PREDICTING AUTISM SPECTRUM DISORDER USING MACHINE LEARNING CLASSIFIER

A Mini Project Report submitted to
JNTU Hyderabad in partial fulfillment
of the requirements for the award of the degree

BACHELOR OF TECHNOLOGY

In

COMPUTER SCIENCE AND ENGINEERING

Submitted by

BHAVANI PRANITHA	**21RG1A05L6**
GOUTI SRINIVAS SREENIDHI	**21RG1A05M7**
METHARI USHARANI	**21RG1A05P3**
VANGA SRUTHIKA	**21RG1A05R4**

Under the Guidance of
Ms .N. Meenakshi
B.Tech,M.Tech
Assistant Professor

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
MALLA REDDY COLLEGE OF ENGINEERING FOR WOMEN
An UGC Autonomous Institution
Approved by AICTE New Delhi and Affiliated to JNTUH
Maisammaguda , Medchal (Dist), Hyderabad -500100, Telangana.
OCTOBER 2024

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

MALLA REDDY COLLEGE OF ENGINEERING FOR WOMEN

An UGC Autonomous Institution

Approved by AICTE New Delhi and Affiliated to JNTUH

Maisammaguda , Medchal (Dist), Hyderabad -500100, Telangana.

OCTOBER 2024

CERTIFICATE

This is to certify that the Mini project entitled **“PREDICTING AUTISM SPECTRUM DISORDER USING MACHINE LEARNING CLASSIFIERS”** has been submitted by **BHAVANI PRANITHA (21RG1A05L6), GOUTI SRINIVAS SREENIDHI (21RG1A05M7), METHARI USHARANI (21RG1A05P3), VANGA SRUTHIKA (21RG1A05R4),** in partial fulfillment of the requirements for the award of **BACHELOR OF TECHNOLOGY** in **COMPUTER SCIENCE & ENGINEERING**. This record of bonafide work carried out by them under my guidance and supervision. ***The result embodied in this mini project report has not been submitted to any other University or Institute for the award of any degree.***

Ms. N. MEENAKSHI
Assistant professor
Project Guide

Mrs. K. SHEETAL KULKARNI
Head of the Department

External Examiner

ACKNOWLEDGEMENT

The Mini Project work carried out by our team in the Department of Computer Science and Engineering, Malla Reddy College of Engineering for Women, Hyderabad. **This work is original and has not been submitted in part or full for any degree or diploma of any other university.**

We wish to acknowledge our sincere thanks to our project guide **Ms. N.Meenakshi,** Assistant Professor, Computer Science & Engineering for formulation of the problem, analysis, guidance and her continuous supervision during the course of work.

We acknowledge our sincere thanks to **Dr.Kanaka Durga Returi**, Principal & Professor of Department of Computer Science and Engineering and **Mrs. Sheetal Kulkarni,** Head of the Department, MRCEW and all our faculties of CSE Department for their kind cooperation in making this Mini Project work a success.

We extend our gratitude to **Sri. Ch. Malla Reddy**, Founder Chairman and **Sri. Ch. Mahender Reddy,** Secretary, **Dr. Vaka Murali Mohan,** Director for their kind cooperation in providing the infrastructure for completion of our Mini Project.

We acknowledge our special thanks to the entire teaching faculty and non-teaching staff members of the Computer Science & Engineering Department for their support in making this project work a success.

BHAVANI PRANITHA	**HT.NO: 21RG1A05L6**
GOUTI SRINIVAS SREENIDHI	**HT.NO: 21RG1A05M7**
METHARI USHARANI	**HT.NO: 21RG1A05P3**
VANGA SRUTHIKA	**HT.NO: 21RG1A05R4**

INDEX

ABSTRACT

People with Autism Spectrum impairment (ASD) have difficulties in many areas of daily living due to this neurological impairment. Even though it's tough to get rid of this condition entirely early interventions can lessen its impact. Four distinct Feature Scaling (FS) algorithms—the Quantile Transformer (QT), the Power Transformer (PT), the Normalizer, and the Max Abs Scaler—are incorporated into the suggested architecture. Afterwards, eight straightforward machine learning algorithms such as Linear Discriminant Analysis (LDA) are used to categorize the feature-scaled datasets Ada Boost (AB), Decision Tree (DT), K-Nearest Neighbors (KNN), Gaussian Naïve Bayes (GNB), Logistic Regression (LR), Support Vector Machine (SVM), and. Four commonly used ASD datasets—Toddlers, Adolescents, Children, and Adults—are used in our investigations. By comparing the results of the classification processes with various statistical evaluation metrics, such as Log loss, Precision, Recall, Mathews Correlation Coefficient (MCC), and Receiver Operating Characteristic (ROC) curve, the top-performing classification methods and FS techniques for each ASD dataset are determined. Based on the results of the experiments conducted on feature-scaled ASD datasets, it was determined that AB achieved the best accuracy in predicting ASD for Toddlers (99.25%) and Children (97.95%), and for Adolescents (97.12%) and Adults (99.03%), respectively. For children and toddlers, the normalizer FS approach yields the best results, whereas for adults and teenagers, the QT FS method yields the best results. After that, we use four Feature Selection Techniques (FSTs)—the Info Gain Attribute Evaluator (IGAE), the Gain Ratio Attribute Evaluator (GRAE), the Relief F Attribute Evaluator (RFAE), and the Correlation Attribute Evaluator—to determine the ASD risk factors. Then, we rank the most important attributes according to their importance values. Thorough experimental evaluations like these show that ML technique fine-tuning is crucial for ASD prediction across age groups. In this study, we present a comprehensive feature importance analysis that we believe will help healthcare providers make better decisions when screening for autism spectrum disorder (ASD). Results from comparing the suggested framework to other methods for ASD early identification are encouraging.

LIST OF FIGURES

CHAPTER-1: SYSTEM ANALYSIS

1.1 EXISTING SYSTEM

The existing system for the detection of Autism Spectrum Disorder (ASD) was developed using the Ada Boost (AB) machine learning algorithm and achieved an impressive accuracy rate of 97.95%. This system represented a significant milestone in the application of machine learning to ASD diagnosis.

The existing system harnessed the power of the Ada Boost (AB) algorithm. which is a popular ensemble learning technique. Ada Boost combines multiple weak learners to create a strong and accurate model. It excels in enhancing classification accuracy by focusing on the instances that were previously misclassified in an iterative manner.

A standout feature of the earlier system was its exceptional accuracy rate of 97.95%. This high accuracy indicated that the Ada Boost model performed exceptionally well in distingung between individuals with ASD and those without. Such accurryviso particularly promising in the field of Healthcare,where the reliability of diagnostic tools is of paramount importance.

The existing system's proficiency in accurately detecting ASD holds significant clinical relevance. Early diagnosis of ASD is crucial for timely intervention and support for affected individuals. The high accuracy rate of the system suggests that it has the potential to contribute to early ASD diagnosis and consequently improved quality of life for affected individuals.

Ada Boost is known for its efficiency in handling both classification and regression problems. This efficiency could have practical implications, particularly in clinical settings where quick and accurate assessments are essential. Ada Boost's ability to learn from multiple weak classifiers and progressively improve its predictive power made it well-suited for the challenging task of ASD detection. This attribute likely played a crucial role in achieving the high accuracy observed in the earlier system.

1.1.1 DISADVANTAGES OF EXISTING SYSTEM:

- **Limited Transparency:** Ada Boost, like other ensemble methods, is not as interpretable as some simpler models like decision trees. Understanding the reasons behind the predictions made by the model can be challenging, which is a potential disadvantage, especially in healthcare applications where interpretability is crucial.

- **Computationally Intensive:** Ada Boost typically requires more computational resources and time compared to simpler algorithms due to its iterative nature. This increased computational demand may not be suitable for real-time or resource-constrained applications.
- **Limited Feature Engineering:** The existing system may not have explored advanced feature engineering techniques. Feature engineering is crucial in extracting the most relevant information from the data. A lack of advanced feature engineering might limit the model's ability to capture the complex relationships between variables.
- **Handling Imbalanced Data:** In healthcare applications, including ASD detection, datasets are often imbalanced, meaning that one class (e.g.. individuals with ASD) may have significantly fewer samples than the other class. Ada Boost can be sensitive to class imbalances, and the existing system may not have adequately addressed this issue.
- **Limited Scope:** The existing system's high accuracy, while impressive, may not address the broader scope of ASD diagnosis. Autism is a complex spectrum with varying degrees and presentations. The system may have limitations in accurately diagnosing different subtypes or stages of ASD.

1.2 PROPOSED SYSTEM

The proposed system for the detection of Autism SpectrumDisorder (ASD) presents an innovative Approach that addresses several key limitations observed in the existing system. Python-based machine learning models, specifically the Random Forest Classifier and the Decision Tree Classifier, to achieve enhanced accuracy and robustness in ASD diagnosis.

The proposed system leverages two distinct machine learning algorithms: the Random Forest Classifier and the Decision Tree Classifier. Random Forest is an ensemble learning method that combines multiple decision trees to create a highly accurate and stable model. Decision Tree, on the other hand, is a simple yet effective algorithm for classification tasks. proposed system has demonstrated impressive results in terms of accuracy.

The proposed system's findings have significant clinical relevance, as they can contribute to early ASD detection and timely intervention. Early diagnosis of ASD is essential for providing support to affected individuals and improving their quality of life. The system's high accuracy underscores its potential in enhancing the diagnostic process for ASD.

In conclusion, the proposed system for ASD detection offers a promising approach to overcoming the limitations of the earlier system. It harnesses the power of Python-based machine learning models, a comprehensive dataset, and a rich feature set to achieve remarkable accuracy in diagnosing ASD. These aspects collectively position the proposed system as a valuable tool for healthcare professionals working with ASD.

1.2.1 ADVANTAGES OF PROPOSED SYSTEM:

- **Robust Generalization**: The system's ability to achieve high accuracy on both the training and testing datasets suggests that it can effectively generalize to new, unseen case his robust generalization is vital in real- world clinical settings may present with diverse ethnicity, and country of residence. This consideration of diversity is essential in ensuring that the system can be applied across different populations and demographic groups.
- **Interpretability:** Decision trees, one of the models used in the system, are inherently more interpretable than complex models. This provides clinicians with insights into how the model is making its predictions, aiding in the understanding of the diagnostic process.
- **Clinical Relevance:** The proposed system's findings have direct clinical relevance and real-world implications. It can assist healthcare professionals in making informed decisions affig ASD diagnosis and intervention, sensory perception, cognitive abilities, demographics, and medical history.

The features in the dataset cover various aspects related to ASD diagnosis, including sensory perception (AI_Score A6_Score), cognitive abilities (A7_Score-A10_Score), demographic information (age, gender, ethnicity). medical history (jundice), autism diagnosis history (austim), country of residence, prior app usage, and other demographic attributes (Age desc, Relation). This rich feature set allows for a more holistic assessment of individuals.

The proposed system has demonstrated impressive results in terms of accuracy. The Random Forest Classifier achieved a remarkable training accuracy score of 100% and a testing accuracy score of 99%, indicating the model's ability to learn from the data and generalize effectively. The Decision Tree Classifier, while Maintaining a training accuracy score of 100%, achieved a testing agree of 96%, highlighting its robust.

- **Flexible Application:** Python-based machine learning models are highly versatile and can be readily adapted to various healthcare settings. The flexibility of the proposed system allows it to be integrated into different clinical environments.
- **Continuous Learning Potential:** Machine learning models like those in the proposed system can be updated and improved as new data and knowledge become available. This adaptability supports ongoing advancements in ASD diagnosis. In summary, the proposed system for ASD detection offers numerous advantages, including high accuracy, robust generalization, a rich feature set, early diagnosis capabilities, interpretability, clinical relevance, flexibility, and the potential for continuous learning. These advantages collectively position the proposed system as a valuable and reliable tool for healthcare professionals working with individuals on the autism spectrum.

1.3 INTRODUCTION

Autism Spectrum Disorder (ASD) is a neurodevelopmental disorder that manifests during the early years of an individual's existence and is deeply ingrained in their social interactions and functioning. Interactional challenges ASD is characterized by repetitive and restricted behavior patterns, and the term spectrum incorporates a extensive spectrum of symptoms and severity. Despite the absence of a sustainable cure for ASD, early intervention and appropriate medical care can significantly impact a child's development by directing attention towards enhancing behavioral patterns and communication abilities.

Despite this, the identification and diagnosis of ASD using conventional behavioral science remains a challenging and intricate process. In addition to being diagnosed at approximately two years of age, autism may also manifest later in life, depending on the severity of the condition. An assortment of treatment approaches are accessible to expedite the detection of ASD. In practice, these diagnostic procedures are not consistently implemented unless there is a significant likelihood of developing Autism Spectrum Disorder (ASD). A concise and observable inventory, presented by the authors, encompasses various developmental stages such as that of toddlers, children, adolescents, and adults. Following this, the authors developed the ASDTests mobile applications system, which utilizes a variety of questionnaire surveys, Q-CHAT, and AQ-10 methods, to identify ASD as quickly as feasible. As a result, they generated an open-source dataset incorporating data from mobile phone applications and uploaded the datasets to Kaggle and the University of California–Irvine (UCI) machine learning repository, both of which are publicly accessible, in order to further investigate this field. In an effort to analyze and diagnose ASD and other diseases, including diabetes, stroke, and heart failure prediction, as rapidly as feasible, numerous studies have been undertaken in recent years that employ diverse Machine Learning (ML) approaches.

By applying Rule-based ML (RML) techniques to the analysis of ASD attributes, the authors confirmed that RML assists classification models in improving their accuracy. By combining the Iterative Dichotomiser 3 (ID3) and Random Forest (RF) algorithms, the authors developed predictive models applicable to the demographics of children, adolescents, and adults. Integrating ADI-R and ADOS ML methods, the authors of this study presented a novel evaluation tool and implemented distinct attribute encoding strategies to address issues of data insufficiency, non-linearity, and inconsistency. An additional investigation carried out by the authors utilizes cognitive computing to establish feature-to-feature and feature-to-class correlation values. As diagnostic and prognostic classifiers for Autism Spectrum Disorder (ASD), they implemented Support Vector Machines (SVM), Decision Tree (DT), and Logistic

Regression (LR). Furthermore, the authors investigated cases involving traditionally formed (TD) (N = 19) and ASD (N = 11) situations, where the significance of the attributes was determined using a correlation-based attribute selection method. In 2015, while investigating ASD and TD in children, the authors identified fifteen preschool ASDs using only seven characteristics.

In addition, they stated that cluster analysis could be utilized to predict the phenotype and diversity of ASD by effectively analyzing complex patterns. For predicting ASD in adults, the authors compared and contrasted the performance of Linear Discrimination Analysis (LDA), K-Nearest Neighbors (KNN), SVM, Naive Bayes (NB), and K-Nearest Neighbors (KNN) classifiers. An ML model for autism detection was proposed in the paper via induction of rules; however, it was tested on a single dataset and comparisons were limited. The authors constructed an ML autism classification approach using LR analysis; however, this approach is also deficient in extensive validation and comparison. Five of the overall sixty-five characteristics examined by the authors in their analysis of autism data are adequate to diagnose ASD via attention deficit hyperactivity disorder. The authors developed an RF-based model in 2019 that utilized behavioral features to predict ASD. Furthermore, the authors employed LDA and KNN techniques to detect children with ASD ranging in age from four to eleven years. In 2018, the authors proposed an ASD model for children aged 4 to 11 based on the RF classifier.

By employing two distinct adult datasets, the authors assessed the predictive performance of the Deep Neural Network (DNN) in the diagnosis of Autism Spectrum Disorder (ASD). The authors of [18] developed a smartphone application programming interface for the diagnosis of ASDs of all ages using RF-CART and RF-ID3. In comparing the performance of various SVM kernels for classifying ASD data in children, the authors discovered that the polynomial kernel performed significantly better. The authors of [1] applied multiple feature selection techniques to four ASD datasets and discovered that the SVM classifier outperformed the child subset and adult subset based on the Boruta CFS intersect (BIC) method and the RIPPER-based toddler subset, correlation-based feature selection (CFS), and CFS-based adult subset. In addition, they ranked their features according to performance and applied the Shapley Additive Explanations (SHAP) method to different feature subsets, which produced the highest accuracy. To classify Parkinson's disease and ASD, the authors in [30] utilized ensemble ML techniques including Fuzzy K-Nearest Neighbor (FKNN), Kernel Support Vector Machines (KSVM), Fuzzy Convolution Neural Network (FCNN), and Random Forest (RF). The classification outcomes are subsequently validated through the implementation of Leave-One-Person-Out Cross Validation (LOPOCV). Utilizing an evolutionary cultural optimization algorithm, the authors of [31] optimized the weights of Artificial Neural Networks (ANN) used for autism screening on three benchmark datasets: toddlers, children, and adults. The experimental analysis conducted by the authors in reference [32] involved the utilization of sixteen distinct machine learning models. Among these models, four were bio-inspired algorithms: Gray Wolf Optimization (GWO), Flower Pollination Algorithm (FPA), Bat Algorithms (BA), and Artificial Bee Colony (ABC). The objective of these algorithms was to optimize the wrapper feature selection method so as to increase the accuracy of the classification models with respect to genetic and personal characteristics by selecting the most informative features. The authors also combined three benchmark datasets containing toddlers, adolescents, and adults in a study [33] in which they implemented a Light Gradient Boosting Machine (LGBM) classifier to categorize individuals with ASD. Extreme Learning Machines (ELM) and Random Vector Function Link (RVFL) generalization were implemented by the authors in [34].

