CEREBRAL STROKE CLASSIFICATION USING SUPERVISED LEARNING APPROACHES

A PROJECT REPORT

Submitted by

NAVEEN KUMAR N [211420104179]

KAVILASH R [211420104126]

NAVEEN S [211420104178]

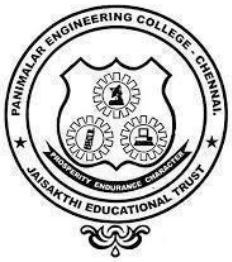
in partial fulfillment for the award of the degree

of

BACHELOR OF ENGINEERING

in

COMPUTER SCIENCE AND ENGINEERING



PANIMALAR ENGINEERING COLLEGE

(An Autonomous Institution, Affiliated to Anna University, Chennai)

MARCH 2024

PANIMALAR ENGINEERING COLLEGE

(An Autonomous Institution, Affiliated to Anna University, Chennai)

BONAFIDE CERTIFICATE

Certified that this project report "**CEREBRAL STROKE CLASSIFICATION USING SUPERVISED LEARNING APPROACHES**" is the bonafide work of "**NAVEEN KUMAR N [211420104179] , KAVILASH R [211420104126], NAVEENS[211420104178]**" who carried out the project work under my supervision.

Signature of the HOD with date

Dr. L. JABASHEELA M.E. Ph.D.

PROFESSOR AND HEAD
Department of Computer Science and Engineering,
Panimalar Engineering College,
Chennai - 123

Signature of the Supervisor with the date

Mr. M MAHENDRAN M. Tech
SUPERVISOR
ASSISTANT PROFESSOR
Department of Computer Science and Engineering,
Panimalar Engineering College,
Chennai - 123

Certified that the above candidate(s) was examined in the End Semester Project Viva-Voce Examination held on **25.03.2024 (FN)**

INTERNAL EXAMINER **EXTERNAL EXAMINER**

DECLARATION BY THE STUDENT

We **NAVEEN KUMAR N [211420104179], KAVILASH R [211420104126], NAVEEN S [211420104178]** hereby declare that this project report titled "**CEREBRAL STROKE CLASSIFICATION USING SUPERVISED LEARNING APPROACHES",** under the guidance of **Mr. M MAHENDRAN M.Tech.,** is the original work done by us and we have not plagiarized or submitted to any other degree in any university by us.

NAVEEN KUMAR N [211420104179]

KAVILASH R [211420104126]

NAVEEN S [211420104178]

ACKNOWLEDGEMENT

Our profound gratitude is directed towards our esteemed Secretary and Correspondent, **Dr. P. CHINNADURAI, M.A., Ph.D**., for his fervent encouragement. His inspirational support proved instrumental in galvanizing our efforts, ultimately contributing significantly to the successful completion of this project

We wish to express our deep gratitude to our Directors, **Tmt. C. VIJAYARAJESWARI, Dr. C. SAKTHIKUMAR, M.E., Ph.D., and Dr. SARANYASREE SAKTHIKUMAR, B.E., M.B.A., Ph.D.,** for graciously affordingus the essential resources and facilities forundertaking this project.

Our gratitude is also extended to our Principal, **Dr. K. MANI M.E., Ph.D.,** whose facilitation proved pivotal in the successful completion of this project.

We express our heartfelt thanks to **Dr. L. JABASHEELA M.E., Ph.D.,** Head of the Department of Computer Science and Engineering, for granting the necessary facilities thatcontributed to the timely and successful completion of the project.

We would like to gladly express our sincere thanks to **Project Coordinator Dr. G. SENTHIL KUMAR, M.C.A., M.Phil., M.B.A., M.E., Ph.D., and Project Guide Mr. M. MAHENDRAN, M.Tech., (Ph.D).,** all the faculty members of the Department of CSE for their unwavering support for the successful completion of the project.

NAVEEN KUMAR N [211420104179]

KAVILASH R [211420104126]

NAVEEN S [211420104178]

ABSTRACT

Cerebral stroke is a life-threatening medical condition that requires immediate diagnosis and intervention to minimize its devastating effects. Timely and accurate classification of stroke subtypes is crucial for determining appropriate treatment strategies and improving patient outcomes. In this research, we explore the application of supervised learning approaches to classify cerebral stroke subtypes based on relevant medical data. The study involves the collection of a diverse dataset comprising anonymized patient records, including clinical indicators, medical history, risk factors, and diagnostic imaging results. The data is carefully pre-processed to handle missing values, normalize numerical features, and address any potential biases. The experimental results demonstrate the effectiveness of supervised learning approaches in accurately classifying cerebral stroke subtypes. The model with the highest performance metrics is identified, and its potential integration into clinical settings is discussed. Additionally, the research explores potential challenges and limitations of the proposed approach, along with potential strategies to improve model accuracy and generalizability. Keywords: Cerebral Stroke, Supervised Learning, Machine Learning, Stroke Subtypes, Classification, Predictive Models, Healthcare, Neurology, Diagnostic Imaging.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF ABBREVIATIONS

AI	Artificial Intelligence
ML	Machine Learning
GUI	Graphical User Interface
SVM	Support Vector Machine
CVA	Cerebral Vascular Accident
TIA	Transient Ischemic Attack
NLP	Natural language Processing

CHAPTER 1
INTRODUCTION

1 INTRODUCTION

Stroke occurs when the blood flow to various areas of the brain is disrupted or diminished, resulting in the cells in those areas of the brain not receiving the nutrients and oxygen they require and dying. A stroke is a medical emergency that requires urgent medical attention. Early detection and appropriate management are required to prevent further damage to the affected area of the brain and other complications in other parts of the body. The World Health Organization (WHO) estimates that fifteen million people worldwide suffer from strokes each year, with one person dying every four to five minutes in the affected population.

Stroke is the sixth leading cause of mortality in the United States according to the Centers for Disease Control and Prevention (CDC) [1]. Stroke is a non-communicable disease that kills approximately 11% of the population. In the United States, approximately 795,000 people suffer from the disabling effects of strokes regularly [2]. It is India's fourth leading cause of death. Strokes are classified as ischemic or hemorrhagic. In a chemical stroke, clots obstruct the drainage; in a hemorrhagic stroke, a weak blood vessel bursts and bleeds into the brain. Stroke may be avoided by leading a healthy and balanced lifestyle that includes abstaining from un-healthy behaviors, such as smoking and drinking, keeping a healthy body mass index (BMI) and an average glucose level, and maintaining excellent heart and kidney function. Stroke prediction is essential and must be treated promptly to avoid irreversible damage or death. With the development of technology in the medical sector, it is now possible to anticipate the onset of a stroke by utilizing ML techniques. The algorithms included in ML are beneficial as they allow for accurate prediction and proper analysis. The majority of previous stroke-related research has focused on, among other things, the prediction of heart attacks. Brain stroke has been the subject of very few studies. The main motivation of this paper is to demonstrate how ML may be used to forecast the onset of a brain stroke.

findings achieved is that among the four distinct classification algorithms tested, Random Forest fared the best, achieving

a higher accuracy metric in comparison to the others. One downside of the model is that it is trained on textual data rather than real-time brain images. The implementation of four ML classification methods is shown in this paper.

1.1 DOMAIN OVERVIEW:

Data Science

Data science is an interdisciplinary field that uses scientific methods, processes, algorithms, and systems to extract knowledge and insights from structured and unstructured data and apply knowledge and actionable insights from data across a broad range of application domains.

The term "data science" has been traced back to 1974, when Peter Naur proposed it as an alternative name for computer science. In 1996, the International Federation of Classification Societies became the first conference to specifically feature data science as a topic. However, the definition was still in flux.

The term “data science” was first coined in 2008 by D.J. Patil, and Jeff Hammerbacher, the pioneer leads of data and analytics efforts at LinkedIn and Facebook. In less than a decade, it has become one of the hottest and most trending professions in the market.

Data science is the field of study that combines domain expertise, programming skills, and knowledge of mathematics and statistics to extract meaningful insights from data.

Data science can be defined as a blend of mathematics, business acumen, tools, algorithms, and machine learning, all of which help us in finding out the hidden insights or patterns from raw data that can be of major use in the formation of big business decision

Data Scientist:

Data scientists examine which questions need answering and where to find the related data. They have business acumen and analytical skills as well as the ability to mine, clean, and present data.

Businesses use data scientists to source, manage, and analyze large amounts of unstructured data.

Required Skills for a Data Scientist:

- **Programming**: Python, SQL, Scala, Java, R, MATLAB.
- **Machine Learning**: Natural Language Processing, Classification, Clustering.
- **Data Visualization**: Tableau, SAS, D3.js, Python, Java, R libraries.
- **Big data platforms**: MongoDB, Oracle, Microsoft Azure, Cloudera

1.2 MACHINE LEARNING

Machine learning is to predict the future from past data. Machine learning (ML) is a type of artificial intelligence (AI) that provides computers with the ability to learn without being explicitly programmed. Machine learning focuses on the development of Computer Programs that can change when exposed to new data and the basics of Machine Learning, implementation of a simple machine learning algorithm using Python. The process of training and prediction involves the use of specialized algorithms. It feeds the training data to an algorithm, and the algorithm uses this training data to give predictions on new test data. Machine learning can be roughly separated into three categories. There are supervised learning, unsupervised learning, and reinforcement learning. A supervised learning program is both given the input data and the corresponding labeling to learn data has to be labeled by a human being beforehand. Unsupervised learning is no labels.

Data scientists use many different kinds of machine learning algorithms to discover patterns in Python that lead to actionable insights. At a high level, these different algorithms can be classified into two groups based on the way they "learn" about data to make predictions: supervised and unsupervised learning.

Classification predictive modeling is the task of approximating a mapping function from input variables(X) to discrete output variables(y). In machine learning and statistics, classification is a supervised learning approach in which the computer program learns from the data input given to it and then uses this learning to classify new observations.

Process Of Machine Learning

Supervised Machine Learning is the majority of practical machine learning uses supervised learning. Supervised learning is where have input variables (X) and an output variable (y) and use an algorithm to learn the mapping function from the input to the output is $y = f(X)$. The goal is to approximate the mapping function so well that when you have new input data (X) that you can predict the output variables (y) for that data. This problem has as goal the construction of a succinct model that can predict the value of the dependent attribute from the attribute variables.

The difference between the two tasks is the fact that the dependent attribute is numerical for categorical for classification. A classification model attempts to draw some conclusion from observed values. Given one or more inputs a classification model will try to predict the value of one or more outcomes. A classification problem is when the output variable is a category, such as “red” or “blue”.

1.3 ARTIFICIAL INTELLIGENCE

Artificial intelligence (AI) refers to the simulation of human intelligence in machines that are programmed to think like humans and mimic their actions. The term mayalso be applied to any machine that exhibits traits associated with a human mind such as learning and problem-solving.

Artificial intelligence (AI) is intelligence demonstrated by machines, as opposed to the natural intelligence displayed by humans or animals. Leading AI textbooks define the field as the study of "intelligent agents" any system that perceives its environment and takes actions that maximize its chance of achieving its goals. Some popular accounts use the term "artificial intelligence" to describe machines that mimic "cognitive" functions that humans associate with the human mind, such as "learning" and "problem solving", however, this definition is rejected by major AI researchers.

The various sub-fields of AI research are centered around particular goals and the use of particular tools. The traditional goals of AI research include reasoning, knowledge representation, planning, learning, natural language processing, perception and the ability to move and manipulate objects. General intelligence (the ability to solve an arbitrary problem) is among the field's long-term goals.

