**ABSTRACT**

This work utilizes wearable devices for real-time stress detection and investigates the effectiveness of meditation audio in reducing stress levels after academic exposure. Physiological data, including Interbeat Interval (IBI)-derived Heart Rate Variability (HRV), Blood Volume Pulse (BVP), and electrodermal activity (EDA), are collected during the Montreal Imaging Stress Task (MIST). The stress classification methodology employs an integrated approach using Genetic Algorithm and Mutual Information to reduce feature set redundancy. It further uses Bayesian optimization to fine-tune machine learning hyperparameters. The results indicate that the combination of EDA, BVP, and HRV achieves the highest classification accuracy of 98.28% and 97.02% using the Gradient Boosting (GB) algorithm for 2-level and 3-level stress classification. In contrast, EDA and HRV alone achieve a comparable accuracy of 97.07% and 95.23% for 2-level and 3-level stress classification, respectively. Furthermore, the SHAP Explainable AI (XAI) analysis confirms that HRV and EDA are the most significant features for stress classification. The study also finds evidence that listening to meditation audio reduces stress levels. These findings highlight the potential of wearable technology combined with machine learning for real-time stress monitoring and management in academic environments.

**Keywords** - Academic environment, Explainable AI, Feature Selection, Machine Learning, Mental stress, Wearable Device

**CHAPTER-01**

**INTRODUCTION**

Students frequently face stressful situations in today’s demanding academic environment, from class presentations to academic examinations. According to Prabu et al., [1], "academic stress is the psychological and emotional strain that arises from the demands and expectations associated with formal education and scholarly pursuits". This combination of rigorous academic workloads and competitive social dynamics can lead to significant mental stress, affecting their overall well-being and academic performance [2]. Studies have shown that unmanaged academic stress can lead to decreased motivation, anxiety, sleep problems, and even physical health issues [3] [4] [5]. Recognizing and addressing students’ challenges in managing academic stress is crucial, as this has become a critical aspect of their educational journey. However, the current focus in healthcare often lies in addressing and treating existing mental health issues and their associated symptoms. This approach is vital, but it is equally important to acknowledge the subtle indicators of stress and anxiety before they escalate. Recognizing these early signs

empowers students to take action and manage their stress effectively, potentially preventing the need for more intensive interventions later. To effectively identify these early signs of stress, re

searchers are increasingly emphasizing the relationship between variations in physiological signals and stress. When individuals undergo these physical and mental reactions to stress, their physiological signals also change [6] [7]. For example, Holtz et al. [8] introduced a sudden air horn sound and discovered that stress responses, such as an increase in Respiration Rate (RESP), Heart Rate (HR), and Electrodermal Activity (EDA) signals, occurred. Hence, it is vital to

understand the connections between variations in physiological signals and stress to identify stress effectively. Concurrently, there is a growing emphasis on using scientific and

technological methods to monitor and analyze physiological signals, making it a significant research area in academia and industrial settings. Wearable technology is promising for continuously and remotely monitoring mental health [9] [10]. These devices can gather physiological patient data and offer valuable contextual information. Additionally, the power of machine learning has enhanced data processing speed and the ability to derive meaningful insights from this physiological data. Recent advancements in machine learning have introduced ensemble techniques such as DeepAVP-TPPred [11], iAFPs-MvBiTCN [12], Deepstacked-AVPs [13], AIPs-SnTCN [14], and pAtbP-EnC [15], which have shown promising results in

various applications. These models precisely predict peptide functionality by extracting features from sequential and evolutionary descriptors through ensemble learning techniques. These advanced predictors offer robust frameworks for analyzing physiological signals, providing more accurate and early detection of stress and anxiety indicators. One notable category of wearable devices is smartwatches. These devices function like miniaturized smartphones, boasting impressive computational capabilities. Moreover, they come equipped with various sensors capable of collecting physiological signals such as EDA, Photoplethysmography (PPG), Electrocardiography (ECG), and Skin Temperature (ST). This combination of technology and sensors makes them a valuable tool for monitoring mental health remotely and continuously.

Moreover, researchers have employed various stressinducing tasks, including public speaking [16], questionnaires, programming contests [17], ice tasks, and games [18] while recording physiological signals. Some researchers have also delved into stress classification using logical thinking based tasks such as pursuing mental arithmetic and the Montreal Imaging Stress Task (MIST) test [19]. The MIST, a well-validated and progressive stressinducing test, has been extensively used for stress classification. It involves three distinct stages: resting, control, and

experimental. The participants solve a series of mathematical tasks without time constraints during the control stage. However, in the experimental stage, participants are presented with

challenging arithmetic problems that are time-constrained and aimed at inducing a higher stress level. Various researchers [20] [21] explored EEG-based stress classification experiments using MIST.

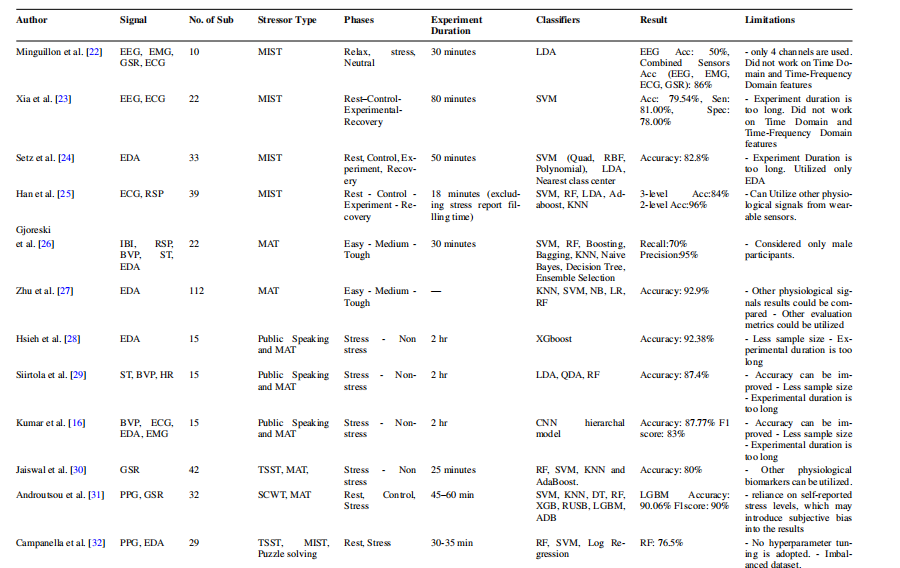
**CHAPTER- 02**  
 **LITERATURE SURVEY**

Minguillon et al.[22] proposed a portable system for real-time detection of stress levels. They utilized Electroencephalography (EEG), ECG, Electromyography (EMG), and Galvanic Skin Response (GSR) signals recorded during the MIST task. Using EEG alone, they achieved only 50% accuracy, but when combined with multimodal data, the accuracy increased to 86%. The results show the efficacy of multimodality, as it provides a comprehensive understanding of stress levels. Although EEG offers valuable insights into brain activity, it has some practical limitations, including noise susceptibility and discomfort, which hinder its everyday use. Wearable devices, especially smartwatches, emerge as a promising alternative for continuous, non-invasive monitoring of physiological signals like heart rate variability (HRV) and EDA. The unobtrusive nature of the signals makes them well-suited for real-time stress assessment in academic environents. studies during tasks like MIST and incorporates wearable devices to record physiological signals such as EEG, ECG, EDA, and RESP. Notably, numerous studies explored EDA signals for stress classification.

Setz et al. [24] employed EDA and MIST as a stressor. The authors investigated six classifiers, including SVM (linear, Quad, RBF, Polynomial), LDA (linear discriminant analysis), and KNN (K-nearest Neighbours). They concluded that monitoring EDA allows discrimination between cognitive load and stress with an accuracy larger than 80% with leave-one person-out cross-validation.