CHAPTER-2: LITERATURE SURVEY

Fusar[1]-Poli L., Brondino N., Politi P., Aguglia E. Missed diagnoses and misdiagnoses of adults with autism spectrum disorder. Eur. Arch. Psychiatry Clin. Neurosci. 2020:1-12. doi: 10.1007/s00406-020-01189-w. [PMC free article] [PubMed] [CrossRef].

Lorah[2] E.R., Karnes A., Miller J., Welch-Beardsley J. Establishing Peer Manding in Young Children with Autism Using a Speech-Generating Device. J. Dev. Phys. Disabil. 2019;31:791-801. doi: 10.1007/s10882-019-09679-z. [CrossRef].

Vásquez[3] B., Del Sol M. Neuronal Morphology in Autism Spectrum Disorder. Int. J. Morphol. 2020;38:1513-1518.

Fernández[4]-Prieto M., Moreira C., Cruz S., Campos V., Martínez-Regueiro R., Taboada M., Carracedo A., Sampaio A. Executive Functioning: A Mediator Between Sensory Processing and Behaviour in Autism Spectrum Disorder.

Hofvander B[5], Bering S., Tärnhäll A., Wallinius M., Billstedt E. Few Differences in the Externalizing and Criminal History of Young Violent Offenders with and Without Autism Spectrum Disorders.

Au-Yeung SK[6], Kaakinen JK. Liversedge SP**,** et al Processing written irony in autism spectrum disorder: an eye-movement study in the year 2021 Participants with ASD and controls had the same performance while reading ironic nonironic texts. Participants of both groups required more time to read ironic statements.

Wei X. Christiano E. Yu W[7], et al.Reading and math in the year 2021 achievement profiles and longitudinal growth trajectories of children ASD Children with hyperlexia had slower improvement in conversational skills when compared with children with hypercalculia.

Davidson M. Weismer S[8] Characterization and prediction of early reading abilities in children on the autism spectrum in the year 2021 Children had reading levels within the normal range. Nonverbal cognition and expressive language abilities predicted reading performance.

Williamson P. Carnahan C. Jacobs J[9] Reading comprehension profiles of high- functioning students on the autism spectrum: a grounded theory in 2020 Approach based on the constructivist theory to study influences on reading comprehension of children with ASD.

Treffert D[10] Hyperlexia III: separating 'autistic- like behaviors from autistic disorder; assessing children who read early or speak late in 2020 Describes different conditions within the autism spectrum and the characteristics as hyperlexia and language delay.

Whalon K. Hart J[11] Children with autism spectrum disorder and literacy instruction: an exploratory study of elementary inclusive settings in 2020 The study shows the challenges faced by children with ASD in literacy instruction: few experiences of comprehension instruction is one of the most relevant.

Caruana N, Brock J[12] No association between autistic traits and contextual influences on eye-movements during reading in 2020 No support to the weak central coherence hypothesis nor to the 'comprehension monitoring alternative.

CHAPTER-3: SYSTEM DESIGN

3.1 System Modules

This project evaluates the performance of various data mining algorithm such as SVM, Logistic Regression, Gaussian Mixture Model (GMM), Artificial Neural Network (ANN) and EML (Extreme Machine Learning) to predict diabetes disease. Among all algorithms ANN is giving better prediction Accuracy.

3.1.1 SVM working details

Machine learning involves predicting and classifying data and to do so we employ various machine learning algorithms according to the dataset. SVM or Support Vector Machine is a linear model for classification and regression problems. It can solve linear and non-linear problems and work well for many practical problems. The idea of SVM is simple: The algorithm creates a line or a hyperplane which separates the data into classes. In machine learning, the radial basis function kernel, or RBF kernel, is a popular kernel function used in various kernelized learning algorithms. In particular, it is commonly used in support vector machine classification. As a simple example, for a classification task with only two features (like the image above), you can think of a hyperplane as a line that linearly separates and classifies a set of data.

3.1.2 ANN Working Procedure

To demonstrate how to build an ANN neural network-based image classifier, we shall build a 6 layer neural network that will identify and separate one image from other. This network that we shall build is a very small network that we can run on a CPU as well. Traditional neural networks that are very good at doing image classification have many more parameters and take a lot of time if trained on normal CPU. However, our objective is to show how to build a real-world convolutional neural network using TENSORFLOW.

Neural Networks are essentially mathematical models to solve an optimization problem. They are made of neurons, the basic computation unit of neural networks. A neuron takes an input (say x), do some computation on it (say: multiply it with a variable w and adds another variable b) to produce a value (say; z= wx + b). This value is passed to a non-linear function called activation function (f) to produce the final output (activation) of a neuron. There are many kinds of activation functions. One of the popular activation functions is Sigmoid. The neuron which uses sigmoid function as an activation function will be called sigmoid neuron. Depending on the activation functions, neurons are named and there are many kinds of them like RELU, TanH.

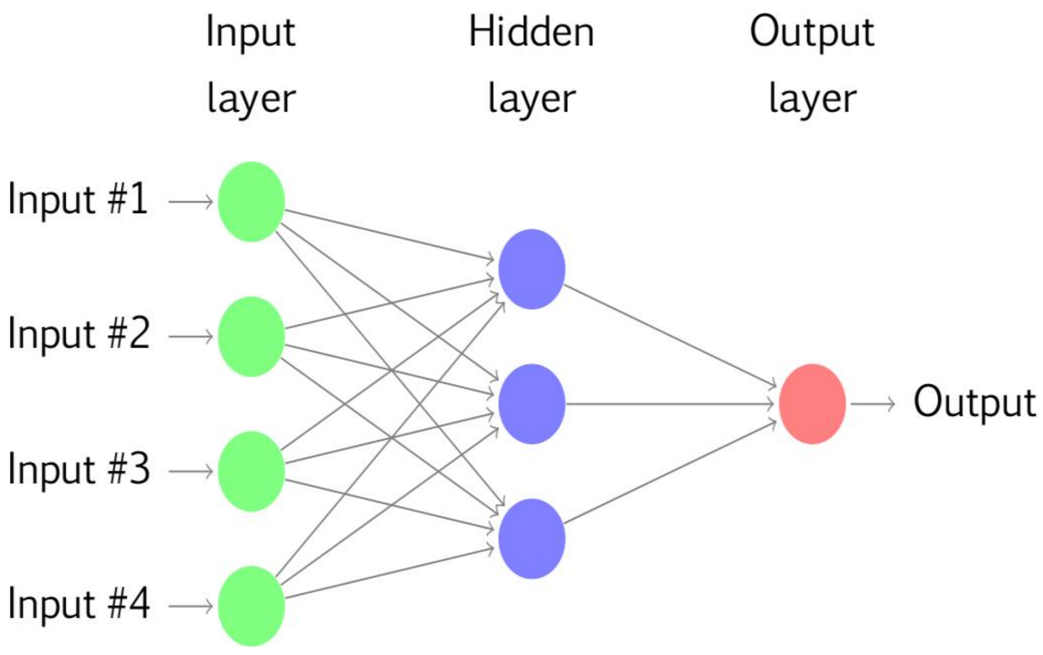


Fig 3.1 system module

To predict disease class multiple layers operate on each other to get best match layer and this process continues till no more improvement left. Extreme machine learning (EML) are feed forward neural networks for classification, regression, clustering, sparse approximation, compression and feature learning with a single layer or multiple layers of hidden nodes, where the parameters of hidden nodes (not just the weights connecting inputs to hidden nodes) need not be tuned. These hidden nodes can be randomly assigned and never updated (i.e. they are random projection but with nonlinear transforms), or can be inherited from their ancestors without being changed. In most cases, the output weights of hidden nodes are usually learned in a single step, which essentially amounts to learning a linear model. The name "extreme learning machine" (ELM) was given to such models by its main inventor Guang-Bin Huang. According to their creators, these models are able to produce good generalization performance and learn thousands of times faster than networks trained using backpropagation.

Dataset description

To implement this project we are using diabetes dataset and below are the dataset example

Pregnancies,Glucose,BloodPressure,SkinThickness,Insulin,BMI,DiabetesPedigreeFunction,Age,Outcome

6,148,72,35,0,33.6,0.627,50,1

1,85,66,29,0,26.6,0.351,31,0

8,183,64,0,0,23.3,0.672,32,1

1,89,66,23,94,28.1,0.167,21,0

In above dataset all bold names are the dataset column names and all integer numbers are the values and in last column we have value either 0 or 1 where 0 means values are normal and 1 means diabetes disease is present. We will build train model using above data mining algorithms and then apply new test records on trained model to predict whether disease is present in test data or not. Below are some records from test dataset

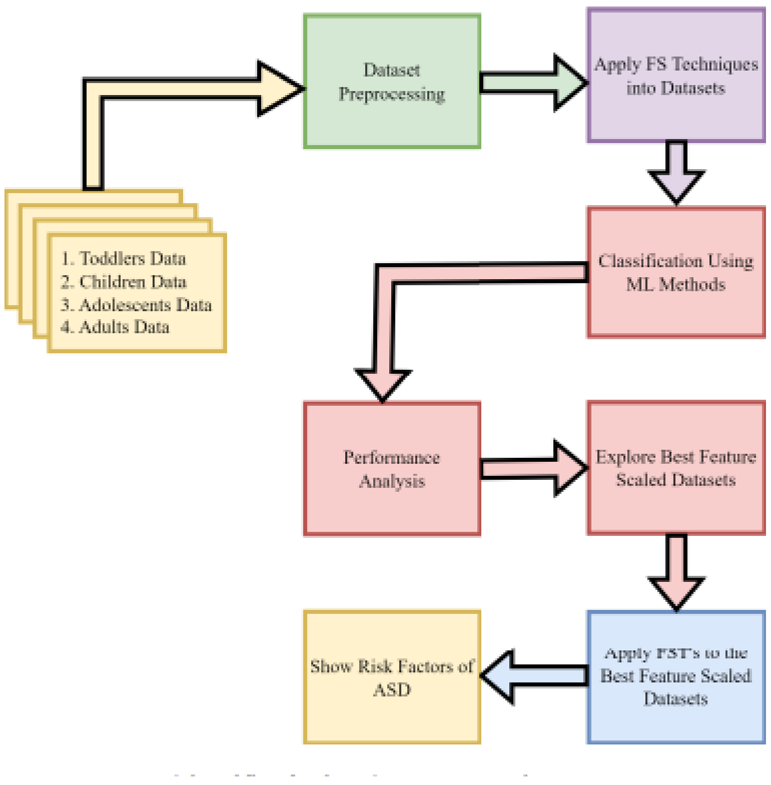
```
Pregnancies,Glucose,BloodPressure,SkinThickness,Insulin,BMI,DiabetesPedigreeFunction,Age

11,138,76,0,0,33.2,0.42,35
9,102,76,37,0,32.9,0.665,46
2,90,68,42,0,38.2,0.503,27
4,111,72,47,207,37.1,1.39,56
3,180,64,25,70,34,0.271,26
7,133,84,0,0,40.2,0.696,37
```

In above test data we have values but not class label as 0 or 1 and when we apply above test data on data mining algorithms then it will predict class label for above test records.

3.1.3 MODULES

- Upload Diabetes Dataset: In this module user upload dataset.
- Pre-process & Generate Train &Test Data: This module is used to read dataset and then divide dataset into train and test.
- Run GMM Algorithm: This module is used to Run GMM Algorithm.
- Run SVM Algorithm: This module is used to Run SVM Algorithm.
- Run ELM Algorithm: This module is used to Run ELM Algorithm.
- Run Logistic Regression Algorithm: This module is used to Run Logistic Regression Algorithm.
- Run ANN Algorithm: This module is used to Run ANN Algorithm.
- Run Accuracy Comparison Graph: This module is used to Accuracy Comparison Graph.
- Predict Disease on Test Data: In this module we predict diseases on test data.

3.2 System Architecture



1. Toddlers Data
2. Children Data
3. Adolescents Data
4. Adults Data
Dataset Preprocessing
Apply FS Techniques into Datasets
Classification Using ML Methods
Performance Analysis
Explore Best Feature Scaled Datasets
Apply FST's to the Best Feature Scaled Datasets
Show Risk Factors of ASD

Fig.3.2: System Architecture

3.3 System Requirements

3.3.1 Hardware Requirements

• System	:	Intel i-3, 5, 7 Processor.
• Hard Disk	:	500 GB.
• Floppy Drive	:	1.44 Mb.
• Monitor	:	14’ Color Monitor.
• Mouse	:	Optical Mouse.
• Ram	:	2GB.

3.3.2 Software Requirements:

• Operating system	:	Windows 7,8,10 Ultimate, Linux, Mac.
• Front-End	:	Python.
• Coding Language	:	Python.
• Software Environment	:	Anaconda (Jupyter or Spyder).

3.4 UML Diagrams

3.4.1 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.

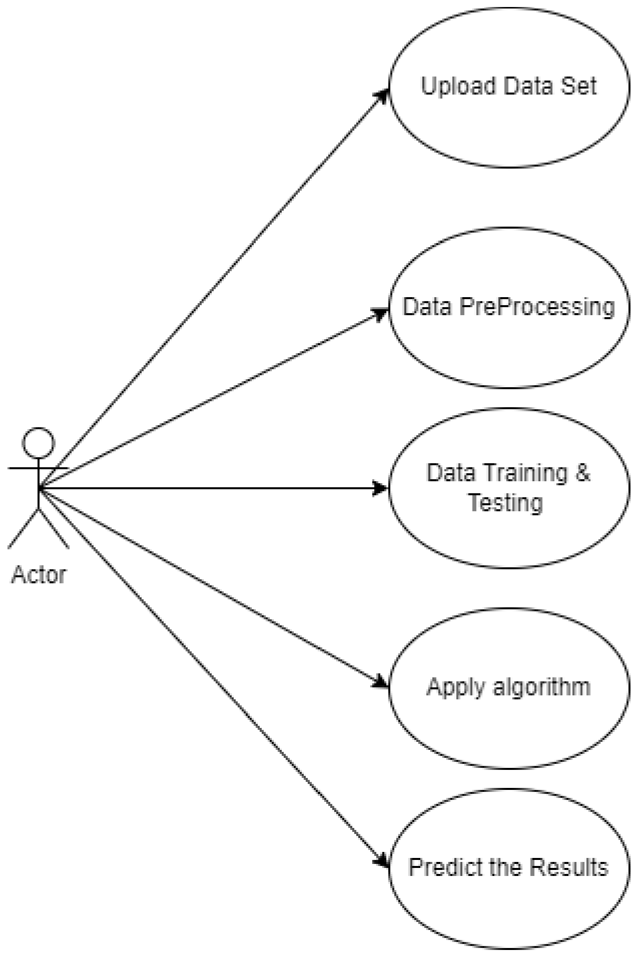


Fig. 3.3 : Use Case Diagrams

3.4.2 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.

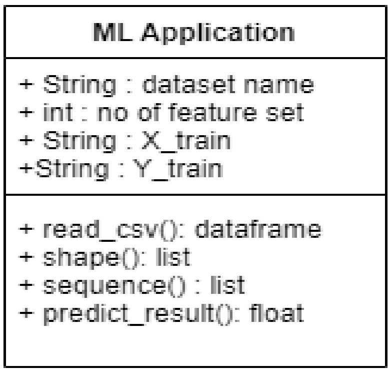


Fig. 3.4: Class Diagram

3.4.3 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 diagrams.

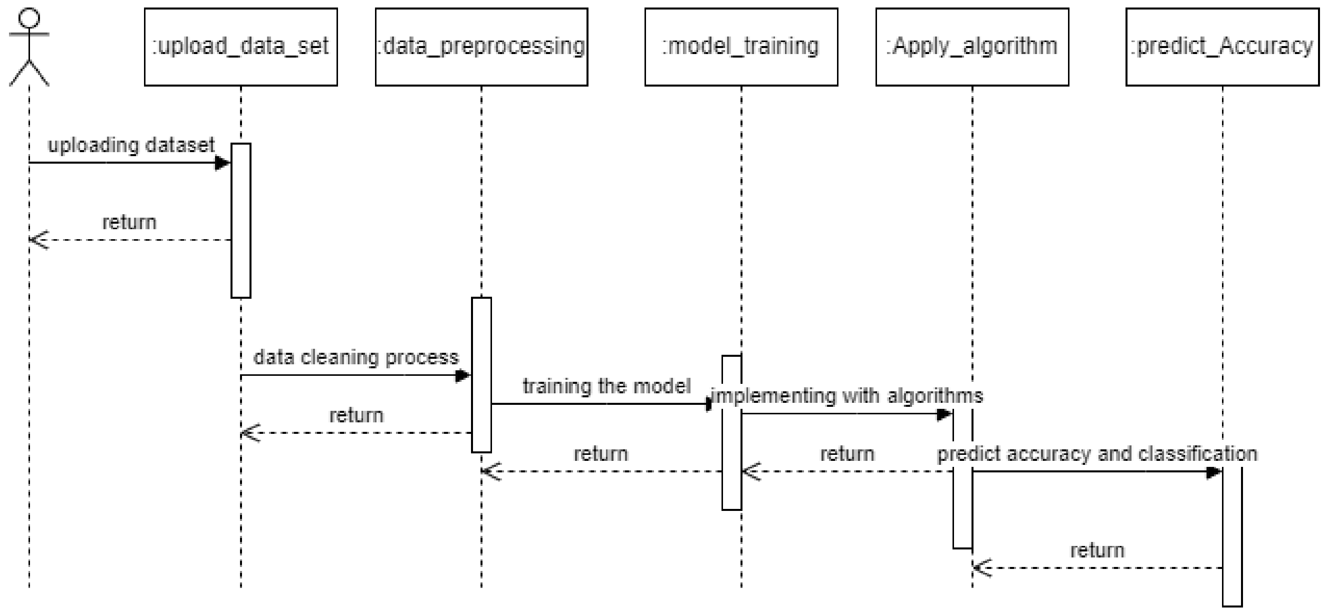


Fig. 3.5 : Sequence Diagram

CHAPTER-4: INPUT AND OUTPUT DESIGN

4.1 INPUT DESIGN

The input design for a machine learning classifier predicting Autism Spectrum Disorder (ASD) involves careful consideration of various data sources and preprocessing techniques. First, data collection should encompass a range of factors, including demographics (age, gender, ethnicity, socioeconomic status), clinical data (previous diagnoses, family history of ASD, developmental milestones), and behavioral assessments, such as standardized questionnaires like the Autism Diagnostic Observation Schedule (ADOS) and the Modified Checklist for Autism in Toddlers (M-CHAT). Neuropsychological tests measuring cognitive abilities, language skills, and social interactions should also be included.

