci. Sleep*, vol. 11, pp. 387–399, 2019, doi: 10.2147/NSS.S220716.
- [7] L. Rachakonda, A. K. Bapatla, S. P. Mohanty, and E. Kougianos, "SaYoPillow: Blockchain-Integrated Privacy-Assured IoMT Framework for Stress Management Considering Sleeping Habits," *IEEE Trans. Consum. Electron.*, vol. 67, no. 1, pp. 20–29, 2021, doi: 10.1109/TCE.2020.3043683.
- [8] V. C. Magana and M. Munoz-Organero, "Reducing stress on habitual journeys," *5th IEEE Int. Conf. Consum. Electron. - Berlin, ICCEBerlin 2015*, pp. 153–157, 2016, doi: 10.1109/ICCEBerlin.2015.7391220.
- [9] A. R. Subhani, W. Mumtaz, M. N. B. M. Saad, N. Kamel, and A. S. Malik, "Machine learning framework for the detection of mental stress at multiple levels," *IEEE Access*, vol. 5, no. c, pp. 13545–13556, 2017, doi: 10.1109/ACCESS.2017.2723622.
- [10] E. Garcia-Ceja, M. Riegler, T. Nordgreen, P. Jakobsen, K. J. Oedegaard, and J. Tørresen, "Mental health monitoring with multimodal sensing and machine learning: A survey," *Pervasive Mob. Comput.*, vol. 51, pp. 1–26, 2018, doi: 10.1016/j.pmcj.2018.09.003.
- [11] F. Akhtar, M. B. Bin Heyat, J. P. Li, P. K. Patel, Rishipal, and B. Guragai, "Role of Machine Learning in Human Stress: A Review," *2020 17th Int. Comput. Conf. Wavelet Act. Media Technol. Inf. Process. ICCWAMTIP 2020*, pp. 170–174, 2020, doi: 10.1109/ICCWAMTIP51612.2020.9317396.
- [12] R. K. Nath, H. Thapliyal, A. Caban-Holt, and S. P. Mohanty, "Machine Learning Based Solutions for Real-Time Stress Monitoring," *IEEE Consum. Electron. Mag.*, vol. 9, no. 5, pp. 34–41, 2020, doi: 10.1109/MCE.2020.2993427.
- [13] N. Keshan, P. V. Parimi, and I. Bichindaritz, "Machine learning for stress detection from ECG signals in automobile drivers," *Proc. – 2015 IEEE Int. Conf. Big Data, IEEE Big Data 2015*, pp. 2661–2669, 2015, doi: 10.1109/BigData.2015.7364066.
- [14] A. Muaremi, A. Bexheti, F. Gravenhorst, B. Arnrich, and G. Troster, "Monitoring the impact of stress on the sleep patterns of pilgrims using wearable sensors," *2014 IEEE-EMBS Int. Conf. Biomed. Heal. Informatics, BHI 2014*, pp. 185–188, 2014, doi: 10.1109/BHI.2014.6864335.
- [15] Kaggle, <https://www.kaggle.com/> (accessed: Feb. 27, 2022).

BIOGRAPHY



Mr. D. Srinivas is working as Assistant Professor at Guru Nanak Institute of Technology, Hyderabad. He completed B. Tech and M. Tech CSE .He has 15+ years of teaching experience. He has published research articles in 06 international journals. He attended 20 FDPs, 05 National level workshops. He filled 6 Indian Patents. He completed 6 ATAL Certificates on various emerging technologies topics. He completed 12 NPTEL Certificates. His research interests include Machine Learning and Deep learning and IoT. He has Qualified GATE with 78 percentile. He is pursuing Ph.D in Deep Learning with IOT.