In another study, Zhu et al. [27] explored stress classification using EDA signal and MIST. They obtained the highest accuracy of 92.9% using the SVM classifier. Furthermore, Hsieh et al. [28] proposed a feature selection framework based on EDA signal. They conclude that the EDA signal contains the most significant information for stress detection and obtained the highest accuracy of 92.38%. Despite the extensive use of EDA in stress classification studies, there needs to be more literature regarding the combined use of ECG and EDA in conjunction with MIST. Han et al. [25] explored the combination of ECG and RESP with MIST for stress classification and obtained the highest accuracy of 96% using RF. However, RESP may be less reliable as a stress biomarker due to challenges in accurately measuring breath rate in real-life environments and its susceptibility to confounding factors beyond stress, such as physical exertion, respiratory illnesses, changes in posture, or environmental conditions [35]. To the best of our knowledge, in the context of academic learning, no research has specifically investigated the combination of ECG and EDA during MIST. ECG provides insight into autonomic nervous system activity through HRV features, while GSR reflects sympathetic nervous system arousal through skin conductance changes [36]. Stress activates the sympathetic nervous system, leading to increased EDA and decreased HRV. Therefore, changes in these signals can help detect stress when an individual encounters a stressor. EDA is more sensitive when capturing acute stress [37] [38]. In contrast, HRV provides a more stable measure of chronic stress [39]. Combining data from EDA and IBI-derived HRV enables the development of more precise and resilient stress detection systems.

TABLE 1. Previous work on MIST and MAT stressors using Physiological Signals



**CHAPTER-03**

**SYSTEM ANALYSIS**

**3.1 EXISTING SYSTEM**

The existing system utilizes various machine learning algorithms, including Random Forest (RF), Decision Trees (DT), Support Vector Machine (SVM), K-Nearest Neighbors (KNN), Extreme Gradient Boosting (XGB), and Gradient Boosting (GB) for stress classification based on physiological signals like Electrodermal Activity (EDA), Blood Volume Pulse (BVP), and Heart Rate Variability (HRV). Among these, GB provides the highest classification accuracy at 98.28% for 2-level classification and 97.02% for 3-level classification, demonstrating superior performance over other models. This system also employs a hybrid feature selection approach, combining Genetic Algorithm (GA) and Mutual Information (MI), to reduce feature redundancy and optimize performance. Bayesian optimization is used for fine-tuning hyperparameters, further enhancing model accuracy and efficiency.

While the existing system demonstrates high classification accuracy, it has limitations. It relies heavily on pre-selected physiological signals, which may not capture all aspects of stress under different scenarios. Additionally, the system’s interpretability remains limited, even though the use of SHAP for explainability has improved insight into key features like HRV and EDA. Lastly, while effective in structured testing environments, this system may not generalize well in uncontrolled real-world settings, such as daily academic activities.

**Disadvantages**

* The system is optimized for controlled testing conditions, potentially reducing accuracy in varied real-world environments.
* Relies mainly on HRV and EDA, potentially overlooking other influential stress indicators.
* GB may still limit explainability for end-users.

**3.2 PROPOSED SYSTEM**

The proposed system aims to address the limitations of existing heart disease prediction models by integrating a comprehensive set of advanced machine learning algorithms and expanding data inputs to improve accuracy, robustness, and real-time performance. Key algorithms employed include **Extra Trees (ET)**, which efficiently handle high-dimensional data and provide feature importance for better interpretability, and **AdaBoost**, an ensemble method that boosts weak learners to create a robust predictive model. The system also incorporates **K-Nearest Neighbors (KNN)** for its simplicity and effectiveness in handling small to medium-sized datasets, alongside the **Multi-Layer Perceptron (MLP)**, a neural network model that captures complex nonlinear relationships within the data.

To further enhance predictive performance, the proposed model utilizes **CatBoost**, **XGBoost (XGB)**, and **LightGBM**, which are gradient boosting-based algorithms known for their speed, efficiency, and ability to handle imbalanced data. CatBoost effectively handles categorical features, while XGBoost provides advanced regularization techniques to minimize overfitting, and LightGBM ensures scalability with large datasets through its leaf-wise growth strategy. Additionally, the **Random Forest (RF)** algorithm is employed for its ensemble learning capability, ensuring stability and reducing variance in predictions. For complex decision boundaries, **Support Vector Classifier (SVC)** is included, offering high accuracy in cases with small datasets and well-defined margins.

The system leverages these diverse algorithms to maximize predictive accuracy and adaptability across various clinical and real-world settings. By integrating multiple models, the ensemble approach balances bias and variance, ensuring consistent and reliable performance. Furthermore, **AutoML techniques** are utilized to automate model selection and hyperparameter tuning, optimizing each algorithm's performance without manual intervention. This ensures that the system adapts efficiently to diverse datasets and healthcare environments. By incorporating advanced algorithms, the system offers improved real-time detection capabilities, robust handling of data imbalances, and enhanced interpretability. Such a solution provides healthcare professionals with a highly accurate, data-driven tool for early heart disease risk prediction and decision-making across a range of dynamic clinical scenarios.

**Advantages**

• Advanced algorithms like XGBoost, LightGBM, and CatBoost ensure precise predictions by effectively handling complex, high-dimensional, and imbalanced clinical data.

· : Ensemble methods such as Random Forest and Extra Trees provide reliable and consistent results, reducing variance and enhancing model robustness.

· The system is optimized for speed and scalability, enabling seamless processing of large datasets and delivering real-time predictions for critical healthcare scenarios.

· Incorporating diverse data sources, including physiological and environmental factors, provides a holistic approach to risk prediction and decision-making.

· AutoML techniques automate model selection and tuning, ensuring adaptability to various datasets and healthcare environments with minimal manual intervention.

**3.3 FUNCTIONAL REQUIREMENTS**

The Functional requirements for a system describe the functionality or the services that the system is expected to provide. These are the statements of services the system should provide and how the system should react to particular inputs and how the system should behave in particular situation.

User Registartion: User Register with their Registration details.

User Login: User Login their account using password

Live Inputs: Inputs Given By the User requirement.

Load Model : Trained or Tested Model will be load .

Predict Output : Output will be predict based on parameters.

**3.4 NON-FUNCTIONAL REQUIREMENTS**

The non-functional requirements describe the system constraints.

**Performance**: The application should have better accuracy and should provide prediction in less time.

**Scalability:** The system must have the potential to be enlarged to accommodate the growth. **Capability:** The capability of the storage should be high so the large amount of data can be stored in order to train the model.

**3.5 FEASIBILITY ANALYSIS**

The feasibility of the project is analyzed in this phase and business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company. For feasibility analysis, some understanding of the major requirements for the system is essential. Three key considerations are involved in the feasibility analysis are:

* ECONOMICAL FEASIBILITY
* TECHNICAL FEASIBILITY
* SOCIAL FEASIBILITY

**3.5.1 ECONOMICAL FEASIBILITY**

This study is carried out to check the economic impact that the system will have on the organization. The amount of fund that the company can pour into the research and development of the system is limited. The expenditures must be justified. Thus the developed system as well within the budget and this was achieved because most of the technologies used are freely available. Only the customized products had to be purchased.

**3.5.2 TECHNICAL FEASIBILITY**

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. This will lead to high demands on the available technical resources. This will lead to high demands being placed on the client. The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system.

**3.5.3 SOCIAL FEASIBILITY**

The aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity. The level of acceptance by the users solely depends on the methods that are employed to educate the user about the system and to make him familiar with it. His level of confidence must be raised so that he is also able to make some constructive criticism, which is welcomed, as he is the final user of the system.