Next, feature engineering is crucial. Quantitative features can include scores from the assessments and metrics derived from questionnaires, while qualitative features may involve categorical data that reflect the presence or absence of specific behaviors. Additionally, derived features can be created based on domain expertise, such as interaction terms or composite scores that enhance the model's predictive capabilities.

Data preprocessing is an essential step in preparing the input for the model. This includes normalization or standardization of numerical features to ensure consistent scales, handling missing values through imputation or exclusion, and encoding categorical variables using methods like one-hot encoding or label encoding. Ultimately, the input should be structured in a tabular format (e.g., CSV or DataFrame), with each row representing an individual case and each column corresponding.

4.2 OUTPUT DESIGN

The output design focuses on providing meaningful predictions and performance metrics. The classifier can generate outputs for both binary and multiclass classifications. In a binary classification scenario, the model predicts the presence (1) or absence (0) of ASD, while in a multiclass setting, it may differentiate between types or severity levels of ASD. The model's predictions should include not just the final class label but also associated probabilities indicating the likelihood of an individual being diagnosed with ASD. For instance, the output could indicate a probability score between 0 and 1.

Performance metrics are crucial for evaluating the classifier's effectiveness. Key metrics include accuracy (the overall correctness of the model), precision and recall (indicating true positive and false positive rates), F1 score (the harmonic mean of precision and recall), and ROC-AUC (the area under the receiver operating characteristic curve, useful for threshold analysis).

CHAPTER-5: SOFTWARE ENVIRONMENT

5.1 What is Machine Learning ?

Before we take a look at the details of various machine learning methods, let's start by looking at what machine learning is, and what it isn't. Machine learning is often categorized as a subfield of artificial intelligence, but I find that categorization can often be misleading at first brush. The study of machine learning certainly arose from research in this context, but in the data science application of machine learning methods, it's more helpful to think of machine learning as a means of building models of data.

Fundamentally, machine learning involves building mathematical models to help understand data. "Learning" enters the fray when we give these models tunable parameters that can be adapted to observed data; in this way the program can be considered to be "learning" from the data. Once these models have been fit to previously seen data, they can be used to predict and understand aspects of newly observed data. I'll leave to the reader the more philosophical digression regarding the extent to which this type of mathematical, model-based "learning" is similar to the "learning" exhibited by the human brain. Understanding the problem setting in machine learning is essential to using these tools effectively, and so we will start with some broad categorizations of the types of approaches we'll discuss here.

5.1.1 Categories of Machine Leaning

At the most fundamental level, machine learning can be categorized into two main types: supervised learning and unsupervised learning. Supervised learning involves somehow modeling the relationship between measured features of data and some label associated with the data; once this model is determined, it can be used to apply labels to new, unknown data. This is further subdivided into classification tasks and regression tasks: in classification, the labels are discrete categories, while in regression, the labels are continuous quantities. We will see examples of both types of supervised learning in the following section.

Unsupervised learning involves modeling the features of a dataset without reference to any label and is often described as "letting the dataset speak for itself." These models include tasks such as clustering and dimensionality reduction. Clustering algorithms identify distinct groups of data, while dimensionality reduction algorithms search for more succinct representations of the data. We will see examples of both types of unsupervised learning in the following section.

5.1.2 Need for Machine Learning

Human beings, at this moment, are the most intelligent and advanced species on earth because they can think, evaluate, and solve complex problems. On the other side, AI is still in its initial stage and have not surpassed human intelligence in many aspects. Then the question is that what is the need to make machine learn? The most suitable reason for doing this is, “to make decisions, based on data, with efficiency and scale”.

Lately, organizations are investing heavily in newer technologies like Artificial Intelligence, Machine Learning and Machine Learning to get the key information from data to perform several real-world tasks and solve problems. We can call it data-driven decisions taken by machines, particularly to automate the process. These data-driven decisions can be used, instead of using programing logic, in the problems that cannot be programmed inherently. The fact is that we can’t do without human intelligence, but other aspect is that we all need to solve real-world problems with efficiency at a huge scale. That is why the need for machine learning arises.

5.1.3 Challenges in Machines Learning

While Machine Learning is rapidly evolving, making significant strides with cybersecurity and autonomous cars, this segment of AI as whole still has a long way to go. The reason behind is that ML has not been able to overcome number of challenges. The challenges that ML is facing currently are −

1. Quality of data − Having good-quality data for ML algorithms is one of the biggest challenges. Use of low-quality data leads to the problems related to data preprocessing and feature extraction.
2. Time-Consuming task − Another challenge faced by ML models is the consumption of time especially for data acquisition, feature extraction and retrieval.
3. Lack of specialist persons − As ML technology is still in its infancy stage, availability of expert resources is a tough job.
4. No clear objective for formulating business problems − Having no clear objective and well-defined goal for business problems is another key challenge for ML because this technology is not that mature yet.
5. Issue of overfitting & underfitting − If the model is overfitting or underfitting, it cannot be represented well for the problem.
6. Curse of dimensionality − Another challenge ML model faces is too many features of data points. This can be a real hindrance.

7. Difficulty in deployment − Complexity of the ML model makes it quite difficult to be deployed in real life.

5.1.4 Applications of Machines Learning

Machine Learning is the most rapidly growing technology and according to researchers we are in the golden year of AI and ML. It is used to solve many real-world complex problems which cannot be solved with traditional approach. Following are some real-world applications of ML −

- Emotion analysis
- Sentiment analysis
- Error detection and prevention
- Weather forecasting and prediction
- Stock market analysis and forecasting
- Speech synthesis
- Speech recognition
- Customer segmentation
- Object recognition
- Fraud detection
- Fraud prevention
- Recommendation of products to customer in online shopping

5.1.5 How to Start Learning Machine Learning?

Arthur Samuel coined the term "Machine Learning" in 1959 and defined it as a "Field of study that gives computers the capability to learn without being explicitly programmed". And that was the beginning of Machine Learning! In modern times, Machine Learning is one of the most popular (if not the most!) career choices. According to Indeed, Machine Learning Engineer Is The Best Job of 2019 with a 344% growth and an average base salary of $146,085 per year.

But there is still a lot of doubt about what exactly is Machine Learning and how to start learning it? So this article deals with the Basics of Machine Learning and also the path you can follow to eventually become a full-fledged Machine Learning Engineer. Now let's get started!!!

How to start learning ML?

This is a rough roadmap you can follow on your way to becoming an insanely talented Machine Learning Engineer. Of course, you can always modify the steps according to your needs to reach your desired end-goal!

Step 1 – Understand the Prerequisites

In case you are a genius, you could start ML directly but normally, there are some prerequisites that you need to know which include Linear Algebra, Multivariate Calculus, Statistics, and Python. And if you don't know these, never fear! You don't need a Ph.D. degree in these topics to get started but you do need a basic understanding.

(a) Learn Linear Algebra and Multivariate Calculus:

Both Linear Algebra and Multivariate Calculus are important in Machine Learning. However, the extent to which you need them depends on your role as a data scientist. If you are more focused on application heavy machine learning, then you will not be that heavily focused on maths as there are many common libraries available. But if you want to focus on R&D in Machine Learning, then mastery of Linear Algebra and Multivariate Calculus is very important as you will have to implement many ML algorithms from scratch.

(b) Learn Statistics:

Data plays a huge role in Machine Learning. In fact, around 80% of your time as an ML expert will be spent collecting and cleaning data. And statistics is a field that handles the collection, analysis, and presentation of data. So it is no surprise that you need to learn it!!! Some of the key concepts in statistics that are important are Statistical Significance, Probability Distributions, Hypothesis Testing, Regression, etc. Also, Bayesian Thinking is also a very important part of ML which deals with various concepts like Conditional Probability, Priors, and Posteriors, Maximum Likelihood, etc.

(c) Learn Python:

Some people prefer to skip Linear Algebra, Multivariate Calculus and Statistics and learn them as they go along with trial and error. But the one thing that you absolutely cannot skip is Python! While there are other languages you can use for Machine Learning like R, Scala, etc. Python is currently the most popular language for ML. In fact, there are many Python libraries that are specifically useful for Artificial Intelligence and Machine Learning such as Keras, TensorFlow, Scikit-learn, etc. So if you want to learn ML, it's best if you learn Python! You can do that using various online resources and courses such as Fork Python available Free on GeeksforGeeks.

Step 2 – Learn Various ML Concepts

Now that you are done with the prerequisites, you can move on to actually learning ML (Which is the fun part!!!) It's best to start with the basics and then move on to the more complicated stuff. Some of the basic concepts in ML are:

(a) Terminologies of Machine Learning

- Model – A model is a specific representation learned from data by applying some machine learning algorithm. A model is also called a hypothesis.
- Feature – A feature is an individual measurable property of the data. A set of numeric features can be conveniently described by a feature vector. Feature vectors are fed as input to the model. For example, in order to predict a fruit, there may be features like color, smell, taste, etc.
- Target (Label) – A target variable or label is the value to be predicted by our model. For the fruit example discussed in the feature section, the label with each set of input would be the name of the fruit like apple, orange, banana, etc.
- Training – The idea is to give a set of inputs(features) and it's expected outputs(labels), so after training, we will have a model (hypothesis) that will then map new data to one of the categories trained on.
- Prediction – Once our model is ready, it can be fed a set of inputs to which it will provide a predicted output(label).

(b) Types of Machine Learning

- Supervised Learning – This involves learning from a training dataset with labeled data using classification and regression models. This learning process continues until the required level of performance is achieved.
- Unsupervised Learning – This involves using unlabelled data and then finding the underlying structure in the data in order to learn more and more about the data itself using factor and cluster analysis models.
- Semi-supervised Learning – This involves using unlabelled data like Unsupervised Learning with a small amount of labeled data. Using labeled data vastly increases the learning accuracy and is also more cost-effective than Supervised Learning.
- Reinforcement Learning – This involves learning optimal actions through trial and error. So the next action is decided by learning behaviors that are based on the current state and that will maximize the reward in the future.

5.1.6 Advantages of Machine Learning

1. Easily identifies trends and patterns -

Machine Learning can review large volumes of data and discover specific trends and patterns that would not be apparent to humans. For instance, for an e-commerce website like Amazon, it serves to understand the browsing behaviours and purchase histories of its users to help cater to the right products, deals, and reminders relevant to them. It uses the results to reveal relevant advertisements to them.

2. No human intervention needed (automation)

With ML, you don't need to babysit your project every step of the way. Since it means giving machines the ability to learn, it lets them make predictions and also improve the algorithms on their own. A common example of this is anti-virus software; they learn to filter new threats as they are recognized. ML is also good at recognizing spam.

3. Continuous Improvement

As ML algorithms gain experience, they keep improving in accuracy and efficiency. This lets them make better decisions. Say you need to make a weather forecast model. As the amount of data you have keeps growing, your algorithms learn to make more accurate predictions faster.

4. Handling multi-dimensional and multi-variety data

Machine Learning algorithms are good at handling data that are multi-dimensional and multi-variety, and they can do this in dynamic or uncertain environments.

5. Wide Applications

You could be an e-tailer or a healthcare provider and make ML work for you. Where it does apply, it holds the capability to help deliver a much more personal experience to customers while also targeting the right customers.

5.1.7 Disadvantages of Machine Learning

1. Data Acquisition

Machine Learning requires massive data sets to train on, and these should be inclusive/unbiased, and of good quality. There can also be times where they must wait for new data to be generated.

2. Time and Resources

ML needs enough time to let the algorithms learn and develop enough to fulfill their purpose with a considerable amount of accuracy and relevancy. It also needs massive resources to function. This can mean additional requirements of computer power for you.

3. Interpretation of Results

Another major challenge is the ability to accurately interpret results generated by the algorithms. You must also carefully choose the algorithms for your purpose.

4. High error-susceptibility

Machine Learning is autonomous but highly susceptible to errors. Suppose you train an algorithm with data sets small enough to not be inclusive. You end up with biased predictions coming from a biased training set. This leads to irrelevant advertisements being displayed to customers. In the case of ML, such blunders can set off a chain of errors that can go undetected for long periods of time. And when they do get noticed, it takes quite some time to recognize the source of the issue and even longer to correct it.

5.2 What is Python?

Below are some facts about Python.

- Python is currently the most widely used multi-purpose, high-level programming language.
- Python allows programming in Object-Oriented and Procedural paradigms. Python programs generally are smaller than other programming languages like Java.
- Programmers have to type relatively less and indentation requirement of the language, makes them readable all the time.
- Python language is being used by almost all tech-giant companies like – Google, Amazon, Facebook, Instagram, Dropbox, Uber… etc.

The biggest strength of Python is huge collection of standard library which can be used for the following –

- Machine Learning
- GUI Applications (like Kivy, Tkinter, PyQt etc.)
- Web frameworks like Django (used by YouTube, Instagram, Dropbox)
- Image processing (like Opencv, Pillow)
- Web scraping (like Scrapy, BeautifulSoup, Selenium)

5.2.1 Advantages of Python

Let’s see how Python dominates over other languages.

1. Extensive Libraries

Python downloads with an extensive library and it contain code for various purposes like regular expressions, documentation-generation, unit-testing, web browsers, threading, databases, CGI, email, image manipulation, and more. So, we don't have to write the complete code for that manually.

2. Extensible

As we have seen earlier, Python can be extended to other languages. You can write some of your code in languages like C++ or C. This comes in handy, especially in projects.

3. Embeddable

Complimentary to extensibility, Python is embeddable as well. You can put your Python code in your source code of a different language, like C++. This lets us add scripting capabilities to our code in the other language.

4. Improved Productivity

The language's simplicity and extensive libraries render programmers more productive than languages like Java and C++ do. Also, the fact that you need to write less and get more things done.

5. IOT Opportunities

Since Python forms the basis of new platforms like Raspberry Pi, it finds the future bright for the Internet Of Things. This is a way to connect the language with the real world.

6. Simple and Easy

When working with Java, you may have to create a class to print 'Hello World'. But in Python, just a print statement will do. It is also quite easy to learn, understand, and code. This is why when people pick up Python, they have a hard time adjusting to other more verbose languages like Java.

7. Readable

Because it is not such a verbose language, reading Python is much like reading English. This is the reason why it is so easy to learn, understand, and code. It also does not need curly braces to define blocks, and indentation is mandatory. This further aid's the readability of the code.

8. Object-Oriented

This language supports both the procedural and object-oriented programming paradigms. While functions help us with code reusability, classes and objects let us model the real world. A class allows the encapsulation of data and functions into one.

9. Free and Open-Source

Like we said earlier, Python is freely available. But not only can you download Python for free, but you can also download its source code, make changes to it, and even distribute it. It downloads with an extensive collection of libraries to help you with your tasks.

10. Portable

When you code your project in a language like C++, you may need to make some changes to it if you want to run it on another platform. But it isn't the same with Python. Here, you need to code only once, and you can run it anywhere. This is called Write Once Run Anywhere (WORA). However, you need to be careful enough not to include any system-dependent features.

11. Interpreted

Lastly, we will say that it is an interpreted language. Since statements are executed one by one, debugging is easier than in compiled languages.

Any doubts till now in the advantages of Python? Mention in the comment section.

5.2.2 Advantages of Python Over Other Languages

1. Less Coding

Almost all of the tasks done in Python requires less coding when the same task is done in other languages. Python also has an awesome standard library support, so you don't have to search for any third-party libraries to get your job done. This is the reason that many people suggest learning Python to beginners.

2. Affordable

Python is free therefore individuals, small companies or big organizations can leverage the free available resources to build applications. Python is popular and widely used so it gives you better community support.

The 2019 Github annual survey showed us that Python has overtaken Java in the most popular programming language category.

3. Python is for Everyone

Python code can run on any machine whether it is Linux, Mac or Windows. Programmers need to learn different languages for different jobs but with Python, you can professionally build web apps, perform data analysis and machine learning, automate things, do web scraping and also build games and powerful visualizations. It is an all-rounder programming language.

5.2.3 Disadvantages of Python

So far, we've seen why Python is a great choice for your project. But if you choose it, you should be aware of its consequences as well. Let's now see the downsides of choosing Python over another language.

1. Speed Limitations

We have seen that Python code is executed line by line. But since Python is interpreted, it often results in slow execution. This, however, isn't a problem unless speed is a focal point for the project. In other words, unless high speed is a requirement, the benefits offered by Python are enough to distract us from its speed limitations.

2. Weak in Mobile Computing and Browsers

While it serves as an excellent server-side language, Python is much rarely seen on the client-side. Besides that, it is rarely ever used to implement smartphone-based applications. One such application is called Carbonnelle.

The reason it is not so famous despite the existence of Brython is that it isn't that secure.

3. Design Restrictions

As you know, Python is dynamically-typed. This means that you don't need to declare the type of variable while writing the code. It uses duck-typing. But wait, what's that? Well, it just means that if it looks like a duck, it must be a duck. While this is easy on the programmers during coding, it can raise run-time errors.