The field was founded on the assumption that human intelligence "can be so precisely described that a machine can be made to simulate it". This raises philosophical arguments about the mind and the ethics of creating artificial beings endowed with human- like intelligence.

AI requires a foundation of specialized hardware and software for writing and training machine learning algorithms. No one programming language is synonymous with AI, but a few, including Python, R, and Java, are popular.

In general, AI systems work by ingesting large amounts of labeled training data, analyzing the data for correlations and patterns, and using these patterns to make predictions about future states. In this way, a chatbot that is fed examples of text chats can learn to produce life-like exchanges with people, or an image recognition tool can learn to identify and describe objects in images by reviewing millions of examples.

AI programming focuses on three cognitive skills: learning, reasoning and self-correction.

Learning processes. This aspect of AI programming focuses on acquiring data and creating rules for how to turn the data into actionable information. The rules, which are called algorithms, provide computing devices with step-by-step instructions for how to complete a specific task.

Reasoning processes. This aspect of AI programming focuses on choosing the right algorithm to reach a desired outcome.

Self-correction processes. This aspect of AI programming is designed to continually fine-tune algorithms and ensure they provide the most accurate results possible.

AI is important because it can give enterprises insights into their operations that they may not have been aware of previously and because, in some cases, AI can perform tasks better than humans. Particularly when it comes to repetitive, detail-oriented tasks like analyzing large numbers of legal documents to ensure relevant fields are filled in properly, AI tools often complete jobs quickly and with relatively few errors.

Artificial neural networks and deep learning artificial intelligence technologies are quickly evolving, primarily because AI processes large amounts of data much faster and makes predictions more accurately than humanly possible.

Natural Language Processing (NLP):

Natural language processing (NLP) allows machines to read and understand human language. A sufficiently powerful natural language processing system would enable natural-language user interfaces and the acquisition of knowledge directly from human- written sources, such as newswire texts. Many current approaches use word co-occurrence frequencies to construct syntactic representations of text. "Keyword spotting" strategies for search are popular and scalable but dumb; a search query for "dog" might only match documents with the literal word "dog" and miss a document with the word "poodle". "Lexical affinity" strategies use the occurrence of words such as "accident "to assessthe sentiment of a document.

CHAPTER 2
LITERATURE SURVEY

2 LITERATURE SURVEY

In the research conducted by Manisha Sirsat, Eduardo Ferme, Joana Camara, the main aim of the research was to classify state-of-arts on ML techniques for brain stroke into 4 categories based on their functionalities or similarity, and then review studies of each category systematically. The study further discusses the outcomes and accuracies obtained by using different Machine Learning models using text and image-based datasets. In this study, the authors discussed many stroke related problems from the state-of-art. The reviewed studies were grouped in several categories based on their similarities. The study notes that it is difficult to compare studies as they employed different performance metrics for different tasks, considering different datasets, techniques, and tuning parameters. Hence, it only mentions the research areas which were targeted in more than one study and the studies which report highest classification accuracy in each section [1]. Harish Kamal, Victor Lopez, Sunil A. Sheth, in their study discuss how Machine Learning (ML) through pattern recognition algorithms is currently becoming an essential aid for the diagnosis, treatment, and prediction of complications and patient outcomes in several neurological diseases. The evaluation and treatment of Acute Ischemic Stroke (AIS) have experienced a significant advancement over the past few years, increasingly requiring the use of neuroimaging for decision making. This study offers an insight into the recent developments and applications of ML in neuroimaging focusing on acute ischemic stroke. The implementations of machine learning are numerous, from early identification of imaging diagnostic findings, estimating time of onset, lesion segmentation, and fate of salvageable tissue, to the analysis of cerebral edema, and predicting complications and patient outcomes after treatment.

2.1 REVIEW OF LITERATURE SURVEY:

1. **Title: Brain Stroke Prediction Using Machine Learning Approach**
Author: DR. AMOL K. KADAM, PRIYANKA AGARWAL
Year: 2022

A Stroke is an ailment that causes harm by tearing the veins in the mind. Stroke may likewise happen when there is a stop in the blood stream and different supplements to the mind. As per the WHO, the World Health Organization, stroke is one of the world's driving reasons for death and incapacity. The majority of the work has been completed on heart stroke forecast however not many works show the gamble of a cerebrum stroke. Subsequently, the AI models are worked to foresee the chance of cerebrum stroke. The project is pointed towards distinguishing the familiarity with being in danger of stroke and its determinant factors amongst victims. The research has taken numerous factors and utilized ML calculations like Logistic Regression, Decision Tree Classification, Random Forest Classification, KNN, and SVM for accurate prediction.

2. **Title**: **Brain Stroke Prediction using Machine Learning**
Author: Mrs. Neha Saxena, Mr. Deep Singh Bhamra, Mr. Arvind Choudhary
Year: 2014

A stroke, also known as a cerebrovascular accident or CVA is when part of the brain loses its blood supply and the part of the body that the blood-deprived brain cells control stops working. This loss of blood supply can be ischemic because of lack of blood flow, or hemorrhagic because of bleeding into brain tissue. There are opportunities to treat ischemic strokes but that treatment needs to be started in the first few hours after the signs of a stroke begin.

The patient, family, or bystanders should activate emergency medical services immediately should a stroke be suspected. A transient ischemic attack (TIA or mini-stroke) describes an ischemic stroke that is short-lived where the symptoms resolve spontaneously. This situation also requires emergency assessment to try to minimize the risk of a future stroke. By definition, a stroke would be classified as a TIA if all symptoms resolved within 24 hours. According to the World Health Organization (WHO), stroke is the 2nd leading cause of death globally, responsible for approximately 11% of total deaths. For survival prediction, our ML model uses a dataset to predict whether a patient is likely to get stroke based on the input parameters like gender, age, various diseases, and smoking status. Unlike most of the datasets, our dataset focuses on attributes that would have a major risk factor of a Brain Stroke.

3. **Title**: Machine Learning for Brain Stroke: A Review
Author: Manisha Sanjay Sirsat, Eduardo Ferme , and Joana Camara,
Year : 2020

Machine Learning (ML) delivers an accurate and quick prediction outcome and it has become a powerful tool in health settings, offering personalized clinical care for stroke patients. An application of ML and Deep Learning in health care is growing however, some research areas do not catch enough attention for scientific investigation though there is real need of research. Therefore, the aim of this work is to classify state-of-arts on ML techniques for brain stroke into 4 categories based on their functionalities or similarity, and then review studies of each category systematically. A total of 39 studies were identified from the results of ScienceDirect web scientific database on ML for brain stroke from the year 2007 to 2019. Support Vector Machine (SVM) is obtained as optimal models in 10 studies for stroke problems.

4. **Title**: Predicting Stroke Risk with an Interpretable Classifier

Author: SERGIO PEÑAFIEL, NELSON BALOIAN, HORACIO SANSON, AND JOSÉ A. PINO

Year : 2021

Predicting an individual's risk of getting a stroke has been a research subject for many authors worldwide since it is a frequent illness and there is strong evidence that early awareness of having that risk can be beneficial for prevention and treatment. Many Governments have been collecting medical data about their own population with the purpose of using artificial intelligence methods for making those predictions. The most accurate ones are based on so called black-box methods which give little or no information about why they make a certain prediction. However, in the medical field the explanations are sometimes more important than the accuracy since they allow specialists to gain insight about the factors that influence the risk level. It is also frequent to find medical information records with some missing data. In this work, we present the development of a prediction method which not only outperforms some other existing ones but it also gives information about the most probable causes of a high stroke risk and can deal with incomplete data records. It is based on the Dempster-Shafer theory of plausibility. For the testing we used data provided by the regional hospital in Okayama, Japan, a country in which people are compelled to undergo annual health checkups by law. This article presents experiments comparing the results of the Dempster-Shafer method with the ones obtained using other well-known machine learning methods like Multilayer perceptron, Support Vector Machines and Naive Bayes. Our approach performed the best in these experiments with some missing data

5. **Title**: Towards stroke prediction using electronic health records

Author: Douglas Teoh

Year: 2014

Background: As of 2014, stroke is the fourth leading cause of death in Japan. Predicting a future diagnosis of stroke would better enable proactive forms of healthcare measures to be taken. We aim to predict a diagnosis of stroke within one year of the patient's last set of exam results or medical diagnoses. Methods: Around 8000 electronic health records were provided by Tsuyama Jifukai Tsuyama Chuo Hospital in Japan. These records contained non-homogeneous temporal data which were first transformed into a form usable by an algorithm.

The transformed data were used as input into several neural network architectures designed to evaluate efficacy of the supplied data and also the networks' capability at exploiting relationships that could underlie the data. The prevalence of stroke cases resulted in imbalanced class outputs which resulted in trained neural network models beingbiased towards negative predictions. To address this issue, we designed and incorporated regularization terms into the standard cross-entropy loss function. These terms penalized false positive and false negative predictions. We evaluated the performance of our trained models using Receiver Operating Characteristic. Results: The best neural network incorporated and combined the different sources of temporal data through a dual-input topology. This network attained area under the Receiver Operating Characteristic curve of 0.669. The custom regularization terms had a positive effect on the training process when compared against the standard cross-entropy loss function

2.2 SYSTEM STUDY:

AIM:

Stroke prediction is essential and must be treated promptly to avoid irreversible damage or death. With the development of technology in the medical sector, it is now possible to anticipate the onset of a stroke by utilizing ML techniques So this project can easily find out the Attack.

Objectives:

The goal of this study is to develop a novel decision-making tool for predicting strokes. This study compares the accuracy, precision, and execution time of convolution neural networks, dense net, and VGG 16 for stroke prediction. The performance of the CNN classifier is superior than Descent and VGG 16, according to the findings of the experiments.

The repository is a learning exercise to:

- Apply the fundamental concepts of machine learning from an available dataset and evaluate and interpret my results and justify my interpretation based on observed dataset.
- Create notebooks that serve as computational records and document my thought process and investigate the network connection whether attacked or not to analyses the data set.
- Evaluate and analyses statistical and visualized results, which find the standard patterns for all regiments.

Project Goals

- Loading the given dataset
- Import required libraries packages
- Analyze the general properties
- Find duplicate and missing values
- Checking unique and count values
- Rename, add data and drop the data
- To specify data type
- Pre-processing the given dataset
- Splitting the test and training dataset
- Comparing the Random Forest and Logistic regression model and Decision Tree
- Based on the best accuracy

Scope:

The scope of this project is to investigate Brain stroke prediction classification technique using machine learning technique. To identifying Brain stroke, occur or NOT

2.3 FEASIBILITY STUDY:

Data Wrangling

In this section of the report will load in the data, check for cleanliness, and then trim and clean given dataset for analysis. Make sure that the document steps carefully and justify for cleaning decisions.

Data collection

The data set collected for predicting given data is split into Training set and Test set. Generally, 7:3 ratios are applied to split the Training set and Test set. The Data Model which was created using Random Forest, logistic, Decision tree algorithms and Support vector classifier (SVC) are applied on the Training set and based on the test result accuracy, Test set prediction is done.

