

**VISVESVARAYA TECHNOLOGICAL UNIVERSITY  
BELAGAVI-590018, KARNATAKA.**



**PHASE II PROJECT REPORT  
ON  
“IDENTIFICATION AND PREDICTION OF AUTISM DISORDER”**

**Project Associates**

<b>Sneha S Chikkaraddi</b>	<b>4BD22CD043</b>
<b>Tanushree S Chakrasali</b>	<b>4BD22CD047</b>
<b>Varshini N H</b>	<b>4BD22CD050</b>

*Submitted in the partial fulfillment of the requirement  
for the award of the degree*

in

**COMPUTER SCIENCE AND ENGINEERING (DATA SCIENCE)**

**UNDER THE GUIDANCE OF**

**Prof. Usha C** M.Tech.  
**Project Guide**  
**Assistant Professor**  
**Department of CS&E (Data Science)**



**2025-2026**

**Department of Computer Science and Engineering (Data Science),  
Bapuji Institute of Engineering and Technology,  
Davangere -577004**

**VISVESVARAYA TECHNOLOGICAL UNIVERSITY  
BELAGAVI - 590018, KARNATAKA**



**BAPUJI INSTITUTE OF ENGINEERING AND TECHNOLOGY  
DAVANGERE – 577004, KARNATAKA, INDIA.**



**Department of Computer Science and Engineering (Data  
Science)**

**CERTIFICATE**

This is to certify that **Sneha S Chikkaraddi, Tanushree S Chakrasali and Varshini N H** bearing **USN 4BD22CD043, 4BD22CD047 and 4BD22CD050** respectively of Computer Science and Engineering (Data Science) department have satisfactorily submitted the Final Year Project Work Phase-II entitled “**Identification And Prediction Of Autism Disorder**” in the partial fulfillment of the requirements for the award of Degree of Bachelor of Engineering (B.E.) in Computer Science and Engineering (Data Science), under the VTU during the academic year 2024-25.

.....  
**Prof. Usha C** M.Tech.  
**Project Guide**  
**Assistant Professor**

.....  
**Prof. Gowramma B H**  
**Project Co-ordinator**

.....  
**Dr. Pradeep N** Ph.D  
**Dean Academics**  
**Professor & HoD**

.....  
**Principal**

**External Viva:**  
**Name of Examiners:**

- 1)
- 2)

**Signature with Date**

.....  
.....

# Acknowledgment

Salutation to our beloved and highly esteemed institute **Bapuji Institute of Engineering and Technology** for having well-qualified staff and labs furnished with necessary equipment.

We express our sincere thanks to the **Guide Mrs. Usha C** and **Project Coordinator Mrs. Gowramma B H** for giving constant encouragement, support and valuable guidance throughout the course of the project, without whose stable guidance this project report would not have been achieved.

We express wholehearted gratitude to **Dr. Pradeep N, HOD of CS&E (Data Science)**. I wish to acknowledge his help who made my task easy by providing with her valuable help and encouragement.

We would like to thank our beloved **Principal, Dr. H B Aravind** and the **Director Prof. Y. Vrushabhendrappa** of this institute for giving us the opportunity and guidance to work on the project.

We would like to extend our gratitude to all Staff of Computer Science and Engineering (Data Science) department for the help and support rendered to us. We have benefited a lot from the feedback, and suggestions given by them.

We would like to extend our gratitude to the family members and friends, especially for their advice and moral support.

**TANUSHREE S CHAKRASALI**  
**SNEHA S CHIKKARADDI**  
**VARSHINI N H**

**4BD22CD047**  
**4BD22CD043**  
**4BD22CD050**

**Department of Computer Science and Engineering**  
**(Data Science)**

**VISION**

To provide a quality and holistic education in data science, data analytics, data visualization, industry collaborations and research by empowering individuals to derive knowledge, thereby transforming the potentials in data for the betterment of society.”

**MISSION**

**M1:** Educate and prepare students with a strong foundation in data science, equipping them with the skills, knowledge, and ethical principles needed to excel in data-driven fields.

**M2:** Foster collaborations with industries to adopt modern data science and visualization tools which solves the real-world problems that have societal benefits.

**M3:** Cultivate a culture of life-long learning with intellectual curiosity in data science and nurturing individuals who are passionate about data-driven decision-making.

**COURSE OUTCOMES**

<b>Sub. Code</b>	<b>Course Outcomes</b>
<b>BCD786.1</b>	Apply appropriate theories, tools, technologies, and methodologies to design an effective and technically feasible solution for the identified problem.
<b>BCD786.2</b>	Develop and implement a fully functional system/using suitable programming languages, frameworks, libraries, or hardware platforms.
<b>BCD786.3</b>	Test, validate, and evaluate the performance of the developed solution using appropriate metrics.
<b>BCD786.4</b>	Prepare a comprehensive technical project report and submit the project outcomes to refereed journals for publication.

# LIST OF PROGRAM OUTCOMES

PO Code	Short Description	Full Description
PO1	Engineering knowledge	Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.
PO2	Problem analysis	Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.
PO3	Design/development of solutions	Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations
PO4	Conduct investigations of complex problems	Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.
PO5	Modern tool usage	Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations
PO6	The engineer and society	Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice
PO7	Environment and sustainability	Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development
PO8	Ethics	Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.
PO9	Individual and team work	Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.
PO10	Communication	Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions
PO11	Project management and finance	Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.
PO12	Life-long learning	Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change

## **LIST OF PROGRAM SPECIFIC OUTCOMES**

<b>PSO Code</b>	<b>Full Description</b>
<b>PSO1</b>	Students will be able to develop sustainable and efficient algorithm solutions for the real time problems by applying their problems solving skills.
<b>PSO2</b>	Students will be able to develop solutions for given problems in the areas of Artificial Intelligence, Data Analytics, and other societal domains through a conducive environment and infrastructure.

### **CO-PO-PSO Mapping**

	<b>PO1</b>	<b>PO2</b>	<b>PO3</b>	<b>PO4</b>	<b>PO5</b>	<b>PO6</b>	<b>PO7</b>	<b>PO8</b>	<b>PO9</b>	<b>PO10</b>	<b>PO11</b>	<b>PO12</b>	<b>PSO1</b>	<b>PSO2</b>
<b>BCD786.1</b>	3	3	1	1	1	1	1	1	3	1	1	3	1	1
<b>BCD786.2</b>	3	3	1	3	1	1	1	1	3	1	2	3	1	1
<b>BCD786.3</b>	3	3	2	3	1	1	1	1	3	1	1	3	1	1
<b>BCD786.4</b>	3	3	3	3	1	1	1	1	3	2	2	3	2	2

# ABSTRACT

Transformer models, which are well-known in the fields of natural language processing and computer vision, are increasingly being utilized in healthcare, especially for the identification and prediction of autism spectrum disorder (ASD). ASD is a multifaceted neurodevelopmental disorder characterized by difficulties in social interaction, communication, and behavior. Transformers are particularly adept at capturing long-range dependencies, making them suitable for processing a variety of ASD-related data, including neuroimaging, genetic sequences, and speech. Vision Transformers (ViTs) are capable of analyzing MRI/fMRI scans to identify unusual brain patterns, while sequence transformers communication, and behavior. Transformers are particularly adept at capturing long-range dependencies, making them suitable for processing a variety of ASD-related data, including neuroimaging, genetic sequences, and speech. Vision Transformers (ViTs) are capable of analyzing MRI/fMRI scans to identify unusual brain patterns, while sequence transformers.

# CONTENTS

<b>ABSTRACT</b>	<b>I</b>
<b>LIST OF ABBREVIATIONS</b>	<b>II</b>
<b>LIST OF TABLES</b>	<b>III</b>
<b>LIST OF FIGURES</b>	<b>IV</b>
<b>CHAPTERS</b>	<b>PAGE NO</b>
<b>1. INTRODUCTION</b>	<b>01-03</b>
1.1. Description	01
1.2. Existing System	02
1.3. Problem Statement	02
1.4. Proposed System	02
1.5. Objectives	03
<b>2. LITERATURE SURVEY</b>	<b>04-06</b>
2.1. Related background study	06
<b>3. SYSTEM REQUIREMENT AND SPECIFICATION</b>	<b>07-09</b>
3.1 Functional Requirements	07
3.2 Non-Functional Requirements	08
3.3 Hardware Requirements	08
3.4 Software Requirements	09
<b>4. SYSTEM DESIGN</b>	<b>10-16</b>
4.1 Module Description	10-11
4.2 System Architecture	11-12
4.3 Use case Diagram	12-13
4.4 Sequence Diagram	13-14
4.6.Data Flow Diagram	14-15
4.6.Data Flow Diagram	15-16
<b>5. IMPLEMENTATION</b>	<b>17-30</b>
5.1.Modules	17-18
5.2.Pseudocode	18-30
<b>6. TESTING</b>	<b>31-33</b>
6.1. Overview	31
6.2. Test Cases	32-33

<b>7.</b>	<b>RESULTS AND DISCUSSION</b>	<b>34-40</b>
7.1.	Snapshots	34-39
7.2.	Advantages and Disadvantages	40
	<b>CONCLUSION AND FUTURE SCOPE</b>	<b>41</b>
	<b>REFERENCES</b>	<b>42</b>

## LIST OF ABBREVIATIONS

Abbreviation No	Abbreviation	Full Form
1	ASD	Autism Spectrum Disorder
2	ML	Machine Learning
3	DL	Deep Learning
4	RF	Random Forest
5	LR	Logistic Regression
6	CNN	Convolutional Neural Network
7	QDA	Quadratic Discriminant Analysis
8	MLP	Multilayer Perceptron
9	SVM	Support Vector Machines
10	KNN	K-Nearest Neighbors
11	SQL	Structured Query Language

## LIST OF TABLES

Table No.	Description	Page No.
2.1	Literature Survey Table	05
6.2	Test Cases	32-33

## LIST OF FIGURES

Figure No.	Description	Page No.
4.2	System Architecture Diagram	11
4.3	Use Case Diagram	12
4.4	Sequence Diagram	13
4.5	Activity Diagram	14
4.6	Data Flow Diagram	15
7.1.1	Login Page	34
7.1.2	ASD Prediction System	34
7.1.3	Visualization of Class & Age	35
7.1.4	Visualization of Gender Score	35
7.1.5	Model Performance Analysis	35
7.1.6	Model Performance Metrics	36
7.1.7	Prediction Tool	36
7.1.8	Changes In Environment	36
7.1.9	Repetitive Behaviors	37
7.1.10	Review Answers	37
7.1.11	Assessment Results	38
7.1.12	Question By Question Scores	38
7.1.13	Behavioral Pattern Analysis	38
7.1.14	Autism Support Assistance	39
7.1.15	Assessment History	39
7.1.16	Admin Panel	39

---

## CHAPTER 1

# INTRODUCTION

### 1.1 DESCRIPTION

Autism Spectrum Disorder (ASD) is a complex neurodevelopmental condition that affects social interaction, communication, and behavior. The early identification and accurate prediction of ASD are critical, as early intervention has been shown to significantly improve developmental outcomes for children. Traditional methods of diagnosing ASD, which often involve clinical interviews, behavioral observations, and standardized diagnostic assessments, can be time-consuming and subjective, leaving room for misdiagnosis or delayed diagnosis [1].

The etiology of the disease remains unclear; nonetheless, it is believed to be associated with biological factors including genetic anomalies, neuroinflammation, and adverse prenatal conditions. The significant increase in the number of children diagnosed with Autism Spectrum Disorder (ASD) highlights the necessity for further research on this population. In particular, appropriate clinical practices are crucial [2].

In recent years, machine learning (ML) and deep learning (DL) models, particularly transformer models, have shown great potential in various fields, including healthcare. Transformers are a class of deep learning models that excel at handling sequential data and capturing long-range dependencies within datasets. Originally developed for natural language processing (NLP), transformer models have expanded into other domains, such as image recognition, genetics, and medical data analysis, due to their flexibility and power [4][5].

The issue of autism spectrum disorder (ASD) has been rapidly increasing across all age groups within the human population. Timely identification of this neurological condition can significantly contribute to the preservation of the individual's mental and physical well-being. Autism spectrum disorder pertains to challenges associated with the development of the human brain. An individual affected by autism spectrum disorder typically faces difficulties in social interactions and communication with others[1].