4. Underdeveloped Database Access Layers

Compared to more widely used technologies like JDBC (Java Database Connectivity) and ODBC (Open Database Connectivity), Python's database access layers are a bit underdeveloped. Consequently, it is less often applied in huge enterprises.

5. Simple

No, we're not kidding. Python's simplicity can indeed be a problem. Take my example. I don't do Java, I'm more of a Python person. To me, its syntax is so simple that the verbosity of Java code seems unnecessary.

This was all about the Advantages and Disadvantages of Python Programming Language.

5.2.4 History of Python

What do the alphabet and the programming language Python have in common? Right, both start with ABC. If we are talking about ABC in the Python context, it's clear that the programming language ABC is meant. ABC is a general-purpose programming language and programming environment, which had been developed in the Netherlands, Amsterdam, at the CWI (Centrum Wiskunde &Informatica). The greatest achievement of ABC was to influence the design of Python. Python was conceptualized in the late 1980s. Guido van Rossum worked that time in a project at the CWI, called Amoeba, a distributed operating system. In an interview with Bill Venners[1], Guido van Rossum said: "In the early 1980s, I worked as an implementer on a team building a language called ABC at Centrum voor Wiskunde en Informatica (CWI). I don't know how well people know ABC's influence on Python. I try to mention ABC's influence because I'm indebted to everything I learned during that project and to the people who worked on it. "Later on in the same Interview, Guido van Rossum continued: "I remembered all my experience and some of my frustration with ABC. I decided to try to design a simple scripting language that possessed some of ABC's better properties, but without its problems. So I started typing. I created a simple virtual machine, a simple parser, and a simple runtime. I made my own version of the various ABC parts that I liked. I created a basic syntax, used indentation for statement grouping instead of curly braces or begin-end blocks, and developed a small number of powerful data types: a hash table (or dictionary, as we call it), a list, strings, and numbers."

5.2.5 Python Development Steps:

Guido Van Rossum published the first version of Python code (version 0.9.0) at alt. sources in February 1991. This release included already exception handling, functions, and the core data types of list, dict, str and others. It was also object oriented and had a module system. Python version 1.0 was released in January 1994. The major new features included in this release were the functional programming tools lambda, map, filter and reduce, which Guido Van Rossum never liked. Six and a half years later in October 2000, Python 2.0 was introduced. This release included list comprehensions, a full garbage collector and it was supporting unique code. Python flourished for another 8 years in the versions 2.x before the next major release as Python 3.0 (also known as "Python 3000" and "Py3K") was released. Python 3 is not backwards compatible with Python 2.x. The emphasis in Python 3 had been on the removal of duplicate programming constructs and modules, thus fulfilling or coming close to fulfilling the 13th law of the Zen of Python: "There should be one -- and preferably only one -- obvious way to do it. "Some changes in Python 7.3:

- Print is now a function.
- Views and iterators instead of lists
- The rules for ordering comparisons have been simplified. E.g., a heterogeneous list cannot be sorted, because all the elements of a list must be comparable to each other.
- There is only one integer type left, i.e., int. long is int as well.
- The division of two integers returns a float instead of an integer. "//" can be used to have the "old" behaviour.
- Text Vs. Data Instead of Unicode Vs. 8-bit

Purpose :

We demonstrated that our approach enables successful segmentation of intra-retinal layers-even with low-quality images containing speckle noise, low contrast, and different intensity ranges throughout—with the assistance of the ANIS feature.

Python :

Python is an interpreted high-level programming language for general-purpose programming. Created by Guido van Rossum and first released in 1991, Python has a design philosophy that emphasizes code readability, notably using significant whitespace.
Python features a dynamic type system and automatic memory management. It supports multiple programming paradigms, including object-oriented, imperative, functional and procedural, and has a large and comprehensive standard library.

- Python is Interpreted − Python is processed at runtime by the interpreter. You do not need to compile your program before executing it. This is similar to PERL and PHP.
- Python is Interactive − you can actually sit at a Python prompt and interact with the interpreter directly to write your programs.

Python also acknowledges that speed of development is important. Readable and terse code is part of this, and so is access to powerful constructs that avoid tedious repetition of code. Maintainability also ties into this may be an all but useless metric, but it does say something about how much code you have to scan, read and/or understand to troubleshoot problems or tweak behaviours. This speed of development, the ease with which a programmer of other languages can pick up basic Python skills and the huge standard library is key to another area where Python excels. All its tools have been quick to implement, saved a lot of time, and several of them have later been patched and updated by people with no Python background - without breaking.

5.2.6 Modules Used in Project :

TensorFlow :

TensorFlow is a free and open-source software library for dataflow and differentiable programming across a range of tasks. It is a symbolic math library and is also used for machine learning applications such as neural networks. It is used for both research and production at Google. TensorFlow was developed by the Google Brain team for internal Google use. It was released under the Apache 2.0 open-source license on November 9, 2015.

NumPy :

NumPy is a general-purpose array-processing package. It provides a high-performance multidimensional array object, and tools for working with these arrays.
It is the fundamental package for scientific computing with Python. It contains various features including these important ones:

- A powerful N-dimensional array object
- Sophisticated (broadcasting) functions
- Tools for integrating C/C++ and Fortran code
- Useful linear algebra, Fourier transform, and random number capabilities

Besides its obvious scientific uses, NumPy can also be used as an efficient multi-dimensional container of generic data. Arbitrary datatypes can be defined using NumPy which allows NumPy to seamlessly and speedily integrate with a wide variety of databases.

Pandas:

Pandas is an open-source Python Library providing high-performance data manipulation and analysis tool using its powerful data structures. Python was majorly used for data munging and preparation. It had very little contribution towards data analysis. Pandas solved this problem. Using Pandas, we can accomplish five typical steps in the processing and analysis of data, regardless of the origin of data load, prepare, manipulate, model, and analyze. Python with Pandas is used in a wide range of fields including academic and commercial domains including finance, economics, Statistics, analytics, etc.

Matplotlib

Matplotlib is a Python 2D plotting library which produces publication quality figures in a variety of hardcopy formats and interactive environments across platforms. Matplotlib can be used in Python scripts, the Python and I Python shells, the Jupyter Notebook, web application servers, and four graphical user interface toolkits. Matplotlib tries to make easy things easy and hard things possible. You can generate plots, histograms, power spectra, bar charts, error charts, scatter plots, etc., with just a few lines of code. For examples, see the sample plots and thumbnail gallery. For simple plotting the pyplot module provides a MATLAB-like interface, particularly when combined with I Python. For the power user, you have full control of line styles, font properties, axes properties, etc, via an object oriented interface or via a set of functions familiar to MATLAB users.

Scikit – learn

Scikit-learn provides a range of supervised and unsupervised learning algorithms via a consistent interface in Python. It is licensed under a permissive simplified BSD license

and is distributed under many Linux distributions, encouraging academic and commercial use. Python Python is an interpreted high-level programming language for general-purpose programming. Created by Guido van Rossum and first released in 1991, Python has a design philosophy that emphasizes code readability, notably using significant whitespace.
Python features a dynamic type system and automatic memory management. It supports multiple programming paradigms, including object-oriented, imperative, functional and procedural, and has a large and comprehensive standard library.

- Python is Interpreted − Python is processed at runtime by the interpreter. You do not need to compile your program before executing it. This is similar to PERL and PHP.
- Python is Interactive − you can actually sit at a Python prompt and interact with the interpreter directly to write your programs.

Python also acknowledges that speed of development is important. Readable and terse code is part of this, and so is access to powerful constructs that avoid tedious repetition of code. Maintainability also ties into this may be an all but useless metric, but it does say something about how much code you have to scan, read and/or understand to troubleshoot problems or tweak behaviours. This speed of development, the ease with which a programmer of other languages can pick up basic Python skills and the huge standard library is key to another area where Python excels. All its tools have been quick to implement, saved a lot of time, and several of them have later been patched and updated by people with no Python background - without breaking.

5.2.7 Install Python Step-by-Step in Windows and Mac

Python a versatile programming language doesn’t come pre-installed on your computer devices. Python was first released in the year 1991 and until today it is a very popular high-level programming language. Its style philosophy emphasizes code readability with its notable use of great whitespace. The object-oriented approach and language construct provided by Python enables programmers to write both clear and logical code for projects. This software does not come pre-packaged with Windows.

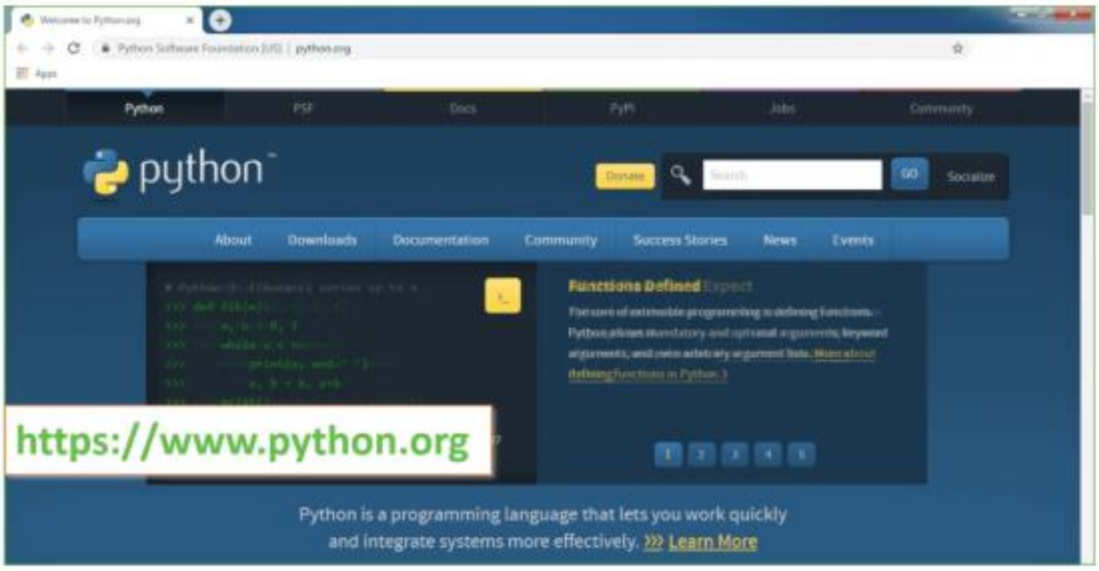
How to Install Python on Windows and Mac

There have been several updates in the Python version over the years. The question is how to install Python? It might be confusing for the beginner who is willing to start learning Python but this tutorial will solve your query. The latest or the newest version of Python is version 3.7.4 or in other words, it is Python 3.

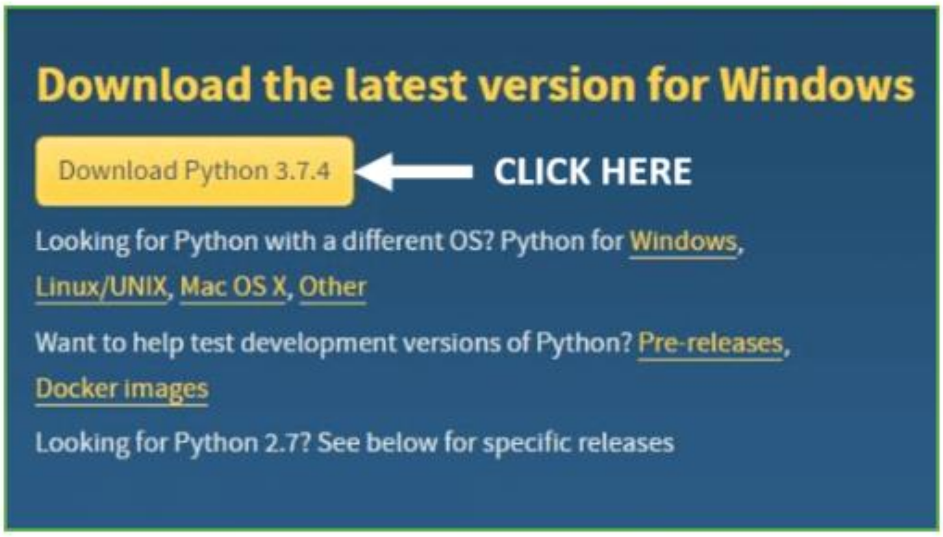
Note: The python version 3.7.4 cannot be used on Windows XP or earlier devices.
Before you start with the installation process of Python. First, you need to know about your System Requirements. Based on your system type i.e. operating system and based processor, you must download the python version. My system type is a Windows 64-bit operating system. So the steps below are to install python version 3.7.4 on Windows 7 device or to install Python 3. Download the Python Cheat sheet here The steps on how to install Python on Windows 10, 8 and 7 are divided into 4 parts to help understand better.

Download the Correct version into the system

Step 1: Go to the official site to download and install python using Google Chrome or any other web browser. OR Click on the following link: https://www.python.org


python
About
Downloads
Documentation
Community
Success Stories
News
Events
Functions Defined
https://www.python.org
Python is a programming language that lets you work quickly and integrate systems more effectively. >>> Learn More

Now, check for the latest and the correct version for your operating system.
Step 2: Click on the Download Tab.


Download the latest version for Windows

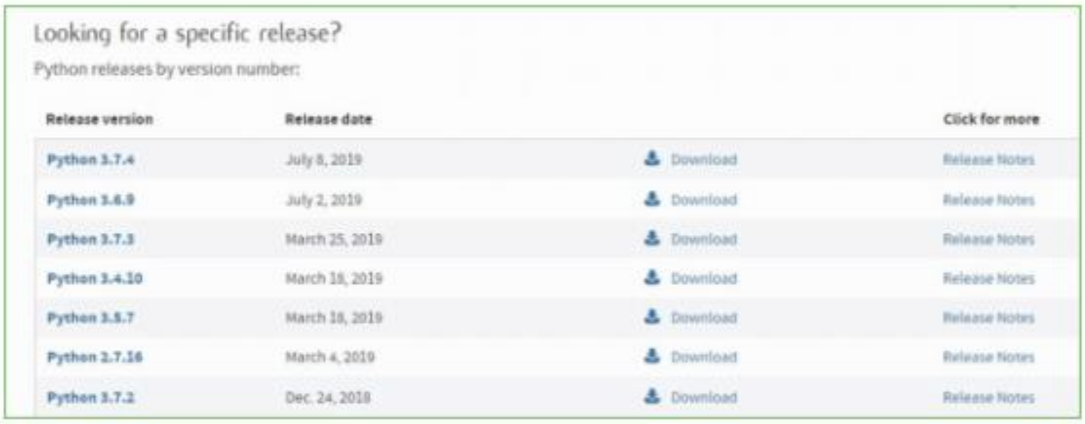
Download Python 3.7.4 CLICK HERE

Looking for Python with a different OS? Python for Windows, Linux/UNIX, Mac OS X, Other

Want to help test development versions of Python? Pre-releases, Docker images

Looking for Python 2.7? See below for specific releases

Step 3: You can either select the Download Python for windows 3.7.4 button in Yellow Color or you can scroll further down and click on download with respective to their version. Here, we are downloading the most recent python version for windows 3.7.4 .

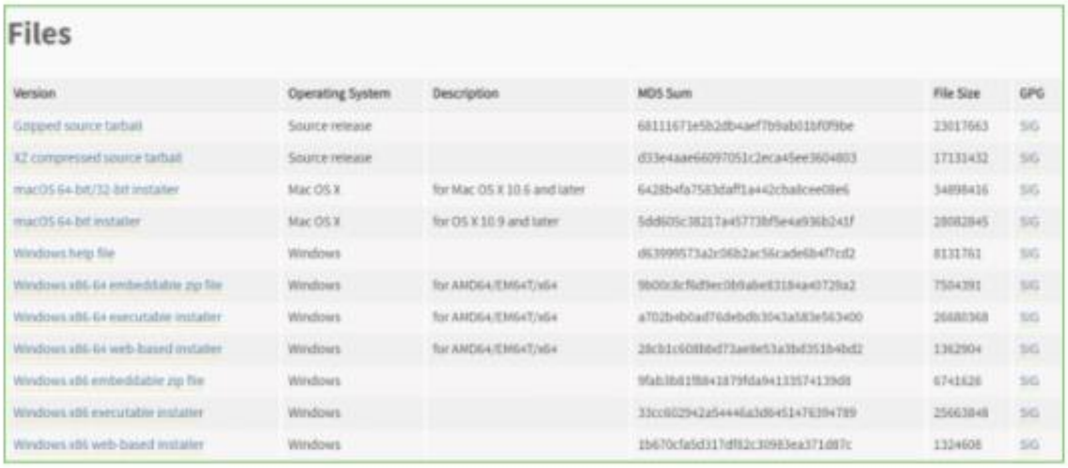

Looking for a specific release?

Python releases by version number:

Release version	Release date		Click for more
Python 3.7.4	July 8, 2019	Download	Release Notes
Python 3.6.9	July 2, 2019	Download	Release Notes
Python 3.7.3	March 25, 2019	Download	Release Notes
Python 3.4.10	March 18, 2019	Download	Release Notes
Python 3.5.7	March 18, 2019	Download	Release Notes
Python 2.7.16	March 4, 2019	Download	Release Notes
Python 3.7.2	Dec. 24, 2018	Download	Release Notes

Step 4: Scroll down the page until you find the Files option.

Step 5: Here you see a different version of python along with the operating system.