Preprocessing

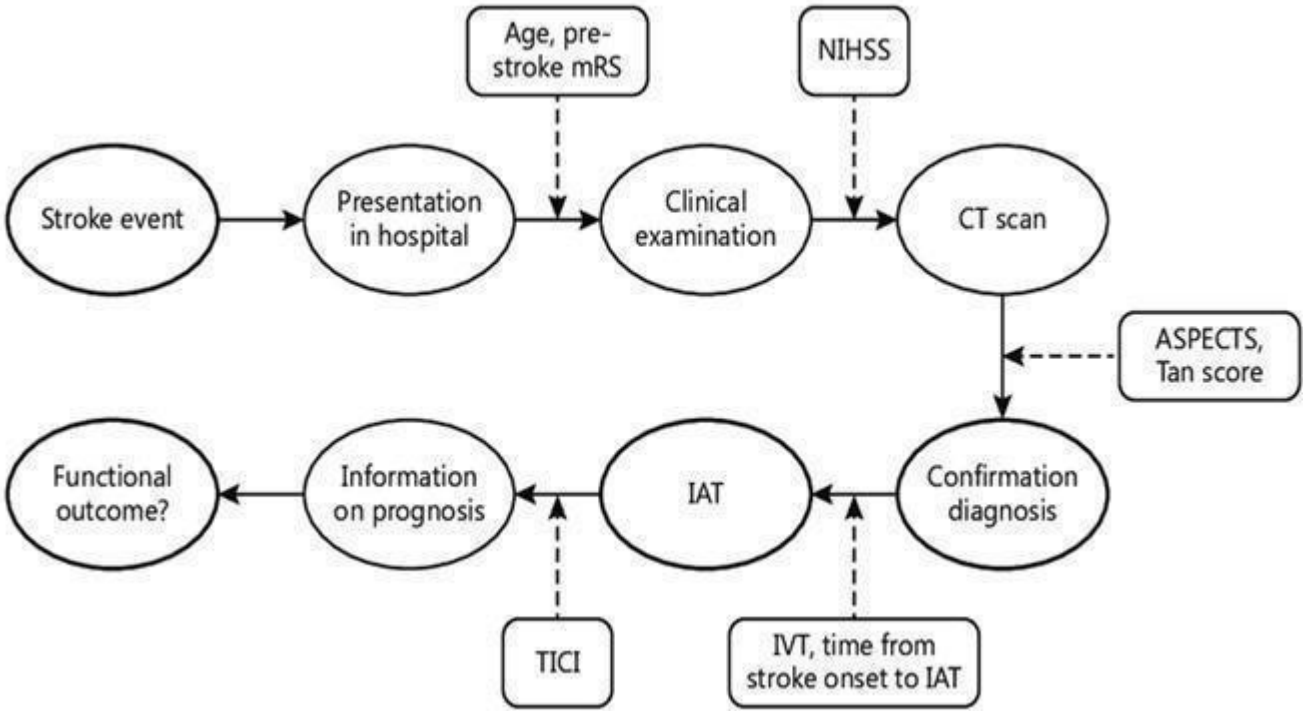
The data which was collected might contain missing values that may lead to inconsistency. To gain better results data need to be preprocessed so as to improve the efficiency of the algorithm. The outliers have to be removed and also variable conversion need to be done.

Building the classification model

The prediction of win attack, A Random Forest Algorithm prediction model is effective because of the following reasons: It provides better results in classification problem.

- It is strong in preprocessing outliers, irrelevant variables, and a mix of continuous, categorical and discrete variables.
- It produces out of bag estimate error which has proven to be unbiased in many tests and it is relatively easy to tune with.

- sufficient historical data and raw data. Before data pre-processing, raw data can't be used directly. It's used to preprocess then, what kind of algorithm with model.Training and testing this model working and predicting correctly with minimum errors. Tuned model involved by tuned time to time with improving the accuracy.



Fg:2.3.1 Process of dataflow diagram

CHAPTER 3
THEORETICAL BACKGROUND

3.1 Implementation Environment:

Anaconda Package versions are been managed by the package managements system "Conda". The Anaconda distribution is used by over 12 million users and includes more than 1400 popular data science packages suitable for Windows, Linux, and MacOS. So, Anaconda distribution comes with more than 1,400 packages as well as the Conda package and virtual environment manager called Anaconda Navigator and iteliminates the need to learn to install each library independently. The open source packages can be individually installed from the Anaconda repository with the condo install commandor using the pip install command that is installed with Anaconda. Pip packages provide many of the features of condo packages and in most cases they can work together. Custompackages can be made using the command, and can be shared with others by uploading them to Anaconda Cloud, Py-PI or other repositories. The default installation of Anaconda2includes Python
2.7 and Anaconda3 includes Python 3.7.However, you can create new environments that include any version of Python packaged with conda.

ANACONDA NAVIGATOR

Anaconda Navigator is a desktop graphical user interface (GUI) included in Anaconda® distribution that allows you to launch applications and easily manage condo packages, environments, and channels without using command-line commands. Navigator can search for packages on Anaconda.org or in a local Anaconda Repository.

Anaconda is a distribution of the Python and R programming languages for scientific computing (data science, machine learning applications, large-scale data processing, predictive analytics, etc.), that aims to simplify package management and deployment.

In order to run, many scientific packages depend on specific versions of other packages. Data scientists often use multiple versions of many packages and use multiple environments to separate these different versions.

The command-line program conda is both a package manager and an environment manager. This helps data scientists ensure that each version of each package has all the dependencies it requires and works correctly

The best way to get started using Python for machine learning is to complete a project.

- It will force you to install and start the Python interpreter (at the very least).
- It will give you a bird's eye view of how to step through a small project.
- It will give you confidence, maybe to go on to your own small projects.

3.2 Existing System:

Neural-like P systems are membrane computing models inspired by natural computing and are viewed as third-generation neural network models. Although real neurons have complex structures, classical neural-like P systems simplify the structures and corresponding mechanisms to two-dimensional graphs or tree-based firing and forgetting communications, which limit the real applications of these models. In this paper, we propose a hypergraph-based numerical neural-like (HNN) P system containing five types of neurons to describe the high-order correlations among neuron structures.

Three new kinds of communication mechanisms among neurons are also proposed to address numerical variables and functions. Based on the new neural-like P system, a tumor/organ segmentation model for medical images is developed. The experimental results indicate that the proposed models outperform the state-of-the-art methods based ontwo hippocampal datasets and a multiple brain metastases dataset, thus verifying the effectiveness of the HNN P system in correctly segmenting tumors/organs.

Advantages:

High Accuracy: Random forests are known for their high accuracy in prediction tasks. They can handle large datasets with high dimensionality and nonlinear relationships effectively, making them suitable for analyzing complex medical data.

Feature Importance: Random forests provide a measure of variable importance, indicating which features (risk factors, biomarkers, etc.) are most relevant for predicting stroke outcomes. This can help in understanding the underlying mechanisms and identifying potential targets for intervention.

Robustness to Overfitting: Random forests are less prone to overfitting compared to other models like decision trees, especially when the number of trees (ensemble size) is adequately chosen. This makes them more reliable for generalizing to unseen data.

Disadvantages:

Black Box Model: Random forests are often considered black box models, meaning they lack interpretability compared to simpler models like logistic regression. While they provide insights into feature importance, understanding the exact decision process of each tree in the forest can be challenging**.**

Computational Complexity: Training a random forest model can be computationally expensive, especially with large datasets and a high number of trees in the ensemble. This may limit its applicability in real-time or resource-constrained environments.

Memory Consumption: Random forests can consume a significant amount of memory, particularly for large datasets or ensembles with numerous trees. This could be a concern when working with limited computational resources

3.2 Proposed System:

The proposed system for cerebral stroke classification using supervised learning approaches aims to enhance the accuracy and efficiency of stroke diagnosis through advanced machine learning techniques. Leveraging the power of super-vised learning algorithms, such as Support Vector Machines (SVM), Random Forest, or Neural Networks, the system will be trained on a diverse dataset of medical images and patient data. This comprehensive approach allows the model to learn patterns and relationships within the data, enabling it to accurately classify different types of cerebral strokes based on imaging features, clinical indica-tors, and other relevant parameters. By harnessing the capabilities of supervised learning, the system can provide timely and reliable predictions, aiding healthcare professionals in making quicker and more informed decisions for effective stroke management.

Moreover, the proposed system will contribute to the ongoing efforts in leveraging artificial intelligence for medical diagnosis and treatment planning. The incorporation of supervised learning not only enhances the accuracy of stroke classification but also enables continuous learning and adaptation as new data becomes available.

This adaptive nature ensures that the system stays up-to-date with the latest medical knowledge and can effectively handle variations in stroke presentations.

Ultimately, the implementation of this system has the potential to significantly improve the speed and precision of stroke diagnosis, leading to better patient outcomes and more efficient allocation of healthcare resources

The proposed system for cerebral stroke encompasses a holistic approach, integratingprevention, early detection, diagnosis, treatment, rehabilitation, data analysis, and public policy advocacy.

It includes education on risk factors, screening programs for early detection, advanced imaging for accurate diagnosis, rapid access to thrombolytic therapy and endovascular procedures, comprehensive rehabilitation services, data collection and analysis for continuous improvement, and advocacy for supportive public policies. By implementing this system, the aim is to reduce the incidence and impact of cerebral stroke through proactive measures, timely intervention, and comprehensive care

Data collection and analysis drive continuous improvement, while advocacy efforts aim to influence supportive public policies. This integrated system aims to mitigate the incidence andimpact of cerebral stroke through proactive measures, timely interventions, and holistic care approaches.

Advantages:

Accuracy: Random forests can offer high accuracy in segmentation tasks, particularly when dealing with complex and varied structures in medical images.

Robustness to Noise: Random forests can handle noisy data effectively, making them suitable for segmenting medical images that may contain artifacts or imperfections.

Feature Importance: Random forests can provide insights into the importance of different features (such as intensity gradients, texture, etc.) for segmentation, aiding in feature selection and understanding of the underlying characteristics of the segmented regions.

CHAPTER 4
SYSTEM DESIGN

4.1 System Architecture

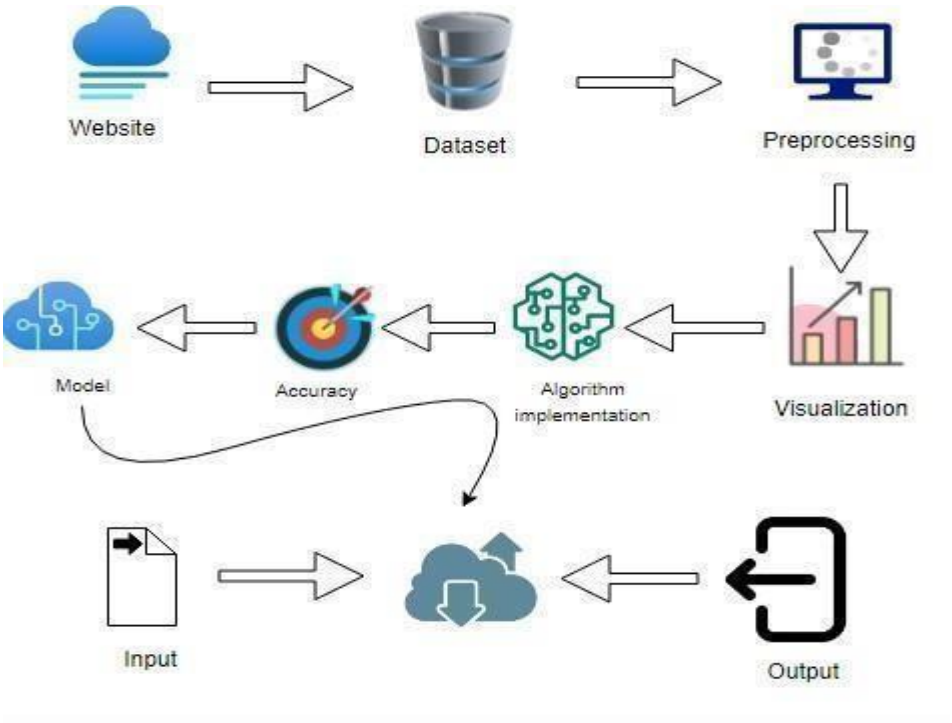


Fig 4.1.2 :work flow diagram

Random forest algorithm is a powerful machine learning technique used for classification andregression tasks. In the context of predicting cerebral stroke, a random forest model can be trained using a dataset containing various features such as demographic information, medicalhistory, lifestyle factors, and possibly results from diagnostic tests.

