Volume 10, Issue 8, August – 2025 International Journal of Innovative Science and Research Technology
ISSN No: -2456-2165 https://doi.org/10.38124/ijisrt/25aug1562

Automated Stroke Prediction and Prevention Recommendations: Development of an Android Application

Md. Jahirul Islam[1]; Sukhdeb Chandra Das[2*]; Md.Golam Mostofa[3]

[1]Department of Information &amp; Communication Engineering, Noakhali Science and Technology University, Noakhali 3814, Bangladesh.

[2]Lecturer (Information and Communication Technology) Rajshahi Cantonment Board School and College,Rajshahi Cantonment,Rajshahi.

[3]Lecturer (Information and Communication Technology) Rajshahi Cantonment Board School and College,Rajshahi Cantonment,Rajshahi.

Corresponding Author: Sukhdeb Chandra Das[*]

Publication Date: 2025/09/08

Abstract: Stroke is currently one of the leading causes of mortality worldwide. Extensive research has identified several risk factors associated with stroke, which have been extensively studied to enhance prediction and categorization of the disease. Machine learning (ML) has proven to be a powerful tool for analyzing vast amounts of data, enabling accurate predictions and informed decision-making. Researchers are actively working to develop automated ML models for stroke prediction with the aim of facilitating early interventions and saving lives. As the global population ages and the number of individuals at risk for stroke continues to rise, the need for accurate prediction algorithms has become increasingly critical. The widespread adoption of Android applications provides an opportunity to make predictive tools accessible to a broader audience. This study investigates the application of ML algorithms in stroke risk prediction and demonstrates their integration into a functional Android application. The dataset used for model development was sourced from Kaggle, where a significant imbalance was observed with a ratio of 19:1 between instances of no stroke and stroke. To address this imbalance, Synthetic Minority Oversampling Technique (SMOTE) analysis was employed, ensuring a balanced dataset for training the models. Eight robust ML algorithms were utilized to develop predictive models. Among these, the Ensemble method leveraging a voting classifier achieved the best performance, attaining an accuracy of 95% and a recall of 95%. TensorFlow was used to integrate the machine learning model into an Android application, enabling real-time predictions. The Android app, developed in Java and built using the Android Studio platform, is designed to offer recommendations for stroke prevention and management. This user-friendly application aims to enhance accessibility to stroke risk predictions, empowering users to take proactive measures to safeguard their health..

Keywords: *SMOTE Analysis, Android App, Machine Learning, Stroke, Recommendations, Android Studio.*

How to Cite: Md. Jahirul Islam; Sukhdeb Chandra Das; Md.Golam Mostofa (2025) Automated Stroke Prediction and Prevention Recommendations: Development of an Android Application. *International Journal of Innovative Science and Research Technology*, 10(8), 2661-2674. https://doi.org/10.38124/ijisrt/25aug1562

I. INTRODUCTION

A stroke occurs when the brain's blood supply is disrupted, causing brain cells to die due to a lack of oxygen and nutrients [1]. Strokes primarily occur in two forms: ischemic strokes, caused by reduced blood flow to the brain, and hemorrhagic strokes, resulting from bleeding within the brain. These classifications are vital for determining appropriate treatment strategies and predicting outcomes. Both types can cause certain regions of the brain to cease functioning, as noted in medical literature. Globally, stroke is a significant health concern. As per the World Stroke Organization, an estimated 13 million individuals are afflicted by strokes annually, culminating in approximately 5.5 million fatalities worldwide [2-4]. This ailment represents one of the foremost contributors to global mortality and morbidity, exerting a profound and multifaceted influence on individuals' lives, encompassing familial dynamics, social engagements, and professional endeavors.

A common misconception is that strokes primarily impact specific demographics, such as older adults or individuals with pre-existing conditions. Nevertheless, strokes are indiscriminate in their occurrence, impacting individuals irrespective of age, gender, or general health status. They manifest as abrupt and critical disruptions in cerebral blood flow, effectively depriving neural cells of essential oxygen and nutrients. While ischemic strokes are the most prevalent type, caused by blockages or narrowing of arteries, hemorrhagic strokes are less common and result from ruptured blood vessels in the brain. Both forms can lead to temporary or permanent damage, depending on their severity.

Risk factors for stroke include age (particularly those over 55), a history of stroke or transient ischemic attacks (TIA), high blood pressure, atrial fibrillation, carotid stenosis due to atherosclerosis, smoking, high cholesterol, diabetes, obesity, physical inactivity, blood clotting disorders, estrogen therapy, substance abuse (e.g., cocaine or illicit drugs), and heart conditions such as myocardial infarction or cardiac arrest [5,6]. These factors significantly increase the likelihood of stroke occurrence.

Strokes often present with sudden and unexpected symptoms. Typical indicators of a stroke encompass unilateral paralysis, facial, arm, or leg numbness, impaired speech or ambulation, vertigo, disorientation, intense headaches, nausea, facial drooping, and, in extreme instances, loss of consciousness or coma. These manifestations may emerge abruptly or evolve incrementally, with certain individuals retaining full consciousness throughout the episode. Early recognition and prompt medical intervention are crucial for improving outcomes and minimizing long-term damage.

➢ *Motivation*

The global incidence of stroke is on the rise, underscoring the critical need for early identification to reduce mortality and long-term disability. Conventional approaches to evaluating stroke risk are frequently labor-intensive and prone to inaccuracies, which can delay timely interventions and negatively influence patient outcomes. In contrast, machine learning algorithms offer a transformative solution by generating interpretable models and enabling the creation of an Android-based application for stroke prediction and personalized recommendations. These algorithms enable accurate risk assessment based on various clinical factors, allowing for the rapid identification and timely treatment of high-risk patients.

This research seeks to meet the escalating need for transparent and dependable machine learning frameworks within the healthcare domain by designing a lucid, precise, and user-centric system. By leveraging a smartphone application, the proposed solution seeks to enhance stroke prediction accuracy and provide actionable recommendations to patients. The overarching objective is to alleviate the impact of stroke-related morbidity and mortality by enabling early intervention and enhancing clinical decision-making through the integration of advanced technological solutions.

➢ *Objectives*

- *The Primary Contributions of this Study are as Follows:*

✓ This research harnesses machine learning methodologies to enhance the precision of stroke prediction, offering a significant advancement in the field of medical diagnostics. Ten key stroke risk factors, each strongly correlated with stroke occurrence, and were utilized to train the predictive models.
✓ The machine learning model, trained on various stroke-related features, predicts whether an individual is at risk of experiencing a stroke. A positive prediction indicates a potential stroke risk, prompting the provision of tailored recommendations to mitigate this risk. Conversely, a negative prediction reassures the user of being stroke-free.
✓ An Android application was developed to identify individuals at risk of stroke and provide actionable recommendations. The app features ten input fields corresponding to the risk factors, enabling users to input their data. Based on the entered information, the app determines the user's stroke risk status. Additionally, it offers appropriate recommendations for those identified as at risk, facilitating early intervention and preventive care.

II. LITERATURE REVIEW

In recent years, various methods for forecasting strokes have been developed and published. Nevertheless, the majority of studies have predominantly utilized a restricted range of ML techniques for stroke prediction. While deep learning models have been employed in various investigations, the application of image processing for stroke detection continues to be relatively unexplored and underdeveloped.

Aminul Haque et al. [6] proposed a machine learning-based method to predict asthma risk. While their study focused on asthma, our research emphasizes stroke prediction, marking a significant difference in application areas. Krishna et al. [7] developed a stroke prediction model using Kaggle data and six machine learning algorithms. Unlike their approach, which focused solely on prediction, our research extends to developing an automated Android app for stroke prediction with high recall and actionable recommendations for patients. Md. Milon Islam et al. [8] utilized 10-fold cross-validation to develop a breast cancer prediction model. Other models for breast cancer were proposed in [9] and [10]. While these studies focused on cancer prediction, our work is centered on stroke risk assessment, underscoring the different applications. Islam, S. I. Ayon et al. [11] introduced a deep neural network-based method for diabetes diagnosis using five- and ten-fold cross-validation. In contrast, our study employs eight ML algorithms, providing a broader analysis of stroke prediction models. Saumya Gupta et al. [12] applied various ML models, including Gaussian Naïve Bayes, Logistic Regression, Decision Tree, and others, for stroke prediction. Their study focused on comparing methods, while our work integrates stroke prediction with recommendations via an Android app. Viswa Priya S. E. et al. [13] proposed a hybrid Artificial Neural Network-Random Forest (ANN-RF) model for stroke prediction with 94% accuracy. Our study builds on this by incorporating an Android app to deliver predictions and recommendations, improving accessibility and usability.