In the context of ASD, transformers can be applied to a wide variety of data types, including behavioral patterns, neuroimaging data, genetic information, and even speech or audio recordings. These models can identify subtle patterns, extract meaningful features, and make accurate predictions that might not be immediately apparent to human clinicians. As such, transformer-based systems are emerging as promising tools for the identification, classification, and prediction of ASD.

The application of transformer models in ASD presents a transformative opportunity to enhance early diagnosis, predict developmental trajectories, and personalize intervention strategies. This introduction will explore how transformer models are being used to improve the identification and prediction of autism spectrum disorder, highlighting their potential benefits and challenges in the

healthcare field [6].

## **1.2 EXISTING SYSTEM**

Traditional methods for identifying Autism Spectrum Disorder (ASD) rely heavily on clinical observations, behavioral assessments, and standardized psychological tests, which can be time-consuming, subjective, and prone to variations in interpretation across clinicians. Existing machine learning approaches have attempted to automate ASD detection using algorithms such as Logistic Regression, Support Vector Machines, Random Forests, K-Nearest Neighbors, and Convolutional Neural Networks. Although these models have achieved high accuracy in several research studies, they often struggle to capture complex relationships within ASD datasets and may be limited by challenges such as imbalanced samples, insufficient feature representation, and restricted ability to handle large or diverse data types. As a result, current systems lack a highly robust and generalizable solution for early and accurate ASD prediction, highlighting the need for more advanced deep learning models such as transformers.

## **1.3 PROBLEM STATEMENT**

Autism Spectrum Disorder (ASD) is a neurodevelopmental disorder that impacts communication, behavior, and social interactions. Timely and precise identification of ASD is essential for effective intervention and support. Nevertheless, conventional diagnostic approaches are often lengthy, subjective, and necessitate professional expertise. This initiative seeks to tackle the difficulties associated with ASD identification and prediction by creating a Transformer-based machine learning model. By utilizing behavioral screening data alongside demographic information, the model will identify patterns that aid in determining whether individuals have ASD. The objective is to establish a dependable, automated system that can support healthcare professionals in the early detection of ASD and enhance decision-making processes.

## **1.4 PROPOSED SYSTEM**

The proposed system introduces a transformer-based deep learning approach to enhance the accuracy and efficiency of Autism Spectrum Disorder (ASD) prediction. By utilizing behavioral, demographic, and screening-related datasets, the system applies advanced preprocessing techniques and feature engineering to prepare high-quality input for model training. The transformer model leverages its self-attention mechanism to learn complex patterns and relationships within the data that traditional machine learning models may overlook. The system evaluates performance using key metrics such as accuracy, precision, recall, and F1-score, ensuring reliable and robust detection results. Additionally, feature importance analysis is incorporated to support early diagnosis and clinical decision-making, enabling personalized intervention strategies. This proposed system aims to provide an automated, scalable, and objective diagnostic support tool for healthcare professionals.

## 1.5 OBJECTIVES

- To collect a high-quality and relevant dataset that includes behavioral, demographic, and screening information related to Autism Spectrum Disorder (ASD).
- To perform preprocessing on the collected data, including handling missing values, encoding categorical variables, normalization, and data splitting to prepare it for training the Transformer model.
- To design and implement a Transformer-based deep learning model that can learn from tabular autism related data. The model will be trained to identify patterns that distinguish individuals with ASD from those without.
- To evaluate the model using key metrics like accuracy, precision, recall, and F1- score to determine its effectiveness in identifying ASD.
- To explore the possibility of early autism prediction through feature importance analysis

---

## CHAPTER 2

# LITERATURE SURVEY

### 2.1 RELATED BACKGROUND STUDY

Machine learning (ML) techniques have increasingly become central to identifying meaningful patterns within medical data, enabling enhanced disease detection and decision-making. In the context of Autism Spectrum Disorder (ASD), ML models are widely used to distinguish between affected and non-affected individuals using behavioral and diagnostic features.

Azian A. et al. [1] investigated multiple feature selection and learning techniques including Least Absolute Shrinkage and Selection Operator (LASSO) and Chi-square methods to evaluate their effectiveness in regression and classification. Models such as K-Nearest Neighbors (K-NN), Random Forest (RF), and Logistic Regression (LR) were tested, with Logistic Regression achieving the highest accuracy of 97.54%, utilizing 13 features selected from the Chi-square method.

Similarly, Suman R. and Sarfaraz M. [2] explored ML-based prediction approaches to estimate ASD risk using classifiers such as Logistic Regression, Naïve Bayes, K-NN, Support Vector Machines, Convolutional Neural Networks (CNN), and Artificial Neural Networks. Their study used three datasets representing children, adults, and adolescents with sample sizes of 292, 740, and 104 respectively. CNN demonstrated superior performance with accuracy rates of 98.30%, 99.53%, and 96.88% for the three age groups. Nishat MM et al. [3] developed a predictive model using Quadratic Discriminant Analysis (QDA) and linear analysis techniques to identify ASD and associated psychological disorders. Using datasets from the UCI repository, model performance was assessed using metrics such as sensitivity, F1-score, accuracy, and the Youden index. Their results showed that QDA achieved a high accuracy of 99.77% following hyperparameter optimization. Md. Mokhlesur R. et al. [4] emphasized the importance of selecting meaningful autistic traits and reducing feature redundancy to improve ML-based ASD detection. They highlighted challenges affecting accuracy, including imbalanced datasets, inappropriate sampling, and irrelevant data dimensions, and suggested enhancing feature selection to achieve more reliable classification outcomes.

In another study, Nurul A. et al. [5] used various ML classifiers, including SVM, K-NN, Naïve Bayes, J48, Bagging, and Stacking, to classify individuals using a dataset of 703 subjects with 16 input features. Implementing these models using the WEKA platform, they reported that Stacking, J48, SVM, and Bagging demonstrated exceptional performance with up to 100% accuracy and minimal error rates. Md. Delowar H. and Muhammad Ak. [6] evaluated the ability of modern classification algorithms to automate ASD diagnosis using datasets across different age categories.

Their results showed that the Multilayer Perceptron (MLP) classifier achieved 100% accuracy across

all datasets, and the Relief-F method proved most effective in identifying and ranking discriminative attributes.

Sl.No	Title	Author	Description	Publicati On Details
1	Autism Spectrum Disorder Screening: Machine Learning Approach	Fadi Thabtah et al.	Applied Decision Tree and Naive Bayes on ASD screening dataset; achieved 96.7% accuracy.	Health Informatics Journal
2	Machine Learning Based Autism Spectrum Disorder Detection	Amanpreet Kaur, Richa Sharma	Compared classifiers (SVM, RF, NB) on UCI autism data; ensemble models performed best.	International Journal of Engineering Research
3	Predicting Autism Spectrum Disorder using ML Algorithms	Khalid Barukab et al.	Used boosting algorithms (XGBoost, LightGBM); XGBoost achieved 97.4% accuracy.	Procedia Computer Science, Elsevier
4	Deep Learning for Autism Spectrum Disorder Detection in MRI Scans	S. Suresh, A. Ghosh	Applied CNN to ABIDE MRI data; achieved 89% accuracy using spatial brain features.	IEEE Access
5	Hybrid ML Framework for Autism Detection	M. Ahmed et al.	Combined behavioral and neuroimaging data using Auto ML and ensemble techniques.	Springer – Neural Computing and Applications

**Figure 2.1 Literature Survey Table**

- Autism Spectrum Disorder (ASD) is a neurodevelopmental condition characterized by challenges in communication, social interaction, and repetitive behaviors. Over the past decade, researchers and healthcare professionals have focused on integrating data science and machine learning (ML) techniques to assist in the early identification and diagnosis of autism. The application of ML in healthcare has proven effective in analyzing large datasets, identifying complex patterns, and providing predictive insights that traditional methods often overlook.
- Machine learning models have been widely used in healthcare for disease diagnosis, patient risk prediction, and behavioral pattern recognition. In the context of autism, researchers have explored behavioral, genetic, and demographic data to train algorithms capable of distinguishing between autistic and non-autistic individuals. Datasets such as the UCI Autism Brain Imaging Data Exchange (ABIDE) have served as valuable resources for model

development and testing.

- Various algorithms have been applied to autism prediction, including Decision Trees, Random Forests, Support Vector Machines (SVM), Logistic Regression, and Neural Networks. These models analyze features such as age, gender, family history, social interaction patterns, and questionnaire responses to identify autism tendencies. Studies have shown that ensemble learning models like Random Forest and boosting techniques often outperform single models in terms of accuracy and generalization.
- In addition to structured data analysis, some researchers have explored the use of deep learning and computer vision for detecting autism-related behaviors through facial expressions, speech patterns, and eye-tracking data.
- However, challenges such as data scarcity, class imbalance, and ethical considerations in handling sensitive medical information still exist. Addressing these limitations through proper data preprocessing, model optimization, and privacy-preserving techniques is essential for reliable deployment of ML-based systems in real-world healthcare environments.
- The current study builds upon these foundational works by leveraging machine learning algorithms to design a reliable and efficient autism prediction model. The goal is to assist clinicians and families in early screening, thereby enabling timely intervention.

---

## CHAPTER 3

# SYSTEM REQUIREMENT AND SPECIFICATION

### 3.1 FUNCTIONAL REQUIREMENTS

The functional requirements describe the specific behaviors, features, and operations that the proposed Autism Prediction System must perform. These requirements define how the system should function to achieve its intended objectives efficiently and accurately.

- **User Data Input:** The system must allow users, such as clinicians or parents, to input data including age, gender, family history, behavioral information, and questionnaire responses. The interface must validate all user inputs to ensure the data is complete and accurate before processing.
- **Data Preprocessing:** The system must automatically clean and preprocess the dataset by handling missing values, encoding categorical variables, and normalizing numerical data. It must prepare the dataset for both the training and testing phases without manual intervention.
- **Model Training and Testing:** It must split the data into training and testing sets to evaluate the performance of each model. The system identify the comparing metrics such as accuracy, precision, recall, and F1-score.
- **Autism Prediction:** The system must be able to predict the likelihood of an individual having autism based on the provided input features. The prediction output must clearly state the result (e.g., Autistic or Non Autistic) and include a confidence score or probability level.
- **Result Visualization:** The system must display key model performance metrics, such as the confusion matrix and accuracy percentage. All results should be presented to the user in a clear, simple, and understandable format.
- **User Interface Functionality:** The user interface must be simple, interactive, and responsive. It must provide input forms, a button to initiate the prediction, and a dedicated section to display the results instantly.
- **Data Storage and Retrieval:** The system is required to securely store all datasets, trained model files, and prediction results. Users should have the capability to retrieve previous results.
- **System Security and Authentication:** If applicable, the system must enforce access control to ensure only authorized users can access it. It must protect the confidentiality of sensitive health information through robust security measures.

### 3.2 NON-FUNCTIONAL REQUIREMENTS

The non-functional requirements define the quality attributes, performance criteria, and operational standards that the Autism Prediction System must meet. These requirements ensure that the system operates efficiently, securely, and reliably in real-world conditions.

- **Performance Requirements:** The system should deliver prediction results within a few seconds after data submission. Machine learning model training and testing processes must be optimized for efficiency to minimize processing time. Response times for all user interactions should be consistently fast across the entire application.
- **Accuracy and Reliability:** The system must achieve high prediction accuracy, ideally above 85%, by using appropriate machine learning algorithms. Results generated by the model must be consistent and reproducible across multiple executions. The system should be capable of handling noisy or incomplete data without a significant drop in performance.
- **Usability Requirements:** The user interface must be simple, intuitive, and easy for non-technical users to navigate. Clear instructions and tooltips should be provided to guide users through data entry and result interpretation. The layout should be designed with accessibility in mind to accommodate all users.
- **Scalability:** The system must be designed to handle large datasets and manage multiple user requests simultaneously. It should support future enhancements, such as the integration of deep learning models or new datasets, without requiring major architectural changes.
- **Security Requirements:** The system must ensure that all sensitive user data is stored and transmitted securely. Robust authentication and authorization mechanisms must be implemented to prevent unauthorized access. Data encryption techniques should be applied to protect the confidentiality of all stored information.
- **Maintainability:** The codebase must be modular and well-documented to facilitate easy updates, debugging, and the integration of new features. The system should allow for the retraining of models with new data without needing a complete redevelopment of the system.
- **Portability:** The system should be platform-independent, enabling it to run on various operating systems like Windows, Linux, and macOS. It must be deployable in both local and cloud-based environments.
- **Availability and Robustness:** The system must be highly available, ensuring minimal downtime for users. It should handle unexpected inputs and system errors gracefully to prevent crashes.