Files

Version	Operating System	Description	MD5 Sum	File Size	GPG
Gzipped source tarball	Source release		68111671e5b2db4aef7b9ab01bf0f9be	23017663	SIG
XZ compressed source tarball	Source release		d33e4aae66097051c2eca45ee3604803	17131432	SIG
macOS 64-bit/32-bit installer	Mac OS X	for Mac OS X 10.6 and later	6428b4fa7583daff1a442cba8cee08e6	34898416	SIG
macOS 64-bit installer	Mac OS X	for OS X 10.9 and later	5dd605c38217a45773bf5e4a936b241f	28082845	SIG
Windows help file	Windows		d63999573a2c06b2ac56cade6b4f7cd2	8131761	SIG
Windows x86-64 embeddable zip file	Windows	for AMD64/EM64T/x64	9b00c8cf6d9ec0b9abe83184a40729a2	7504391	SIG
Windows x86-64 executable installer	Windows	for AMD64/EM64T/x64	a702b4b0ad76debdb3043a583e563400	26680368	SIG
Windows x86-64 web-based installer	Windows	for AMD64/EM64T/x64	28cb1c608bbd73ae8e53a3bd351b4bd2	1362904	SIG
Windows x86 embeddable zip file	Windows		9fab3b81f8841879fda94133574139d8	6741626	SIG
Windows x86 executable installer	Windows		33cc602942a54446a3d6451476394789	25663848	SIG
Windows x86 web-based installer	Windows		1b670cfa5d317df82c30983ea371d87c	1324608	SIG

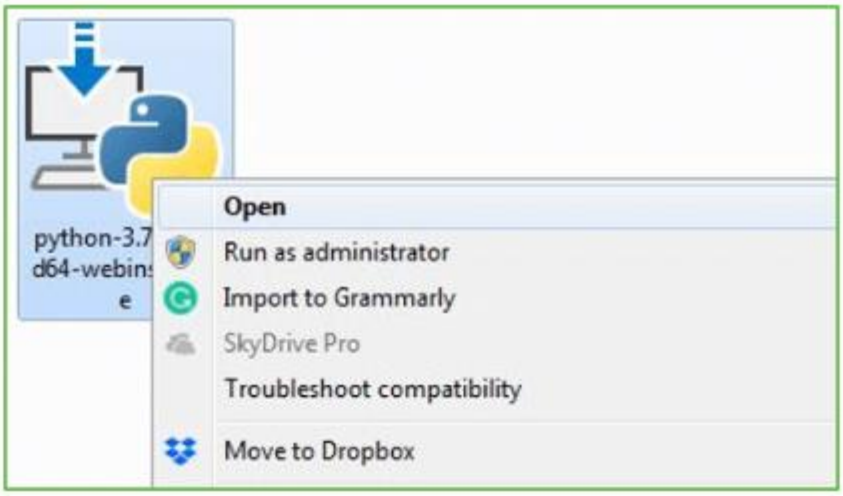
- To download Windows 32-bit python, you can select any one from the three options: Windows x86 embeddable zip file, Windows x86 executable installer or Windows x86 web-based installer.
- To download Windows 64-bit python, you can select any one from the three options: Windows x86-64 embeddable zip file, Windows x86-64 executable installer or Windows x86-64 web-based installer.

Here we will install Windows x86-64 web-based installer. Here your first part regarding which version of python is to be downloaded is completed. Now we move ahead with the second part in installing python i.e. Installation

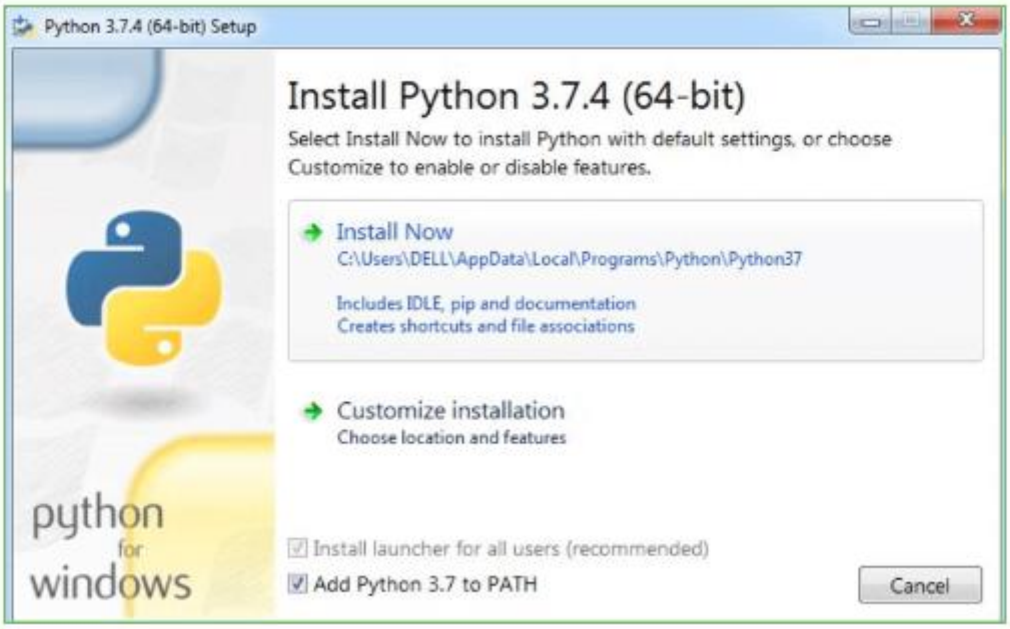
Note: To know the changes or updates that are made in the version you can click on the Release Note Option.

Installation of Python

Step 1: Go to Download and Open the downloaded python version to carry out the installation process.



Step 2: Before you click on Install Now, Make sure to put a tick on Add Python 3.7 to PATH.



Python 3.7.4 (64-bit) Setup

Install Python 3.7.4 (64-bit)

Select Install Now to install Python with default settings, or choose Customize to enable or disable features.

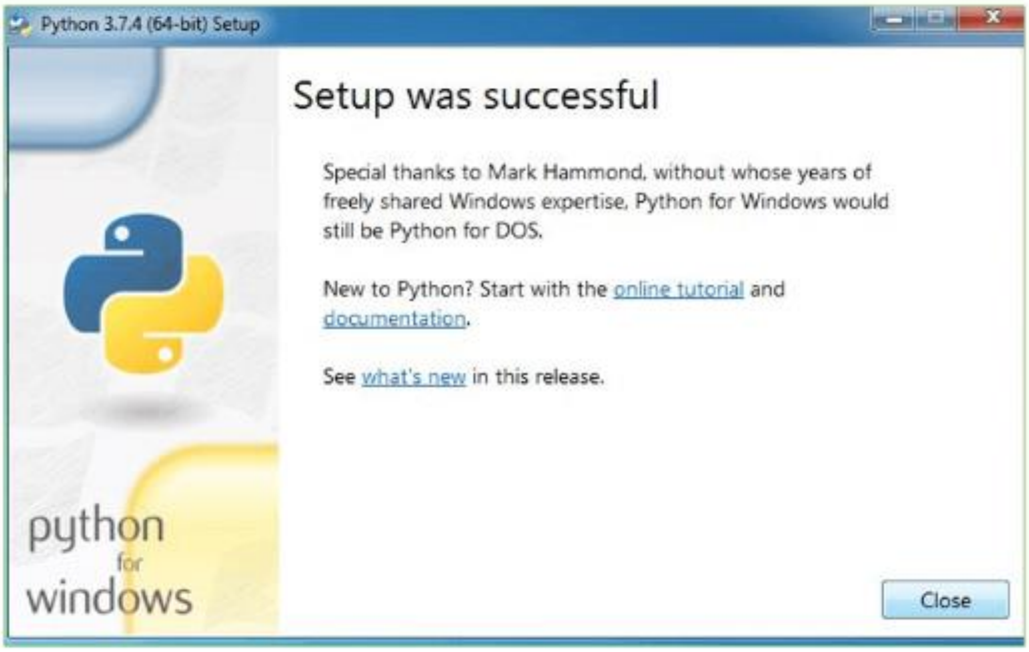
Install Now
C:\Users\DELL\AppData\Local\Programs\Python\Python37

Includes IDLE, pip and documentation
Creates shortcuts and file associations

Customize installation
Choose location and features

[✓] Install launcher for all users (recommended)
[✓] Add Python 3.7 to PATH

Step 3: Click on Install NOW After the installation is successful. Click on Close.



Python 3.7.4 (64-bit) Setup

Setup was successful

Special thanks to Mark Hammond, without whose years of freely shared Windows expertise, Python for Windows would still be Python for DOS.

New to Python? Start with the online tutorial and documentation.

See what's new in this release.

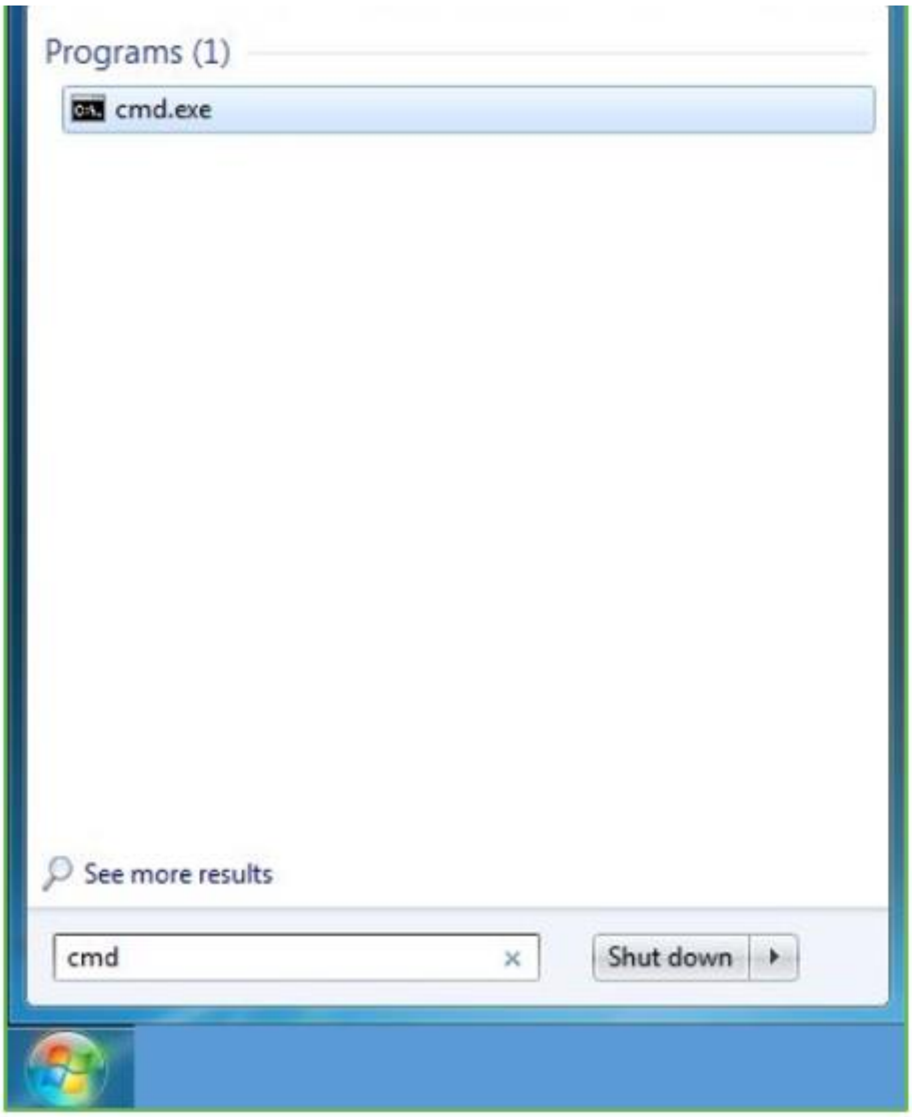
With these above three steps on python installation, you have successfully and correctly installed Python. Now is the time to verify the installation.

Note: The installation process might take a couple of minutes.

Verify the Python Installation

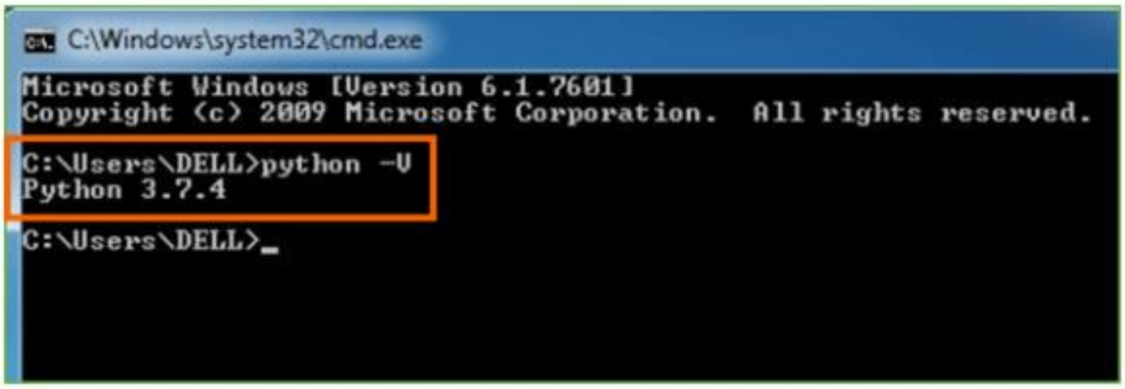
Step 1: Click on Start

Step 2: In the Windows Run Command, type “cmd”.



Step 3: Open the Command prompt option.

Step 4: Let us test whether the python is correctly installed. Type python –V and press Enter.



```
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation.  All rights reserved.

C:\Users\DELL>python -V
Python 3.7.4

C:\Users\DELL>_
```

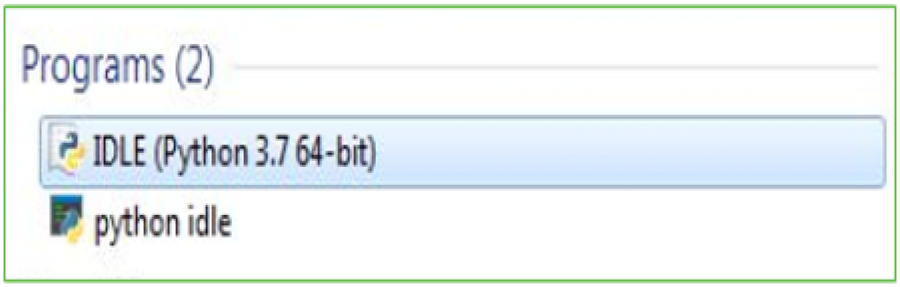
Step 5: You will get the answer as 3.7.4

Note: If you have any of the earlier versions of Python already installed. You must first uninstall the earlier version and then install the new one.

Check how the Python IDLE works

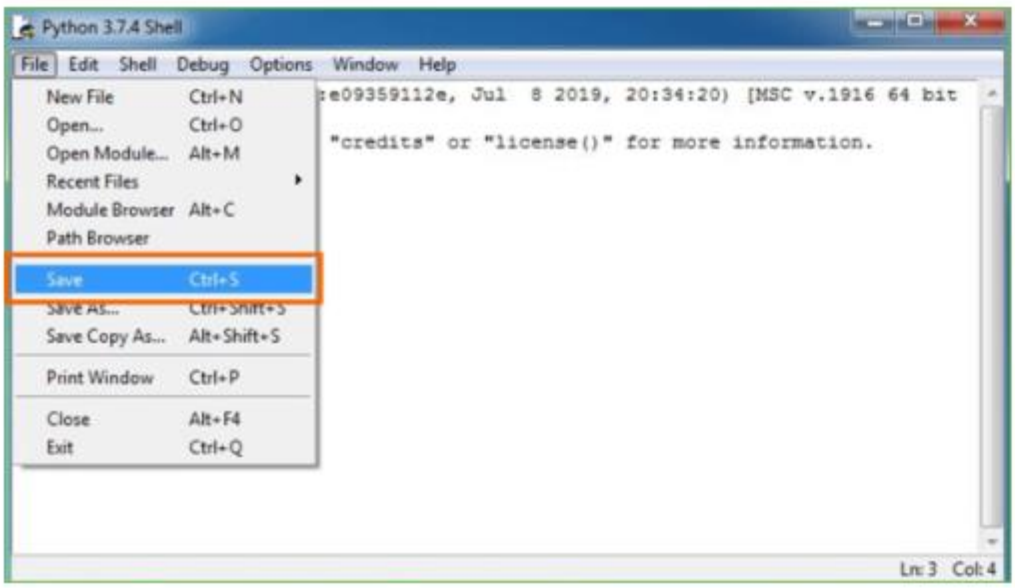
Step 1: Click on Start

Step 2: In the Windows Run command, type "python idle".



Step 3: Click on IDLE (Python 3.7 64-bit) and launch the program

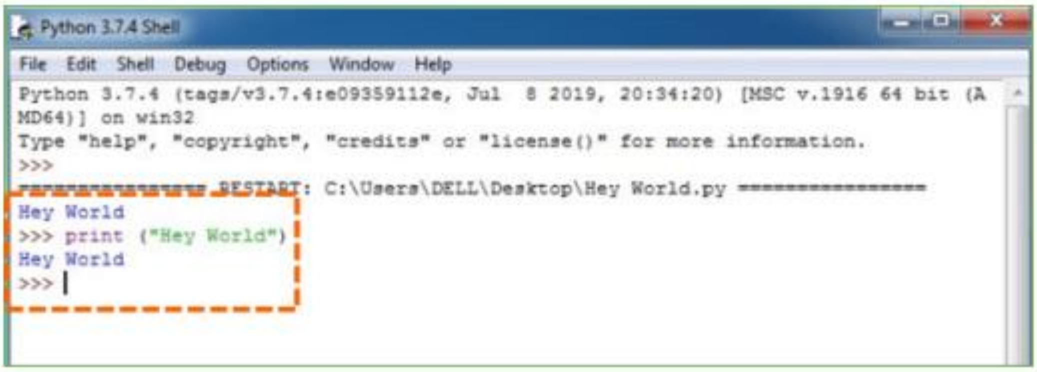
Step 4: To go ahead with working in IDLE you must first save the file. Click on File > Click on Save



```
:e09359112e, Jul  8 2019, 20:34:20) [MSC v.1916 64 bit

 "credits" or "license()" for more information.
```

Step 5: Name the file and save as type should be Python files. Click on SAVE. Here I have named the files as Hey World.