Tasfia Ismail Shoily et al. [14] employed four ML algorithms to identify stroke types. While their study was limited to classification, our research provides tailored recommendations using an Android app, alongside predictions. Soumyabrata Dev et al. [15] utilized statistical methods to identify critical variables for stroke prediction. Despite achieving high accuracy, their study did not include patient recommendations, which is a focus of our work. Mohammed Saidul Islam et al. [16] explored Explainable AI (XAI) to predict stroke outcomes using EEG data, achieving 80% accuracy. While innovative, our study focuses on utilizing demographic and clinical data for more practical and scalable predictions. Elias Dritsas et al. [17] identified stacking as the most effective strategy for stroke prediction, achieving 98.9% AUC. Our research builds on such approaches by incorporating ensemble methods and focusing on recall improvements.

Redwanul Islam et al. [18] compared classifiers like DT, SVM, and XGBoost for stroke prediction using hospital data. Our study differs by incorporating eight algorithms and offering recommendations via an Android app. Youngkeun Choi et al. [19] achieved high accuracy using decision trees. Our work advances this by integrating an ensemble voting classifier and developing an Android app. Tahia Tazin et al. [20] employed SMOTE to balance datasets for stroke prediction. While effective, their study lacked a recommendation component, which is a key feature of our Android app. Hager Saleh et al. [22] utilized distributed ML systems for stroke prediction, achieving 90% accuracy with Random Forest. Our study achieves 95% accuracy using an ensemble voting classifier with high recall. Hanifa and Raja [23] applied advanced mathematical techniques to enhance stroke prediction accuracy. Our study focuses on practical implementations via Android applications for wider accessibility.

Therefore, this study utilizes ten specific characteristics to forecast stroke risk accurately. By employing an ensemble voting classifier, we achieve superior accuracy, recall, F1 score, and precision metrics compared to prior works. Furthermore, the Android app developed in this research provides a user-friendly interface for predicting stroke risks and delivering personalized recommendations, addressing the growing need for accessible healthcare technologies.

III. METHODOLOGY

This research was conducted in two primary stages: the first involved the development of machine learning models to analyze stroke-related data and produce predictions, while the second focused on creating a Java-based Android application to provide personalized recommendations based on the prediction results. Google Colab was employed for data preprocessing and model training, following a structured workflow comprising four essential steps: data sourcing, preparation, algorithm training, and prediction. Data sourcing ensured a sufficient number of samples with diverse variations, while preparation involved preprocessing techniques such as imputation to handle missing values and feature scaling to standardize variable distributions. Feature selection was performed during training to eliminate redundant features with minimal impact on prediction accuracy, thereby improving the model's overall performance. Eight reliable machine learning algorithms were compared and applied to ensure robust predictions. The trained machine learning models were exported using TensorFlow and subsequently integrated into Android Studio for use in the mobile application. The Android application was developed using Java and XML, with XML managing the design aspects and Java implementing the core logic. This application predicts an individual’s stroke risk using the trained machine learning models and offers appropriate recommendations based on the results. Figure 1 illustrates the proposed methodology, encompassing both the implementation of machine learning models and their integration into the Android app.

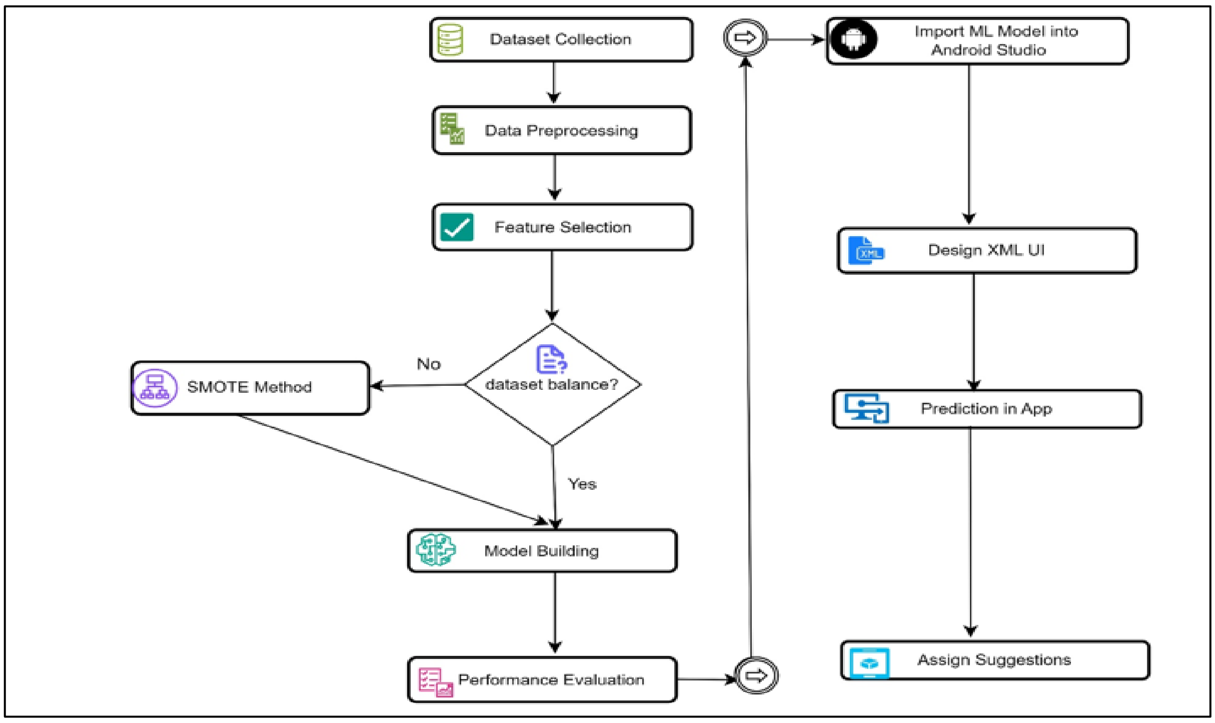


Fig 1 Proposed Methodology

➢ *Dataset Collection*

Machine learning algorithms necessitate a well-organized and meticulously prepared dataset to optimize predictive performance. These algorithms depend on detecting patterns and characteristics within the data to make accurate forecasts. Therefore, the quality and structure of the dataset play a pivotal role in the successful implementation of machine learning models. In this study, a dataset consisting of 5,110 rows and 12 columns was employed for stroke prediction, sourced from the reputable data science platform Kaggle [24]. Notably, the dataset exhibited an inherent imbalance, with 4,861 rows representing non-stroke cases (labeled as 0) and only 249 rows representing stroke cases (labeled as 1). To mitigate this imbalance and improve the model's predictive accuracy, preprocessing techniques, including the Synthetic Minority Over-sampling Technique (SMOTE), were applied to balance the dataset. Figure 2 displays the distribution of target samples, and Table 1 offers a comprehensive overview of the dataset.