**CHAPTER-4**

**SYSTEM REQUIREMENTS**

**4.1 HARDWARE REQUIREMENTS:**

➢  **Processor**  - Pentium –IV

**➢ RAM** - 4 GB (min)

➢ **Hard Disk** - 20 GB

➢ **Key Board** - Standard Windows Keyboard

**➢ Mouse** - Two or Three Button Mouse

➢  **Monitor** - SVGA

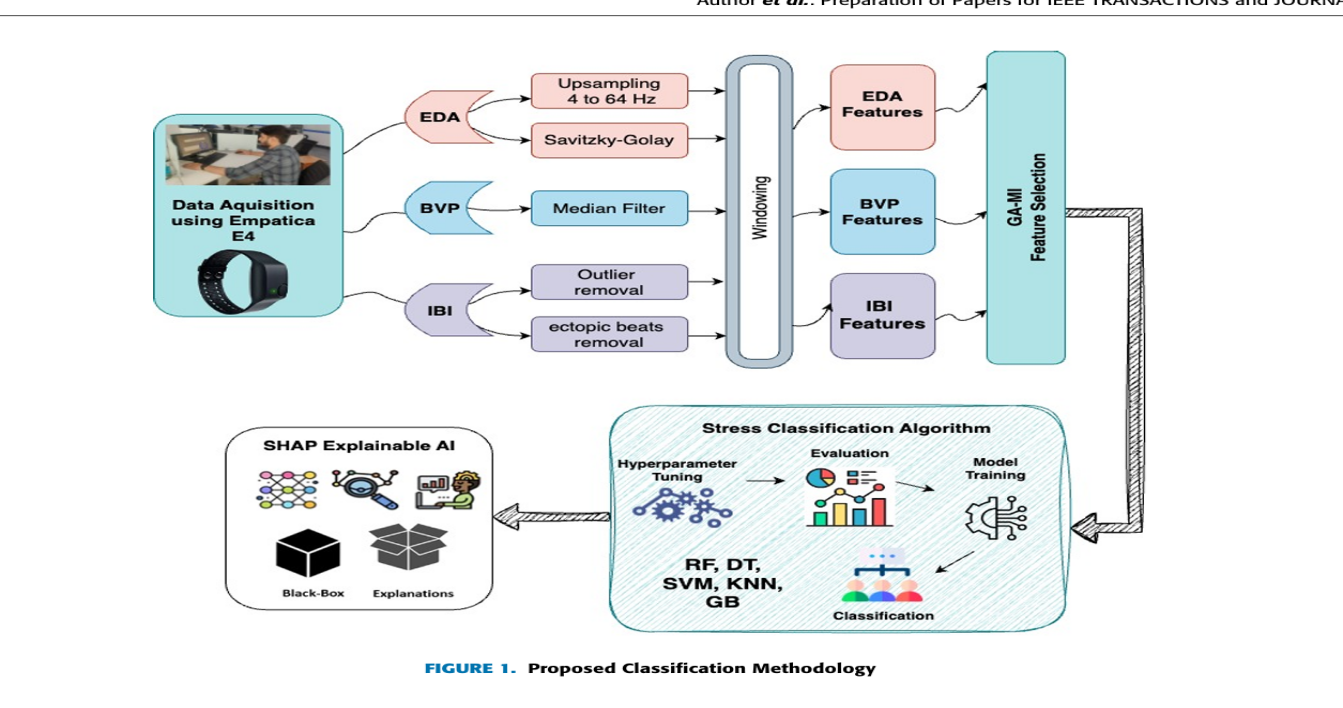
**4.2 SOFTWARE REQUIREMENTS:**

* **Operating system :** Windows 7 Ultimate.
* **Coding Language :** Python.
* **Front-End :** Flask.
* **Back-End :** Python
* **Designing :** Html, css, javascript.
* **Data Base :** MySQL (WAMP Server).

**CHAPTER-05**

**SOFTWARE DESIGN**

**5.1 SYSTEM ARCHITECTURE:**



**Fig 5.1 Proposed block diagram.**

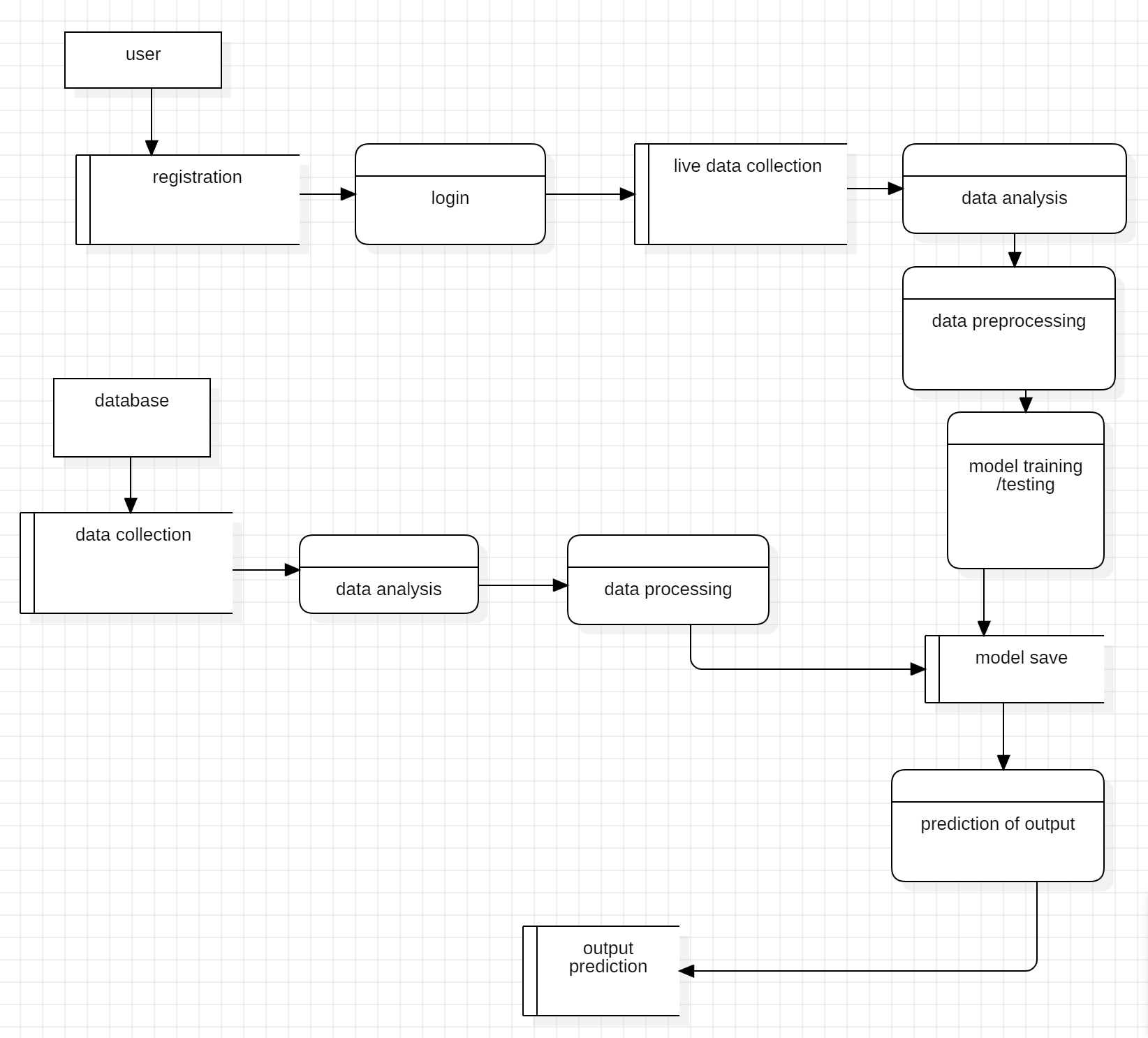
**DATA FLOW DIAGRAM:**

1. The DFD is also called as bubble chart. It is a simple graphical formalism that can be used to represent a system in terms of input data to the system, various processing carried out on this data, and the output data is generated by this system.

2. The data flow diagram (DFD) is one of the most important modeling tools. It is used to model the system components. These components are the system process, the data used by the process, an external entity that interacts with the system and the information flows in the system.

3. DFD shows how the information moves through the system and how it is modified by a series of transformations. It is a graphical technique that depicts information flow and the transformations that are applied as data moves from input to output.

4. DFD is also known as bubble chart. A DFD may be used to represent a system at any level of abstraction. DFD may be partitioned into levels that represent increasing information flow and functional detail.