Step 6: Now for e.g. enter print ("Hey World") and Press Enter.


Python 3.7.4 Shell

```
Python 3.7.4 (tags/v3.7.4:e09359112e, Jul  8 2019, 20:34:20) [MSC v.1916 64 bit (A
MD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
================ RESTART: C:\Users\DELL\Desktop\Hey World.py ================
Hey World
>>> print ("Hey World")
Hey World
>>> 
```

You will see that the command given is launched. With this, we end our tutorial on how to install Python. You have learned how to download python for windows into your respective operating system.

Note: Unlike Java, Python does not need semicolons at the end of the statements otherwise it won’t work.

CHAPTER-6: SYSTEM STUDY

A system study refers to the comprehensive analysis and design of a machine learning system aimed at predicting Autism Spectrum Disorder (ASD). It involves several key stages that collectively help develop a robust and effective prediction model. Here's an outline of the system study:

1. System Objectives

Goal: Develop a machine learning-based system that predicts the likelihood of an individual having Autism Spectrum Disorder based on features such as behavior, demographics, or genetics.

Purpose: Early and accurate diagnosis of ASD to enable timely intervention and treatment.

2.System Design

Input: Data sources, such as clinical records, questionnaires, and possibly genetic markers.

Processing: Data preprocessing, including handling missing values, feature scaling, and encoding categorical variables.

Modeling: The system applies various machine learning classifiers (e.g., Decision Trees, Random Forest, SVM) to build predictive models.

3. Data Flow

Data Collection Module: Handles input data from questionnaires and clinical records

Preprocessing Module: Cleans, transforms, and scales the data, making it ready for modeling.

Feature Engineering Module: Selects and extracts the most important features for training the model.

Classification Module: Applies machine learning classifiers to predict ASD.

Evaluation Module: Assesses model performance using accuracy, F1 scores, and precision.

4. System Evaluation

Validation: The model is validated using techniques such as cross-validation or holdout validation to ensure it performs consistently on unseen data.

Performance Metrics: The system measures how well the model predicts ASD, using evaluation metrics like accuracy, ROC-AUC, precision, recall, and F1 score.

5. Deployment and Monitoring

Deployment: Once the model is trained and evaluated, it is integrated into a decision support system for use in healthcare or clinical environments.

Monitoring: The system needs to be continuously monitored for performance as new data is added to the model, ensuring that the predictions remain reliable over time.

6. System Constraints

Data Sensitivity: Medical and genetic data often have privacy and ethical constraints, requiring the system to adhere to data protection regulations (e.g., HIPAA).

Scalability: The system should be designed to handle increasing amounts of data as it is collected from more sources over time.

Interpretability: The model needs to be interpretable, especially in a medical context, where clinicians need to understand how predictions are made.The system study focuses on analyzing each component of the system, understanding its objectives, constraints, and evaluating the best tools, methodologies, and workflows for ASD prediction. It ensures the system is designed holistically for optimal performance, accuracy, and real-world applicability.

CHAPTER-7: SYSTEM TESTING

A system study for predicting Autism Spectrum Disorder (ASD) using machine learning classifiers would involve several steps, from data collection and preprocessing to model development, evaluation, and interpretation.

7.1 Types of Testing

7.1.1. Unit Testing

Objective: To test individual components or modules of the system to ensure they function correctly. Modules Tested are:

Data Collection Module: Validate that the system accurately collects data from questionnaires, databases, or external sources.

Preprocessing Module: Ensure correct handling of missing data, feature scaling, and encoding of categorical variables.

Feature Engineering Module: Test the extraction and selection of relevant features for the model.

Model Training Module: Verify that the classifiers (e.g., Decision Trees, SVM) are correctly implemented and can be trained without errors.

7.1.2. Integration Testing

Objective: To ensure that the modules of the system interact with each other properly.

Tests: Ensure that data flows seamlessly from data collection to preprocessing, feature engineering, model training, and prediction.

Validate that data preprocessing outputs are correctly fed into the machine learning model.

Test integration between the trained model and the evaluation metrics module to ensure predictions are being correctly evaluated.

7.1.3. Functional Testing

Objective: To validate that the system's functionality meets the specified requirements.

Tests: Ensure that the system can take input data (e.g., questionnaire responses or demographic information) and output predictions regarding the likelihood of ASD.

Verify the system’s ability to generate predictions for various data inputs, including edge cases (e.g., incomplete questionnaires, outliers in age).

Test if the system can process data from different sources and formats.

7.1.4. Security Testing

Objective: To ensure the system protects sensitive data, especially in healthcare applications where patient privacy is critical.

Tests:

Data Encryption: Ensure that sensitive input data, such as medical records or demographic information, is properly encrypted.

Access Control: Test that only authorized users can access the system, particularly the data collection and prediction modules.

7.1.5. Usability Testing

Objective: To ensure that the system is easy to use by clinicians, caregivers, or non-technical users.

User Interface: Test the ease of navigating through the system's interface, ensuring clear instructions for data input and interpretation of predictions.

Accessibility: Ensure the system is accessible for a wide range of users, including those with disabilities, if applicable.

Error Handling: Check how the system responds to incorrect or incomplete data inputs, ensuring that meaningful error messages are provided.

7.2. Test Data and Environment

7.2.1. Test Data

Training and Validation Data: The system should be tested with a combination of real-world and synthetic data, including datasets like the UCI ASD dataset or clinical trial data.

Edge Cases: Include extreme cases, such as very young or old age groups, missing or incorrect data, or unusual behavioral patterns.

Imbalanced Data: Test how the system handles data imbalance, where the number of ASD-positive cases might be much lower than the number of ASD-negative cases.

7.2.2. Test Environment

Hardware Requirements: Test the system on different hardware setups (e.g., high-performance servers vs. standard workstations) to ensure scalability.

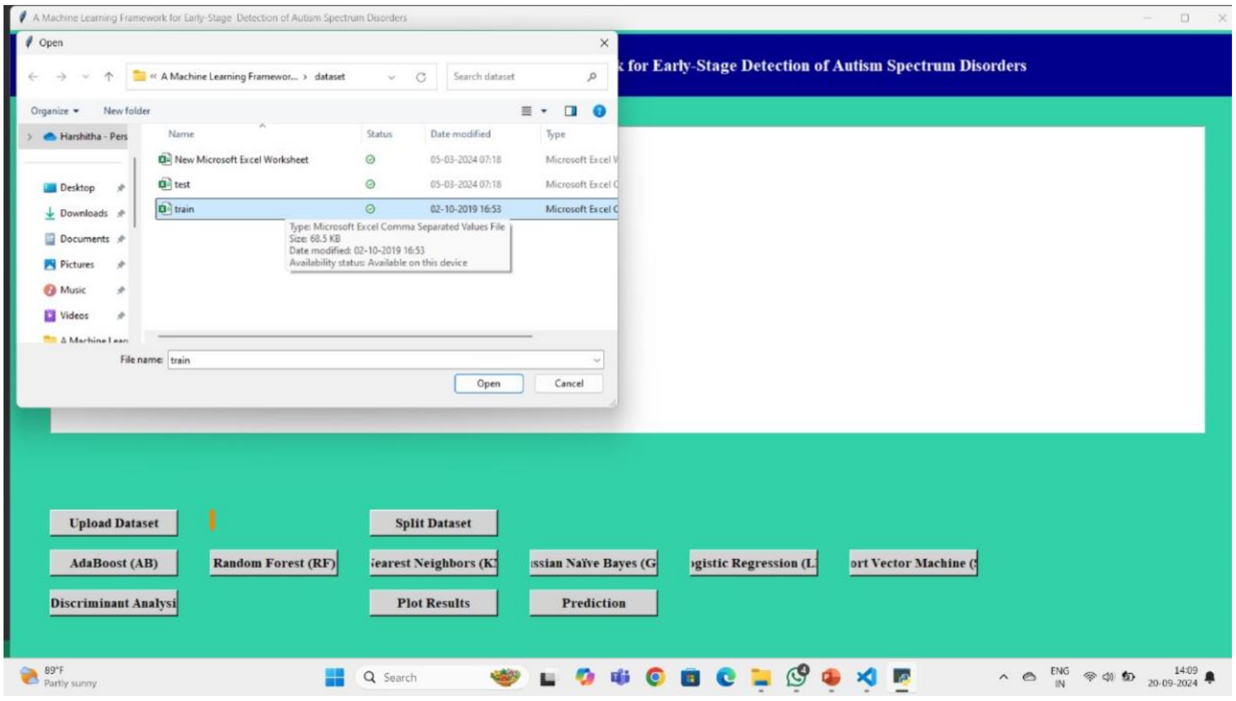
Software Dependencies: Ensure that all machine learning libraries and tools (e.g., TensorFlow, Scikit-learn) function correctly in the deployment environment.

7.2.3. Test Automation

Automated Testing: Where possible, unit and integration tests should be automated to speed up the testing process. Tools like PyTest or Jenkins can automate testing and integration processes, while ML-specific libraries can help automate model validation.

Continuous Testing: Implement continuous integration (CI) pipelines to automatically run tests after each system update, ensuring that new changes do not break existing functionality.

- Click on upload dataset and train file



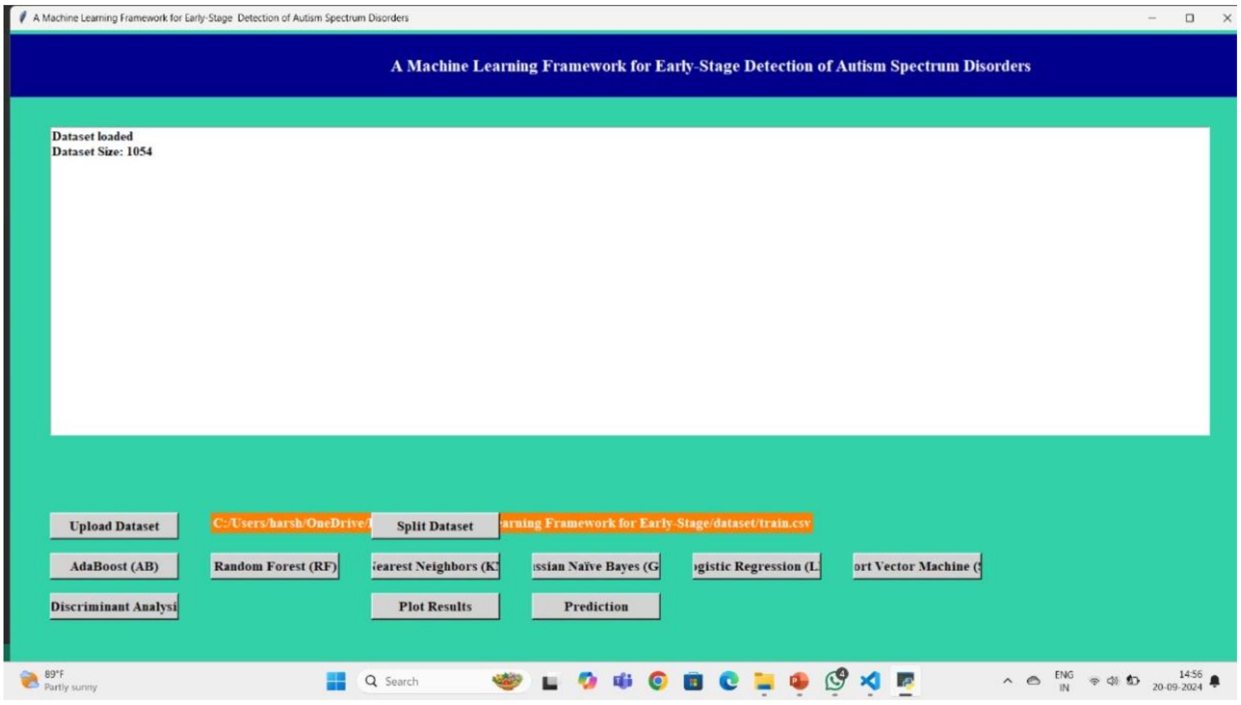
Name	Status	Date modified	Type
New Microsoft Excel Worksheet	✓	05-03-2024 07:18	Microsoft Excel W
test	✓	05-03-2024 07:18	Microsoft Excel C
train	✓	02-10-2019 16:53	Microsoft Excel C

Type: Microsoft Excel Comma Separated Values File
Size: 68.5 KB
Date modified: 02-10-2019 16:53
Availability status: Available on this device

File name: train

Fig 8.1: Screenshot of upload dataset

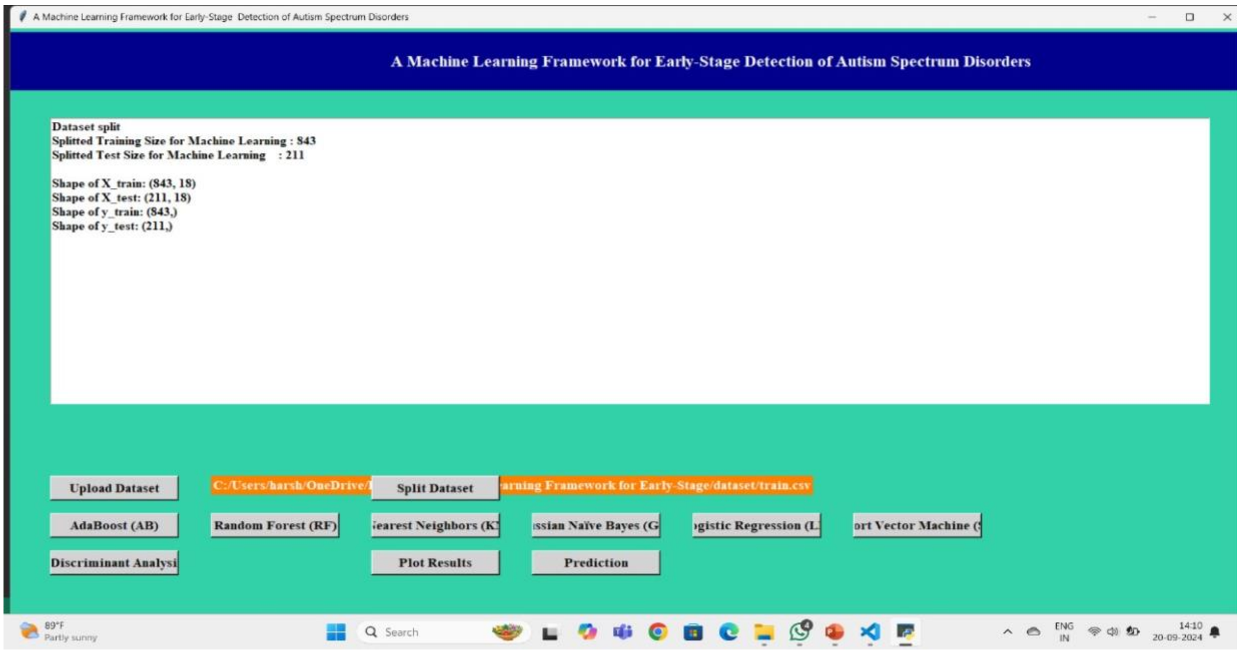
- Data set is uploaded and it displays the size of dataset



```
Dataset loaded
Dataset Size: 1054
```

Fig 8.2: Screenshot of size of dataset

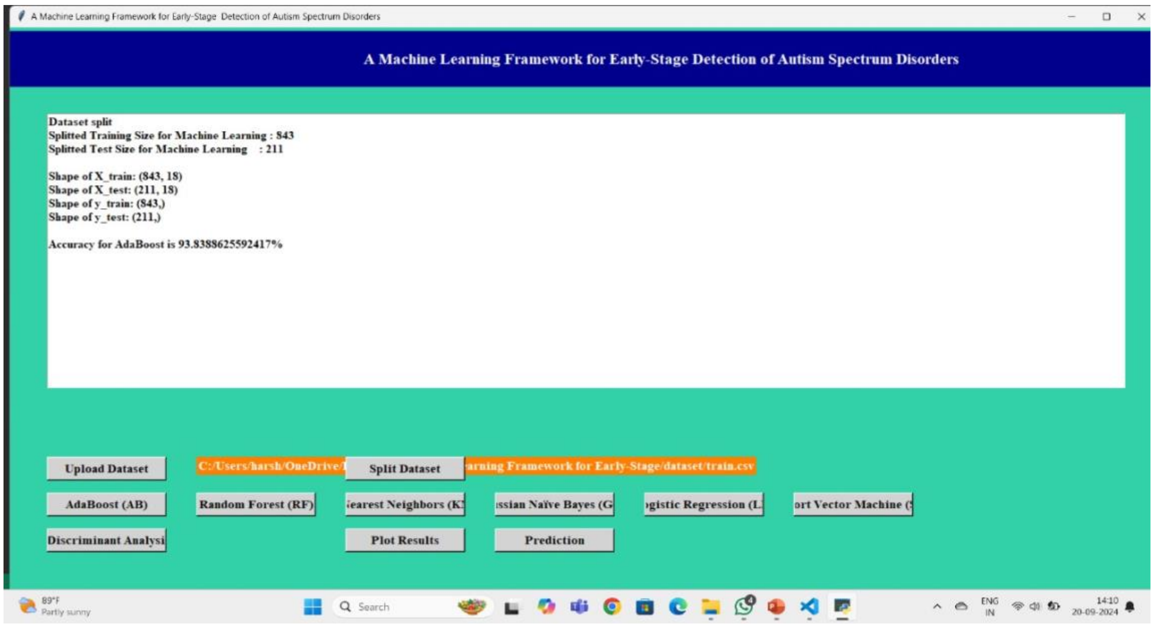
- Click on the split dataset



```
Dataset split
Splitted Training Size for Machine Learning : 843
Splitted Test Size for Machine Learning    : 211

Shape of X_train: (843, 18)
Shape of X_test: (211, 18)
Shape of y_train: (843,)
Shape of y_test: (211,)
```