### 3.3 HARDWARE REQUIREMENTS

The Software Requirements define the tools, platforms, and technologies needed for the development, implementation, and execution of the Autism Prediction System. These components

ensure that the system operates efficiently and supports all necessary machine learning functionalities.

- **Operating System:** The system should be compatible with common operating systems such as Windows 10/11, Linux (Ubuntu), or macOS for both development and deployment.
- **Programming Language:** SQL, Javascript/Typescript or above is required due to its simplicity, readability, and extensive machine learning libraries for data preprocessing, model training, and prediction.
- **Development Environment / IDE:** Google Colab or cloud based platform should be used for data analysis, model training, and testing.
- **Libraries and Frameworks:** React is used for building user interfaces. rechart for comparing charting libraries for react. Supabase is an open-source backend platform that gives you authentication, database, APIs and storage built on top of SQL.
- **Database:** SQLite or MySQL can be used to store user details, dataset records, and prediction history if a backend is implemented.
- **Tools and Utilities:** Git and GitHub should be used for version control and collaborative development. Excel or CSV files will be used for dataset storage and import.
- **Browser:** For a web-based system, compatibility with Google Chrome, Microsoft Edge, or any modern browser is required.
- **Machine Learning Dataset:** The UCI Autism Screening Dataset will be used for training and evaluating the models, as it includes relevant behavioral, demographic, and screening features.

### 3.4 SOFTWARE REQUIREMENTS

- **Processor (CPU):** A multi-core processor is highly recommended. An Intel Core i7/i9 or AMD Ryzen 7/9 with at least 6-8 cores will significantly speed up data preprocessing and other computational tasks.
- **RAM:** 16GB of RAM or more is ideal for working with larger datasets and training more complex models without performance bottlenecks. For very large datasets, 32GB is preferable.
- **Storage:** A Solid-State Drive (SSD) with at least 500GB of space is strongly recommended. An SSD provides much faster data access speeds, which accelerates dataset loading, model saving, and overall system responsiveness.
- **GPU:** A dedicated NVIDIA GPU with at least 8GB of VRAM (e.g., NVIDIA GeForce GTX 1080 or better) is recommended for accelerating model training, especially if deep learning models are considered in the future. For standard machine learning algorithms like the ones specified, a mid-range GPU like the NVIDIA RTX 3060 (12GB) offers excellent performance.

---

## CHAPTER 4

# SYSTEM DESIGN

### INTRODUCTION

System design defines the architecture and structure of the Autism Prediction System, specifying how the components interact to achieve the required functionality. The system integrates a frontend interface for user data input and result display, a backend machine learning engine for data processing, model training, and prediction, and a database for storing datasets and prediction history. It focuses on both logical design, outlining data flow, processing, and interactions, and physical design, detailing software architecture, technology stack, and integration of components. A well-planned system design ensures accuracy, scalability, maintainability, and efficient performance, providing a seamless and reliable tool for early.

### 4.1 MODULE DESCRIPTION

The Autism Prediction System is divided into several functional modules, each responsible for a specific task to ensure efficient and accurate operation.

- User Input Module

Function: Provides an interface for users to enter data such as age, gender, behavioral information, and questionnaire responses.

Validation: It validates all inputs to ensure they are correct and complete before processing.

- Data Preprocessing Module

Function: Cleans and prepares the collected data for analysis.

Tasks: This module handles missing values, encodes categorical variables, normalizes numerical features, and formats the data for the model.

- Feature Selection Module

Function: Identifies and selects the most significant features that affect the prediction of autism.

Benefit: This improves model accuracy and reduces computational complexity by filtering out irrelevant or redundant data.

- Model Training and Evaluation Module

Function: Trains various machine learning models like Logistic Regression, Decision Tree, Random Forest, and SVM using labeled datasets.

Evaluation: It assesses model performance with metrics such as accuracy, precision, recall, and F1-score to choose the best-performing model.

- Prediction Module

Function: Takes new user data and uses the trained model to predict the likelihood of

autism.

Output: Clearly presents the result (e.g., —Autistic|| or —Non- Autistic||) along with a confidence score.

- Result Display and Visualization Module

Function: Presents predictions and performance metrics to users in a clear and understandable format.

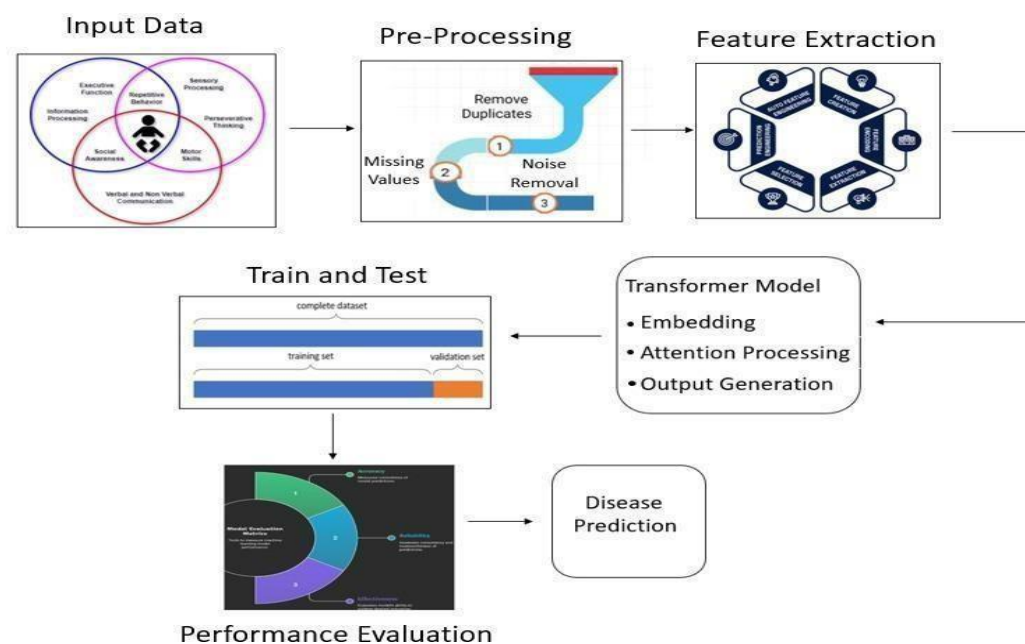
Presentation: Uses tables, charts, or graphical summaries to provide better insight into the results.

- Database/Storage Module

Function: Securely handles the storage of input data, historical predictions, and trained model files.

Capabilities: Allows for the retrieval of past results for analysis or for retraining the model when new data becomes available.

## 4.2 SYSTEM ARCHITECTURE



**Fig. 4.2 System Architecture Diagram**

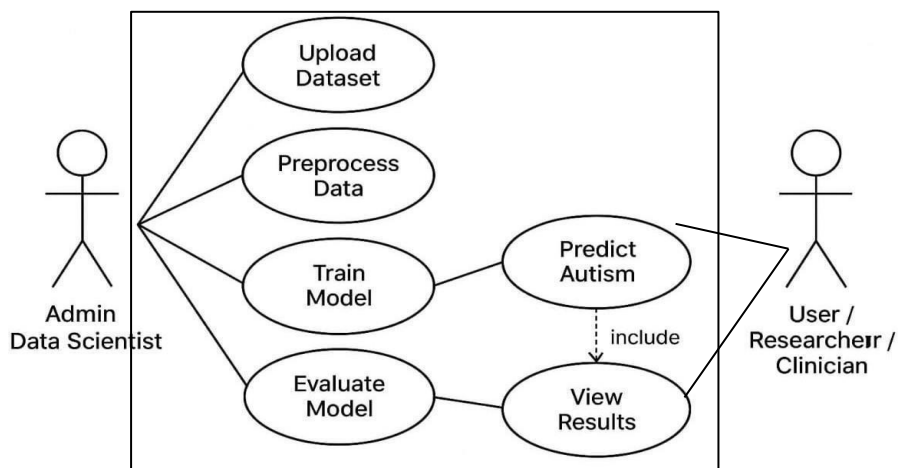
- User Input (Questionnaire / Dataset): The process begins with the input data, which can be collected from autism screening questionnaires or an existing dataset containing behavioral and demographic information. This raw data may contain missing values, noise, or irrelevant information, so it needs further processing.
- Data Preprocessing (Cleaning, Normalization, Feature Selection): This stage prepares the data for model training by performing: Data Cleaning Removing duplicates, filling missing values, and correcting errors. Normalization – Scaling numerical features to ensure

uniformity. Feature Selection- Identifying the most important variables that influence autism prediction. The goal is to produce a clean and structured dataset suitable for analysis.

- **Feature Extraction (Relevant Attributes):** Important features or attributes related to autism are extracted—such as communication skills, social behavior, or age-related factors. This step ensures that only relevant features are used by the machine learning model to improve accuracy and reduce complexity.
- **Machine Learning Model (Training & Testing):** The preprocessed and feature-extracted data is used to train a machine learning model. The model learns patterns and relationships between input features and autism labels (Autistic / Non-Autistic). After training, the model is tested using unseen data to check how well it predicts autism.
- **Prediction Output (Autism / No Autism):** Once trained, the model generates predictions for new data inputs. The output can be: Autism-indicating a high likelihood of autism. No Autism- indicating a low likelihood of autism.
- **Performance Evaluation (Accuracy, Precision, Recall):** The model's performance is evaluated using metrics such as Accuracy -Overall correctness of predictions. Precision- How many predicted autism cases were actually correct. Recall-How well the model identifies all autism cases.

### 4.3 USE CASE DIAGRAM

- **Admin / Data Scientist:** This actor is responsible for handling the technical part of the system. Performs tasks like uploading datasets, preprocessing data, training models, and evaluating them.
- **User / Researcher / Clinician:** This actor uses the system to make predictions and view the results. Typically a non-technical user who interprets the model's predictions for practical or medical purposes.



**Fig. 4.3 Use Case Diagram**

## Use Cases

- **Upload Dataset:** The Admin uploads raw data related to autism screening or symptoms into the system.
- **Preprocess Data:** The Admin cleans and prepares the data (handling missing values, encoding, normalization, etc.) for model training.
- **Train Model:** The Admin trains a machine learning model using the preprocessed dataset to predict autism.
- **Evaluate Model:** The Admin checks the model's accuracy and performance metrics to ensure it predicts reliably.
- **Predict Autism:** Either the Admin or User can use the trained model to predict whether an individual is likely to have autism based on input data.
- **View Results:** The User/Researcher/Clinician views the prediction results (e.g., Autisticl or —Non-Autisticl) and related analysis.

## 4.4 SEQUENCE DIAGRAM

- **User Submits Data:** The User initiates the process by submitting input data to the Interface (Web/Software).
- **Interface Sends Raw Data:** The Interface forwards the raw data directly to the Preprocessing Module.
- **Data Preprocessing:** The Preprocessing Module executes the necessary task to clean & transform data, making it usable for the model.
- **Module Sends Processed Data:** The Preprocessing Module sends the resulting processed data to the ML Model.