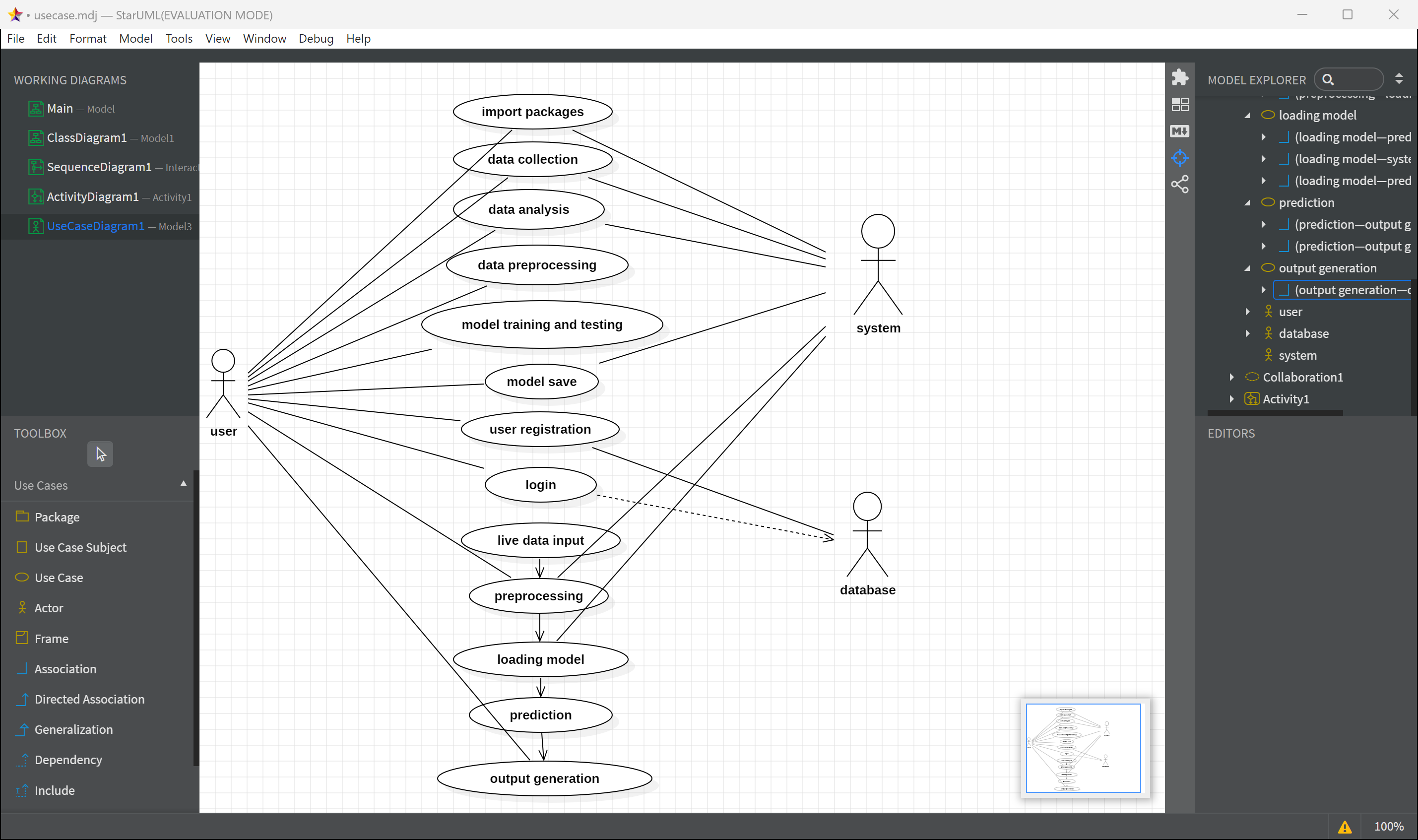
**Fig 5.2 Data Flow diagram**

**5.2 UNIFIED MODELLING LANGUAGE DIAGRAMS**

UML is a method for describing the system architecture in detail using the blue print. UML represents a collection of best engineering practice that has proven successful in the modeling of large and complex systems. The UML is very important parts of developing object-oriented software and the software development process. The UML uses mostly graphical notations to express the design of software projects. Using the helps UML helps project teams communicate explore potential designs and validate the architectural design of the software.

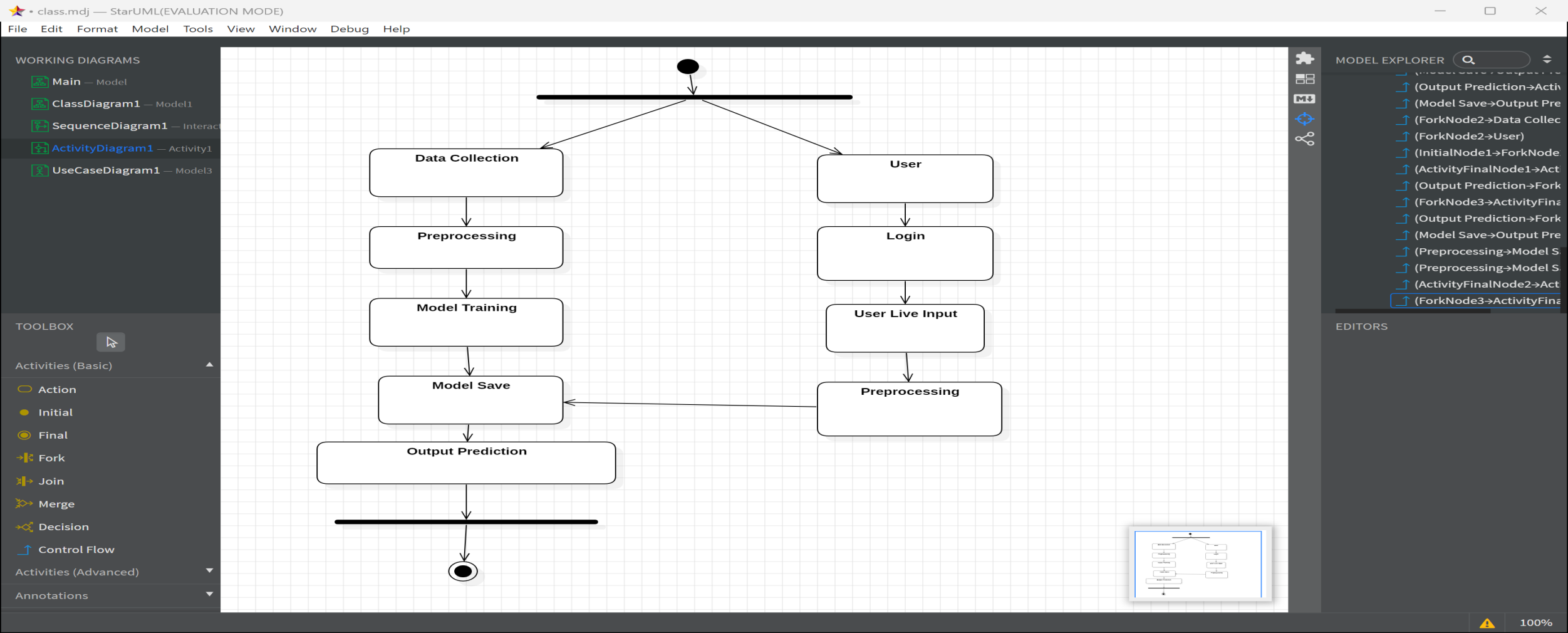
**5.2.1 USE CASE DIAGRAM**

A use case diagram in the Unified Modeling Language (UML) is a type of behavioral diagram defined by and created from a Use-case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases. 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.