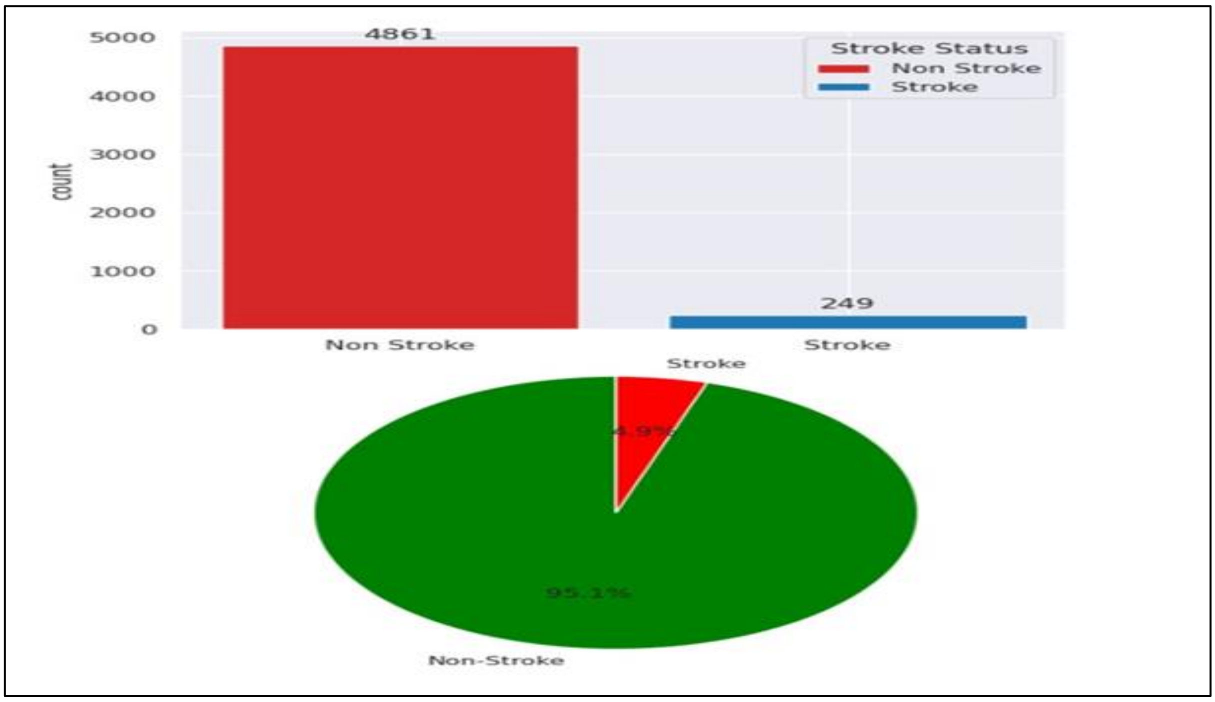


Fig 2 Target Samples Distribution

Table 1 Dataset Description

Feature	Description
Id	Any amount in numbers
Gender	Female or Male or other
Ever Married	No, Yes
Work Type	children or govt_job or never_worked or private or self-employed
Residence Type	rural or urban
Smoking Status	never smoked or previously smoked or currently smokes, or unknown
Hypertension	No hypertension is 0, having hypertension is 1
Heart Disease	0 for no heart diseases, 1 for heart disease
Stroke	Stroke = 1, no stroke = 0
Age	The patient's age in number
Average Glucose Level	blood glucose level
BMI	Body Mass Index

➢ *Data Pre-Processing*

Data preprocessing is crucial to optimize machine learning models by addressing inconsistencies, missing values, and irrelevant information. In this study, the dataset contained twelve attributes, with the 'id' column excluded as it was irrelevant. Missing data in the BMI column was imputed by replacing the missing values with the column's mean. Additionally, categorical variables such as gender, marital status, work type, residence type, and smoking status were encoded using one-hot encoding, a technique that converts categorical data into a numerical format for efficient model training.

➢ *Feature Correlation to Target*

The Figure 3 displays a horizontal bar chart showing the correlation coefficients between various features and the target variable (presumably stroke prediction). Each feature is listed on the vertical axis, and the corresponding correlation coefficient is plotted on the horizontal axis.

- Age has the highest correlation with the target, followed by heart disease and hypertension, which indicates that these factors are more strongly related to the likelihood of a stroke in the dataset.

- Ever married and smoking status also show moderate correlations, suggesting some influence but not as significant as the health-related features.

- Residence type, gender, and work type exhibit very low correlations with the target, indicating minimal or no direct influence on stroke prediction based on this dataset.

This chart helps in identifying the most influential factors for stroke prediction, guiding feature selection for building a more efficient model.

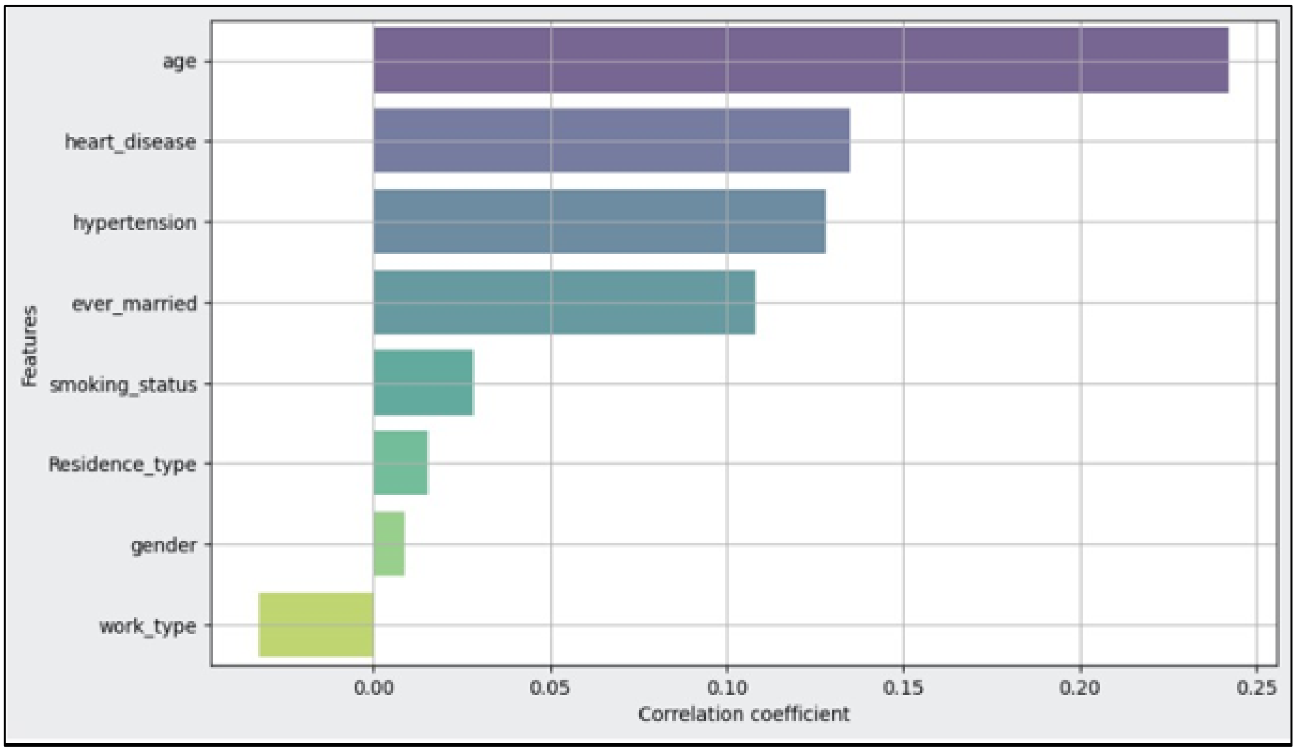


Fig 3 Feature Correlation to target variable

➢ *Model Building*

This research utilizes eight machine learning techniques, each chosen for its ability to effectively process the dataset and deliver high accuracy. The study evaluates the error rates of each method to identify the models with the lowest error, thus facilitating the selection of the most accurate model for stroke prediction.

- *Random Forest:*

The Random Forest classifier combines the predictions of multiple decision trees, each trained on different subsets of the dataset [25]. Generally, as the number of trees in the forest increases, the accuracy of the model improves, while the likelihood of overfitting is reduced. Random Forest employs a technique called bagging, which involves selecting random samples with replacement from the training set ($X = x_1, ..., x_n$) along with corresponding responses ($Y = y_1, ..., y_n$). Multiple decision trees are then fitted to these bootstrapped samples to refine the model's predictions.

$$J = \frac{1}{B}\Sigma_{b=1}^{B} fb(X') \qquad (1)$$

By averaging J approximations from each unique tree on the unseen samples X', the model generates predictions for those unidentified samples. This process involves the aggregation of results from multiple decision trees, each trained on different subsets of the data, leading to more robust and generalized predictions. The Random Forest methodology outlined above is visually represented in Figure 4, which illustrates how individual trees contribute to the final prediction by averaging their outputs on the unseen samples.

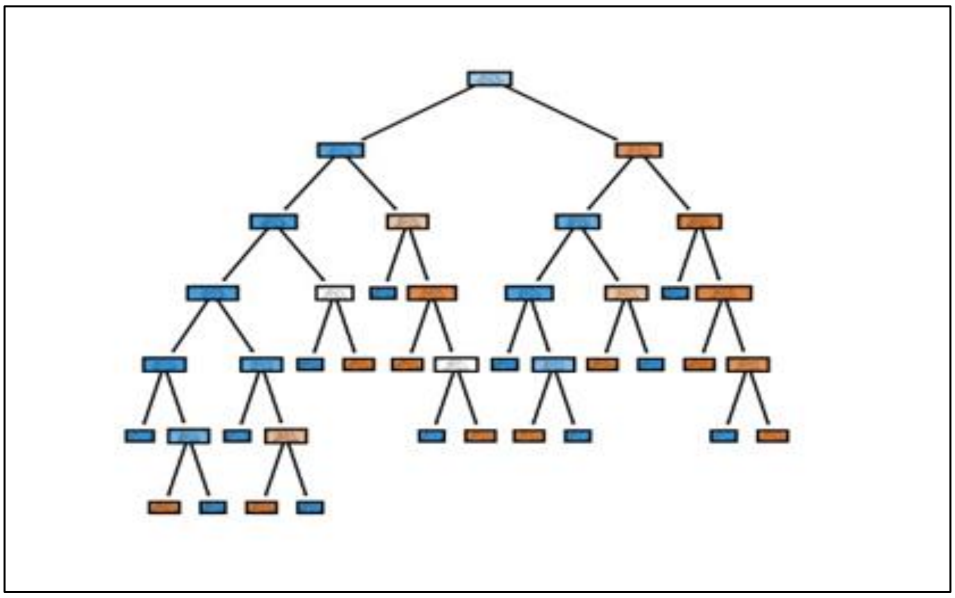


Fig 4 Diagram of Random Forest

- *K-Nearest Neighbor Classifier:*

The K-Nearest Neighbors (K-NN) technique is widely regarded as one of the fundamental algorithms in Machine Learning, grounded in the Supervised Learning paradigm. It works by classifying a new data point based on its similarity to previously labeled data, using all available data for comparison. This allows K-NN to efficiently categorize new data into predefined classes. The algorithm identifies the closest neighbors by calculating the shortest distance between data points, with the Euclidean distance being the most commonly used metric [26]. There are several methods for calculating the Euclidean distance, including:

$$\text{Euclid Distance} = \sqrt{\Sigma_{i=0}^{k}(x_i - y_i)^2} \qquad (2)$$

This process is visually represented in Figure 5, which illustrates how the K-Nearest Neighbors algorithm accurately classifies a new data point by evaluating its proximity to neighboring points within the dataset. The diagram showcases how K-NN computes the distance between the unknown data point and other existing points, ultimately assigning it to the most relevant category based on the majority class of its nearest neighbors.

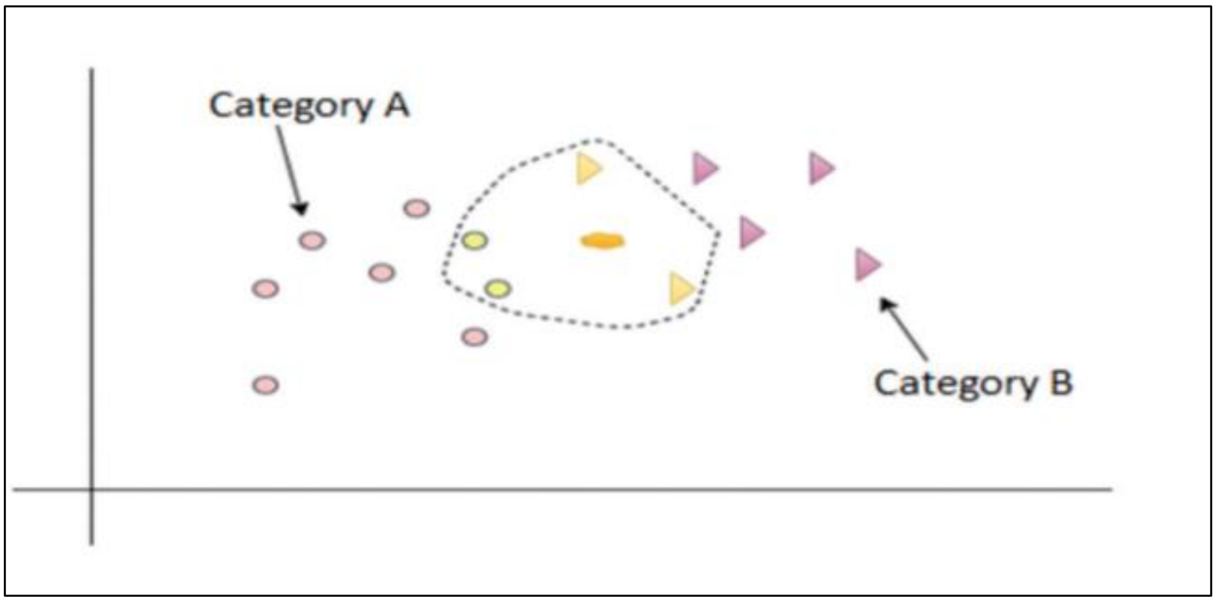


Fig 5 Diagram of K-Nearest Neighbor Classifier

- *MLP Classifier:*

The Multi-Layer Perceptron (MLP) network is composed of fully connected layers that extend from the input layer to the output layer. The depth of the network is determined by the number of hidden layers situated between the input and output layers. As illustrated in Figure 6, the network's depth can be augmented by incorporating additional hidden layers. This expansion enables the model to capture more intricate patterns within the data while preserving the input and output layers as constant elements.

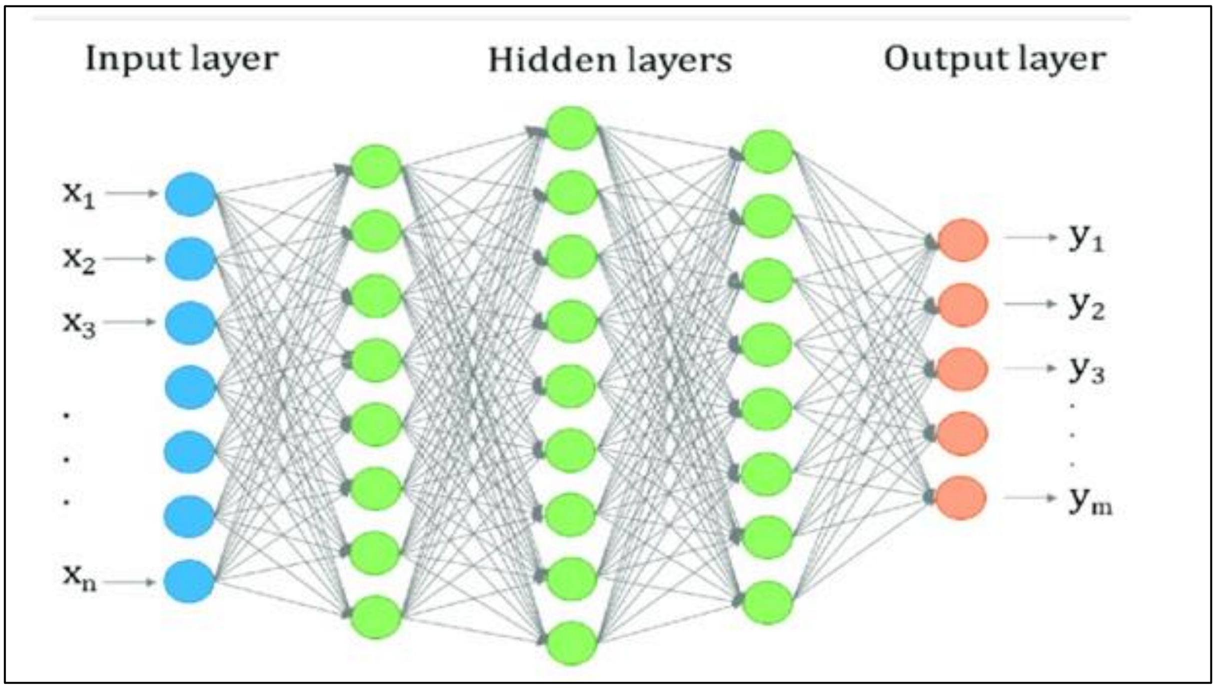


Fig 6 Configuration of MLP Classifier Network

- *AdaBoost Classifier Algorithm:*

AdaBoost algorithm works by adjusting the weights of the classifier and the training data samples in each iteration, with a focus on improving predictions for misclassified observations. This method can be utilized as a base classifier in any machine learning algorithm that accepts weighted training data. The AdaBoost process begins by selecting a random subset of the training set, and then iteratively enhances the model by focusing on the data points misclassified in previous iterations. As each iteration progresses, AdaBoost increases the weights of these misclassified instances, giving them more importance in subsequent rounds [27]. It also prioritizes accurate classifications, adjusting classifier weights based on the performance of previous rounds. To make final decisions, AdaBoost assigns a voting mechanism to each learning method in the ensemble, combining their outputs for a more accurate classification. Through this iterative process, AdaBoost is able to significantly improve classification accuracy, particularly in complex tasks.

- *SVM Classifier:*

The fundamental principle underlying Support Vector Machines (SVM) is the identification of the optimal decision boundary, or hyperplane, within an n-dimensional feature space that adeptly segregates distinct classes of data points [28]. This hyperplane is meticulously chosen to maximize the margin, defined as the greatest distance between the hyperplane and the closest data points from each class, referred to as support vectors. By this maximization process, SVM endeavors to construct a classifier that not only achieves the highest possible classification accuracy on the training dataset but also possesses robust generalizability, ensuring effective performance on previously unseen data.