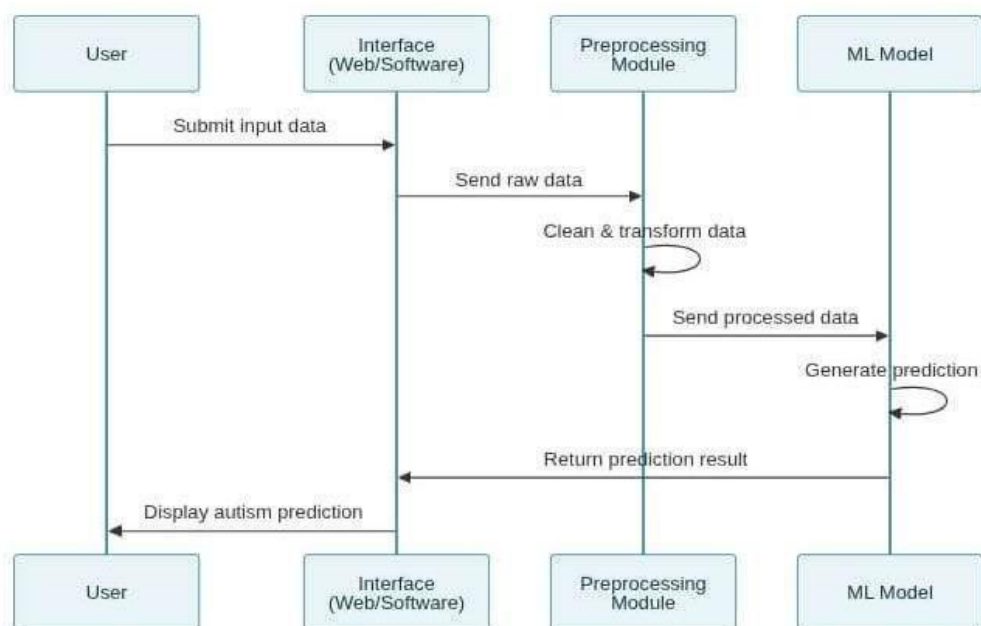
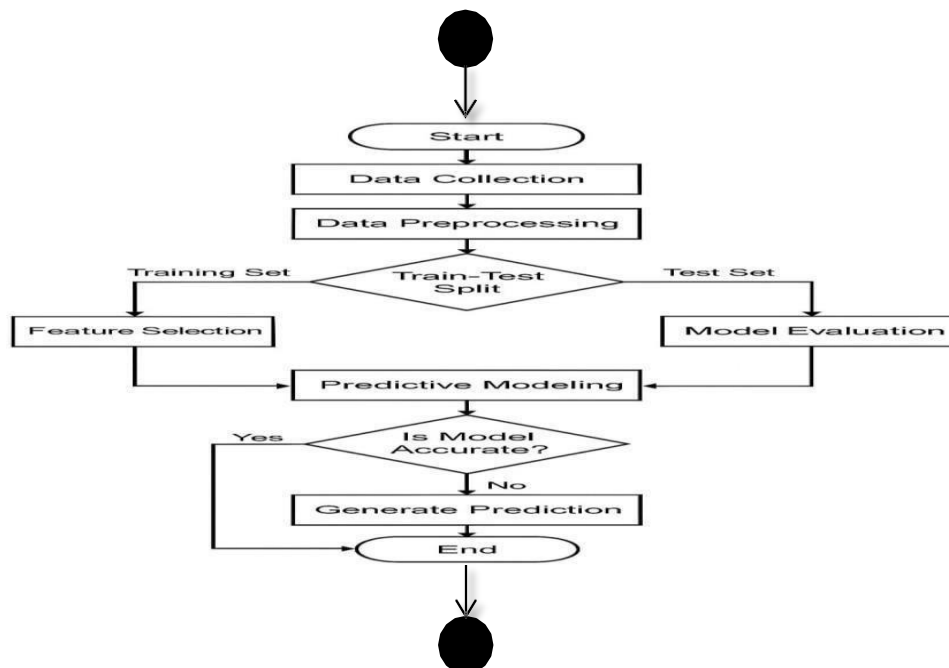


Fig. 4.4 Sequence Diagram

- **Model Generates Prediction:** The ML Model uses the processed data to generate a prediction (the inference step).
- **Model Returns Result:** The ML Model sends the final prediction result back to the Interface (Web/Software).
- **Interface Displays Result:** The Interface (Web/Software) presents the final autism prediction result back to the User.

#### 4.5 Activity Diagram

- **Start:** The process begins.
- **Data Collection:** The initial step is gathering the necessary raw data for the prediction task.
- **Data Preprocessing:** The collected data is cleaned, transformed, and prepared for use in the modeling phase.
- **Train-Test Split (Decision Point):** The preprocessed data is divided into two parts: a Training Set and a Test Set.
- **Following the Training Set path:** This leads to Feature Selection, where the most relevant variables are chosen for the model.
- **Following the Test Set path:** This leads to Model Evaluation, where metrics will be used to assess the model's performance on unseen data.

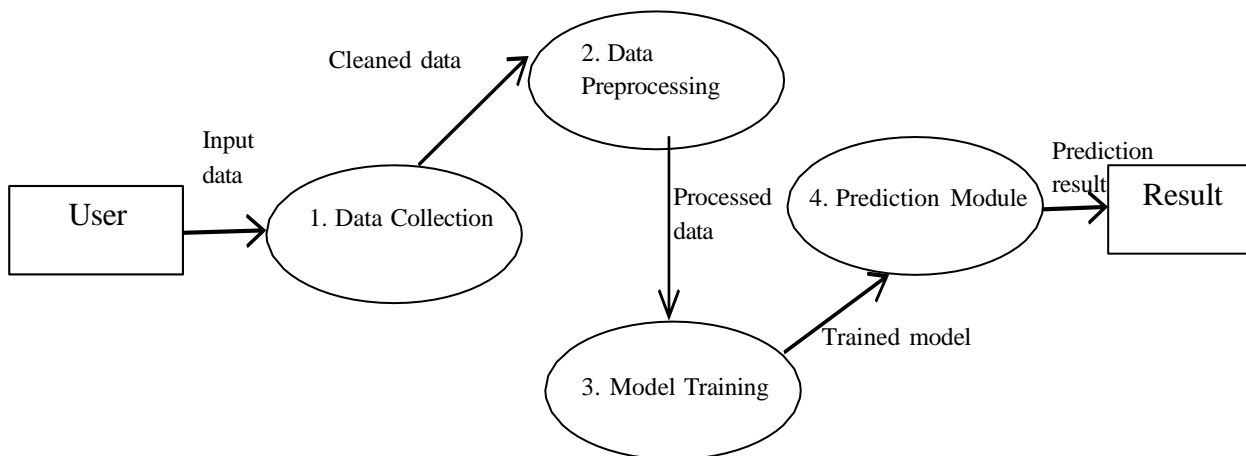


**Fig. 4.5 Activity Diagram**

- **Predictive Modeling:** The process converges here. The chosen features from the Training Set are used to build and train the predictive model. The Model Evaluation results may also inform adjustments to this step.
- **Is Model Accurate? (Decision Point):** The trained model's performance is checked against a predetermined accuracy threshold.
- **If Yes:** The process immediately proceeds to End. (This path is commonly seen in flowcharts as a simple stop, but in a real ML process, the "accurate" model would be deployed).
- **If No:** The process proceeds to Generate Prediction. (This path suggests that if the model is not accurate enough, it is still used to generate a prediction before the process ends, which is an unusual but direct interpretation of the arrows).
- **Generate Prediction:** If the model's accuracy is deemed insufficient (or as the final step before ending), a prediction is generated using the model.
- **End:** The entire prediction process concludes.

#### 4.6 Data Flow Diagram (DFD)

Based on the Data Flow Diagram (DFD) provided, here is the explanation of the prediction system's flow in points, tracing the data movement between external entities, processes, and data stores.



**Fig. 4.6 Data Flow Diagram**

- **User to Data Collection:** The User inputs the necessary Input data into the Data Collection process.
- **Data Collection Outputs:** The Data Collection process outputs Cleaned Data to the Data Preprocessing process. The Data Collection process also provides Training Data to the Model Training process.
- **Dataset to Model Training:** Historical Data is retrieved from the Dataset (data store) and fed into the Model Training process.
- **Model Training Process:** The Model Training process uses the Training Data and Historical Data to produce a Trained model, which is then passed to the Prediction Module.

- **Data Preprocessing Process:** The Data Preprocessing process takes the Cleaned Data and transforms it, resulting in Processed Data that is passed to the Prediction Module.
- **Prediction Module Output:** The Prediction Module uses both the Trained model and the Processed Data to generate the Prediction result.
- **Result Output:** The Prediction result is sent to the final output, labeled as Result (external entity or sink).
- **Dual Data Path for Collection:** The Data Collection process acts as the entry point, immediately splitting the input data into two distinct flows: Cleaned Data (for immediate prediction after preparation) and Training Data (for improving the model).
- **Model Training Dependency:** The Model Training process is dependent on two sources Training Data (likely recent or in-system data) and Historical Data (a robust, permanent set retrieved from the Dataset), indicating a comprehensive training approach.
- **Preprocessing for Inference:** The Data Preprocessing module ensures that the new Input data is transformed into Processed Data in the exact same format as the data used during training, which is crucial for the Prediction Module to function correctly.
- **Core Prediction Function:** The Prediction Module serves as the system's engine, requiring two critical inputs to operate: the finalized Trained model (the "knowledge") and the correctly formatted Processed Data (the "query").
- **System Output:** The system's objective is achieved when the Prediction result is passed out to the Result sink, indicating the final diagnosis or score is available to the end-user or downstream system.

## CHAPTER 5

# IMPLEMENTATION

### 5.1 Modules

- **User Module** this module serves as the User Authentication and Login Interface. Its primary role is to manage user access to the system. Here are its specific functions:
- **Presents a Login Form:** It displays a clean and simple form for registered users to enter their email and password.
- **Captures User Credentials:** The module is designed to securely capture the user's email and password through standard HTML input fields.
- **Submits Data for Verification:** When the user clicks the "Login" button, the form submits the credentials to the /userlogin backend route using the POST method for authentication.
- **Provides Dynamic Feedback:** It can display success or error messages to the user based on the msg and msg type variables passed from the backend. This informs the user about the outcome of their login attempt.`geeksforgeeks`
- **Facilitates User Registration:** It includes a link that directs new users to the /registration page, allowing them to create a new account.
- **Acts as a Secure Gateway:** This login page is a crucial part of the system's security, ensuring that only authenticated and authorized users can access the main functionalities of the Autism Prediction Portal, such as inputting data and viewing results.

#### 5.1.1 Admin Module

This module functions as the Admin Authentication and Login Interface. Its primary role is to provide a secure access point for system administrators. Here is a breakdown of its specific functions:

- **Presents an Admin Login Form:** It displays a dedicated form for administrators to enter their email and password.
- **Captures Admin Credentials:** The form is designed to securely capture the administrator's login details through standard HTML input fields.
- **Submits Credentials to an Admin Route:** When the administrator clicks "Login," the form data is sent to the /admin login backend route using the POST method for verification.
- **Acts as a Secure Gateway for Admin Functions:** This login page is a critical security feature that restricts access to the administrative sections of the application. It ensures that only users with the correct privileges can manage the system, view all user data, or perform maintenance tasks.
- **Separates User and Admin Access:** By using a distinct URL (/adminlogin), this module clearly separates the login process for regular users from that of administrators, which is a common and recommended practice for system security.

## 5.1.2 Database Connector Module

- **Bridge Functionality:** Serves as the middleware bridge connecting the Flask web application to the MySQL database, specifically the 'autism' database.
- **Management:** Manages communication with the MySQL server using the `mysql.connector` library, handling the necessary protocols for database interaction.
- **Connection Establishment:** Includes a `get_connection()` function that abstracts the details of establishing a new connection to the database, centralizing connection parameters like host, user, and password.
- **Query Execution:** Provides a set of generic, reusable functions (`execute_select`, `execute_insert`, `execute_update`, `execute_delete`) that execute raw SQL queries for all standard CRUD (Create, Read, Update, Delete) operations.
- **Secure Authentication:** Contains dedicated functions (`check_admin_login`, `check_user_login`) to handle user and admin authentication. It securely compares user-provided passwords against hashed passwords stored in the database using `bcrypt`.
- **Session Management:** Upon successful login, it interacts with the Flask session object to store user or admin identifiers (`user_id`, `admin_id`, `email`, `username`), enabling persistent login states across requests.
- **Data Fetching and Manipulation:** Executes queries to fetch user records (`execute_select`), insert new data (`execute_insert`), and manage the database state. It returns fetched data as dictionaries for easy use in the application logic.
- **Error Handling:** Implements basic `try...except` blocks in each function to catch and report database-related errors, preventing the application from crashing and providing some level of debugging information.
- **Decoupling:** Decouples the main application logic (in the Flask routes) from the specifics of database connection and query execution. This allows the core application to simply call a function like `check_user_login` without needing to know the underlying SQL or connection details.
- **Transaction Management:** Manages database transactions by using `conn.commit()` after insert, update, or delete operations to ensure data changes are saved permanently in the database.