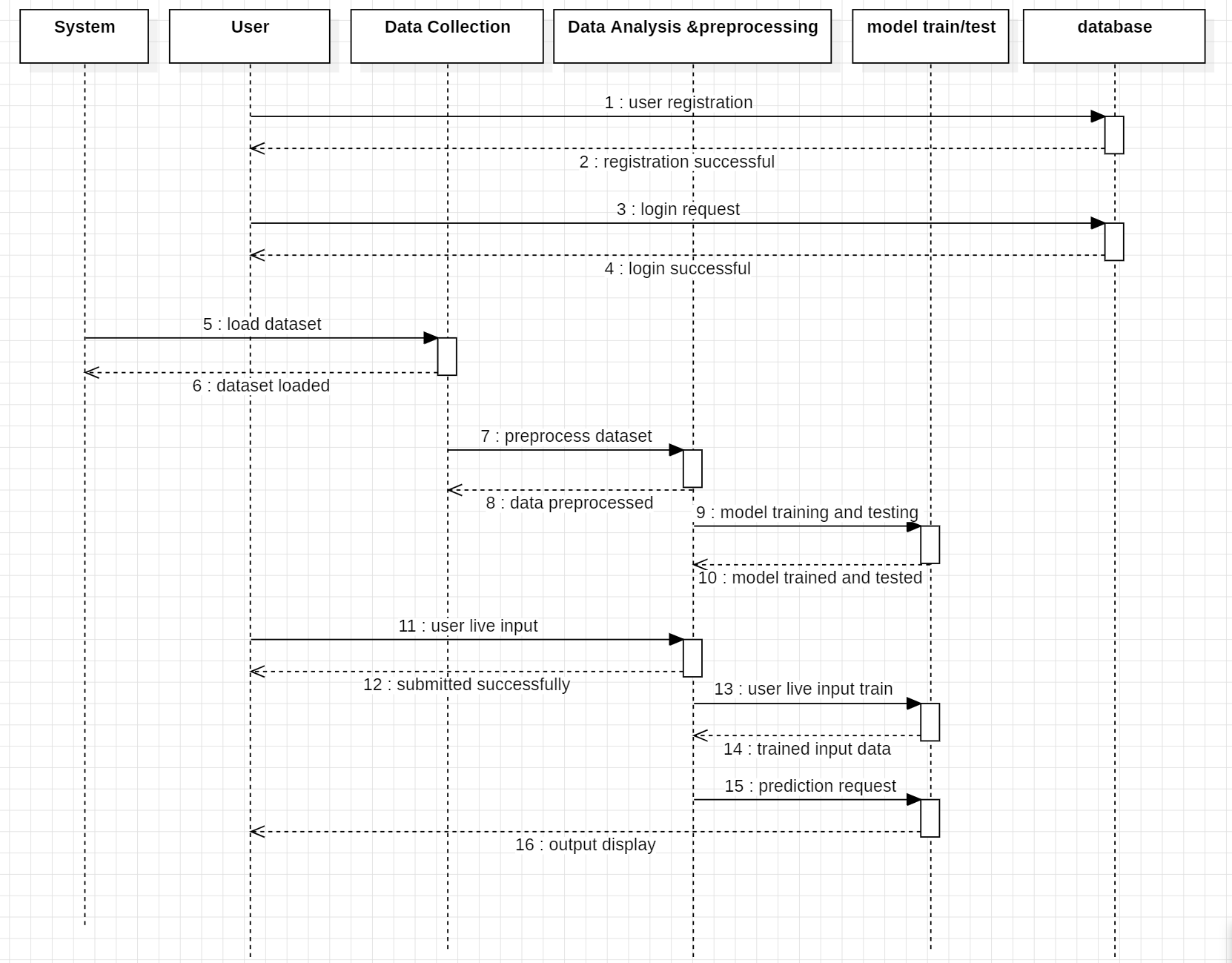
**5.1.2 ACTIVITY DIAGRAM**

Activity diagrams are graphical representations of work flows 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 work flows of components in a system. An activity diagram shows the overall flow of control.



**5.1.3 SEQUENCE DIAGRAM**

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. Sequence diagrams are sometimes called event diagrams, event scenarios, and timing diagram



**5.1.4 CLASS DIAGRAM**

In software engineering, a class diagram in the Unified Modeling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among the classes. It explains which class contains information.

A screenshot of a computer

Description automatically generated

**CHAPTER -06**

**SYSTEM IMPLEMENTATION**

The methodology for the proposed system starts with **data acquisition**, which utilizes the Empatica E4 wearable device to collect raw physiological signals. These signals include Electrodermal Activity (EDA), Blood Volume Pulse (BVP), and Inter-Beat Interval (IBI), all of which are crucial in understanding physiological responses associated with stress. The Empatica E4 is a sophisticated device designed to provide high-resolution data, forming the basis for accurate and reliable stress detection. These raw signals, however, are prone to noise and artifacts, making preprocessing a vital step before moving toward feature extraction and analysis.

In the **signal preprocessing phase**, each signal undergoes specific cleaning and smoothing techniques tailored to its unique characteristics. The EDA signal is upsampled to 4 Hz to standardize the sampling frequency, and the Savitzky-Golay filter is applied to smooth the signal while retaining essential features. Similarly, the BVP signal is cleaned using a median filter to remove noise and ensure that heart-rate-related metrics are precise. The IBI signal undergoes outlier and ectopic beat removal, which helps eliminate irregularities caused by noise or transient physiological states, thereby enhancing the accuracy of heart rate variability measurements. These preprocessing steps ensure that the data is of high quality and suitable for feature extraction.

The system then transitions to **feature extraction and classification**. For each preprocessed signal, specific features are computed—EDA features capture patterns such as amplitude and frequency, BVP features focus on metrics like heart rate variability, and IBI features represent trends in heartbeat intervals. These extracted features are combined into a unified global feature space, offering a holistic view of physiological responses. Machine learning algorithms, including Random Forest (RF), Decision Tree (DT), Support Vector Machine (SVM), K-Nearest Neighbors (KNN), and Gradient Boosting (GB), are employed for stress classification. Key steps like hyperparameter tuning and model evaluation ensure the models are optimized for high performance. Additionally, the system incorporates SHAP (SHapley Additive exPlanations) to provide interpretability, making the predictions of these black-box models more transparent. This integration of advanced preprocessing, feature engineering, machine learning, and explainable AI creates a robust and reliable framework for stress classification.

**6.1 MODULES**Load Data   
Data collection   
Data pre-processing   
Feature Selection   
Feature Extraction   
Machine Learning

**6.1.1 LOAD DATA:**

Pandas allows you to import data from a wide range of data sources directly into a dataframe. These can be static files, such as CSV, TSV, fixed width files, Microsoft Excel, JSON, SAS and SPSS files, as well as a range of popular databases, such as MySQL, PostgreSQL and Google BigQuery. You can even scrape data directly from web pages into Pandas dataframes.

**6.1.2 DATA COLLECTION :**

Data collection means pooling data by scraping, capturing, and loading it from multiple sources, including offline and online sources. High volumes of data collection or data creation can be the hardest part of a machine learning project, especially at scale.Data collection allows you to capture a record of past events so that we can use data analysis to find recurring patterns. From those patterns, you build predictive models using machine learning algorithms that look for trends and predict future changes.Predictive models are only as good as the data from which they are built, so good data collection practices are crucial to developing high-performing models. The data needs to be error-free and contain relevant information for the task at hand. For example, a loan default model would not benefit from tiger population sizes but could benefit from gas prices over time.

**6.1.3 DATA PRE-PROCESSING :**

Data preprocessing is a process of preparing the raw data and making it suitable for a machine learning model. It is the first and crucial step while creating a machine learning model. When creating a machine learning project, it is not always a case that we come across the clean and formatted data. And while doing any operation with data, it is mandatory to clean it and put in a formatted way. So for this, we use data preprocessing task.

A real-world data generally contains noises, missing values, and maybe in an unusable format which cannot be directly used for machine learning models. Data preprocessing is required tasks for cleaning the data and making it suitable for a machine learning model which also increases the accuracy and efficiency of a machine learning model.

It involves below steps:

* Getting the dataset
* Importing libraries
* Importing datasets
* Finding Missing Data
* Encoding Categorical Data
* Splitting dataset into training and test set
* Feature scaling

**6.1.4 FEATURE SELECTION:**

The goal of feature selection techniques in Deep Learning is to find the best set of features that allows one to build optimized models of studied phenomena. The techniques for feature selection in Deep Learning can be broadly classified into the following categories: Supervised Techniques: These techniques can be used for labeled data and to identify the relevant features for increasing the efficiency of supervised models like classification and regression. For Example- linear regression, decision tree, SVM, etc. Unsupervised Techniques: These techniques can be used for unlabeled data. For Example- K-Means Clustering, Principal Component Analysis, Hierarchical Clustering, etc. From a taxonomic point of view, these techniques are classified into filter, wrapper, embedded, and hybrid methods

**6.1.5 FEATURE EXTRACTION:**

Feature extraction is a part of the dimensionality reduction process, in which, an initial set of the raw data is divided and reduced to more manageable groups. So when you want to process it will be easier. The most important characteristic of these large data sets is that they have a large number of variables. These variables require a lot of computing resources to process. So Feature extraction helps to get the best feature from those big data sets by selecting and combining variables into features, thus, effectively reducing the amount of data. These features are easy to process, but still able to describe the actual data set with accuracy and originality. Color features are obtained by extracting statistical features from image histograms. They are used to provide a general description of color statistics in the image.

**6.1.6 MACHINE LEARNING**

.Machine Learning is one of the booming technologies across the world that enables computers/machines to turn a huge amount of data into predictions. However, these predictions highly depend on the quality of the data, and if we are not using the right data for our model, then it will not generate the expected result. In machine learning projects, we generally divide the original dataset into training data and test data. We train our model over a subset of the original dataset, i.e., the training dataset, and then evaluate whether it can generalize well to the new or unseen dataset or test set. Therefore, train and test datasets are the two key concepts of machine learning, where the training dataset is used to fit the model, and the test dataset is used to evaluate the model.

**Training Dataset**

The training data is the biggest (in -size) subset of the original dataset, which is used to train or fit the machine learning model. Firstly, the training data is fed to the ML algorithms, which lets them learn how to make predictions for the given task.