- *Decision Tree:*

Decision Trees are a highly effective and versatile predictive modeling technique, utilized in a broad range of applications. A Decision Tree is composed of two primary types of nodes: Decision Nodes and Leaf Nodes [29]. Decision Nodes serve as pivotal points where choices are made based on the values of specific features, and these nodes subsequently branch into multiple paths, each corresponding to different potential outcomes. In contrast, Leaf Nodes symbolize the ultimate predictions or classifications derived from the sequence of decisions at the preceding nodes. These Leaf Nodes do not have any further branches, representing the final decision or outcome of the tree's logic.

- *XGBoost Algorithm:*

XGBoost is an advanced gradient boosting algorithm designed to minimize the model’s cost function by emphasizing the exploration of functional space [30]. This strategic approach enhances its predictive accuracy and efficiency. XGBoost improves the performance of weaker learners by leveraging advanced parallel processing and optimized algorithms, enabling faster and more efficient training. Its iterative process involves sequentially adding new decision trees that focus on predicting the residuals or errors of the previously trained trees. The predictions from these newly added trees are then combined with the outputs from earlier trees, gradually refining the model's final prediction and enhancing overall performance.

$$F_{xt+1} = F_{xt} + \varepsilon_{xt} \frac{\delta F}{\delta x}(xt) \tag{3}$$

- *Ensemble Method (Voting classifier):*

Ensemble learning has garnered considerable prominence in the machine learning domain due to its capacity to synergize the predictions of multiple base models, thereby augmenting forecast precision. A particularly prominent ensemble technique is the Voting Classifier, which amalgamates the outputs of several base estimators (machine learning models) and assigns the class label based on a democratic voting mechanism [31]. This versatile approach can be effectively employed in both regression and classification tasks. Numerous strategies, including parametric, nonparametric, heuristic, logical, and probabilistic methodologies, have been proposed to design robust voting systems. A prevalent strategy entails the utilization of diverse learning algorithms—such as Random Forest, K-Nearest Neighbors (KNN), and XGBoost—on the same training dataset. Each classifier, informed by its distinct learning paradigm and feature set, generates separate predictions. To derive the most accurate forecast, a voting procedure is implemented in which the class receiving the majority of votes is chosen as the final output. This ensemble paradigm mitigates the risk of all classifiers concurrently making erroneous predictions, thereby enhancing the robustness of the model. Figure 7 visually represents the ensemble method employed in this study, wherein the integration of three distinct machine learning algorithms culminates in a potent classifier, capitalizing on the complementary strengths of each constituent model to elevate predictive efficacy.

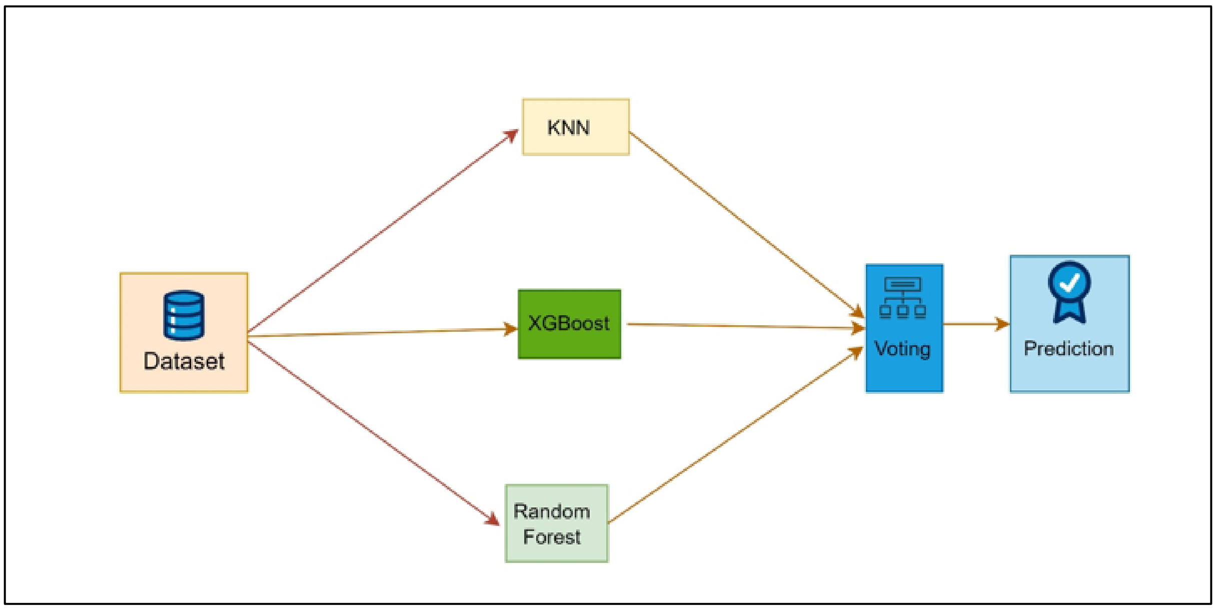


Fig 7 Ensemble Method

➢ *Android App Development*

The program was developed using the Java programming language, with version 10.0 of the platform. XML was utilized to implement the design, while Android Studio was employed for both coding and design completion. Figure 8 demonstrates the outcome and the developmental journey of an Android application created for stroke prediction.

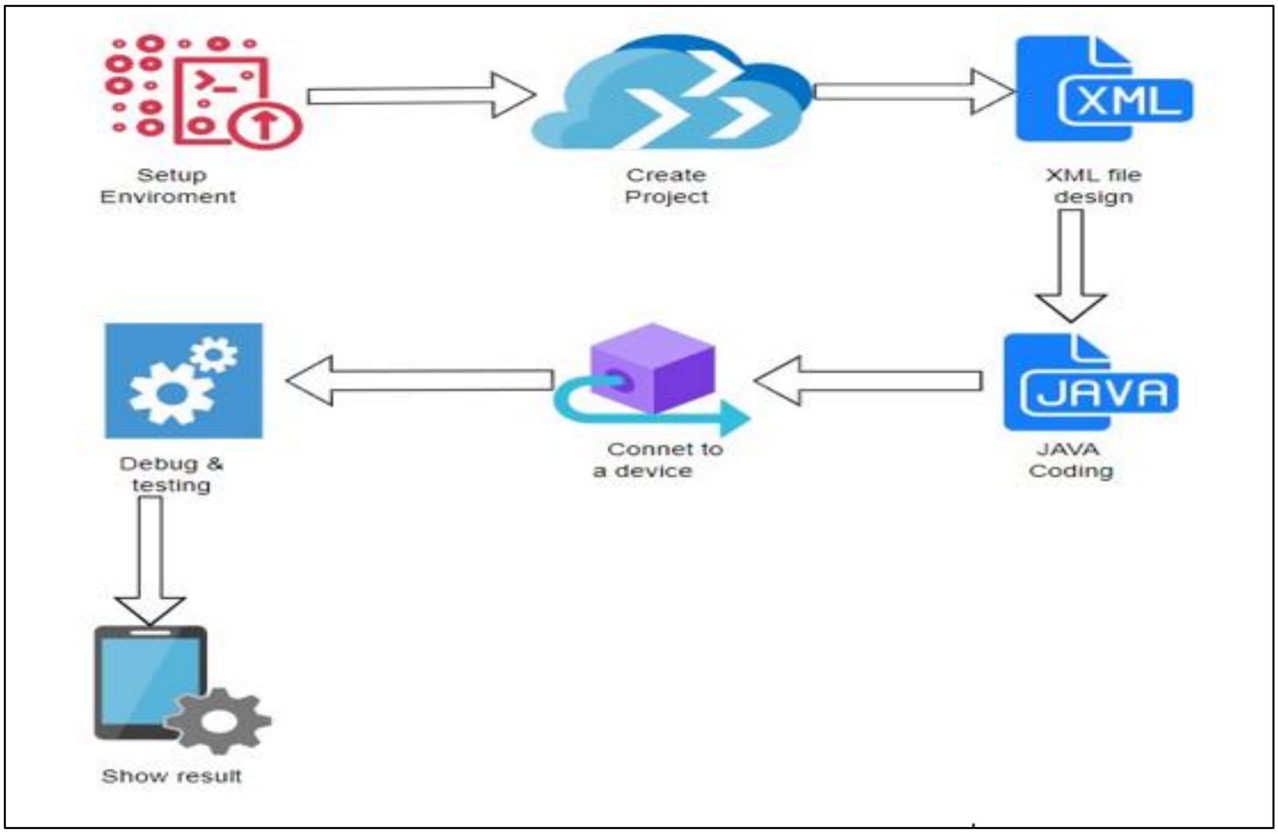


Fig 8 Development Process of Stroke Prediction App

✓ *Create Project*

The project development begins with the download and installation of Android Studio, which includes the Android SDK and Android Virtual Device (AVD) [32]. Once the installation is complete, we proceed to configure the API level to meet the project's requirements. After setting up the necessary components, Android Studio is opened by selecting the "Quick Start" button and choosing the "Start a new Android Studio project" option. During this stage, the corporate domain and the application name are defined, which are then combined to form the package name for the project. This package file is prepared for upload to Google Play, and the location for saving the project file is specified.