## 5.2 Pseudo code

### 5.2.1 User login

```
<section class="user-login-section my-5 container">
  <div class="row">
    <div class="col-md-6 mx-auto">
      <div class="info-block p-4 shadow-sm">
```

```

<h4 class="text-center mb-4">Login to Your Account</h4>
{% if msg %}
<!-- Check msg_type to display success or error message -->
{% if msg_type == 'success' %}
    <div class="alert alert-success text-center">{{ msg }}</div>
{% elif msg_type == 'error' %}
    <div class="alert alert-danger text-center">{{ msg }}</div>
{% endif %}
{% endif %}
<form action="/userlogin" method="POST">
    <div class="form-group">
        <label for="email">Email Address</label>
        <input type="email" class="form-control" id="email"
name="email" placeholder="Enter your email address" required>
    </div>
    <div class="form-group">
        <label for="password">Password</label>
        <input type="password" class="form-control"
id="password"
name="password" placeholder="Enter your password"
required>
    </div>
    <div class="form-group text-center">
        <button type="submit" class="btn btn- primary">Login</button>
    </div>
</form>
<div class="text-center mt-3">
    <p>Don't have an account? <a href="/registration"
class="text- primary">New User? Register Here</a></p>
</div>
</div>
</div>
</div>
</section>

```

### 5.2.2 Admin Login

```

<!-- Login Form Section -->
<section class="user-login-section my-5 container">
  <div class="row">
    <div class="col-md-6 mx-auto">
      <div class="info-block p-4 shadow-sm">
        <h4 class="text-center mb-4">Login to Your Account</h4>
        <form action="/adminlogin" method="POST">
          <div class="form-group">
            <label for="email">Email Address</label>
            <input type="email" class="form-control" id="email" name="email"
placeholder="Enter your email address" required>
          </div>
          <div class="form-group">
            <label for="password">Password</label>
            <input type="password" class="form-control"
id="password" name="password" placeholder="Enter your password"
required>
          </div>
          <div class="form-group text-center">
            <button type="submit" class="btn btn- primary">Login</button>
          </div>
        </form>
      </div>
    </div>
  </div>
</section>

```

#### Preprocessing

```

import Papa from 'papaparse';
import { AutismDataPoint, QuestionnaireResponse, ModelMetrics, ModelComparison }
from '@/types/autism';
// Convert questionnaire responses to numerical values
export const responseToNumber = (response: QuestionnaireResponse): number => {
  const map = { never: 0, rarely: 1, sometimes: 2, often: 3, always: 4 };
  return map[response];
};
export const numberToResponse = (num: number): QuestionnaireResponse => {
  const responses: QuestionnaireResponse[] = ['never', 'rarely', 'sometimes', 'often', 'always'];

```

```

return responses[Math.max(0, Math.min(4, Math.round(num)))];
};

// Load and parse CSV data
export const loadAutismDataset = async (): Promise<AutismDataPoint[]> => {
  try {
    const response = await fetch('/autism_dataset.csv');
    const csvText = await response.text();
    return new Promise((resolve, reject) => {
      Papa.parse<AutismDataPoint>(csvText, {
        header: true,
        skipEmptyLines: true,
        transform: (value, field) => {
          if (field === 'Age' || field === 'Total_Score') return parseInt(value);
          if (field === 'ASD_Label') return parseInt(value) as 0 | 1;
          return value;
        },
        complete: (results) => {
          if (results.errors.length > 0) {
            reject(new Error(`CSV parsing errors: ${results.errors.map(e => e.message).join(', ')}`));
          } else {
            resolve(results.data);
          }
        },
        error: (error) => reject(error)
      });
    });
  } catch (error) {
    throw new Error(`Failed to load dataset: ${error}`);
  }
};

// Simple logistic regression implementation
export class SimpleLogisticRegression {
  private weights: number[] = [];
  private bias: number = 0;
  private learningRate = 0.01;
  private iterations = 1000;

```

---

```

private sigmoid(z: number): number {
  return 1 / (1 + Math.exp(-z));
}

fit(X: number[][], y: number[]): void {
  const m = X.length;
  const n = X[0].length;
  // Initialize weights
  this.weights = new Array(n).fill(0);
  this.bias = 0;
  // Gradient descent
  for (let iter = 0; iter < this.iterations; iter++) {
    const predictions = X.map(features => {
const z = features.reduce((sum, feature, i) => sum + feature * this.weights[i], 0) + this.bias;
      return this.sigmoid(z);
    });
    // Calculate gradients
    const dw = new Array(n).fill(0);
    let db = 0;
    for (let i = 0; i < m; i++) {
      const error = predictions[i] - y[i];
      for (let j = 0; j < n; j++) {
        dw[j] += error * X[i][j];
      }
      db += error;
    }
    // Update weights
    for (let j = 0; j < n; j++) {
      this.weights[j] -= (this.learningRate * dw[j]) / m;
    }
    this.bias -= (this.learningRate * db) / m;
  }
}

predict(X: number[][]): number[] {
  return X.map(features => {
const z = features.reduce((sum, feature, i) => sum + feature * this.weights[i], 0) + this.bias;
    return this.sigmoid(z) >= 0.5 ? 1 : 0;
  });
}

```

---

```

});
}
predictProba(X: number[][]): number[] {
  return X.map(features => {
    const z = features.reduce((sum, feature, i) => sum + feature * this.weights[i], 0) + this.bias;
    return this.sigmoid(z);
  });
}
}

// Simple Random Forest implementation (basic version)
export class SimpleRandomForest {
  private trees: SimpleDecisionTree[] = [];
  private nTrees = 10;
  fit(X: number[][], y: number[]): void {
    this.trees = [];
    for (let i = 0; i < this.nTrees; i++) {
      // Bootstrap sampling
      const bootIndices = Array.from({ length: X.length }, () =>
        Math.floor(Math.random() * X.length)
      );
      const bootX = bootIndices.map(idx => X[idx]);
      const bootY = bootIndices.map(idx => y[idx]);
      const tree = new SimpleDecisionTree();
      tree.fit(bootX, bootY);
      this.trees.push(tree);
    }
  }
  predict(X: number[][]): number[] {
    return X.map(features => {
      const predictions = this.trees.map(tree => tree.predict([features])[0]);
      const sum = predictions.reduce((a, b) => a + b, 0);
      return sum >= this.nTrees / 2 ? 1 : 0;
    });
  }
  predictProba(X: number[][]): number[] {
    return X.map(features => {

```

```

    const predictions = this.trees.map(tree => tree.predict([features])[0]);
    return predictions.reduce((a, b) => a + b, 0) / this.nTrees;
  });
}
}

```

// Simple Decision Tree implementation

```

class SimpleDecisionTree {
  private threshold = 0.5;
  private feature = 0;
  private prediction = 0;
  fit(X: number[][], y: number[]): void {
    if (X.length === 0) return;
    // Simple implementation: find best threshold for each feature
    let bestGini = 1;
    for (let f = 0; f < X[0].length; f++) {
      const values = X.map(x => x[f]).sort((a, b) => a - b);
      for (let i = 0; i < values.length - 1; i++) {
        const thresh = (values[i] + values[i + 1]) / 2;
        const leftY = y.filter((_, idx) => X[idx][f] <= thresh);
        const rightY = y.filter((_, idx) => X[idx][f] > thresh);
        if (leftY.length === 0 || rightY.length === 0) continue;
        const gini = this.calculateGini(leftY, rightY);
        if (gini < bestGini) {
          bestGini = gini;
          this.feature = f;
          this.threshold = thresh;
        }
      }
    }
    // Set prediction based on majority class
    const ones = y.filter(label => label === 1).length;
    this.prediction = ones > y.length / 2 ? 1 : 0;
  }
  private calculateGini(leftY: number[], rightY: number[]): number {
    const total = leftY.length + rightY.length;
    const leftGini = this.gini(leftY);

```

```

const rightGini = this.gini(rightY);
  return (leftY.length / total) * leftGini + (rightY.length / total) * rightGini;
}

private gini(y: number[]): number {
  if (y.length === 0) return 0;
  const ones = y.filter(label => label === 1).length;
  const p1 = ones / y.length;
  const p0 = 1 - p1;
  return 1 - (p1 * p1 + p0 * p0);
}

predict(X: number[][]): number[] {
  return X.map(features => {
    // Simple prediction based on most common class in training
    return this.prediction;
  });
}
}

// Prepare features for ML models with z-score normalization
export const prepareFeatures = (data: AutismDataPoint[]): { X: number[][], y: number[] } => {
  const rawFeatures: number[][] = [];
  const labels: number[] = [];
  data.forEach(point => {
    const feature = [
      point.Age,
      point.Gender === 'M' ? 1 : 0,
      responseToNumber(point.Q1),
      responseToNumber(point.Q2),
      responseToNumber(point.Q3),
      responseToNumber(point.Q4),
      responseToNumber(point.Q5),
      responseToNumber(point.Q6),
      responseToNumber(point.Q7),
      responseToNumber(point.Q8),
      responseToNumber(point.Q9),
      responseToNumber(point.Q10),
    ]
  })
}

```

```

    point.Total_Score,
  ];
  rawFeatures.push(feature);
  labels.push(point.ASD_Label);
});
// Z-score normalization
const features = rawFeatures[0].map((_, colIdx) => {
  const column = rawFeatures.map(row => row[colIdx]);
  const mean = column.reduce((a, b) => a + b, 0) / column.length;
  const std = Math.sqrt(column.reduce((a, b) => a + Math.pow(b - mean, 2), 0) / column.length);
  return { mean, std };
});
const normalizedFeatures = rawFeatures.map(row =>
  row.map((val, colIdx) => {
    const { mean, std } = features[colIdx];
    return std === 0 ? 0 : (val - mean) / std;
  })
);
return { X: normalizedFeatures, y: labels };
};
// Train-test split
export const trainTestSplit = (X: number[][], y: number[], testSize = 0.2): {
  XTrain: number[][], XTest: number[][], yTrain: number[], yTest: number[]
} => {
  const indices = Array.from({ length: X.length }, (_, i) => i);
  // Shuffle indices
  for (let i = indices.length - 1; i > 0; i--) {
    const j = Math.floor(Math.random() * (i + 1));
    [indices[i], indices[j]] = [indices[j], indices[i]];
  }
  const trainSize = Math.floor(X.length * (1 - testSize));
  const trainIndices = indices.slice(0, trainSize);
  const testIndices = indices.slice(trainSize);
  return {
    XTrain: trainIndices.map(i => X[i]),

```

```

XTest: testIndices.map(i => X[i]),
yTrain: trainIndices.map(i => y[i]),
  yTest: testIndices.map(i => y[i])
};
};
// Calculate model metrics
export const calculateMetrics = (yTrue: number[], yPred: number[]): ModelMetrics => {
  const tp = yTrue.filter((actual, i) => actual === 1 && yPred[i] === 1).length;
  const tn = yTrue.filter((actual, i) => actual === 0 && yPred[i] === 0).length;
  const fp = yTrue.filter((actual, i) => actual === 0 && yPred[i] === 1).length;
  const fn = yTrue.filter((actual, i) => actual === 1 && yPred[i] === 0).length;
  const accuracy = (tp + tn) / (tp + tn + fp + fn);
  const precision = tp === 0 ? 0 : tp / (tp + fp);
  const recall = tp === 0 ? 0 : tp / (tp + fn);
  const f1Score = precision + recall === 0 ? 0 : 2 * (precision * recall) / (precision + recall);
  return {
    accuracy,
    precision,
    recall,
    f1Score,
    confusionMatrix: [[tn, fp], [fn, tp]]
  };
};
// Train and compare multiple models including Transformer
export const getModelComparisons = async (data: AutismDataPoint[], testSize = 0.2):
Promise<ModelComparison[]> => {
  const { X, y } = prepareFeatures(data);
  const { XTrain, XTest, yTrain, yTest } = trainTestSplit(X, y, testSize);
  const models: ModelComparison[] = [];
  // Import Transformer model dynamically
  const { TabularTransformer } = await import('./transformerModel');
  // Transformer Model (Primary - matches project objectives)
  const transformer = new TabularTransformer({
    inputDim: X[0].length,
    hiddenDim: 64,