**Test Dataset**

Once we train the model with the training dataset, it's time to test the model with the test dataset. This dataset evaluates the performance of the model and ensures that the model can generalize well with the new or unseen dataset. **The test dataset is another subset of original data, which is independent of the training dataset.** However, it has some similar types of features and class probability distribution and uses it as a benchmark for model evaluation once the model training is completed. Test data is a well-organized dataset that contains data for each type of scenario for a given problem that the model would be facing when used in the real world. Usually, the test dataset is approximately 20-25% of the total original data for an ML project.

**RANDOM FOREST ALGORITHM:**

Random Forest is a popular machine learning algorithm that belongs to the supervised learning technique. It can be used for both Classification and Regression problems in ML. It is based on the concept of ensemble learning, which is a process of combining multiple classifiers to solve a complex problem and to improve the performance of the model.

As the name suggests, "Random Forest is a classifier that contains a number of decision trees on various subsets of the given dataset and takes the average to improve the predictive accuracy of that dataset." Instead of relying on one decision tree, the random forest takes the prediction from each tree and based on the majority votes of predictions, and it predicts the final output.

The greater number of trees in the forest leads to higher accuracy and prevents the problem of overfitting.

**K-Nearest Neighbors (KNN) algorithm**

The **K-Nearest Neighbors (KNN) algorithm** is a supervised machine learning method employed to tackle classification and regression problems. Evelyn Fix and Joseph Hodges developed this algorithm in 1951, which was subsequently expanded by Thomas Cover. The article explores the fundamentals, workings, and implementation of the KNN algorithm.It is widely disposable in real-life scenarios since it is non-parametric, meaning it does not make any underlying assumptions about the distribution of data (as opposed to other algorithms such as GMM, which assume a [Gaussian distribution](https://www.geeksforgeeks.org/mathematics-probability-distributions-set-3-normal-distribution) of the given data). We are given some prior data (also called training data), which classifies coordinates into groups identified by an attribute.(K-NN) algorithm is a versatile and widely used machine learning algorithm that is primarily used for its simplicity and ease of implementation. It does not require any assumptions about the underlying data distribution. It can also handle both numerical and categorical data, making it a flexible choice for various types of datasets in classification and regression tasks. It is a non-parametric method that makes predictions based on the similarity of data points in a given dataset. K-NN is less sensitive to outliers compared to other algorithms. The K-NN algorithm works by finding the K nearest neighbors to a given data point based on a distance metric, such as Euclidean distance. The class or value of the data point is then determined by the majority vote or average of the K neighbors. This approach allows the algorithm to adapt to different patterns and make predictions based on the local structure of the data.

**CatBoost (Categorical Boosting):**

CatBoost, developed by Yandex, is a high-performance gradient boosting algorithm designed specifically for handling categorical data. It eliminates the need for extensive preprocessing, such as one-hot encoding or label encoding, by natively supporting categorical features. This feature reduces data preparation time and simplifies model building. One of its key innovations is **Ordered Boosting**, which reduces overfitting by ensuring that leaf values are calculated on a different subset of data than the one used for constructing the tree. Additionally, **Symmetric Trees** are another highlight, where the splits at each tree level occur simultaneously across all branches, ensuring balanced and efficient tree growth. CatBoost works well with minimal hyperparameter tuning, as its default settings are optimized for most tasks. It incorporates advanced regularization techniques to improve generalization and prevent overfitting, making it robust in real-world applications. The algorithm is also optimized for speed and scalability, supporting both CPU and GPU acceleration for faster training on large datasets. CatBoost is widely used for **classification** and **regression** problems in domains such as finance, fraud detection, recommendation systems, and e-commerce. Its ability to handle both numerical and categorical features efficiently makes it a preferred choice for many machine learning practitioners.

### ****XGBoost (Extreme Gradient Boosting):****

XGBoost, developed by Tianqi Chen, is an optimized and scalable implementation of gradient boosting designed for speed and performance. It incorporates both **L1** and **L2 regularization** to control model complexity and reduce overfitting, making it highly effective for predictive tasks. XGBoost supports parallel tree boosting, which speeds up computations, and it is capable of handling missing values effectively without requiring imputation. Its **gradient-based approach** combined with **pruning techniques** ensures optimal tree construction while avoiding overfitting. XGBoost provides a variety of boosting methods, including gradient boosting, stochastic boosting, and dart boosting (Dropouts meet Multiple Additive Regression Trees), enabling flexibility for various applications. It can handle both **classification** and **regression** problems and is also efficient for ranking and survival analysis tasks. XGBoost is highly customizable with parameters like learning rate, tree depth, and subsampling ratios, making it ideal for large-scale datasets. Its success in machine learning competitions, such as Kaggle, highlights its superior accuracy and speed. Due to its scalability, XGBoost has found applications in industries like healthcare, finance, recommendation systems, and anomaly detection.

### **Extra Trees (Extremely Randomized Trees):**

Extra Trees, short for Extremely Randomized Trees, is an ensemble learning technique that builds multiple decision trees to improve accuracy and robustness. Unlike standard random forests, Extra Trees introduces additional randomness during the tree-building process. In random forests, splits are chosen based on the best feature and threshold, while in Extra Trees, the thresholds are selected **randomly** for each feature. This randomness reduces variance and overfitting, especially on noisy datasets, while maintaining computational efficiency. Extra Trees are highly parallelizable, making them scalable for large datasets. By combining predictions from multiple de-correlated trees, Extra Trees improve generalization and deliver high accuracy. The algorithm supports both **classification** and **regression** tasks and works well for datasets with high-dimensional or irrelevant features. Extra Trees are less sensitive to hyperparameter tuning, as they are robust to overfitting due to their high randomness. They are faster to train compared to gradient boosting methods since they do not rely on gradient optimization. Extra Trees are widely used in various domains, including bioinformatics, anomaly detection, and predictive modeling, where accuracy and speed are critical.

### **LightGBM (Light Gradient Boosting Machine):**

LightGBM, developed by Microsoft, is an advanced gradient boosting framework specifically designed for speed and efficiency on large datasets. It is optimized for scenarios with **high-dimensional data** and large-scale training. LightGBM introduces two key innovations: **Gradient-based One-Side Sampling (GOSS)** and **Exclusive Feature Bundling (EFB)**. GOSS reduces computation time by selecting only a subset of gradients with higher magnitudes for tree splitting, while EFB bundles mutually exclusive features to reduce dimensionality. LightGBM constructs trees using a **leaf-wise splitting strategy** instead of the traditional level-wise approach, which allows it to grow deeper trees and achieve better accuracy. This technique ensures that more informative splits are chosen, resulting in faster convergence. LightGBM is highly memory-efficient, making it suitable for devices with limited resources. It supports GPU acceleration for even faster training and is widely used in machine learning competitions and real-world applications such as finance, fraud detection, and recommendation systems. LightGBM is highly customizable with parameters like learning rate, max depth, and feature fraction, providing flexibility for different tasks. Its ability to handle large-scale, sparse data makes it one of the fastest and most accurate boosting frameworks available today.

**Gradient Boosting**:  
Gradient Boosting is a powerful ensemble technique used for both regression and classification tasks. It builds models sequentially in an additive manner by combining weak learners, usually decision trees, to create a strong model. Each new tree is trained to correct the errors made by the previous tree using the gradient of a loss function, which minimizes the prediction error. Gradient Boosting is highly flexible, as it allows the use of custom loss functions and handles overfitting using techniques like regularization and learning rate adjustments. It performs well on structured/tabular data but can be computationally expensive with large datasets.

**AdaBoost (Adaptive Boosting)**:  
AdaBoost is another ensemble method that combines multiple weak learners, such as shallow decision trees (stumps), into a strong classifier. It assigns weights to each training instance and iteratively adjusts them so that misclassified samples get higher weights in the next iteration. This focuses the model on the harder-to-classify examples. The final model is a weighted sum of all weak learners, giving more importance to the better-performing learners. AdaBoost is simple, effective, and less prone to overfitting on clean datasets, but its performance can degrade with noisy data or outliers.

**NAIVE BAYES CLASSIFIER ALGORITHM:**

⦁ Naïve Bayes algorithm is a supervised learning algorithm, which is based on Bayes theorem and used for solving classification problems.

⦁ It is mainly used in text classification that includes a high-dimensional training dataset.

⦁ Naïve Bayes Classifier is one of the simple and most effective Classification algorithms which helps in building the fast machine learning models that can make quick predictions.

⦁ It is a probabilistic classifier, which means it predicts on the basis of the probability of an object.