✓ *XML File Design*

XML (Extensible Markup Language) is a key technology for designing layouts in Android applications. It functions similarly to HTML but differs in its purpose and syntax. While HTML is designed for displaying content on the web, XML was created as a data encoding standard for web-based applications. XML is more rigid in its structure, requiring proper closure of tags, adherence to whitespace conventions, and case sensitivity.

In Android development, various layout types can be utilized within XML files. For this project, we opted for a LinearLayout, which arranges elements either horizontally or vertically based on the specified attributes. To present multiple selectable options to users, a RadioGroup containing individual RadioButton items was employed [33]. The TextView element is used to display the question or prompt to the user, ensuring clear communication within the interface. Additionally, an EditText field was included to capture user input, allowing for interactive data collection. This setup allows the app to gather information from the user while maintaining a clean and organized layout.

✓ *JAVA Coding*

The Android SDK and Java are used to build Android apps. Java is object-oriented, robust, secure, and simple. Android uses the Dalvik Virtual Machine (DVM), optimized for mobile devices, instead of the standard JVM. The design specified in XML is executed through Java code. The onCreate() function is where all code is written. Java calls the views using the unique IDs provided in the XML layout using findViewById().

✓ *Connect to a Device*

To start, the developer connects the device to the development workstation using a USB cable. USB debugging must be enabled on the device by first selecting the Developer options. Once the application module is selected in Android Studio's project window, the developer clicks "Run" from the toolbar. In the "Select Deployment Target" dialog, the developer chooses their own device and clicks the OK button. Android Studio then installs and launches the application on the connected device, making the developer's app visible and ready for use on their personal smartphone.

- ✓ *Debug and Test*

Debugging is a cruscial part of a developer's workflow, allowing them to inspect an application's variables, methods, and the performance of each line of code. This helps identify minor errors in large codebases. Before starting the debugging process, the device must be properly configured and connected via USB for debugging. Once the device is set up, Android Studio initiates the debugging application, and the developer selects the device in the "Device Choice" box. Android Studio can also automatically launch the debug tool. Alternatively, the developer can manually start the debugging process by clicking "Debug" at the bottom of Android Studio. This process helps identify and resolve issues in the app efficiently.

- ✓ *Show the Result*

Once the coding and design are complete and functioning correctly, the final product is displayed on the connected device. In this phase, the user interface will show various fields that the user must fill out in order to input relevant data for stroke prediction. Based on the information provided by the user, the software will process the input and generate predictions, offering recommendations related to stroke risk or diagnosis. This feature ensures that the app functions as intended, providing valuable insights to the user.

IV. RESULT & DISCUSSION

The numerous experiments are divided into subsections in this part, each with a distinct set of findings. To evaluate the dataset, we used eight machine learning methods. One piece of data is regarded as output, and the other ten are considered input. The results for the ML model and the Android application are explained in subsections 4.1 and 4.2.

- ➢ *Result for ML Model*

To ascertain the optimal model for stroke prediction, an experimental framework was employed wherein the dataset was preprocessed using Google Colab, partitioning 20% for testing and the remaining 80% for training. Subsequently, various machine learning algorithms were applied, and classifiers were constructed based on the training set. Their efficacy was gauged using precision, accuracy, recall, and F-measure metrics, tailored to the dataset. The equation for accuracy, as outlined below, quantifies the proportion of correct predictions, encompassing both stroke and non-stroke classifications:

$$\text{Accuracy} = \frac{TP + TN}{TP + FN + TN + FP} \tag{4}$$

Recall, frequently denoted as sensitivity, quantifies the proportion of true positive instances that are accurately identified by the model. It is mathematically expressed as:

$$\text{Recall} = \frac{TP}{TP + FN} \tag{5}$$

Precision gauges the proportion of true positive instances among those predicted as positive by the model. It can be mathematically represented as:

$$\text{Precision} = \frac{TP}{TP + FP} \tag{6}$$

The F1 score is computed by taking the harmonic mean of precision and recall, thus offering a balanced measure of a model's performance, particularly when there is an uneven class distribution. The formula for the F1 score is:

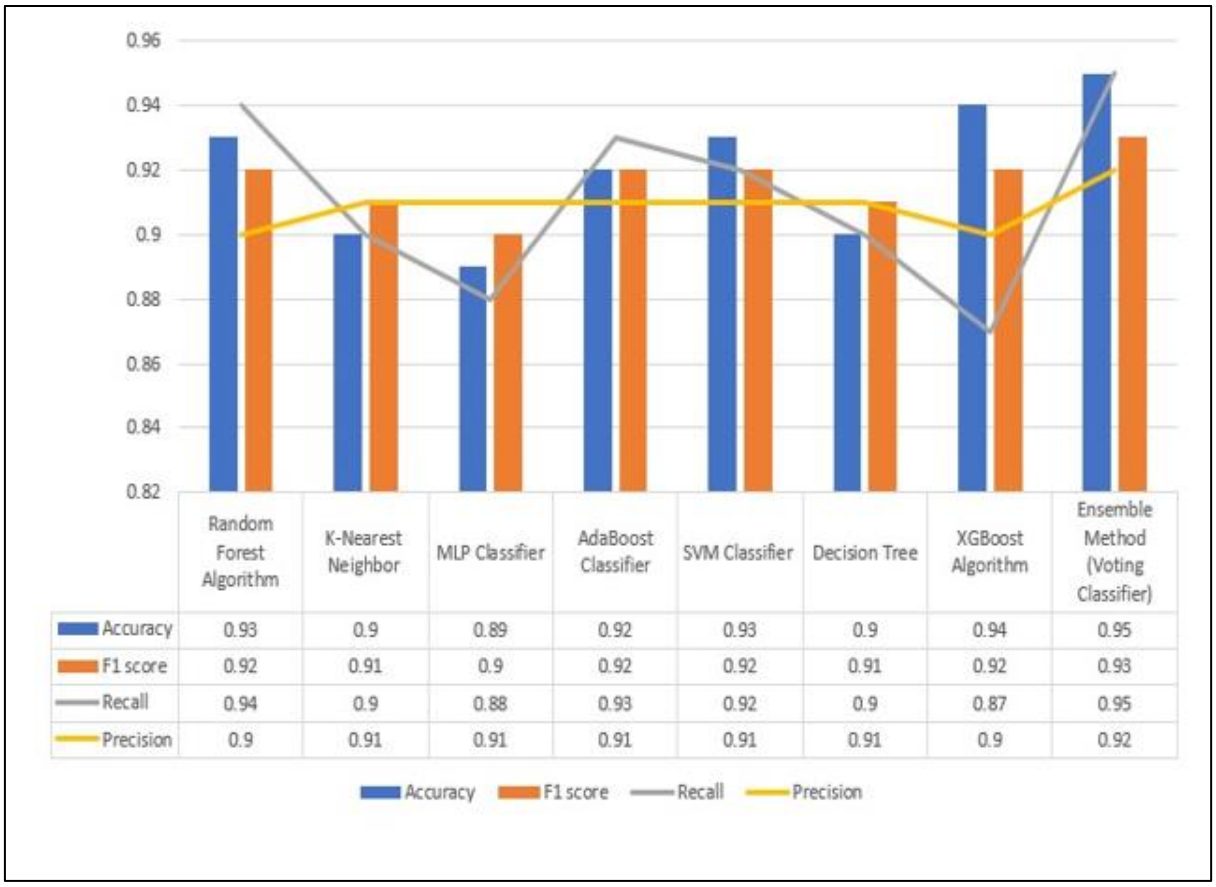
$$\text{F1 score} = \frac{2 \times precision \times recall}{precision + recall} \tag{7}$$

In this study, the ability of eight mathematical algorithms to forecast the severity of stroke was evaluated based on their performance in classifying the outcomes into four categories: True Positive (TP), False Positive (FP), True Negative (TN), and False Negative (FN). These categories reflect the accuracy of the algorithm's predictions. The results of these classifications, for each of the eight algorithms, are presented in Table 2. This table highlights how each algorithm performed in predicting the severity of stroke, enabling a comparison of their effectiveness in classifying stroke severity correctly. The prediction abilities of these algorithms were thoroughly examined to determine the most accurate and reliable method for stroke severity classification.