The process involves:

Data Collection: Gathering a dataset containing relevant information about individuals, including those who have experienced cerebral stroke and those who have not.

Data Preprocessing: Cleaning the data, handling missing values, and encoding categorical variables if necessary. It's crucial to ensure the data is suitable for training the model.

Feature Selection: Identifying the most important features that contribute to predicting cerebralstroke. This step helps in improving model performance and reducing overfitting.

Model Training: Utilizing the random forest algorithm to build a predictive model based on thelabeled dataset. During training, the algorithm constructs multiple decision trees, each trained ona random subset of the features and data samples.

Model Evaluation: Assessing the performance of the trained model using evaluation metricssuch as accuracy, precision, recall, and F1-score. This step helps in determining how well themodel generalizes to unseen data.

Model Deployment: Integrating the trained model into a system where it can make predictionson new data. This could be in the form of a web application, mobile app, or an integrated healthcare system.

Continuous Monitoring and Updating: Monitoring the model's performance over time and updating it as new data becomes available or as the underlying patterns change. This ensures themodel remains effective and accurate in predicting cerebral stroke.

4.2 Work flow diagram:

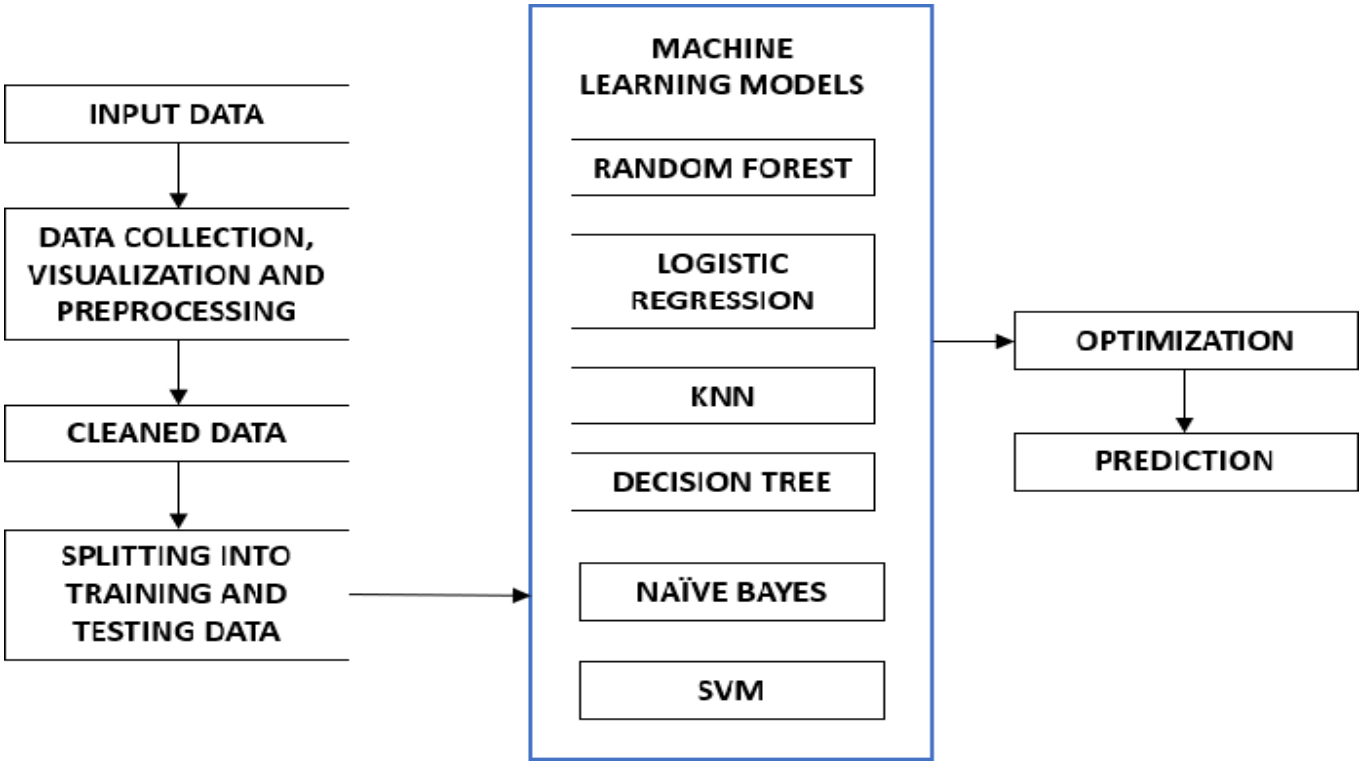


Fig no 4.2.3 WORK FLOW DIAGRAM

A workflow diagram for cerebral stroke would typically begin with the identification of risk factors such as hypertension, diabetes, smoking, and genetic predisposition. This leads to the onset of cerebral stroke, characterized by symptoms such as sudden numbness or weakness in the face, arm, or leg, especially on one side of the body; confusion, trouble speaking, or difficulty understanding speech; and sudden trouble seeing in one or both eyes, among others. The diagnostic process involves a series of steps, including physical examination, neurological evaluation, imaging tests such as CT or MRI scans, and laboratory tests to confirm the diagnosis. Treatment options vary depending on the type and severity of the stroke but may include medication, surgery, or rehabilitation.

4.3 Use Case Diagram

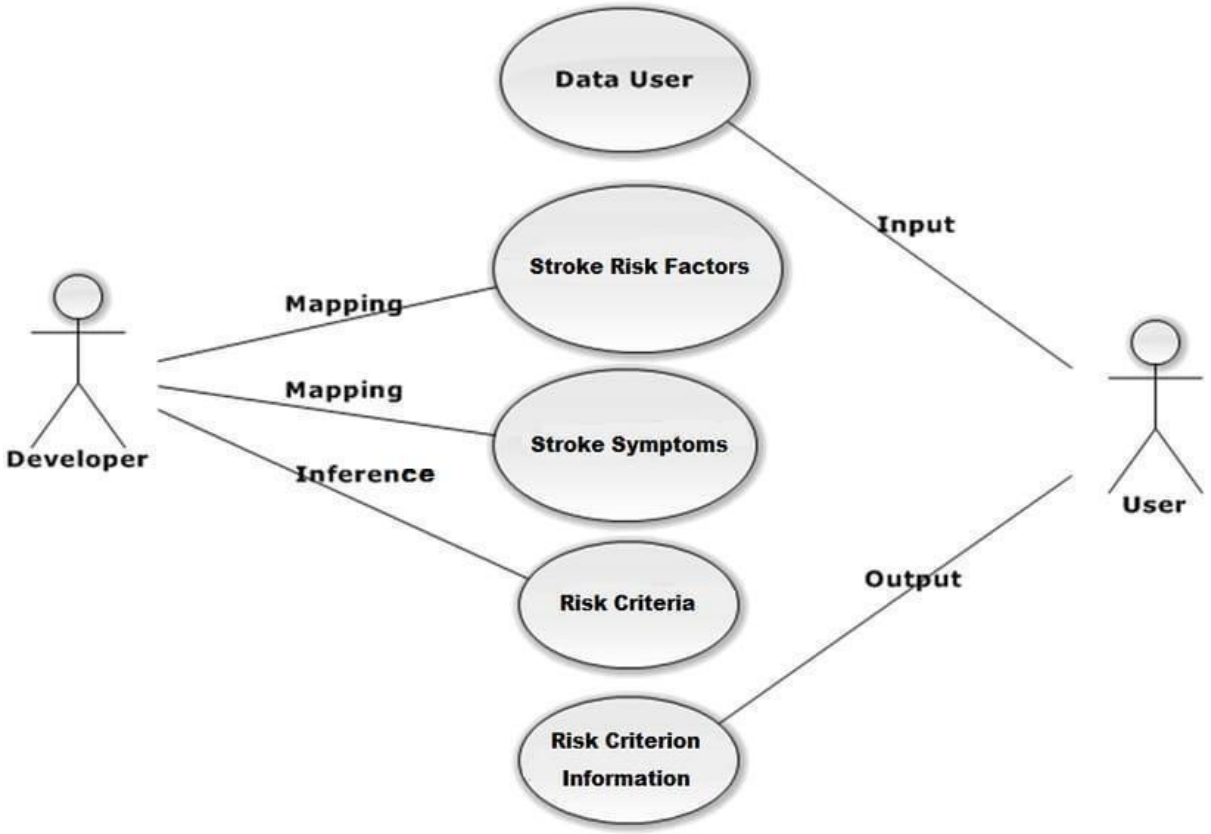
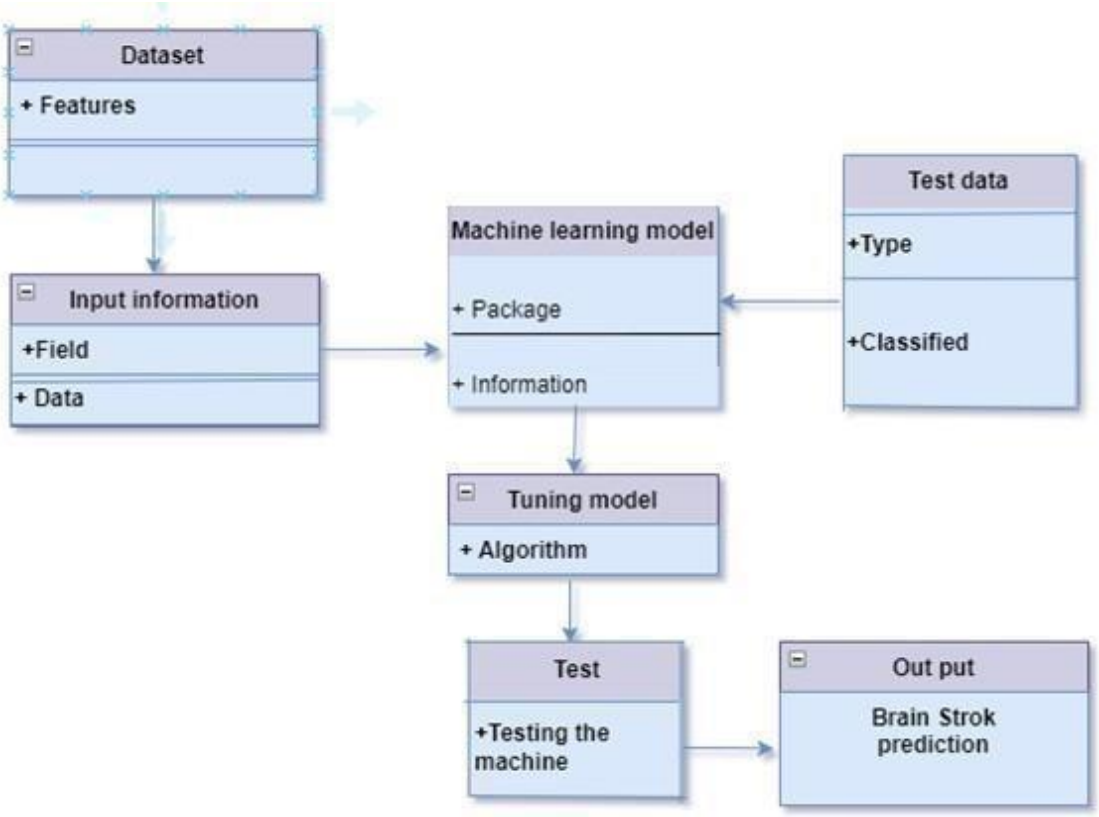


Fig 4.3.4 Use case diagram

Use case diagrams are considered for high level requirement analysis of a system. So when the requirements of a system are analyzed the functionalities are captured in use cases. So, it cansay that uses cases are nothing but the system functionalities written in an organized manner.