⦁ Some popular examples of Naïve Bayes Algorithm are spam filtration, Sentimental analysis, and classifying articles.

**DECISION TREE CLASSIFICATION ALGORITHM :**

⦁ Decision Tree is a Supervised learning technique that can be used for both classification and Regression problems, but mostly it is preferred for solving Classification problems. It is a tree-structured classifier, where internal nodes represent the features of a dataset, branches represent the decision rules and each leaf node represents the outcome.

⦁ In a Decision tree, there are two nodes, which are the Decision Node and Leaf Node. Decision nodes are used to make any decision and have multiple branches, whereas Leaf nodes are the output of those decisions and do not contain any further branches.

⦁ The decisions or the test are performed on the basis of features of the given dataset.

⦁ It is a graphical representation for getting all the possible solutions to a problem/decision based on given conditions.

⦁ It is called a decision tree because, similar to a tree, it starts with the root node, which expands on further branches and constructs a tree-like structure.

⦁ In order to build a tree, we use the CART algorithm, which stands for Classification and Regression Tree algorithm.

⦁ A decision tree simply asks a question, and based on the answer (Yes/No), it further splits the tree into subtrees.

**SUPPORT VECTOR MACHINE ALGORITHM :**

Support Vector Machine or SVM is one of the most popular Supervised Learning algorithms, which is used for Classification as well as Regression problems. However, primarily, it is used for Classification problems in Machine Learning.

The goal of the SVM algorithm is to create the best line or decision boundary that can segregate n-dimensional space into classes so that we can easily put the new data point in the correct category in the future. This best decision boundary is called a hyperplane.

SVM chooses the extreme points/vectors that help in creating the hyperplane. These extreme cases are called as support vectors, and hence algorithm is termed as Support Vector Machine. Consider the below diagram in which there are two different categories that are classified using a decision boundary or hyperplane.

**6.1.7 MODEL SELECTION IN MACHINE LEARNING:**

Model selection in machine learning is the process of selecting the best algorithm and model architecture for a specific job or dataset. It entails assessing and contrasting various models to identify the one that best fits the data & produces the best results. Model complexity, data handling capabilities, and generalizability to new examples are all taken into account while choosing a model. Models are evaluated and contrasted using methods like cross-validation, and grid search, as well as indicators like accuracy and mean squared error. Finding a model that balances complexity and performance to produce reliable predictions and strong generalization abilities is the aim of model selection.

**6.2. TECHNOLGIES**

6.2.1 PYTHON

6.2.2 FLASK

**6.2.1 Python**

Python is a highly interpreted programming language Python provides man GUI development possibilities (Graphical User Interface). flask is, the most frequently used technique of all GUI methods. It's a standard Python interface to the Python Tk GUI toolkit.

Python is the quickest and simplest method for creating GUI apps using Flask outputs. It is a simple job to create a GUI using flask. Python is a common, flexible and popular language of programming.

It is excellent as a first language since it is succinet and simple to understand and also good to use in any programmer's pile because it can be utilized from development of the web to software. It's basic, easy-to-use grammar, making it the ideal language to first learn computer programming.

Most implementations of Python (including C and Python), include a read- eval-print (REPL) loop that enables the user to act as a command-line interpreter that results in sequence and instantaneous intake of instructions. Other shells like as IDLE and Python provide extra features such as auto-completion, session retention and highlighting of syntax.

**Interactive mode programming**

Invoking the interpreter without passing a script file as a parameter brings up the following prompt

− $ python

Python 2.4.3 (#1, Nov 11 2010, 13:34:43)

[GCC 4.1.2 20080704 (Red Hat 4.1.2-48)] on linux2

Type "help", "copyright", "credits" or "license" for more information

Type the following text at the Python prompt and press the Enter –

>>> print "Hello, Python!" If you are running new version of Python, then you would need to use print statement with parenthesis as in print ("Hello, Python!");. However in Python version 2.4.3, this produces the following result

− Hello,

**Script mode programming**

Invoking the interpreter with a script parameter begins execution of the script and continues until the script is finished. When the script is finished, the interpreter is no longer active.

Let us write a simple Python program in a script. Python files have extension .py. Type the following source code in a test.py file –

Live Demo print "Hello, Python!" We assume that you have Python interpreter set in PATH variable. Now, try to run this program as follows –

$ python test.py This produces the following result –

Hello, Python! Let us try another way to execute a Python script. Here is the modified test.py file –

Live Demo

#!/usr/bin/python print "Hello, Python!"

We assume that you have Python interpreter available in /usr/bin directory. Now, try to run this program as follows –

$ chmod +x test.py # This is to make file executable

$./test.py

This produces the following result –

Hello, Python!

**6.2.2 Flask web framework**

Flask is a web application framework written in Python. Armin Ronacher, who leads an international group of Python enthusiasts named Pocco, develops it. Flask is based on Werkzeug WSGI toolkit and Jinja2 template engine. Both are Pocco projects.Unlike the Django framework, Flask is very Pythonic. It’s easy to get started with Flask, because it doesn’t have a huge learning curve.

On top of that it’s very explicit, which increases readability.

It is classified as a microframework because it does not require particular tools or libraries. It has no database abstraction layer, form validation, or any other components where pre- existing third-party libraries provide common functions. However, Flask supports extensions that can add application features as if they were implemented in Flask itself. Extensions exist for object-relational mappers, form validation, upload handling, various open authentication technologies and several common framework related tools

**CHAPTER -07**

**CODING**

**CHAPTER-08**

**OUTPUT SCREENS**

**CHAPTER -09**

**TESTING**

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub assemblies, assemblies and/or a finished product It is the process of exercising software with the intent of ensuring that the Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

**TYPES OF TESTS**

**Unit testing**

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

**Integration testing**

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfaction, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

**Functional test**

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centered on the following items:

Valid Input : identified classes of valid input must be accepted.

Invalid Input : identified classes of invalid input must be rejected.

Functions : identified functions must be exercised.

Output : identified classes of application outputs must be exercised. Systems/Procedures : interfacing systems or procedures must be invoked.

Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined. System Test System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

**White Box Testing**

White Box Testing is a testing in which in which the software tester has knowledge of the inner workings, structure and language of the software, or at least its purpose. It is purpose. Itis used to test areas that cannot be reached from a black box level.

**Black Box Testing**

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or requirements document, such as specification or requirements document. It is a testing in which the software under test is treated, as a black box .you cannot “see” into it. The test provides inputs and responds to outputs without considering how the software works.

**Unit Testing**

Unit testing is usually conducted as part of a combined code and unit test phase of the software lifecycle, although it is not uncommon for coding and unit testing to be conducted as two distinct phases.

**Test strategy and approach**

Field testing will be performed manually and functional tests will be written in detail.

**Test objectives**

∙ All field entries must work properly.

∙ Pages must be activated from the identified link.

∙ The entry screen, messages and responses must not be delayed.

**Features to be tested**

∙ Verify that the entries are of the correct format

∙ No duplicate entries should be allowed

∙ All links should take the user to the correct page.

**Integration Testing**

Software integration testing is the incremental integration testing of two or more integrated software components on a single platform to produce failures caused by interface defects.

The task of the integration test is to check that components or software applications, e.g. components in a software system or – one step up – software applications at the company level – interact without error.

**Test Results:**

All the test cases mentioned above passed successfully. No defects encountered.

**Acceptance Testing**

User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements.

**Test Results:**

All the test cases mentioned above passed successfully. No defects encountered.

**CHAPTER-10**

**CONCLUSION**

In conclusion, the optimization of wearable biosensor data through the application of machine learning and explainable AI offers transformative possibilities for stress classification and management, particularly in academic environments. By effectively integrating diverse environmental and physiological factors, this approach enables a more comprehensive understanding of stress triggers and responses. The inclusion of advanced explainability features ensures that predictions and insights are not only accurate but also interpretable, fostering trust and facilitating actionable decisions. Furthermore, the system’s adaptability to varied conditions makes it highly versatile, allowing for reliable performance across different settings and populations.

This innovative methodology demonstrates immense potential in promoting mental well-being by providing timely and precise stress detection, ultimately supporting improved academic performance. Additionally, its ability to deliver personalized interventions tailored to individual needs underscores its relevance in addressing the unique challenges faced by students and educators. Moving forward, future research should prioritize large-scale implementations to validate the system’s effectiveness in real-world scenarios. Longitudinal studies can provide deeper insights into stress patterns over time, enhancing the system’s robustness and utility. Exploring additional applications in education and healthcare can further expand its impact, paving the way for more holistic and data-driven approaches to mental health management and performance optimization.