Table 2 Eight algorithm and metrics score

Algorithm	Accuracy (%)	F1 score (%)	Recall (%)	Precision (%)
Random Forest	93	92	94	90
K-NN	90	91	90	91
MLP	89	90	88	91
AdaBoost	92	92	93	91
SVM	93	92	92	91
Decision Tree	90	91	90	91
XGBoost	94	92	87	90
Ensemble Method (Voting classifier)	95	93	95	92

In Table 2, we compared the performance measures—accuracy, precision, recall, and F1 score—across eight robust algorithms. The results show that, overall, the algorithms performed well, with high levels of accuracy achieved by all. Among the eight algorithms, the Ensemble method using the voting classifier achieved the highest accuracy at 95% and the highest recall at 95%. The MLP Classifier had the lowest accuracy at 89%, while XGBoost exhibited the lowest recall at 87%. As shown in the Figure 9 below, the Ensemble method using the voting classifier also outperformed the others in terms of F1 score and precision, achieving 93% and 92%, respectively. The MLP Classifier had the lowest F1 score at 90%. Additionally, both the Random Forest and XGBoost algorithms had the lowest precision, with values of 90%. These results highlight the strengths and weaknesses of each algorithm in terms of different performance metrics for stroke severity prediction.



	Random Forest Algorithm	K-Nearest Neighbor	MLP Classifier	AdaBoost Classifier	SVM Classifier	Decision Tree	XGBoost Algorithm	Ensemble Method (Voting Classifier)
Accuracy	0.93	0.9	0.89	0.92	0.93	0.9	0.94	0.95
F1 score	0.92	0.91	0.9	0.92	0.92	0.91	0.92	0.93
Recall	0.94	0.9	0.88	0.93	0.92	0.9	0.87	0.95
Precision	0.9	0.91	0.91	0.91	0.91	0.91	0.9	0.92

Fig 9 Algorithm Performance Comparison

The Ensemble method using the voting classifier emerged as the best algorithm among the eight robust algorithms implemented. This algorithm achieved the highest accuracy and recall, both of which are crucial performance metrics, especially in the medical sector. High accuracy ensures reliable predictions, while high recall is essential for identifying true positive cases, minimizing the risk of missed diagnoses. These qualities make the ensemble method particularly effective and valuable for stroke prediction and other critical medical applications.

➢ *Android Application Development*

Android applications are primarily developed using the Java programming language and make use of core Java libraries for various functionalities [32]. For the machine learning tasks, we employed TensorFlow, a widely recognized open-source framework for machine learning and artificial intelligence. TensorFlow operates as a symbolic math library rooted in the principles of dataflow and differentiable programming [33]. In this study, the model was optimized and converted into a mobile-compatible format using the TensorFlow Lite (TFLite) extension. TFLite is a streamlined version of TensorFlow specifically designed for mobile and embedded systems, enabling efficient model deployment on resource-constrained device To integrate the TensorFlow model into the Android app, we first create an "assets" directory in Android Studio to store the tflite file. Additional dependencies are included to enable the app to execute the TensorFlow file on Android devices. Before launching the application, a splash screen is designed to provide a welcoming interface to the user, as illustrated in Figure 10(a). The XML file for the stroke prediction application is created with a form containing ten fields for user input. This includes five fields where the user can select options and five fields where the user can enter text. Once the user completes the form, they can click the prediction button to initiate the stroke prediction process based on the provided data.

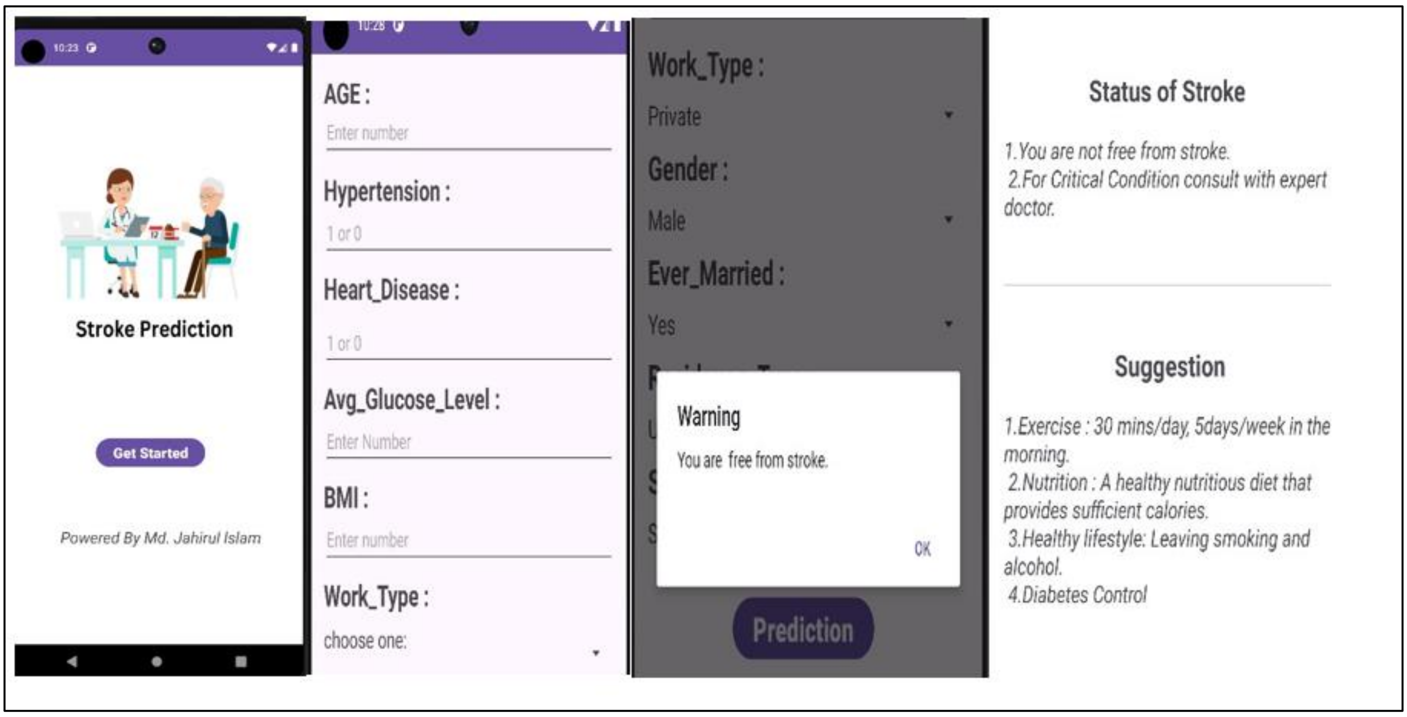

Stroke Prediction
Get Started
Powered By Md. Jahirul Islam
AGE :
Enter number
Hypertension :
1 or 0
Heart_Disease :
1 or 0
Avg_Glucose_Level :
Enter Number
BMI :
Enter number
Work_Type :
choose one:
Work_Type :
Private
Gender :
Male
Ever_Married :
Yes
Warning
You are free from stroke.
OK
Prediction
Status of Stroke
1.You are not free from stroke.
2.For Critical Condition consult with expert doctor.
Suggestion
1.Exercise : 30 mins/day, 5days/week in the morning.
2.Nutrition : A healthy nutritious diet that provides sufficient calories.
3.Healthy lifestyle: Leaving smoking and alcohol.
4.Diabetes Control

Fig 10 Android App output: Screen of Splash and Data Input, Result Screen for Non-Stroke and Stroke.

The Android application provides a prediction result based on the user's input. If the result indicates that the user may have a stroke, the outcome will be displayed as 0, signaling that there is a possibility of stroke. If the user is not at risk, the result will be displayed as 1, indicating no stroke risk. For non-stroke patients, the app will display the message "You are free from stroke" through an AlertDialog, as shown in Figure 10(b). AlertDialog is a method in Android Studio used to display messages or alerts to the user. In case the user is identified as being at risk for a stroke, the application will redirect them to a separate page that provides more detailed information. On this page, two key items are displayed:

- The stroke status of the user, which clearly indicates whether the user is at risk.

- *A Recommendation to Consult a Skilled Doctor if the Condition Appears Serious.*

If the user's condition is not serious, the app will provide suggestions that aim to reduce the likelihood of a stroke occurring in the future. These suggestions are designed to encourage lifestyle changes and preventive measures that may help lower the risk of a stroke.

V. CONCLUSION

Developing an Android app to identify stroke patients can play a pivotal role in the long-term preservation of human life. Predicting stroke risk in medical science is a complex but crucial task, as early identification can lead to timely interventions and potentially save lives. The stroke risk prediction tool presented in this paper leverages an Android application integrated with a machine learning classifier. The dataset for stroke patients was sourced from Kaggle, which provided a comprehensive set of records. We then applied eight different machine learning algorithms to analyze the dataset, achieving impressive accuracy across all models.