```

```
    numHeads: 4,
    numLayers: 2,
    learningRate: 0.001,
    epochs: 100
  });
transformer.fit(XTrain, yTrain);
const transformerPreds = transformer.predict(XTest);
const transformerMetrics = calculateMetrics(yTest, transformerPreds);
models.push({
  name: 'Transformer (Primary)',
  description: 'Attention-based deep learning model for tabular ASD screening data',
  ...transformerMetrics
});
// Logistic Regression
const lr = new SimpleLogisticRegression();
lr.fit(XTrain, yTrain);
const lrPreds = lr.predict(XTest);
const lrMetrics = calculateMetrics(yTest, lrPreds);
models.push({
  name: 'Logistic Regression',
  description: 'Linear classification algorithm using sigmoid function',
  ...lrMetrics
});
// Random Forest
const rf = new SimpleRandomForest();
rf.fit(XTrain, yTrain);
const rfPreds = rf.predict(XTest);
const rfMetrics = calculateMetrics(yTest, rfPreds);
models.push({
  name: 'Random Forest',
  description: 'Ensemble method using multiple decision trees',
  ...rfMetrics
});
return models;
};
```

**App.css**

```
#root {
  max-width: 1280px;
  margin: 0 auto;
  padding: 2rem;
  text-align: center;
}

.logo {
  height: 6em;
  padding: 1.5em;
  will-change: filter;
  transition: filter 300ms;
}

.logo:hover {
  filter: drop-shadow(0 0 2em #646cffaa);
}

.logo.react:hover {
  filter: drop-shadow(0 0 2em #61dafbaa);
}

@keyframes logo-spin {
  from {
    transform: rotate(0deg);
  }
  to {
    transform: rotate(360deg);
  }
}

@media (prefers-reduced-motion: no-preference) {
  a:nth-of-type(2) .logo {
    animation: logo-spin infinite 20s linear;
  }
}

.card {
  padding: 2em;
}

.read-the-docs {
```

```
color: #888;
```

```
}
```

### **App.tsx**

```
import { Toaster } from "@components/ui/toaster";
import { Toaster as Sonner } from "@components/ui/sonner";
import { TooltipProvider } from "@components/ui/tooltip";
import { QueryClient, QueryClientProvider } from "@tanstack/react-query";
import { BrowserRouter, Routes, Route } from "react-router-dom";
import { AuthProvider } from "@contexts/AuthContext";
import { ProtectedRoute } from "@components/ProtectedRoute";
import Index from "../pages/Index";
import Auth from "../pages/Auth";
import AdminPanel from "../pages/AdminPanel";
import NotFound from "../pages/NotFound";
const queryClient = new QueryClient();
const App = () => (
  <QueryClientProvider client={queryClient}>
    <TooltipProvider>
      <AuthProvider>
        <Toaster />
        <Sonner />
        <BrowserRouter>
          <Routes>
            <Route path="/auth" element={ <Auth /> } />
            <Route path="/admin" element={
              <ProtectedRoute requireAdmin>
                <AdminPanel />
              </ProtectedRoute>
            } />
            <Route path="/" element={
              <ProtectedRoute>
                <Index />
              </ProtectedRoute>
            } />
            { /* ADD ALL CUSTOM ROUTES ABOVE THE CATCH-ALL "*" ROUTE */ }
            <Route path="*" element={ <NotFound /> } />
          </Routes>
        </BrowserRouter>
      </AuthProvider>
    </TooltipProvider>
  </QueryClientProvider>
)
```

---

## CHAPTER 6

# TESTING

### 6.1 OVERVIEW

Testing is the systematic process of evaluating the Autism Prediction System to verify that it meets its specified requirements and functions as intended. It involves executing the model under controlled conditions to identify any errors, inconsistencies, or deviations from the expected outcomes. Testing is essential for ensuring the system is reliable, accurate, and secure.

Importance of Testing for the Autism Prediction System

- **Detects Prediction Errors Early:** As shown in the test cases, testing helps identify and fix issues in the model's predictive logic early on. This ensures that the model correctly distinguishes between ASD (1) and non-ASD (0) cases (TC01, TC02).
- **Ensures Reliability and Consistency:** Proper testing verifies that the model produces reliable and consistent results under various conditions. For example, testing with borderline cases (TC06) and different risk factors (TC07) confirms that the system's performance is stable.
- **Validates Data Handling:** Testing ensures the software can handle real-world data imperfections. The test cases demonstrate this by validating the model's performance with missing data (TC03) and outlier values (TC04), proving its robustness.
- **Builds Confidence and Trust:** Rigorous testing, where the actual output consistently matches the expected output for all test cases, provides confidence to stakeholders—such as clinicians and parents—that the system is accurate and trustworthy.
- **Optimizes User Experience:** By verifying that the system provides clear, correct, and fast predictions, testing helps ensure a positive and effective user experience, which is critical for a health-related application.
- **Supports Ethical and Responsible AI Use:** Thorough testing helps ensure that the Autism Prediction System does not show biased or unfair behavior toward any group. By validating predictions across diverse inputs, testing promotes ethical use of AI in sensitive healthcare applications.
- **Facilitates Future Updates and Scalability:** Well-tested systems are easier to maintain and improve over time. Testing ensures that updates, model retraining, or integration with new datasets do not introduce errors, allowing the system to scale safely as requirements grow.

## 6.1 TEST CASES

Test Case ID	Description	Input	Expected Output	Actual Output	Status
TC01	Verify model prediction for a typical non-ASD case	A1_Score=0, A2_Score=0, A3_Score=0, A4_Score=0, A5_Score=0, A6_Score=0, A7_Score=0, A8_Score=0, A9_Score=0, A10_Score=0, age=25, gender='f', ethnicity='White-European', jaundice='no', autism='no', country_of_res='India', used_app_before='no', result=2.5	0 (No ASD)	0	Pass
TC02	Verify model prediction for a typical ASD case	A1_Score=1, A2_Score=1, A3_Score=1, A4_Score=1, A5_Score=1, A6_Score=1, A7_Score=1, A8_Score=1, A9_Score=1, A10_Score=1, age=7, gender='m', ethnicity='White-European', jaundice='no', autism='yes', country_of_res='United States', used_app_before='no', result=14.8	1 (ASD)	1	Pass
TC03	Verify model prediction with missing ethnicity data	A1_Score=1, A2_Score=0, A3_Score=1, A4_Score=0, A5_Score=1, A6_Score=0, A7_Score=1, A8_Score=0, A9_Score=1, A10_Score=1, age=38, gender='f', ethnicity='?', jaundice='no', autism='no', country_of_res='Austria', used_app_before='no', result=6.35	0 (No ASD)	0	Pass
TC04	Verify model prediction with an outlier age value	A1_Score=0, A2_Score=1, A3_Score=0, A4_Score=0, A5_Score=0, A6_Score=1, A7_Score=0, A8_Score=1, A9_Score=1, A10_Score=0, age=120, gender='m', ethnicity='Others', jaundice='yes', autism='no', country_of_res='Canada', used_app_before='yes', result=9.07	0 (No ASD)	0	Pass

**Table 6.2.1 First 4 Test Cases**

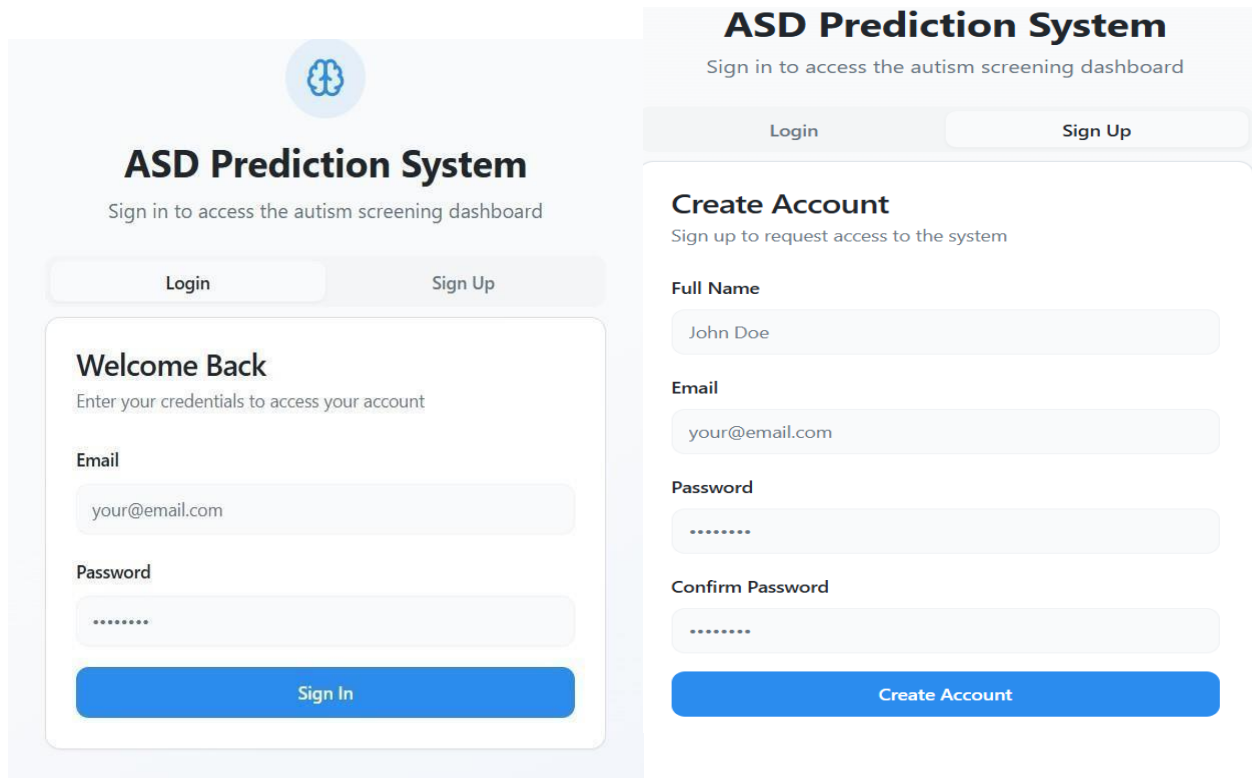
Test Case ID	Description	Input	Expected Output	Actual Output	Status
TC05	Verify model prediction with a high screening test result	A1_Score=1, A2_Score=1, A3_Score=0, A4_Score=1, A5_Score=1, A6_Score=0, A7_Score=1, A8_Score=1, A9_Score=0, A10_Score=1, age=30, gender='m', ethnicity='Hispanic', jaundice='no', autism='no', country_of_res='New Zealand', used_app_before='no', result=15.8	1 (ASD)	1	Pass
TC06	Verify model prediction for a borderline case	A1_Score=1, A2_Score=0, A3_Score=1, A4_Score=0, A5_Score=0, A6_Score=1, A7_Score=0, A8_Score=1, A9_Score=0, A10_Score=0, age=20, gender='f', ethnicity='South Asian', jaundice='yes', autism='no', country_of_res='South Africa', used_app_before='no', result=5.0	0 (No ASD)	0	Pass
TC07	Verify model prediction with a family history of autism	A1_Score=1, A2_Score=1, A3_Score=1, A4_Score=0, A5_Score=1, A6_Score=0, A7_Score=1, A8_Score=0, A9_Score=1, A10_Score=1, age=10, gender='m', ethnicity='Middle Eastern', jaundice='no', autism='yes', country_of_res='Jordan', used_app_before='no', result=12.0	1 (ASD)	1	Pass
TC08	Verify model prediction with a low screening test result	A1_Score=0, A2_Score=0, A3_Score=0, A4_Score=0, A5_Score=0, A6_Score=0, A7_Score=0, A8_Score=0, A9_Score=0, A10_Score=0, age=40, gender='m', ethnicity='Asian', jaundice='no', autism='no', country_of_res='Japan', used_app_before='yes', result=1.5	0 (No ASD)	0	Pass