**CHAPTER-12**

**REFERENCE**

[1] P. S. Prabu, ‘‘A study on academic stress among higher secondary students,’’ International journal of humanities and social science invention, vol. 4, no. 10, pp. 63–68, 2015.

[2] S. D. Ghatol, Academic Stress among School Students. Allied Publishers, 2019.

[3] A. Waqas, S. Khan, W. Sharif, U. Khalid, and A. Ali, ‘‘Association of academic stress with sleeping difficulties in medical students of a Pakistani medical school: a cross sectional survey,’’ PeerJ, vol. 3, p. e840, 2015.

[4] V. M. Bhujade, ‘‘Depression, anxiety and academic stress among college students: A brief review.,’’ Indian Journal of Health & Wellbeing, vol. 8, no. 7, 2017.

[5] S. Maji, A. Chaturmohta, D. Deevela, S. Sinha, S. Tarsolia, and A. Barsaiya, ‘‘Mental health consequences of academic stress, amotivation, and coaching experience: A study of india’s top engineering undergraduates,’’ Psychology in the Schools.

[6] S. Byun, A. Y. Kim, E. H. Jang, S. Kim, K. W. Choi, H. Y. Yu, and H. J. Jeon, ‘‘Detection of major depressive disorder from linear and nonlinear heart rate variability features during mental task protocol,’’ Computers in biology and medicine, vol. 112, p. 103381, 2019.

[7] M. Jafari, A. Shoeibi, M. Khodatars, S. Bagherzadeh, A. Shalbaf, D. L. García, J. M. Gorriz, and U. R. Acharya, ‘‘Emotion recognition in EEG signals using deep learning methods: A review,’’ Computers in Biology and Medicine, p. 107450, 2023.

[8] L. Holtz, M. Martinez, K. Paton, K. Rosich, and E. Schnittka, ‘‘Effects of physiological stress response on short-term memory recall,’’ 2017.

[9] C. Anders and B. Arnrich, ‘‘Wearable electroencephalography and multimodal mental state classification: A systematic literature review,’’ Computers in Biology and Medicine, p. 106088, 2022.

[10] A. Arsalan and M. Majid, ‘‘Human stress classification during public speaking using physiological signals,’’ Computers in biology and medicine, vol. 133, p. 104377, 2021.

[11] M. Ullah, S. Akbar, A. Raza, and Q. Zou, ‘‘DeepAVP-TPPred: identification of antiviral peptides using transformed image-based localized descriptors and binary tree growth algorithm,’’ Bioinformatics, vol. 40, no. 5, p. btae305, 2024.

[12] S. Akbar, Q. Zou, A. Raza, and F. K. Alarfaj, ‘‘iAFPs-Mv-BiTCN: Predicting antifungal peptides using self-attention transformer embedding and transform evolutionary based multi-view features with bidirectional temporal convolutional networks,’’ Artificial Intelligence in Medicine, vol. 151, p. 102860, 2024.

[13] S. Akbar, A. Raza, and Q. Zou, ‘‘Deepstacked-AVPs: predicting antiviral peptides using tri-segment evolutionary profile and word embedding based multi-perspective features with deep stacking model,’’ BMC bioinformatics, vol. 25, no. 1, p. 102, 2024.

[14] A. Raza, J. Uddin, A. Almuhaimeed, S. Akbar, Q. Zou, and A. Ahmad, ‘‘AIPs-SnTCN: Predicting anti-inflammatory peptides using fasttext and transformer encoder-based hybrid word embedding with self-normalized temporal convolutional networks,’’ Journal of chemical information and modeling, vol. 63, no. 21, pp. 6537–6554, 2023.

[15] S. Akbar, A. Raza, T. Al Shloul, A. Ahmad, A. Saeed, Y. Y. Ghadi, O. Mamyrbayev, and E. T. Eldin, ‘‘pAtbP-EnC: identifying anti-tubercular peptides using multi-feature representation and genetic algorithm based deep ensemble model,’’ IEEE Access, 2023.

[16] A. Kumar, K. Sharma, and A. Sharma, ‘‘Hierarchical deep neural network for mental stress state detection using IoT based biomarkers,’’ Pattern Recognition Letters, vol. 145, pp. 81–87, 2021.

[17] Y. S. Can, N. Chalabianloo, D. Ekiz, and C. Ersoy, ‘‘Continuous stress detection using wearable sensors in real life: Algorithmic programming contest case study,’’ Sensors, vol. 19, no. 8, p. 1849, 2019.

[18] B. Egilmez, E. Poyraz, W. Zhou, G. Memik, P. Dinda, and N. Alshurafa, ‘‘UStress: Understanding college student subjective stress using wristbased passive sensing,’’ in 2017 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops), pp. 673–678, IEEE, 2017.

[19] K. Dedovic, R. Renwick, N. K. Mahani, V. Engert, S. J. Lupien, and J. C. Pruessner, ‘‘The Montreal Imaging Stress Task: using functional imaging to investigate the effects of perceiving and processing psychosocial stress in the human brain,’’ Journal of Psychiatry and Neuroscience, vol. 30, no. 5, pp. 319–325, 2005.

[20] Y. Zhang, Q. Wang, Z. Y. Chin, and K. K. Ang, ‘‘Investigating different stress-relief methods using electroencephalogram (EEG),’’ in 2020 42nd Annual International Conference of the IEEE Engineering in Medicine & Biology Society (EMBC), pp. 2999–3002, IEEE, 2020.

[21] F. Al-Shargie, ‘‘Prefrontal cortex functional connectivity based on simultaneous record of electrical and hemodynamic responses associated with mental stress,’’ arXiv preprint arXiv:2103.04636, 2021.

[22] J. Minguillon, E. Perez, M. A. Lopez-Gordo, F. Pelayo, and M. J. SanchezCarrion, ‘‘Portable system for real-time detection of stress level,’’ Sensors,vol. 18, no. 8, p. 2504, 2018.

[23] L. Xia, A. S. Malik, and A. R. Subhani, ‘‘A physiological signal-based method for early mental-stress detection,’’ Biomedical Signal Processing and Control, vol. 46, pp. 18–32, 2018.

[24] C. Setz, B. Arnrich, J. Schumm, R. La Marca, G. Tröster, and U. Ehlert, ‘‘Discriminating stress from cognitive load using a wearable EDA device,’’ IEEE Transactions on information technology in biomedicine, vol. 14, no. 2, pp. 410–417, 2009.

[25] L. Han, Q. Zhang, X. Chen, Q. Zhan, T. Yang, and Z. Zhao, ‘‘Detecting work-related stress with a wearable device,’’ Computers in Industry, vol. 90, pp. 42–49, 2017.

[26] M. Gjoreski, M. Luštrek, M. Gams, and H. Gjoreski, ‘‘Monitoring stress with a wrist device using context,’’ Journal of biomedical informatics, vol. 73, pp. 159–170, 2017.

[27] L. Zhu, P. Spachos, P. C. Ng, Y. Yu, Y. Wang, K. Plataniotis, and D. Hatzinakos, ‘‘Stress Detection Through Wrist-Based Electrodermal Activity Monitoring and Machine Learning,’’ IEEE Journal of Biomedical and Health Informatics, 2023.

[28] C.-P. Hsieh, Y.-T. Chen, W.-K. Beh, and A.-Y. A. Wu, ‘‘Feature selection framework for xgboost based on electrodermal activity in stress detection,’’ in 2019 IEEE International Workshop on Signal Processing Systems (SiPS), pp. 330–335, IEEE, 2019.

[29] P. Siirtola, ‘‘Continuous stress detection using the sensors of commercial smartwatch,’’ in Adjunct Proceedings of the 2019 ACM International Joint Conference on Pervasive and Ubiquitous Computing and Proceedings of the 2019 ACM International Symposium on Wearable Computers, pp. 1198–1201, 2019.

[30] D. Jaiswal, D. Chatterjee, R. Gavas, R. K. Ramakrishnan, and A. Pal, ‘‘Person and Stressor Independent Generic Model for Stress Detection Using GSR,’’ in 2021 43rd Annual International Conference of the IEEE Engineering in Medicine & Biology Society (EMBC), pp. 7195–7198,

IEEE, 2021.