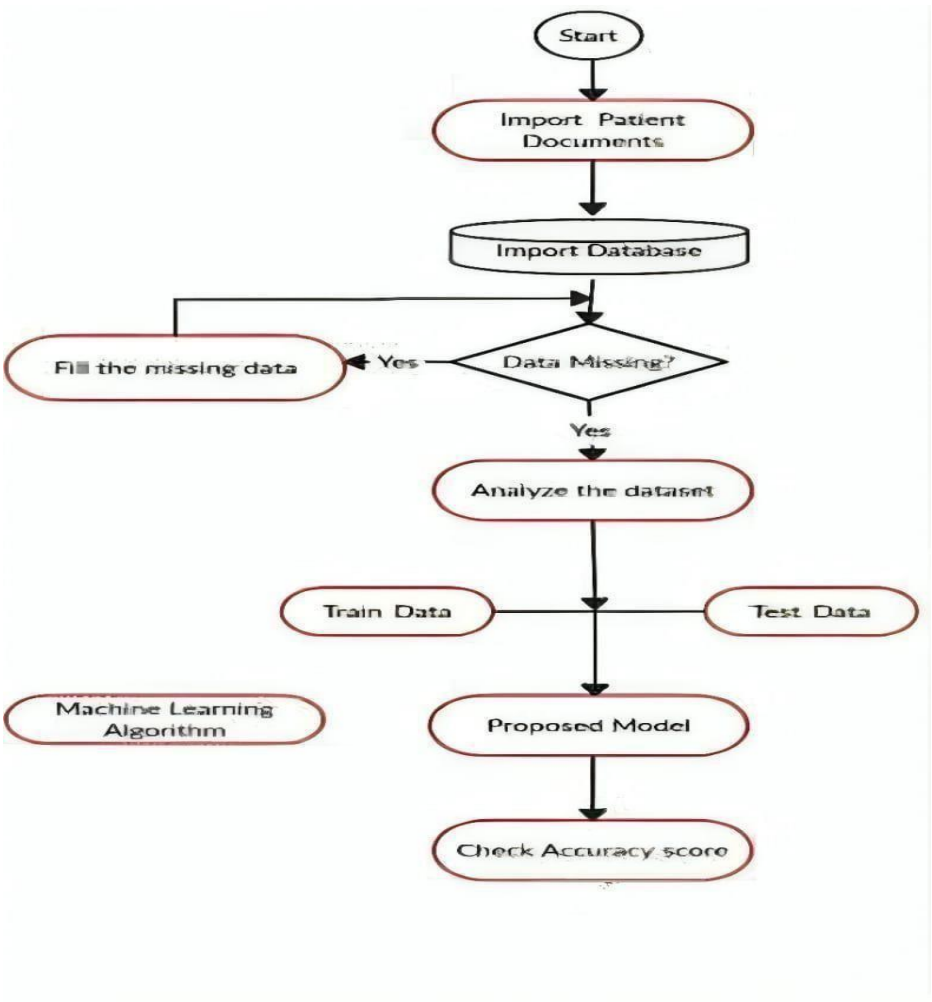
4.4 Class Diagram:



Fg :4.4.5 Class diagram

Class diagram is basically a graphical representation of the static view of the system and represents different aspects of the application. So, a collection of class diagrams represent the whole system. The name of the class diagram should be meaningful to describe the aspect of the system. Each element and their relationships should be identified in advance Responsibility (attributes and methods) of each class should be clearly identified for each class minimum number of properties should be specified and because, unnecessary properties will make the diagram complicated. Use notes whenever required to describe some aspect of the diagram and at the end of the drawing it should be understandable to the developer/coder.

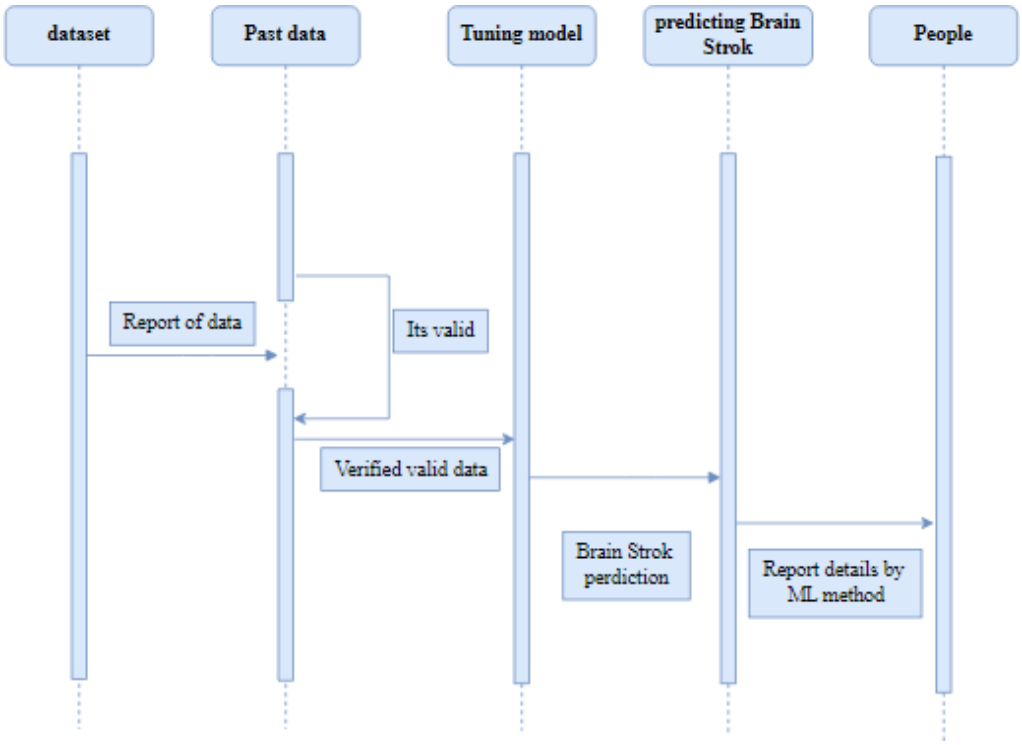
4.5 Activity Diagram:



Fg 4.5.6 Activity diagram

Activity is a particular operation of the system. Activity diagrams are not only used for visualizing dynamic nature of a system but they are also used to construct the executable system by using forward and reverse engineering techniques. The only missing thing in activity diagram is the message part. It does not show any message flow from one activity to another. Activity diagram is some time considered as the flow chart. Although the diagrams looks like a flow chart but it is not. It shows different flow like parallel, branched, concurrent and single.

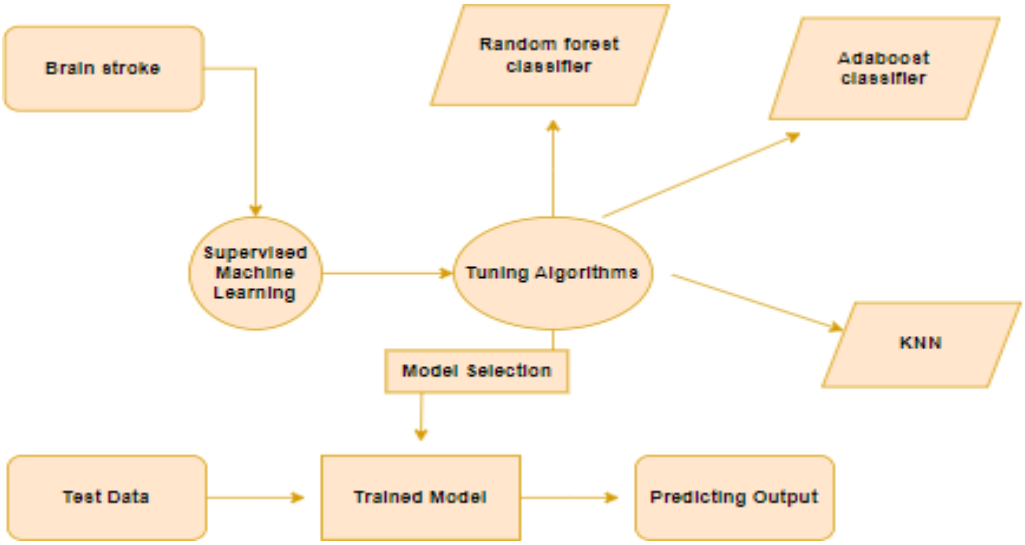
4.6 Sequence Diagram:



Fg 4.6.7 Sequence diagram

Sequence diagrams model the flow of logic within your system in a visual manner, enabling you both to document and validate your logic, and are commonly used for both analysis and design purposes. Sequence diagrams are the most popular UML artifact for dynamic modeling, which focuses on identifying the behavior within your system. Other dynamic modeling techniques include activity diagramming, communication diagramming, timing diagramming, and interaction overview diagramming. Sequence diagrams, along with class diagrams and physical data models are in my opinion the most important design-level models for modern business application development.

4.7 Entity Relationship Diagram (ERD)



Fg 4.7.8 ENTITY RELATIONSHIP DIAGRAM

An entity relationship diagram (ERD), also known as an entity relationship model, is a graphical representation of an information system that depicts the relationships among people, objects, places, concepts or events within that system.

An ERD is a data modelingtechnique that can help define business processes and be used as the foundation fora entity relationship diagrams provide a visual starting point for database design that can also be used to help determine information system requirements throughout an organization.

After a relational database is rolled out, an ERD can still serveas a referral point, should any debugging or business process re-engineering be needed later.

CHAPTER 5
SYSTEM IMPLEMENTATION

Module description:

Data Pre-processing

Validation techniques in machine learning are used to get the error rate of the Machine Learning (ML) model, which can be considered as close to the true error rate of the dataset. If the data volume is large enough to be representative of the population, you may not need the validation techniques. However, in real-world scenarios, to work with samples of data that may not be a true representative of the population of given dataset. To finding the missing value, duplicate value and description of data type whether it is float variable or integer. The sample of data used to provide an unbiased evaluation of a modelfit on the training dataset while tuning model hyper parameters.

The evaluation becomes more biased as skill on the validation dataset is incorporatedinto the model configuration. The validation set is used to evaluate a given model, but thisis for frequent evaluation. It as machine learning engineers use this data to fine-tune the model hyper parameters. Data collection, data analysis, and the process of addressing datacontent, quality, and structure can add up to a time-consuming to-do list. During the process of data identification, it helps to understand your data and its properties; thisknowledge will help you choose which algorithm to use to build your model.