Fig 8.3: Screenshot of split dataset

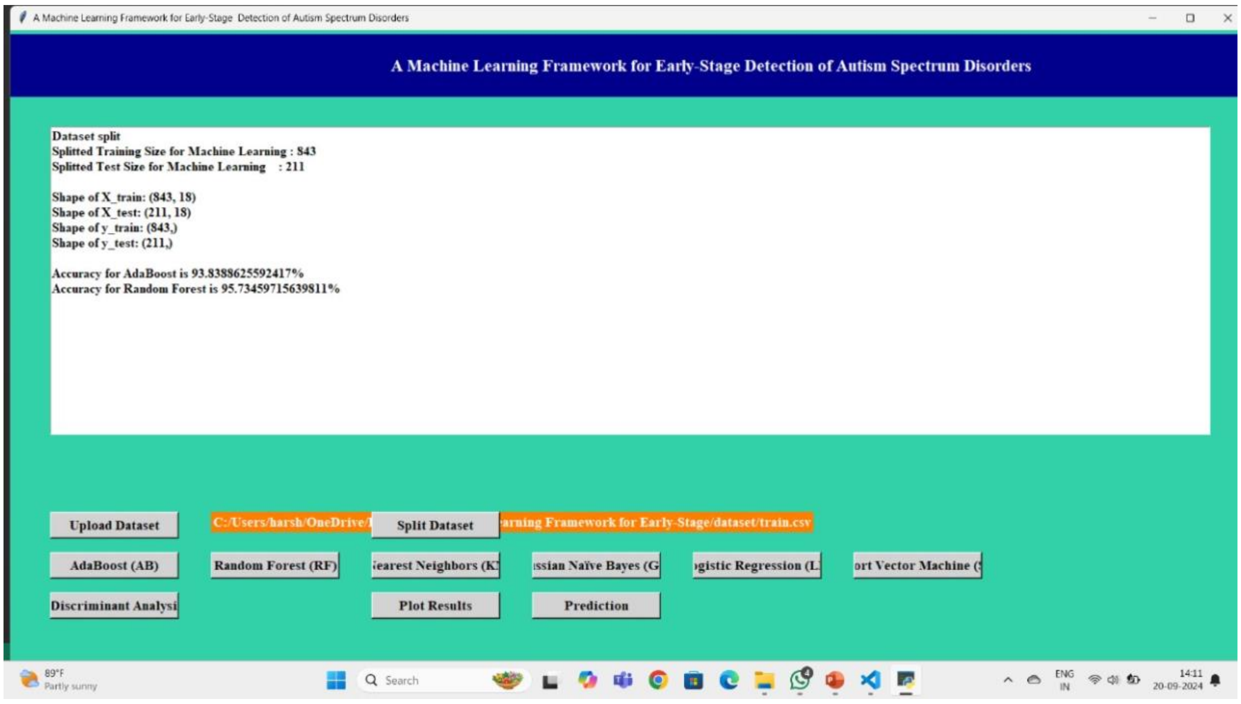
- Click on AdaBoost(AB)


A Machine Learning Framework for Early-Stage Detection of Autism Spectrum Disorders
A Machine Learning Framework for Early-Stage Detection of Autism Spectrum Disorders
Upload Dataset
C:/Users/harsh/OneDrive/
Split Dataset
arning Framework for Early-Stage/dataset/train.csv
AdaBoost (AB)
Random Forest (RF)
learest Neighbors (K
ssian Naïve Bayes (G
gistic Regression (L
ort Vector Machine (
Discriminant Analysi
Plot Results
Prediction

```
Dataset split
Splitted Training Size for Machine Learning : 843
Splitted Test Size for Machine Learning   : 211

Shape of X_train: (843, 18)
Shape of X_test: (211, 18)
Shape of y_train: (843,)
Shape of y_test: (211,)

Accuracy for AdaBoost is 93.8388625592417%
```

Fig 8.4: Screenshot of AdaBoost

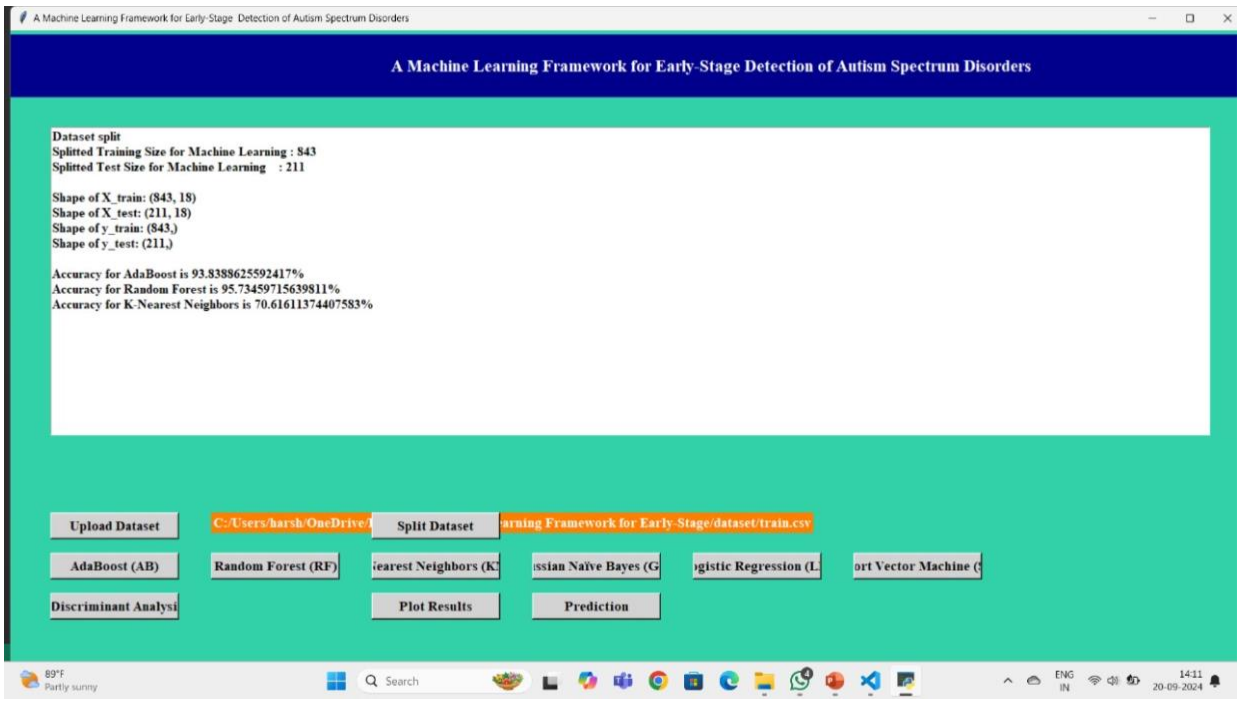
- Click on Random Forest (RF)


A Machine Learning Framework for Early-Stage Detection of Autism Spectrum Disorders
A Machine Learning Framework for Early-Stage Detection of Autism Spectrum Disorders
Upload Dataset
C:/Users/harsh/OneDrive/
Split Dataset
arning Framework for Early-Stage/dataset/train.csv
AdaBoost (AB)
Random Forest (RF)
learest Neighbors (K
ssian Naïve Bayes (G
gistic Regression (L
ort Vector Machine (
Discriminant Analysi
Plot Results
Prediction

```
Dataset split
Splitted Training Size for Machine Learning : 843
Splitted Test Size for Machine Learning   : 211

Shape of X_train: (843, 18)
Shape of X_test: (211, 18)
Shape of y_train: (843,)
Shape of y_test: (211,)

Accuracy for AdaBoost is 93.8388625592417%
Accuracy for Random Forest is 95.73459715639811%
```

Fig 8.5: Screenshot of Random Forest

- Click on K-Nearest Neighbors (KNN)

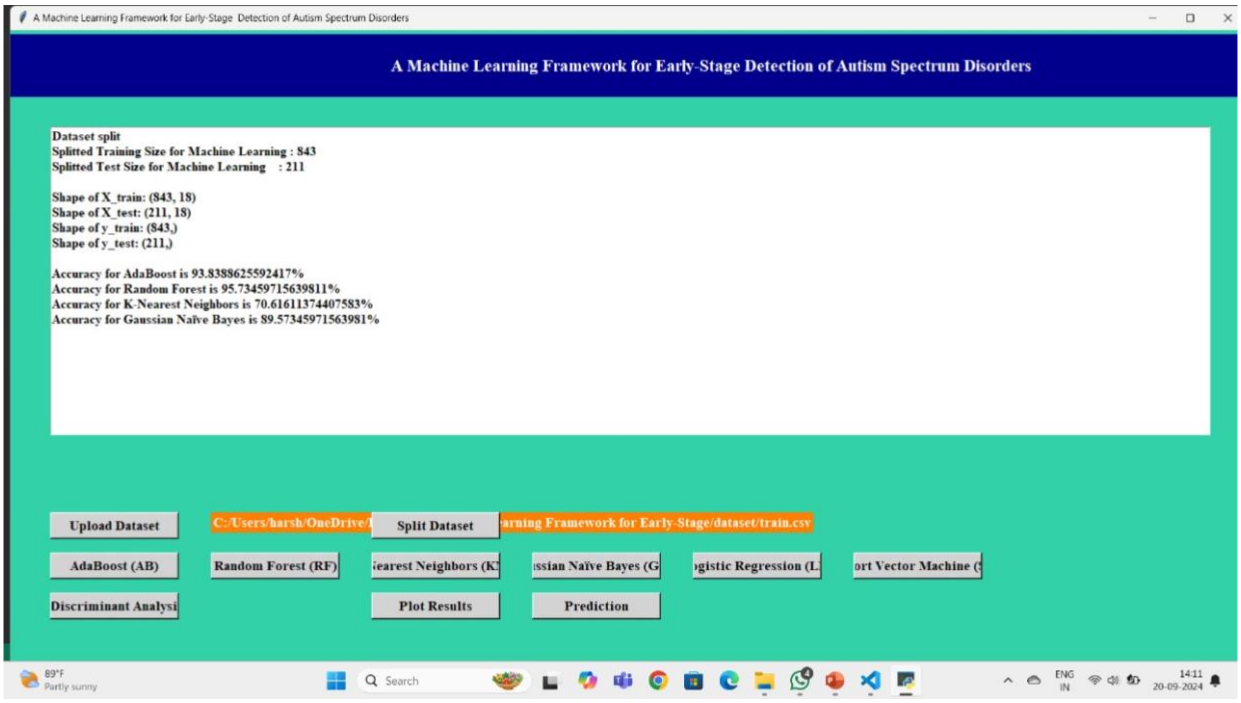


A Machine Learning Framework for Early-Stage Detection of Autism Spectrum Disorders

```
Dataset split
Splitted Training Size for Machine Learning : 843
Splitted Test Size for Machine Learning    : 211

Shape of X_train: (843, 18)
Shape of X_test: (211, 18)
Shape of y_train: (843,)
Shape of y_test: (211,)

Accuracy for AdaBoost is 93.8388625592417%
Accuracy for Random Forest is 95.73459715639811%
Accuracy for K-Nearest Neighbors is 70.61611374407583%
```

Fig 8.6: Screenshot of KNN

- Click on Gaussian Naive Bayes(GNB)

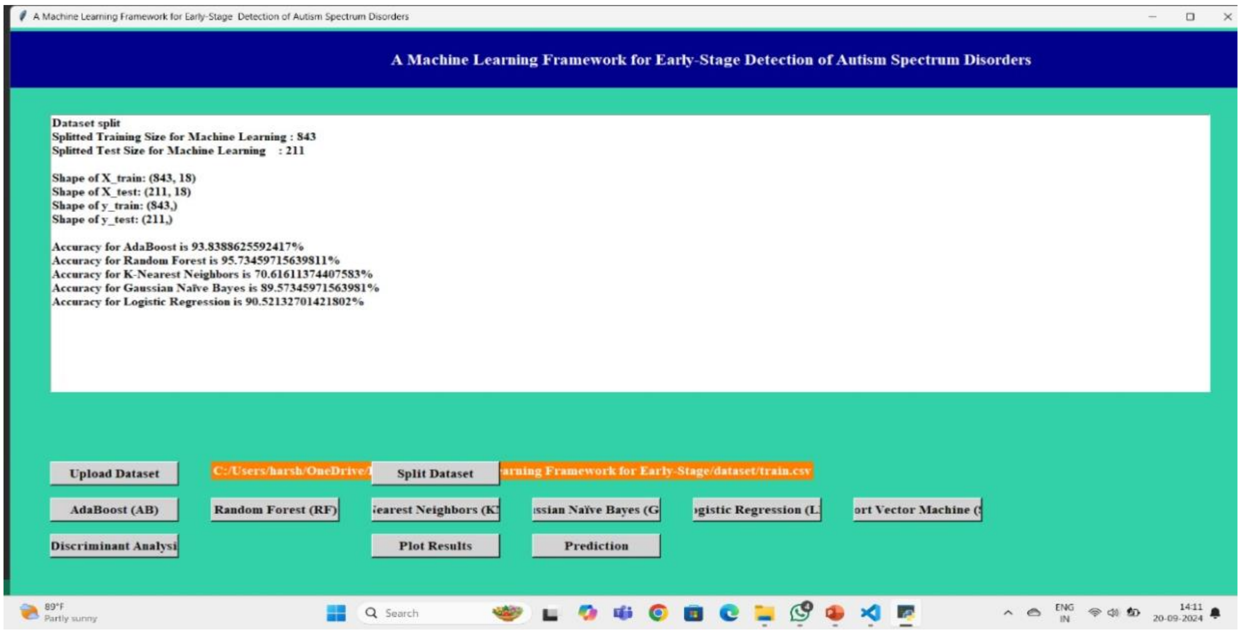


A Machine Learning Framework for Early-Stage Detection of Autism Spectrum Disorders

```
Dataset split
Splitted Training Size for Machine Learning : 843
Splitted Test Size for Machine Learning    : 211

Shape of X_train: (843, 18)
Shape of X_test: (211, 18)
Shape of y_train: (843,)
Shape of y_test: (211,)

Accuracy for AdaBoost is 93.8388625592417%
Accuracy for Random Forest is 95.73459715639811%
Accuracy for K-Nearest Neighbors is 70.61611374407583%
Accuracy for Gaussian Naïve Bayes is 89.57345971563981%
```

Fig 8.7: Screenshot of GNB

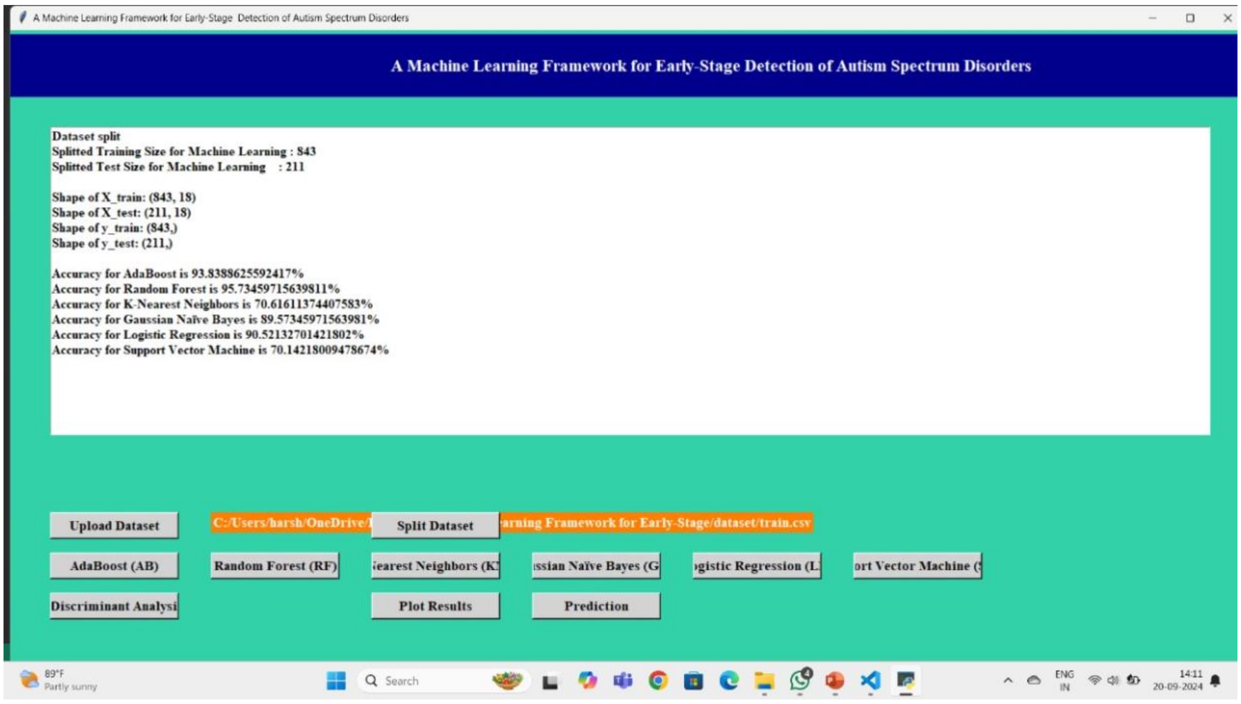
- Click on the Logistic Regression


A Machine Learning Framework for Early-Stage Detection of Autism Spectrum Disorders
A Machine Learning Framework for Early-Stage Detection of Autism Spectrum Disorders
Upload Dataset
C:/Users/harsh/OneDrive/
Split Dataset
arning Framework for Early-Stage/dataset/train.csv
AdaBoost (AB)
Random Forest (RF)
learest Neighbors (K
ssian Naïve Bayes (G
gistic Regression (L
ort Vector Machine (
Discriminant Analysi
Plot Results
Prediction

```
Dataset split
Splitted Training Size for Machine Learning : 843
Splitted Test Size for Machine Learning    : 211

Shape of X_train: (843, 18)
Shape of X_test: (211, 18)
Shape of y_train: (843,)
Shape of y_test: (211,)

Accuracy for AdaBoost is 93.8388625592417%
Accuracy for Random Forest is 95.73459715639811%
Accuracy for K-Nearest Neighbors is 70.61611374407583%
Accuracy for Gaussian Naïve Bayes is 89.57345971563981%
Accuracy for Logistic Regression is 90.52132701421802%
```