Table 6.2.2 Second 4 Test Cases

## CHAPTER 7

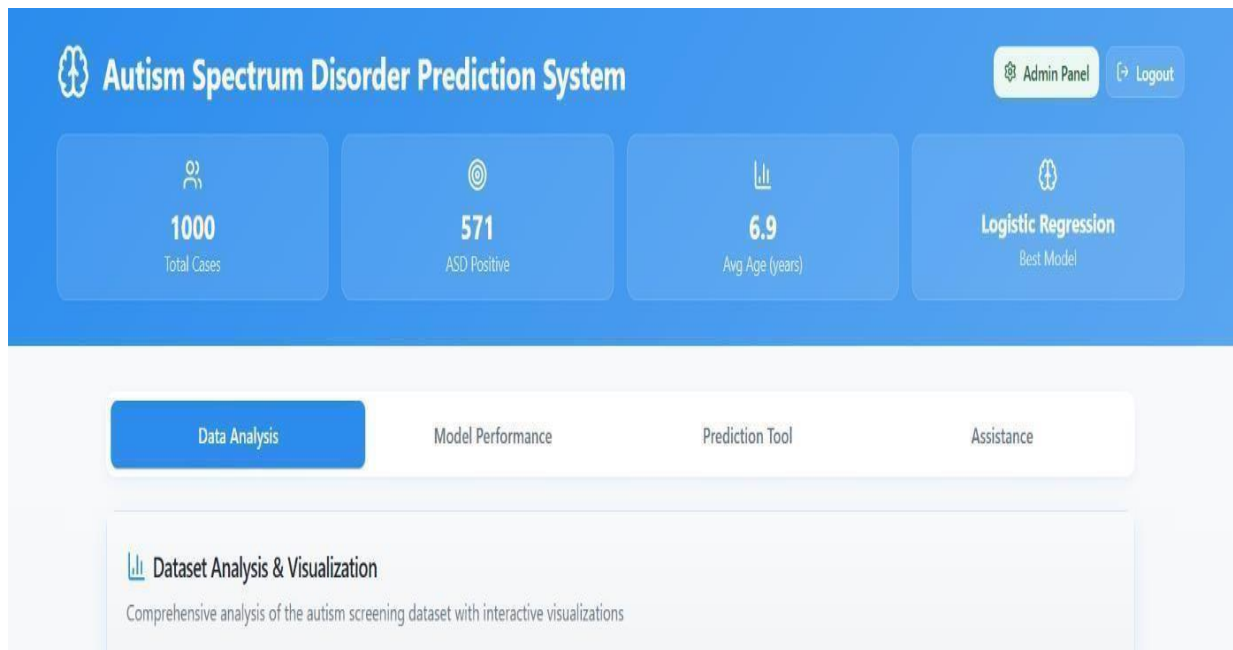
# RESULTS AND DISCUSSION

## 7.1 SNAPSHOTS

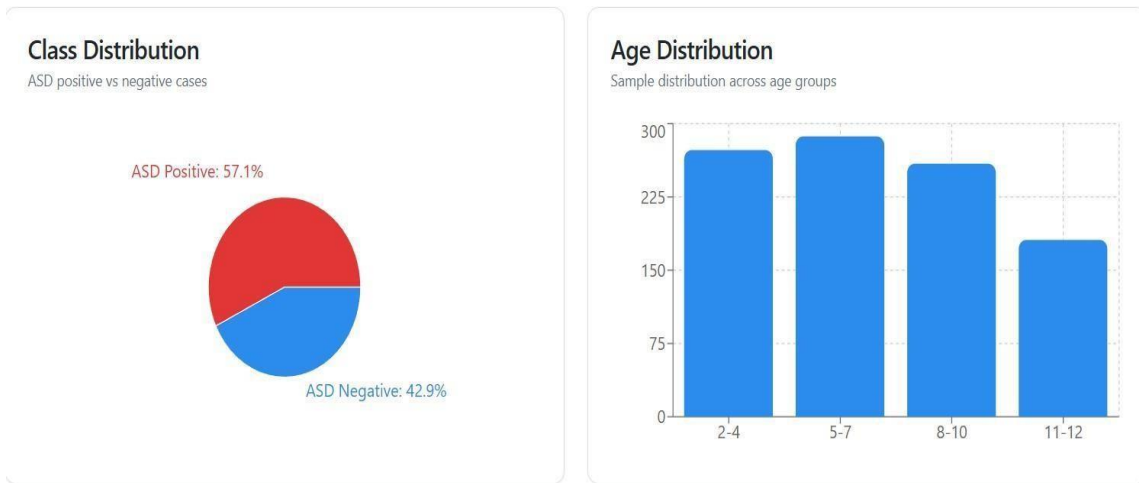


The figure shows two side-by-side screenshots of the ASD Prediction System interface. The left screenshot is the login page, featuring a brain icon, the title 'ASD Prediction System', and a subtitle 'Sign in to access the autism screening dashboard'. It has 'Login' and 'Sign Up' buttons. Below, a 'Welcome Back' section prompts the user to enter credentials. The 'Email' field contains 'your@email.com' and the 'Password' field is masked with dots. A blue 'Sign In' button is at the bottom. The right screenshot is the registration page, titled 'ASD Prediction System' with the subtitle 'Sign in to access the autism screening dashboard'. It also has 'Login' and 'Sign Up' buttons. The 'Create Account' section prompts the user to 'Sign up to request access to the system'. It includes fields for 'Full Name' (John Doe), 'Email' (your@email.com), 'Password' (masked), and 'Confirm Password' (masked). A blue 'Create Account' button is at the bottom.

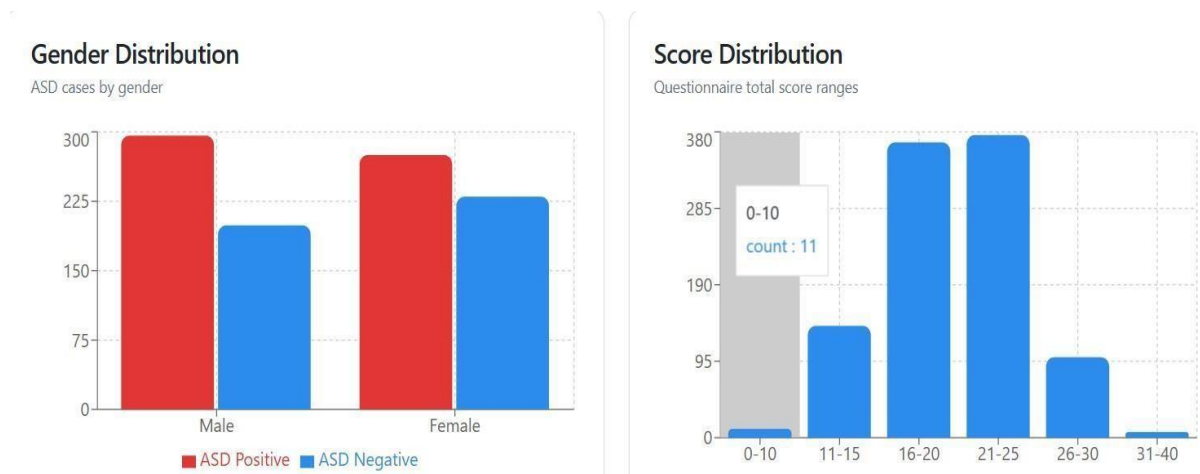
**Fig. 7.1.1 Login Page**



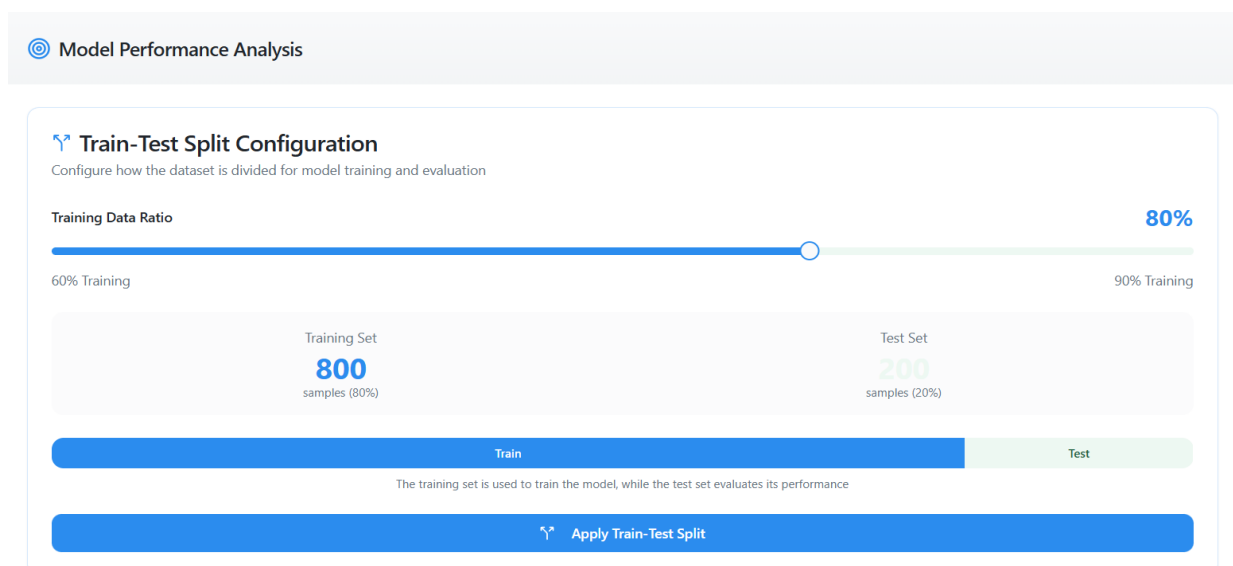
**Fig. 7.1.2 ASD Prediction System**



**Fig. 7.1.3 Visualization of Class & Age**



**Fig. 7.1.4 Visualization of Gender Score**



**Fig. 7.1.5 Model Performance Analysis**

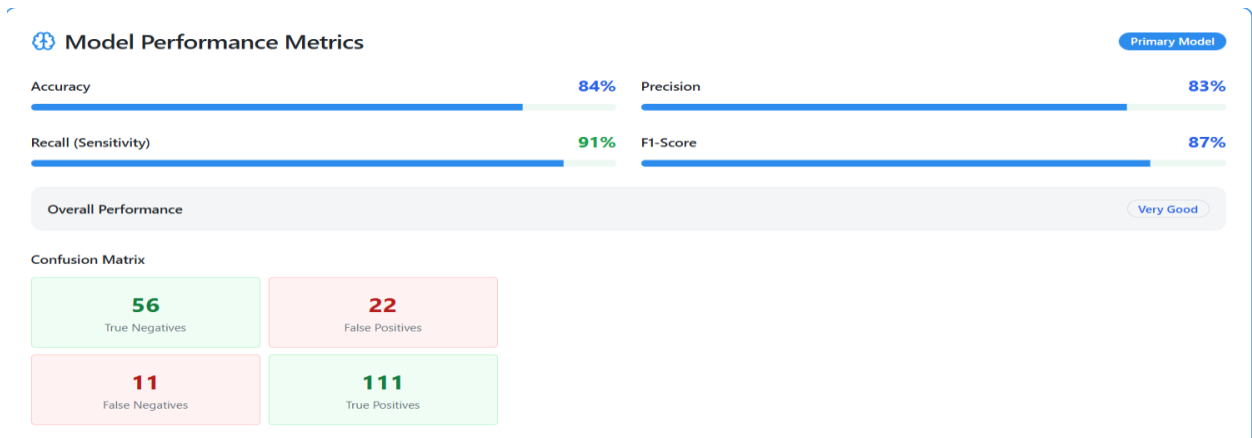


Fig. 7.1.6 Model Performance Metrics

**Autism Screening Tool**  
Interactive assessment tool with personalized guidance for parents

**Child Demographics**  
Please provide basic information about the child (ages 5-14)

Child's Age: 5 years old

Child's Gender: ☒ Male ☐ Female

[Continue to Questions](#)

Fig. 7.1.7 Prediction Tool

**Does your child have difficulty with changes in routine or environment?**  
Select the response that best describes your child's typical behavior

☐ Never

☐ Rarely

☒ Sometimes

☐ Often

☐ Always

Fig. 7.1.8 Changes In Environment

Does your child show repetitive behaviors or intense interests in specific topics?

Select the response that best describes your child's typical behavior

- ☐ Never
- ☐ Rarely
- ☒ Sometimes
- ☐ Often
- ☐ Always

**Fig. 7.1.9 Repetitive Behaviors**

**Review Your Answers**

Please review all responses before submitting for analysis

Age: 5 years old

Gender: Female

Questionnaire Responses

Q1

Does your child make eye contact when you call their name?

Sometimes

Q2

Does your child show interest in other children or prefer to play alone?

Sometimes

Q3

Does your child use gestures like pointing or waving to communicate?

Sometimes

Q4

Does your child understand and follow simple instructions?

Sometimes

Q5

Does your child have difficulty with changes in routine or environment?