A number of different data cleaning tasks using Python's Pandas library and specifically, it focus on probably the biggest data cleaning task, missing values and it ableto more quickly clean data. It wants to spend less time cleaning data, and more time exploring and modeling

DATA PREPROCESSING AND DATA CLEANING

```
import pandas as pd
import numpy as np

import warnings
warnings.filterwarnings('ignore')
```

```
df = pd.read_csv('CEREBRAL.csv')
del df['id']
df.head()
```

	gender	age	hypertension	heart_disease	ever_married	work_type	Residence_type	avg_glucose_level	bmi	smoking_status	stroke
0	Male	3.0	0	0	No	children	Rural	95.12	18.0	NaN	0
1	Male	58.0	1	0	Yes	Private	Urban	87.96	39.2	never smoked	0
2	Female	8.0	0	0	No	Private	Urban	110.89	17.6	NaN	0
3	Female	70.0	0	0	Yes	Private	Rural	69.04	35.9	formerly smoked	0
4	Male	14.0	0	0	No	Never_worked	Rural	161.28	19.1	NaN	0

GIVEN INPUT EXPECTED OUTPUT

Input: data

output: removing noisy data

Exploration data analysis of visualization

Data visualization is an important skill in applied statistics and machine learning. Statistics does indeed focus on quantitative descriptions and estimations of data. Data visualization provides an important suite of tools for gaining a qualitative understanding.

This can be helpful when exploring and getting to know a dataset and can help with identifying patterns, corrupt data, outliers, and much more. With a little domain knowledge, data visualizations can be used to express and demonstrate key relationships in plots and charts that are more visceral and stakeholders than measures of association or significance. Data visualization and exploratory data analysis are whole fields themselves and it will recommend a deeper dive into some the books mentioned at the end.

Sometimes data does not make sense until it can look at in a visual form, such as with charts and plots. Being able to quickly visualize of data samples and others is an important skill both in applied statistics and in applied machine learning. It will discover the many types of plots that you will need to know when visualizing data in Python and how to use them to better understand your own data.

ALGORITHM AND TECHNIQUES

Algorithm

Some examples of classification problems are: speech recognition, handwriting recognition, bio metric identification, document classification etc. In Supervised Learning, algorithms learn from labeled data. After understanding the data, the algorithm determines

which label should be given to new data based on pattern and associating the patterns to the unlabeled new data.

Used Python Packages:

sklearn:

- In python, sklearn is a machine learning package which include a lot of ML algorithms.
- Here, we are using some of its modules like train_test_split, DecisionTreeClassifier or Logistic Regression and accuracy_score.

NumPy:

- It is a numeric python module which provides fast maths functions for calculations.
- It is used to read data in numpy arrays and for manipulation purpose.

Pandas:

- Used to read and write different files.

- Data manipulation can be done easily with data frames. Matplotlib:
- Data visualization is a useful way to help with identify the patterns from given dataset.
- Data manipulation can be done easily with data frames.

5.1 MODULE – 1:

Random forest Algorithm:

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

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

1. Decision Trees:

- Random Forest is built upon the foundation of decision trees. Decision trees are simple models that recursively split the data based on features to make predictions.

2. Ensemble Learning:

- Random Forest is an ensemble learning method that combines the predictions of multiple decision trees to improve overall performance and robustness.

3. Bagging (Bootstrap Aggregating):

- Random Forest uses bagging to build diverse trees. It randomly selects subsets of the training data (with replacement) to train each tree. This helps in reducing overfitting and capturing different patterns in the data.

4. Random Feature Selection:

- In addition to using random subsets of data, Random Forest also randomly selects a

subset of features for each split in a tree. This introduces further randomness, prevents overfitting, and ensures that each tree focuses on different aspects of the data.

5. Decision Tree Training:

- Each decision tree in the Random Forest is trained independently on its subset of data. The training process involves recursively splitting the data based on features, considering the best split at each node according to a specified criterion (commonly Gini impurity or information gain).

6. Voting (Classification):

- For classification tasks, each tree in the forest "votes" for a class. The class with the most votes becomes the final prediction of the Random Forest.

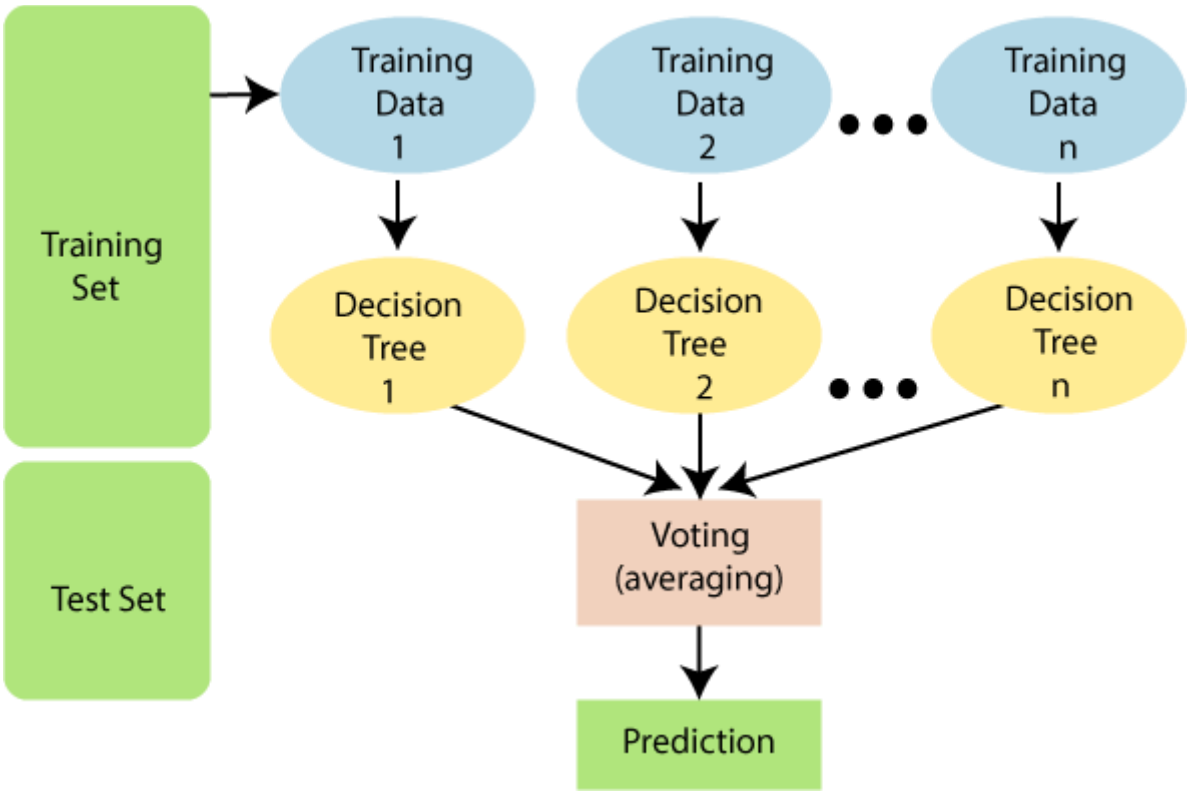
7. Averaging (Regression):

- For regression tasks, each tree predicts a numerical value, and the final prediction is the average of these values.

8. Out-of-Bag (OOB) Score:

- Because each tree is trained on a subset of the data, some data points are not included (out-of-bag) in the training of certain trees. The OOB score is calculated by evaluating each tree on the data it did not see during training. This provides an estimate of the model's performance without the need for a separate validation set.

The below diagram explains the working of the Random Forest algorithm:



GIVEN INPUT EXPECTED OUTPUT

Input: data

output: getting accuracy

5.2 MODULE – 2:

AdaBoost classifier:

This method operates iteratively, identifying misclassified data points and adjusting their weights to minimize the training error. The model continues to optimize sequentially untilit yields the strongest predictor.

AdaBoost is implemented by combining several weak learners into a single strong learner.The weak learners in AdaBoost take into account a single input feature and draw out a single split decision tree called the decision stump.

The results from the first decision stump are analyzed, and if any observations are wrongfully classified, they are assigned higher weights. A new decision stump is drawn byconsidering the higher-weight observations as more significant.

1. **Weak Learners:**

   AdaBoost starts by training a weak learner on the entire dataset . A weak learner is amodel that performs slightly better than random chance. Common weak learners aredecision stumps (simple decision trees with a single split)

2. **Weighted Data:**

   Each instance in the training dataset is assigned an initial weight. Initially, allweights are set equally, so each instance has the same importance.

3. **Training the Weak Learner**:

   The first weak learner is trained on the original data using the initial weights. Itproduces a model and predicts the target variable.

4. **Compute Error:**

   The algorithm computes the error of the weak learner by comparing its predictionsto the actual targets. Instances that are misclassified receive higher weights.

5. **Compute Learner Weight**:

   The weight of the weak learner is calculated based on its classification error. The lower the error, the higher the weight assigned to the weak learner.

6. **Update Instance Weights:**

   The instance weights are updated based on whether they were correctly or incorrectly classified by the weak learner. Misclassified instances receive higher weights.

7. Repeat:

Steps 3-6 are repeated for a predefined number of iterations or until a perfect modelis achieved.

8. Final Model:

The final model is an ensemble of weak learners, each contributing to the final prediction based on their individual weights.

9. Weighted Voting:

During prediction, each weak learner "votes" on the class of a new instance. The weight of each learner influences its contribution to the final prediction

10. Adaptive Learning:

Adaboost is adaptive, meaning it focuses more on instances that are difficult to classify correctly. As the algorithm progresses, it assigns higher weights to misclassified instances, forcing subsequent weak learners to focus on these challenging cases.

11. Combining Weak Models:

The final prediction is a weighted sum of the weak learners' predictions. The weights are determined by the accuracy of each weak learner on the training data.

GIVEN INPUT EXPECTED OUTPUT

Input : data

output : getting accuracy

5.3 MODULE – 3

KNN Classifier:

- K-Nearest Neighbour is one of the simplest Machine Learning algorithms based on Supervised Learning technique.

- K-NN algorithm assumes the similarity between the new case/data and available cases and put the new case into the category that is most similar to the available categories.
- K-NN algorithm stores all the available data and classifies a new data point based on the similarity. This means when new data appears then it can be easily classified into a well suite category by using K- NN algorithm.
- K-NN algorithm can be used for Regression as well as for Classification but mostly it is used for the Classification problems.
- K-NN is a non-parametric algorithm, which means it does not make any assumption on underlying data.
- It is also called a lazy learner algorithm because it does not learn from the trainingset immediately instead it stores the dataset and at the time of classification, it performs an action on the dataset.
- KNN algorithm at the training phase just stores the dataset and when it gets new data, then it classifies that data into a category that is much similar to the new data.

1. Overview:

KNN is a non-parametric, lazy learning algorithm that makes predictions by findingthe majority class of the k-nearest data points in the feature space.

2. Distance Metric:

KNN relies on a distance metric (e.g., Euclidean distance, Manhattan distance) to measure the similarity between data points in the feature space. The choice of distance metric depends on the nature of the data.

3. Training:

KNN is an instance-based algorithm, meaning it memorizes the entire training dataset.

4. Prediction:

To make a prediction for a new instance, KNN calculates the distancebetween the instance and all data points in the training set.

5. Nearest Neighbors:

The algorithm identifies the k-nearest neighbors of the new instance based on the computed distances. The value of k is a user-defined hyperparameter.