Following this, we created an Android application that utilized TensorFlow to implement the predictions, transforming the model's output into a usable tool for real-time forecasting. The Android application was built using Android Studio, ensuring a smooth and efficient user interface. The application not only provides predictions but also offers recommendations based on the forecasted results. To validate the efficacy of the system, we collected data from a small sample of patients and ran predictions using the developed model. The output generated by the system aligned closely with the recommendations provided by medical professionals, confirming the accuracy and reliability of the app. The physicians were particularly pleased with the tool, as it provided valuable insights to support their clinical decisions.

However, one limitation of the current system is that it still requires physician assistance for report evaluation and interpretation. In the future, we plan to enhance the application by integrating a chatbot designed specifically for stroke victims. The chatbot will serve as an accessible and interactive tool, helping stroke victims to better understand their health status and follow up on their recovery process. By offering direct, real-time communication, the chatbot will significantly improve user engagement and support patients on their path to better health and a higher quality of life.

STATEMENTS AND DECLARATIONS

➢ *Funding*

The authors declare that no funds, grants, or other support were received during the preparation of this manuscript.

➢ *Completing Interests*

The authors have no conflict of interests to disclose.

➢ *Ethics Approval and Consent to Participate*

This article does not contain any studies with human participants and animals performed by any of the authors.

➢ *Availability of Data and Material*

The datasets generated during the current study are available from the corresponding author on reasonable request.

➢ *Consent for Publication*

Not applicable

REFERENCES

[1]. "Learn about Stroke. Available on: "[Online]. Available: https://www.world-stroke.org/world-stroke-day-campaign/why- stroke-matters/learnabout-stroke, [Accessed 2023].

[2]. T. Elloker and A. Rhoda, "The relationship between social support and participation in stroke: A systematic review.," pp.1-9, 2018.

[3]. M. Katan and A. Luft, "Global burden of stroke," in In seminars in Neurology, New work, USA, 2018.

[4]. X. Xia, W. Yue, B. Chao, M. Li, L. Cao, L. Wang, Y. Shen and X. Li, "Prevalence and risk factors of stroke in the elderly in Northern China: Data from the National Stroke Screening Survey. J. Neurol," pp. 1-12, 2019.

[5]. J. Lecouturier, M. Murtagh, R. Thomson, G. Ford, M. White, M. Eccles and H. Rodgers, "Response to symptoms of stroke in the UK: A systematic review," pp. 1-9, 2010.

[6]. A. J. M. M. e. a. Aminul Haque, "AI Powered Asthma Prediction Towards Treatment Formulation: An Android App Approach," Intelligent Automation & Soft Computing (IASC), vol. 34, no. 1, pp. 1-18, 2022.

[7]. S. G. e. a. KRISHNA MRIDHA, "Automated Stroke Prediction Using Machine Learning: An Explainable and Exploratory Study with a Web Application for Early Intervention," IEEE Access, vol. 11, pp. 1-24, 2023.

[8]. H. I. M. R. H. a. M. K. H. M. M. Islam, ""Prediction of breast cancer using support vector machine and K-Nearest neighbors," in 2017 IEEE Region 10 Humanitarian Technology Conf. (R10-HTC), Dhaka, Bangladesh, pp. 226-229, 2017.

[9]. M. R. H. H. I. M. M. H. M. H. e. a. M. M. Islam, "Breast cancer prediction: A comparative study using machine learning techniques," SN Computer Science, vol. 1, no. 5, p. 290, 2020.

[10]. M. M. I. a. M. M. A. H. M. K. Hasan, "Mathematical model development to detect breast cancer using multigene genetic programming,"in 5th Int. Conf. on Informatics, Electronics and Vision (ICIEV), Dhaka, Bangladesh, pp. 574-557, 2016.

[11]. S. I. A. a. M. M. Islam, "Diabetes prediction: A deep learning approach," International Journal of Information Engineering and Electronic Business (IJIEEB), vol. 11, no. 2, pp. 21-27, 2019.

[12]. S. R. Saumya Gupta, "Stroke Prediction using Machine Learning Methods,"12th International Conference on Cloud Computing, Data Science & Engineering (IEEE Xplore), pp. 1-6, 2022.

[13]. R. D. Viswa Priya S E, "A Systematic Method of Stroke Prediction Model based on Big Data and Machine Learning," IEEE Access, pp. 1-6, 2023.

[14]. T. I., S. J. S. A. T. M. e. a. Tasfia Ismail Shoily, "Detection of Stroke Disease using Machine Learning Algorithms," IEEE Access, pp. 1-6, 2019.

[15]. H. W. e. a. Soumyabrata Dev, "A predictive analytics approach for stroke prediction using machine learning and neural networks," ELSEVIER, vol. 2, pp. 1-7, 2022.

[16]. M. H. I. R. M. P. S. a. H. M. Islam, "Explainable Artificial Intelligence Model for Stroke Prediction Using EEG Signal. Sensors," MDPI, 2022, 22(24), 9859.

[17]. E. a. T. M. Dritsas, "Stroke risk prediction with machine learning techniques," Sensors, 2022, 22(13), p.4670.

[18]. R. D. S. a. P. T. Islam, "Predictive Analysis for Risk of Stroke Using Machine Learning Techniques," In 2021 International Conference on Computer, Communication, Chemical, Materials and Electronic Engineering (IC4ME2) (pp. 1-4). IEEE.

[19]. J. W. C. Youngkeun Choi, "Stroke Prediction Using Machine Learning based on Artificial Intelligence," International Journal of Advanced Trends in Computer Science and Engineering (IJATCSE), Volume 9, No.5, September - October 2020.

[20]. T. A. M. D. N. e. a. Tazin, "Stroke Disease Detection and Prediction Using Robust Learning Approaches," Journal of Healthcare Engineering, 2021.

[21]. H. P. G. G. V. P. a. P. K. B. H. K V, "STROKE PREDICTION USING MACHINE LEARNING ALGORITHMS," International Journal of Innovative Research in Engineering &amp; Management, vol. 8, no. 4, Jul. 2021, Doi: 10.21276/ijirem.2021.8.4.2.

[22]. E. a. Ali, "Stroke prediction using distributed machine learning based on Apache spark," Stroke, 28(15), pp.89-97.

[23]. S.-M. H. e. al., "Stroke risk prediction through non-linear support vector classification models," Int. J. Adv. Res. Computer. Sci. 1 (3) (2010).

[24]. On, Kaggle, [Online]. Available: https://www.kaggle.com/datasets/fedesoriano/stroke-prediction-dataset, [Accessed 2023].

[25]. Z. R. M. A. Z. H. H. N. J. P. a. M. K. S. A. Murad, "Computer-aided system for extending the performance of diabetes analysis and prediction," in 7th Int. Conf. on Software Engineering and Computer Systems (ICSECS-2021), Pahang, Malaysia, pp. 465–470, 2021.

[26]. R. N. H. T. T. I. H. S. e. a. K. Tomita, "Deep learning facilitates the diagnosis of adult asthma," Allergology International, vol. 68, no. 4, pp. 456–461, 2019.

[27]. M. P. K. R. G. V. M. R. a. K. M. M. P. B. P. Reddy, "Fake data analysis and detection using ensembled

hybrid algorithm," in 3rd Int. Conf. on Computing Methodologies and Communication (ICCMC), Erode, India, pp. 890–897, 2019.
[28]. Cherif and A. Kortebi, "On using Gradient Boosting (XGBoost) Machine Learning algorithm for Home Network Traffic Classification," IEEE, vol. II, pp. 1-6, 2019.
[29]. D. K. P. a. M. P. A. Gupta, "Development of mobile application for laundry services using android studio," International Journal of Applied Engineering Research, India, vol. 13, no. 12, pp. 10623–10626, 2018.
[30]. Abhi Android, "Android App Development Tutorial: Beginners Guide with Examples, Code and Tutorials," India. [Online]. Available: https://abhiandroid.com/java/. [Accessed 2023].
[31]. N. S. D. K. D. J. S. S. S. e. a. G. S. Bhat, "Machine learning-based asthma risk prediction using iot and smartphone applications," IEEE Access, vol. 9, pp. 118708– 118715, 2021.
[32]. N. Y. a. J. E. DeBello, "Recommended practices for python pedagogy in graduate data science courses," IEEE Frontiers in Education Conf. (FIE), Nebraska, Lincoln, USA, pp. 1-7, 2020.
[33]. C. G. G. G. E. M. S. e. a. Kokkotis, "An Explainable Machine Learning Pipeline for Stroke Prediction on Imbalanced Data," 2022, Diagnostics, 12(10), p.2392.