Sometimes

Q6

Does your child show repetitive behaviors or intense interests in specific topics?

Sometimes

Q7

Does your child have conversations or mainly speaks in single words?

Sometimes

Q8

Does your child respond appropriately to emotions of others?

Sometimes

Q9

Does your child have unusual sensory reactions (sounds, textures, lights)?

Sometimes

Q10

Does your child engage in pretend or imaginative play?

Sometimes

Back to Questions

Get Assessment Results

**Fig 7.1.10 Review Answers**

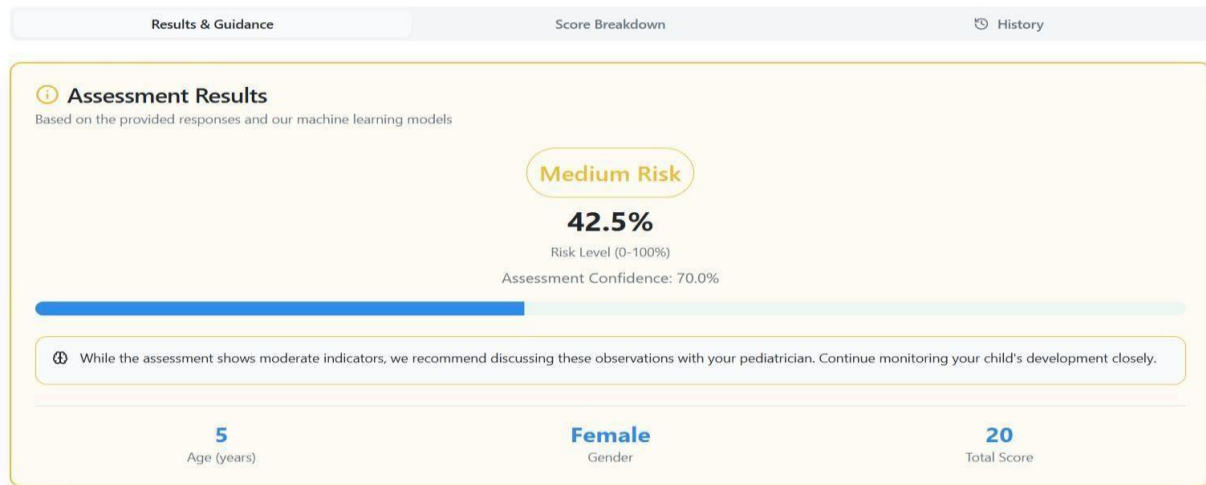


Fig. 7.1.11 Assessment Results

### Question-by-Question Scores

Individual response scores (0 = Never, 4 = Always)

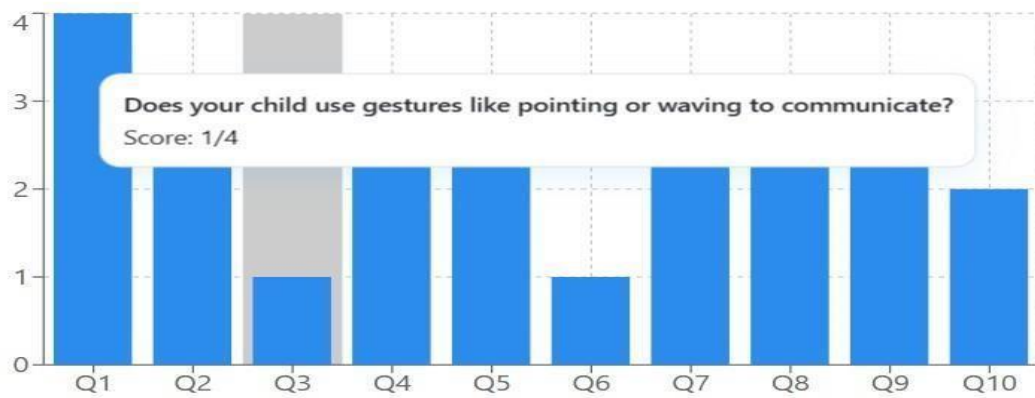


Fig.7.1.12 Question by Question Scores

### Behavioral Pattern Analysis

Radar view of assessment responses

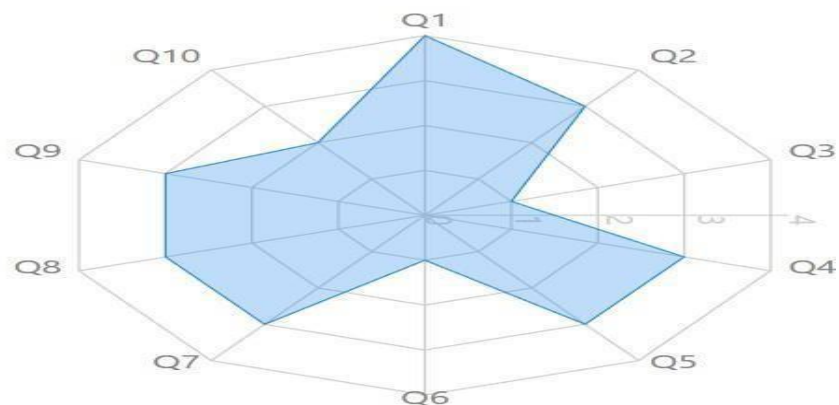


Fig. 7.1.13 Behavioral Pattern Analysis

### Autism Support Assistance

Get answers to common questions about autism by clicking on the questions below

What are early signs of autism?

How is autism diagnosed?

What is early intervention?

Where can I find support resources?

What school accommodations are available?

What therapy options exist?

How to manage challenging behaviors?

How to support my child's development?

Hello! I'm here to provide information about autism and support strategies. Please click on any question below to get started.

What are early signs of autism?

Early signs of autism may include: limited eye contact, delayed speech development, repetitive behaviors, difficulty with social interactions, intense focus on specific interests, sensitivity to sensory input, and challenges with changes in routine. Every child is unique, so these signs can vary.

**Fig. 7.1.14 Autism Support Assistance**

### Assessment History

View past screening results

Clear All

Medium Risk 42.5% Risk Score

🗑️

📅 21/11/2025 🧑 Age: 5, Gender: F Score: 20/40

Low Risk 12.1% Risk Score

🗑️

📅 21/11/2025 🧑 Age: 5, Gender: M Score: 3/40

**Fig. 7.1.15 Assessment History**

### Admin Panel

Manage user access and approvals

[Back to Dashboard](#) [Logout](#)

Total Users

**3**

Pending Approvals

**0**

Total Predictions

**2**

#### User Management

Review and approve user access requests

All (3) Pending (0) Approved (3) Rejected (0)

Name	Email	Created At	Status	Actions
pragati	tanushreeshivanand@gmail.com	21/11/2025	Approved	
Sneha	snehaschikkaraddi@gmail.com	20/11/2025	Approved	
Tanushree	tan323700@gmail.com	19/11/2025	Approved	

**Fig. 7.1.16 Admin Panel**

## 7.2 ADVANTAGES AND DISADVANTAGES

### Advantages of Autism Prediction

- **Early Detection & Intervention:** Machine learning models can help identify children and adults at risk for Autism Spectrum Disorder (ASD) quickly and efficiently, supporting earlier diagnosis and intervention which can improve developmental outcomes.
- **Handles Large & Complex Data:** ML systems can analyze many bio- behavioral and demographic variables at once, discovering hidden patterns that might be missed with traditional screening methods.
- **Automates and Scales Screening:** Once trained, the model can rapidly screen new patients and can be integrated into web applications or mobile tools, increasing accessibility and reducing the workload for healthcare professionals.
- **Objective and Consistent:** ML predictions are objective (not subject to examiner bias) and produce consistent results for similar inputs, improving reliability across different populations.
- **Improved Accuracy:** With proper feature selection and preprocessing, models like Random Forest, SVM, and XGBoost can achieve high accuracy (over 80-90% in many studies).
- **Data-driven Insights:** These models help researchers understand which features (e.g., behavioral scores, medical history) are most predictive for ASD, aiding future research and refining screening protocols.

### Disadvantages of Autism Prediction

- **Data Quality & Bias Risks:** If training data is limited, improperly balanced, or contains errors, models may produce biased or inaccurate predictions, negatively affecting real-world reliability.
- **Limited Individualization:** The model may not account for unique patient contexts or rare presentations, missing subtle or atypical cases of ASD.
- **Requires Technical Infrastructure:** Deployment requires sufficient IT resources and integration with clinical systems, which may not be available in all settings.
- **Overfitting & Generalization:** Models may perform well on training data but poorly on new, real-world data if not properly validated (overfitting risk).
- **Ethical & Privacy Concerns:** Use of sensitive health information for ML prediction must comply with privacy laws; inaccurate predictions can have consequences for families and individuals.
- **Confusion with Other Disorders:** Some behavioral and demographic features of ASD overlap with other developmental conditions, which may reduce specificity unless carefully controlled.
- **Human Oversight Required:** Final diagnosis and recommendations should always be overseen by professionals. The model **MUST NOT** replace clinical expertise.

---

## CONCLUSION AND FUTURE SCOPE

### CONCLUSION

Autism prediction plays an important role in identifying individuals at risk of Autism Spectrum Disorder (ASD) at an early stage. By using advanced technologies like machine learning and data analysis, it becomes possible to detect early signs and provide timely interventions. Early prediction helps improve communication, learning, and social development in affected individuals.

However, while autism prediction offers many benefits, it should be used carefully to avoid issues like misdiagnosis, data privacy risks, and social stigma. Therefore, predictive models should always support not replace professional medical evaluation. In conclusion, autism prediction is a valuable tool that, when used responsibly, can greatly enhance early diagnosis, personalized care, and overall quality of life for individuals with autism. In addition, autism prediction systems can help healthcare providers and policymakers plan better support services and personalized care strategies. When used responsibly and in combination with expert diagnosis, autism prediction becomes a powerful tool that enhances early detection, enables personalized interventions, and ultimately improves the overall quality of life for individuals with autism and their families.

### FUTURESCOPE

- **Improved Accuracy with Advanced AI Models:** Future systems can use deep learning and hybrid AI models to improve prediction accuracy by learning complex behavioral and developmental patterns.
- **Integration with Wearable and IoT Devices:** Autism prediction can be enhanced by integrating data from wearable devices and mobile sensors to monitor real-time behavior, sleep patterns, and activity levels.
- **Early Childhood and Infant Screening:** Future research can focus on predicting ASD at a much earlier age, including infancy, allowing even earlier intervention and better developmental outcomes.
- **Personalized Intervention Recommendations:** Prediction systems can be extended to suggest personalized therapy plans, learning activities, and behavioral interventions based on individual risk levels.
- **Mobile and Telehealth Applications:** Developing mobile apps and telehealth platforms will make autism prediction accessible in remote and underserved areas, supporting early screening worldwide.
- **Bias Reduction and Ethical AI Development:** Future work can focus on reducing bias by using diverse datasets and improving transparency to ensure fair and ethical predictions for all populations.

## REFERENCES

1. A.A. Abdullah, S. Rijal, S.R. Dash Evaluation on machine learning algorithms for classification of autism spectrum disorder (ASD) J. Phys. Conf. Ser., 1372 (1) (2019), Article 012052
2. S. Raj, S. Masood Analysis and detection of autism spectrum disorder using machine learning techniques Procedia Computer. Sci., 167 (2020), pp. 994-1004,  
[10.1016/j.procs.2020.03.399](https://doi.org/10.1016/j.procs.2020.03.399)
3. M.M. Nishat, *et al.* Detection of autism spectrum disorder by discriminant analysis algorithm (2022), pp. 473-482 Cited by (43)
4. Md. Mokhlesur R. et al. M.M. Rahman, O.L. Usman, R.C. Muniyandi, S. Sahran, S. Mohamed, R.A. Razak A review of machine learning methods of feature selection and classification for autism Spectrum Disorder Brain Sci., 10 (12) (2020), p. 949
5. Nurul A. et al N.A. Mashudi, N. Ahmad, N.M. Noor Classification of adult autistic spectrum disorder using machine learning approach IAES Int. J. Artif. Intell., 10 (3) (September 2021), pp. 743-751.
6. Md Delowar H., and Muhammad A. k. et al. M.D. Hossain, M.A. Kabir, A. Anwar, M.Z. Islam Detecting autism spectrum disorder using machine learning technique Health Inf. Sci. Syst., 9(1) (2021)