6. Majority Voting:

For classification, KNN uses majority voting among the k-nearest neighbors to determine the predicted class of the new instance. The class with the most occurrences among the neighbors is assigned.

7. Regression:

For regression tasks, the algorithm takes the average of the target values of the k- nearest neighbors as the predicted output.

8. Hyperparameter Tuning:

The choice of the hyperparameter k is critical. A smaller k value may lead to noisy predictions, while a larger k value may smooth the decision boundaries but could ignore local patterns. Cross-validation is often used to find the optimal value of

GIVEN INPUT EXPECTED OUTPUT

Input : data

output: getting accuracy

CHAPTER 6
CONCLUSION AND FUTURE WORK

Conclusion:

After the literature survey, we came to know various pros and cons of different research papers and thus, proposed a system that helps to predict brain strokes in a cost effective and efficient way by taking few inputs from the user side and predicting accurate results with the help of trained Machine Learning algorithms. Thus, the Brain Stroke Prediction system has been implemented using the given Machine Learning algorithm given a Best accuracy.

The system is therefore designed providing simple yet efficient User Interface design with an empathetic approach towards their users and patients. The system has a potential for future scope which

Future Work:

- The added background knowledge from other datasets can also possibly improve the accuracy of stroke prediction models as well.
- We intend to collect our institutional dataset for further benchmarking of these machine learning methods for stroke prediction.
- We also plan to perform external validation of our proposed method, as a part of our upcoming planned work.

CHAPTER 7
APPENDICE

7.1 Source Code:

Module 3 : Performance measurements of Ada-boost classifier:

```
# Import the Necessary packages.
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

import warnings
warnings.filterwarnings('ignore')
In [ ]:
df = pd.read_csv('CEREBRAL.csv')
del df['id']
df.head()
In [ ]:
df.columns
In [ ]:
df=df.dropna()
In [ ]:
df.columns
In [ ]:
from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()
var = ['gender','ever_married','work_type','Residence_type','smoking_status']
```

```
for i in var:
    df[i] =  le.fit_transform(df[i]).astype(int)
```

In [ ]:

```
df.tail()
```

In [ ]:

```
x1 = df.drop(labels='stroke', axis=1)
y1 = df.loc[:,'stroke']
```

In [ ]:

```
import imblearn
from imblearn.over_sampling import RandomOverSampler
from collections import Counter
ros =RandomOverSampler(random_state=42)
x,y=ros.fit_resample(x1,y1)
print("OUR DATASET COUNT       : ", Counter(y1))
print("OVER SAMPLING DATA COUNT : ", Counter(y))
```

In [ ]:

```
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.20, random_state=42,
stratify=y)
print("NUMBER OF TRAIN DATASET   : ", len(x_train))
print("NUMBER OF TEST DATASET    : ", len(x_test))
print("TOTAL NUMBER OF DATASET   : ", len(x_train)+len(x_test))
```

In [ ]:

```
print("NUMBER OF TRAIN DATASET   : ", len(y_train))
print("NUMBER OF TEST DATASET    : ",  len(y_test))
print("TOTAL NUMBER OF DATASET   : ", len(y_train)+len(y_test))
```

In [ ]:

```
from sklearn.ensemble import AdaBoostClassifier
```

In [ ]:

```
ADB = AdaBoostClassifier(random_state=42)
ADB.fit(x_train,y_train)
```

In [ ]:

```
predicted = ADB.predict(x_test)
```

In [ ]:

```
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(y_test,predicted)
print('THE CONFUSION MATRIX SCORE OF ADABOOST CLASSIFIER:\n\n\n',cm)
```

In [ ]:

```
from sklearn.model_selection import cross_val_score
accuracy = cross_val_score(ADB, x, y, scoring='accuracy')
print('THE CROSS VALIDATION TEST RESULT OF ACCURACY :\n\n\n',
accuracy*100)
```

In [ ]:

```
from sklearn.metrics import accuracy_score
a = accuracy_score(y_test,predicted)
print("THE ACCURACY SCORE OF ADABOOST CLASSIFIER IS :",a*100)
```

In [ ]:

```
from sklearn.metrics import hamming_loss
hl = hamming_loss(y_test,predicted)
print("THE HAMMING LOSS OF ADABOOST CLASSIFIER IS :",hl*100)
```

In [ ]:

```
from sklearn.metrics import precision_score
P = precision_score(y_test,predicted)
print("THE PRECISION SCORE OF ADABOOST CLASSIFIER IS :",P*100)
```

In [ ]:

```
from sklearn.metrics import recall_score
```

```
R = recall_score(y_test,predicted)
print("THE RECALL SCORE OF ADABOOST CLASSIFIER IS :",R*100)
In [ ]:
from sklearn.metrics import f1_score
f1 = f1_score(y_test,predicted)
print("THE PRECISION SCORE OF ADABOOST CLASSIFIER IS :",f1*100)
In [ ]:
def plot_confusion_matrix(cm, title='THE CONFUSION MATRIX SCORE OF ADABOOST CLASSIFIER\n\n', cmap=plt.cm.Blues):
    target_names=['']
    plt.imshow(cm, interpolation='nearest', cmap=cmap)
    plt.title(title)
    plt.colorbar()
    tick_marks = np.arange(len(target_names))
    plt.xticks(tick_marks, target_names, rotation=45)
    plt.yticks(tick_marks, target_names)
    plt.tight_layout()
    plt.ylabel('True label')
    plt.xlabel('Predicted label')
cm=confusion_matrix(y_test, predicted)
print('THE CONFUSION MATRIX SCORE OF ADABOOST CLASSIFIER:\n\n')
print(cm)
sns.heatmap(cm/np.sum(cm), annot=True, cmap = 'Blues', annot_kws={"size": 16},fmt='.2%')
plt.show()
In [ ]:
def graph():
import matplotlib.pyplot as plt
```

```
data=[a]
    alg=" ADABOOST CLASSIFIER"
    plt.figure(figsize=(5,5))
    b=plt.bar(alg,data,color=("GREEN"))
    plt.title("THE ACCURACY SCORE OF ADABOOST CLASSIFIER IS\n\n\n")
    plt.legend(b,data,fontsize=9)
graph()
```

In [ ]:

Module 2: Implementing KNN classifier

```
# Import the neccsary packages.
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings('ignore')
```

In [ ]:

```
df = pd.read_csv('CEREBRAL.csv')
del df['id']
df.head()
```

In [ ]:

```
df.columns
```

In [ ]:

```
df=df.dropna()
```

In [ ]:

```
from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()
```

```
var = ['gender','ever_married','work_type','Residence_type','smoking_status']

for i in var:
    df[i] = le.fit_transform(df[i]).astype(int)
```

In [ ]:

```
df.tail()
```

In [ ]:

```
x1 = df.drop(labels='stroke', axis=1)
y1 = df.loc[:,'stroke']
```

In [ ]:

```
import imblearn
from imblearn.over_sampling import RandomOverSampler
from collections import Counter
ros =RandomOverSampler(random_state=42)
x,y=ros.fit_resample(x1,y1)
print("OUR DATASET COUNT       : ", Counter(y1))
print("OVER SAMPLING DATA COUNT : ", Counter(y))
```

In [ ]:

```
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.20, random_state=42,
stratify=y)
print("NUMBER OF TRAIN DATASET   : ", len(x_train))
print("NUMBER OF TEST DATASET    : ", len(x_test))
print("TOTAL NUMBER OF DATASET   : ", len(x_train)+len(x_test))
```

In [ ]:

```
print("NUMBER OF TRAIN DATASET   : ", len(y_train))
print("NUMBER OF TEST DATASET    : ", len(y_test))
print("TOTAL NUMBER OF DATASET   : ", len(y_train)+len(y_test))
```

In [ ]:

```
from sklearn.neighbors import KNeighborsClassifier
In [ ]:
KNN = KNeighborsClassifier()
KNN.fit(x_train,y_train)
In [ ]:
predicted = KNN.predict(x_test)
In [ ]:
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(y_test,predicted)
print('THE CONFUSION MATRIX SCORE OF KNEIGHBORS CLASSIFIER:\n\n\n',cm)
In [ ]:
from sklearn.model_selection import cross_val_score
accuracy = cross_val_score(KNN, x, y, scoring='accuracy')
print('THE CROSS VALIDATION TEST RESULT OF ACCURACY :\n\n\n', accuracy*100)
In [ ]:
from sklearn.metrics import accuracy_score
a = accuracy_score(y_test,predicted)
print("THE ACCURACY SCORE OF KNEIGHBORS CLASSIFIER IS :",a*100)
In [ ]:
from sklearn.metrics import hamming_loss
hl = hamming_loss(y_test,predicted)
print("THE HAMMING LOSS OF KNEIGHBORS CLASSIFIER IS :",hl*100)
In [ ]:
from sklearn.metrics import precision_score
P = precision_score(y_test,predicted)
print("THE PRECISION SCORE OF KNEIGHBORS CLASSIFIER IS :",P*100)
```

In [ ]:

```
from sklearn.metrics import recall_score
R = recall_score(y_test,predicted)
print("THE RECALL SCORE OF KNEIGHBORS CLASSIFIER IS :",R*100)
```

In [ ]:

```
from sklearn.metrics import f1_score
f1 = f1_score(y_test,predicted)
print("THE PRECISION SCORE OF KNEIGHBORS CLASSIFIER IS :",f1*100)
```

In [ ]:

```
def plot_confusion_matrix(cm, title='THE CONFUSION MATRIX SCORE OF KNEIGHBORS CLASSIFIER\n\n', cmap=plt.cm.Blues):
    target_names=['']
    plt.imshow(cm, interpolation='nearest', cmap=cmap)
    plt.title(title)
    plt.colorbar()
   tick_marks = np.arange(len(target_names))
   plt.xticks(tick_marks, target_names, rotation=45)
   plt.yticks(tick_marks, target_names)
   plt.tight_layout()
    plt.ylabel('True label')
    plt.xlabel('Predicted label')
cm=confusion_matrix(y_test, predicted)
print('THE CONFUSION MATRIX SCORE OF KNEIGHBORS CLASSIFIER:\n\n')
print(cm)
sns.heatmap(cm/np.sum(cm), annot=True, cmap = 'Blues', annot_kws={"size": 16},fmt='.2%')
plt.show()
```

In [ ]:

```
def graph():
    import matplotlib.pyplot as plt
    data=[a]
    alg=" KNEIGHBORS CLASSIFIER"
    plt.figure(figsize=(5,5))
    b=plt.bar(alg,data,color=("BLUE"))
    plt.title("THE ACCURACY SCORE OF KNEIGHBORS CLASSIFIER IS\n\n\n")
    plt.legend(b,data,fontsize=9)
graph()
```

Module – 1 Implementing the Random forest classifier:

```
# Import the neccessary packages
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings('ignore')
In [ ]:
df = pd.read_csv('CEREBRAL.csv')
del df['id']
df.head()
In [ ]:
df.columns
In [ ]:
df=df.dropna()
In [ ]:
from sklearn.preprocessing import LabelEncoder
```

```
le = LabelEncoder()
var = ['gender','ever_married','work_type','Residence_type','smoking_status'

for i in var:
    df[i] = le.fit_transform(df[i]).astype(int)
In [ ]:
df.tail()
In [ ]:
x1 = df.drop(labels='stroke', axis=1)
y1 = df.loc[:,'stroke']
In [ ]:
#import imblearn
from imblearn.over_sampling import RandomOverSampler
from collections import Counter
ros =RandomOverSampler(random_state=42)
x,y=ros.fit_resample(x1,y1)
print("OUR DATASET COUNT       : ", Counter(y1))
print("OVER SAMPLING DATA COUNT : ", Counter(y))
In [ ]:
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.20, random_state=42,
stratify=y)
print("NUMBER OF TRAIN DATASET   : ", len(x_train))
print("NUMBER OF TEST DATASET    : ", len(x_test))
print("TOTAL NUMBER OF DATASET   : ", len(x_train)+len(x_test))
In [ ]:
print("NUMBER OF TRAIN DATASET   : ", len(y_train))
print("NUMBER OF TEST DATASET    : ", len(y_test))
print("TOTAL NUMBER OF DATASET   : ", len(y_train)+len(y_test))
```