Fig 8.8: Screenshot of Logistic Regression

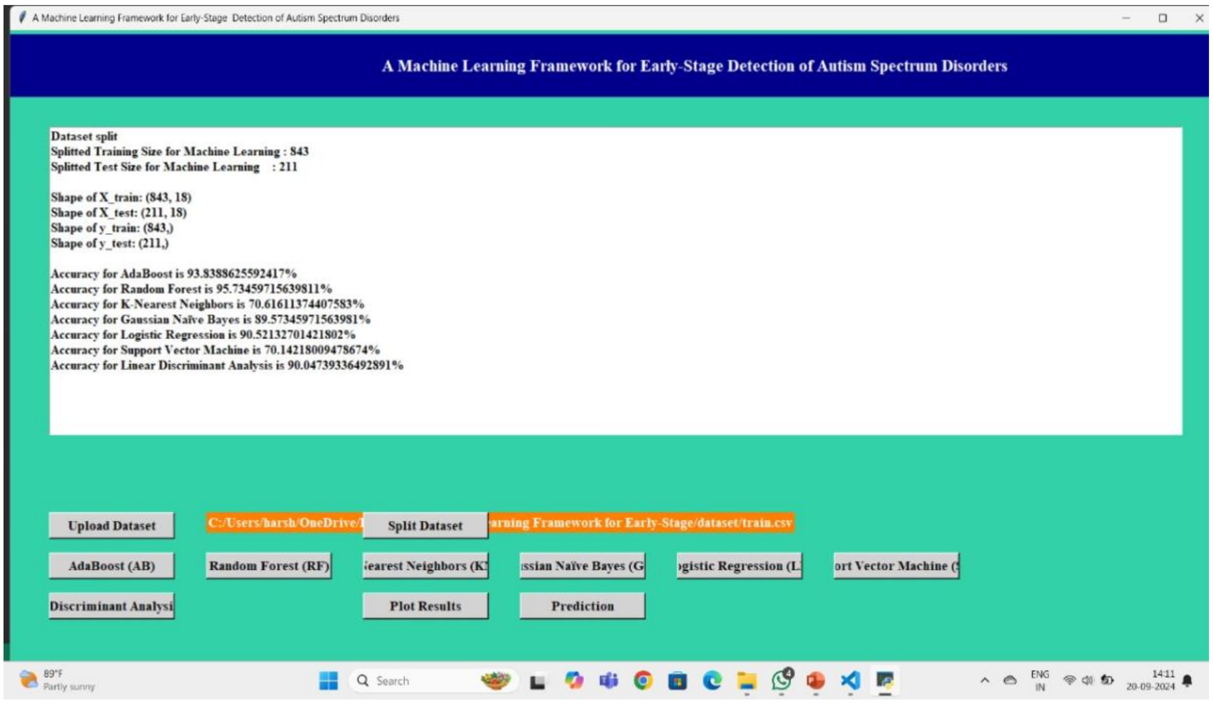
- Click on the Support Vector Machine(SVM)


A Machine Learning Framework for Early-Stage Detection of Autism Spectrum Disorders
A Machine Learning Framework for Early-Stage Detection of Autism Spectrum Disorders
Upload Dataset
C:/Users/harsh/OneDrive/
Split Dataset
arning Framework for Early-Stage/dataset/train.csv
AdaBoost (AB)
Random Forest (RF)
learest Neighbors (K
ssian Naïve Bayes (G
gistic Regression (L
ort Vector Machine (
Discriminant Analysi
Plot Results
Prediction

```
Dataset split
Splitted Training Size for Machine Learning : 843
Splitted Test Size for Machine Learning    : 211

Shape of X_train: (843, 18)
Shape of X_test: (211, 18)
Shape of y_train: (843,)
Shape of y_test: (211,)

Accuracy for AdaBoost is 93.8388625592417%
Accuracy for Random Forest is 95.73459715639811%
Accuracy for K-Nearest Neighbors is 70.61611374407583%
Accuracy for Gaussian Naïve Bayes is 89.57345971563981%
Accuracy for Logistic Regression is 90.52132701421802%
Accuracy for Support Vector Machine is 70.14218009478674%
```

Fig 8.9: Screenshot of SVM

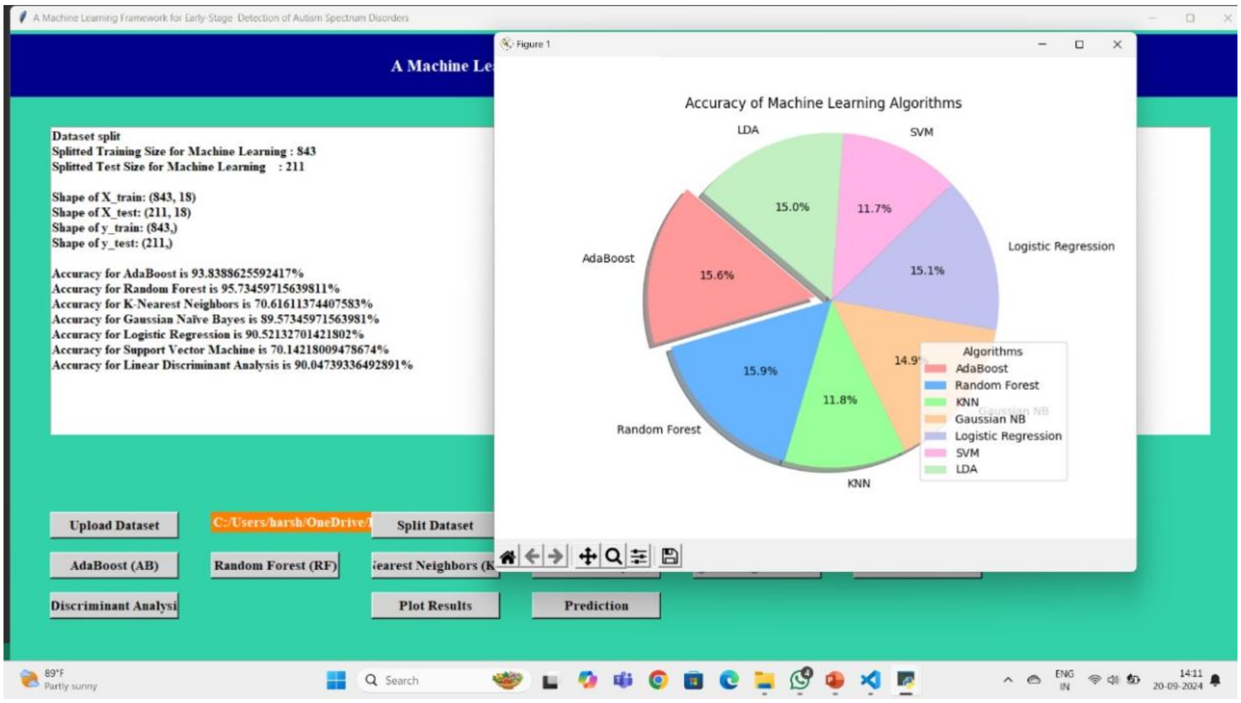
- Click on the Discriminant analysis


A Machine Learning Framework for Early-Stage Detection of Autism Spectrum Disorders
A Machine Learning Framework for Early-Stage Detection of Autism Spectrum Disorders
Upload Dataset
C:/Users/harsh/OneDrive/
Split Dataset
arning Framework for Early-Stage/dataset/train.csv
AdaBoost (AB)
Random Forest (RF)
earest Neighbors (K
ssian Naïve Bayes (G
gistic Regression (L
ort Vector Machine (
Discriminant Analysi
Plot Results
Prediction

```
Dataset split
Splitted Training Size for Machine Learning : 843
Splitted Test Size for Machine Learning    : 211

Shape of X_train: (843, 18)
Shape of X_test: (211, 18)
Shape of y_train: (843,)
Shape of y_test: (211,)

Accuracy for AdaBoost is 93.8388625592417%
Accuracy for Random Forest is 95.73459715639811%
Accuracy for K-Nearest Neighbors is 70.61611374407583%
Accuracy for Gaussian Naïve Bayes is 89.57345971563981%
Accuracy for Logistic Regression is 90.52132701421802%
Accuracy for Support Vector Machine is 70.14218009478674%
Accuracy for Linear Discriminant Analysis is 90.04739336492891%
```

Fig 8.10: Screenshot of Discriminant Analysis

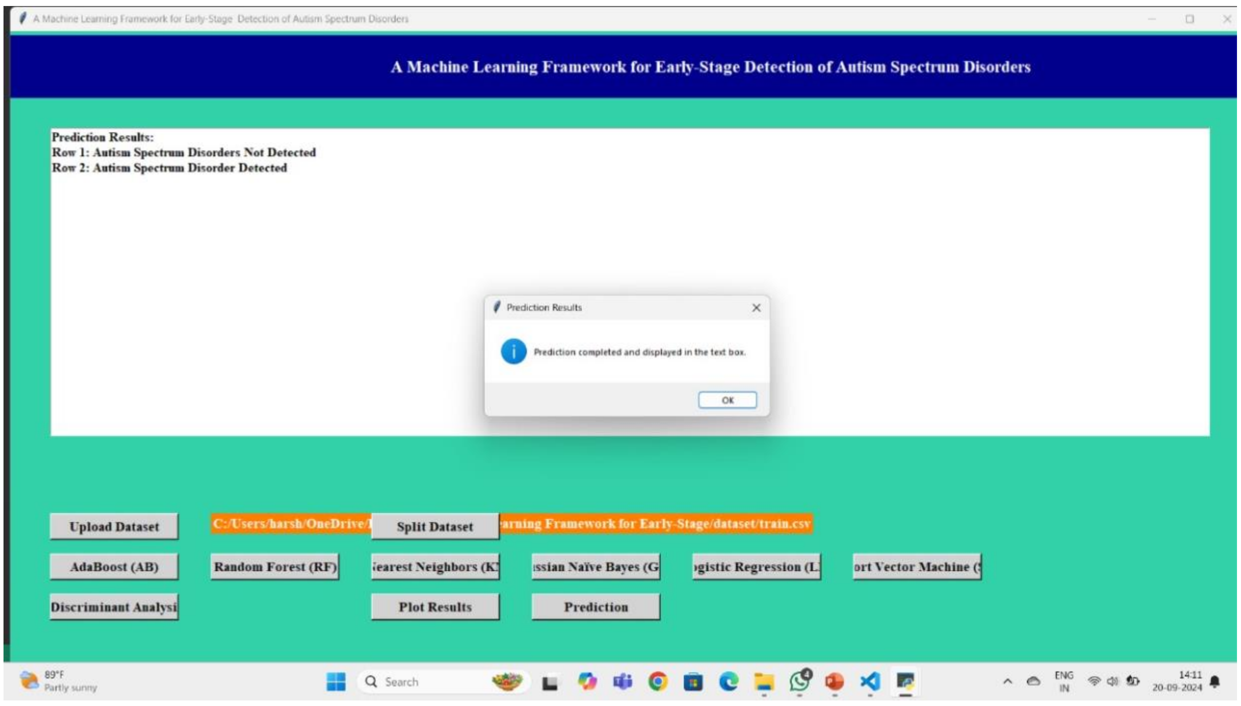
- Click on the Plot Results



```
Dataset split
Splitted Training Size for Machine Learning : 843
Splitted Test Size for Machine Learning    : 211

Shape of X_train: (843, 18)
Shape of X_test: (211, 18)
Shape of y_train: (843,)
Shape of y_test: (211,)

Accuracy for AdaBoost is 93.8388625592417%
Accuracy for Random Forest is 95.73459715639811%
Accuracy for K-Nearest Neighbors is 70.61611374407583%
Accuracy for Gaussian Naïve Bayes is 89.57345971563981%
Accuracy for Logistic Regression is 90.52132701421802%
Accuracy for Support Vector Machine is 70.14218009478674%
Accuracy for Linear Discriminant Analysis is 90.04739336492891%
```

Fig 8.11: Screenshot of Plot results

- Click on the Predictions



Prediction Results:
Row 1: Autism Spectrum Disorders Not Detected
Row 2: Autism Spectrum Disorder Detected

Prediction completed and displayed in the text box.

Fig 8.12: Screenshot of Predictions

CHAPTER 9: CONCLUSION & FUTURE ENHANCEMENT

CONCLUSION:

A framework for machine learning-based ASD detection in individuals of various ages (toddlers, children, adolescents, and adults) was proposed in this paper. We demonstrate that ML-based predictive models are effective instruments for this endeavor. Once the preliminary data processing was finished, the ASD datasets underwent feature scaling using four distinct techniques (QT, PT, normalizer, MAS) and classification using eight distinct ML classifiers (AB, RF, DT, KNN, GNB, LR, SVM, LDA). We then determined the most effective FS and classification approaches by analyzing the classification performance of each feature-scaled dataset. To substantiate the experimental results, we examined various statistical evaluation metrics, including accuracy, ROC, F1-Score, precision, recall, Mathews correlation coefficient (MCC), kappa score, and log loss. Consequently, clinicians may find our proposed ML-based prediction models to be a viable alternative or even a useful instrument when attempting to accurately diagnose ASD in patients of varying ages. Furthermore, the feature importance values were computed utilizing four distinct FSTs (IGAE, GRAE, RFAE, and CAE) in order to ascertain the most prominent features for the purpose of ASD prediction. Consequently, the experimental analysis of this study will enable medical professionals to consider the most critical characteristics when screening for ASD. Our research is limited in that the quantity of data was insufficient to develop a model applicable to individuals at all stages. Moving forward, our objective is to augment our dataset on ASD and develop a predictive model that is applicable to individuals of all ages, with the intention of refining the identification of ASD and other neurodevelopmental disorders.

FUTURE ENHANCEMENT:

1.Enhanced Prediction Models:

Continued research and advancements in machine learning algorithms can lead to more accurate and reliable prediction models for ASD. This can further improve early detection and intervention strategies.

2.Integration with Other Technologies:

Machine learning can be integrated with other technologies like wearable devices, genetic testing, and brain imaging to create comprehensive diagnostic tools for ASD.

3.Early Intervention Strategies:

Future applications of machine learning may focus on developing proactive intervention strategies based on predictive models. This can lead to tailored interventions that address specific needs and challenges of individuals with ASD at an early stage.

4. Big Data Analysis:

As more data becomes available, machine learning can leverage big data analysis to identify new patterns, risk factors, and potential treatment options for ASD. This can contribute to ongoing research efforts and improve our understanding of the disorder.

CHAPTER-10: BIBLIOGRAPHY

[1] M. Bala, M. H. Ali, M. S. Satu, K. F. Hasan, and M. A. Moni, ‘‘Efficient machine learning models for early stage detection of autism spectrum disorder,’’ Algorithms, vol. 15, no. 5, p. 166, May 2022.

[2] D. Pie trucci, A. Teofani, M. Milanesi, B. Fosso, L. Putignani, F. Messina, G. Pesole, A. Desideri, and G. Chillemi, ‘‘Machine learning data analysis highlights the role of parasutterella and allow prevotella in autism spectrum disorders,’’ Biomedicines, vol. 10, no. 8, p. 2028, Aug. 2022.

[3] R. Sreedasyam, A. Rao, N. Sachidanandan, N. Sampath, and S. K. Vasudevan, ‘‘Aarya—A kinesthetic companion for children with autism spectrum disorder,’’ J. Intell. Fuzzy Syst., vol. 32, no. 4, pp. 2971–2976, Mar. 2017.

[4] J. Amudha and H. Nandakumar, ‘‘A fuzzy based eye gaze point estimation approach to study the task behavior in autism spectrum disorder,’’ J. Intell. Fuzzy Syst., vol. 35, no. 2, pp. 1459–1469, Aug. 2018. \

[5] H. Chahkandi Nejad, O. Khayat, and J. Razjouyan, ‘‘Software development of an intelligent spirography test system for neurological disorder detection and quantification,’’ J. Intell. Fuzzy Syst., vol. 28, no. 5, pp. 2149–2157, Jun. 2015. [6] F. Z. Subah, K. Deb, P. K. Dhar, and T. Koshiba, ‘‘A deep learning approach to predict autism spectrum disorder using multisite resting-state fMRI,’’ Appl. Sci., vol. 11, no. 8, p. 3636, Apr. 2021.