In [ ]:

```
from sklearn.ensemble import RandomForestClassifier
```

In [ ]:

```
RFC = RandomForestClassifier(random_state=42)
RFC.fit(x_train,y_train)
```

In [ ]:

```
predicted = RFC.predict(x_test)
```

In [ ]:

```
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(y_test,predicted)
print('THE CONFUSION MATRIX SCORE OF RANDOM FOREST CLASSIFIER:\n\n\n',cm)
```

In [ ]:

```
from sklearn.model_selection import cross_val_score
accuracy = cross_val_score(RFC, x, y, scoring='accuracy')
print('THE CROSS VALIDATION TEST RESULT OF ACCURACY :\n\n\n', accuracy*100)
```

In [ ]: from

```
sklearn.metric
s import
accuracy_scor
ea =
accuracy_scor
e(y_test,predic
ted)
print("THE ACCURACY SCORE OF RANDOM FOREST CLASSIFIER IS :",a*100)
```

In [ ]:

```
from sklearn.metrics import hamming_loss
```

```
hl = hamming_loss(y_test,predicted)
print("THE HAMMING LOSS OF RANDOM FOREST CLASSIFIER IS :",hl*100)
In [ ]:
from sklearn.metrics import precision_score
P = precision_score(y_test,predicted)
print("THE PRECISION SCORE OF RANDOM FOREST CLASSIFIER IS :",P*100)
In [ ]:
from sklearn.metrics import recall_score
R = recall_score(y_test,predicted)
print("THE RECALL SCORE OF RANDOM FOREST CLASSIFIER IS :",R*100)
In [ ]:
from sklearn.metrics import f1_score
f1 = f1_score(y_test,predicted)
print("THE PRECISION SCORE OF RANDOM FOREST CLASSIFIER IS :",f1*100)
In [ ]:
def plot_confusion_matrix(cm, title='THE CONFUSION MATRIX SCORE OF
RANDOM FOREST CLASSIFIER\n\n', cmap=plt.cm.Blues):
    target_names=['']
    plt.imshow(cm, interpolation='nearest', cmap=cmap)
    plt.title(title)
    plt.colorbar()
    tick_marks = np.arange(len(target_names))
    plt.xticks(tick_marks, target_names, rotation=45)
    plt.yticks(tick_marks, target_names)
    plt.tight_layout()
    plt.ylabel('True label')
    plt.xlabel('Predicted label')
cm=confusion_matrix(y_test, predicted)
```

```
print('THE CONFUSION MATRIX SCORE OF RANDOM FOREST CLASSIFIER:\n\n')
print(cm)
sns.heatmap(cm/np.sum(cm), annot=True, cmap = 'Blues', annot_kws={"size": 16},fmt='.2%')
plt.show()
```

In [ ]:

```
def graph()
    import matplotlib.pyplot as pltdata=[a]
    alg="RANDOM FOREST CLASSIFIER"
    plt.figure(figsize=(5,5))
    b=plt.bar(alg,data,color=("YELLOW"))
    plt.title("THE ACCURACY SCORE OF RANDOM FOREST CLASSIFIER IS\n\n\n")
    plt.legend(b,data,fontsize=9)
graph()
```

In [ ]:

```
import joblib
joblib.dump(RFC, 'RFC.pkl')
```

BASE :

```
{% load static %}
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link rel="stylesheet" href="{% static 'css/bootstrap.css' %}">
    <link rel="stylesheet" href="{% static 'css/master.css' %}">
    <title>Doc</title>
<style>
```

```
th,td{
    border-style:solid; border-
    width:1px; border-
    color:#BCBCBC;word-
    wrap:break-word;
}
.feed_table{
        margin-left: 50px;
        margin-right: 50px;
}
.index{
    background:url("/static/image/brain.gif");
    background-repeat: no-repeat; background-
    attachment: fixed; background-size: 100%
    100%;
}
.container1{
  background: #f06d06;
  padding: 40px; margin:
  20%;
  margin-top: 5%;
  margin-bottom: 5%;  border:
  2px solid #61dafb;border-
  radius: 1em;
}
.form1{
  background-color:  #023349;
```

```
}
.legend1{
  box-sizing: content-box;
  padding: 10px;
  border-radius: 1em;
  background-color: #023349;
  font-size: 20px;
  font-weight: 900;
  color: #61dafb;
}
.input1{ width:
  100%;
  padding: 12px;
  border: 1px solid #61dafb;
}

.fieldset1{
  /* background-color: #ffff; */
  border: 2px solid #61dafb;
  border-radius: 1em;
}
.input1[type="submit"],input[type="reset"]{
  font-size: 17px;
  background-color: #61dafb;
  cursor: pointer;
  border-radius: 4px;
}
```

```
.input1[type="submit"]:hover,input[type="reset"]:hover{color:
 white;
 background-color: #049ff8;
}
</style>
</head>
<body>
    {% block content %}

    {% endblock content %}

</body>
</html>
```

INDEX.HTML :

```
{% extends 'new/base.html' %}
{% load static %}
    {% block content %}
    <body class="index">
    <div>
        <div style="background-color: #e9ecef;">
        <a class="navbar-brand text-dark" style="margin-left: 30px;" href="{% url 'apredict' %}">Enter Details</a>
        <a class="navbar-brand text-dark" style="margin-left: 1088px;" href="{% url 'home' %}">Go Back</a>
    </div><br>
    <h1 style="text-align:center; color:cyan;"><b>BRAIN STROKE PREDICTION </b></h1>
  <!-- Main Input For Receiving Query to our ML -->

    <form class="form1 container1" action="{% url 'predict' %}" style="background-color: transparent; box-sizing: border-box;
    font-family: proxima_nova_rgregular, Helvetica, Arial, sans-serif;" method="POST">
    {% csrf_token %}
        <fieldset class="fieldset1">
            <legend class="legend1">Enter the Detail</legend>

    <label class="white"  for="">GENDER</label>
```

<input type="number" class="space form-control" step="0.01" name="gender" placeholder="Press 1 Male 2 Female" required="required" />

<label class="white" for="">AGE</label>

<input type="number" class="space form-control" step="0.01" name="age" placeholder="AGE" required="required" />

<label class="white" for="">HYPERTENSION</label>

<input type="number" class="space form-control" step="0.01" name="hypertension" placeholder="HYPERTENSION" required="required" />

<label class="white" for="">HEART DISEASE</label>

<input type="number" class="spaceform-control" step="0.01" name="heart_disease" placeholder="HEART DISEASE" required="required" />

<label class="white" for="">AVERAGE GLUCOSE LEVEL</label>

<input type="number" class="space form-control" step="0.01" name="avg_glucose_level" placeholder="AVERAGE GLUCOSE LEVEL" required="required" />

<label class="white" for="">BMI</label>

<input type="number" class="space form-control" step="0.01" name="bmi" placeholder="BMI" required="required" />

<label class="white" for="">SMOKING STATUS</label>

<input type="number" class="space form-control" step="0.01" name="smoking_status" placeholder="SMOKING STATUS" required="required" />

<input type="hidden" name="stroke" value="data">

<input class="input1" type="submit" value="Submit">

```
                </fieldset>
            </form>
    <br>
    <br>
<div  style="background:skyblue;padding:2%  40%">
    {{ prediction_text }}
</div>
 </div>
 {% endblock content %}
</body>
</html>
```

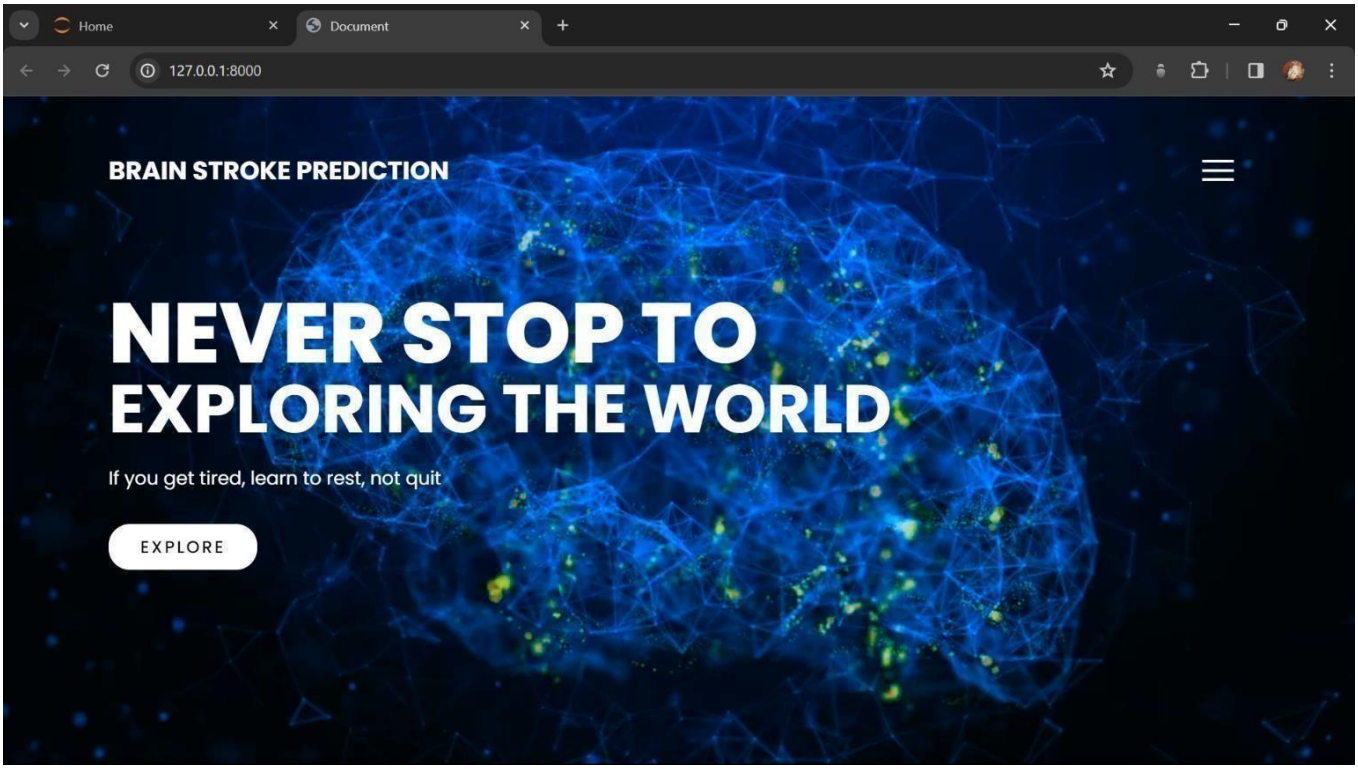
OUTPUT CODE:

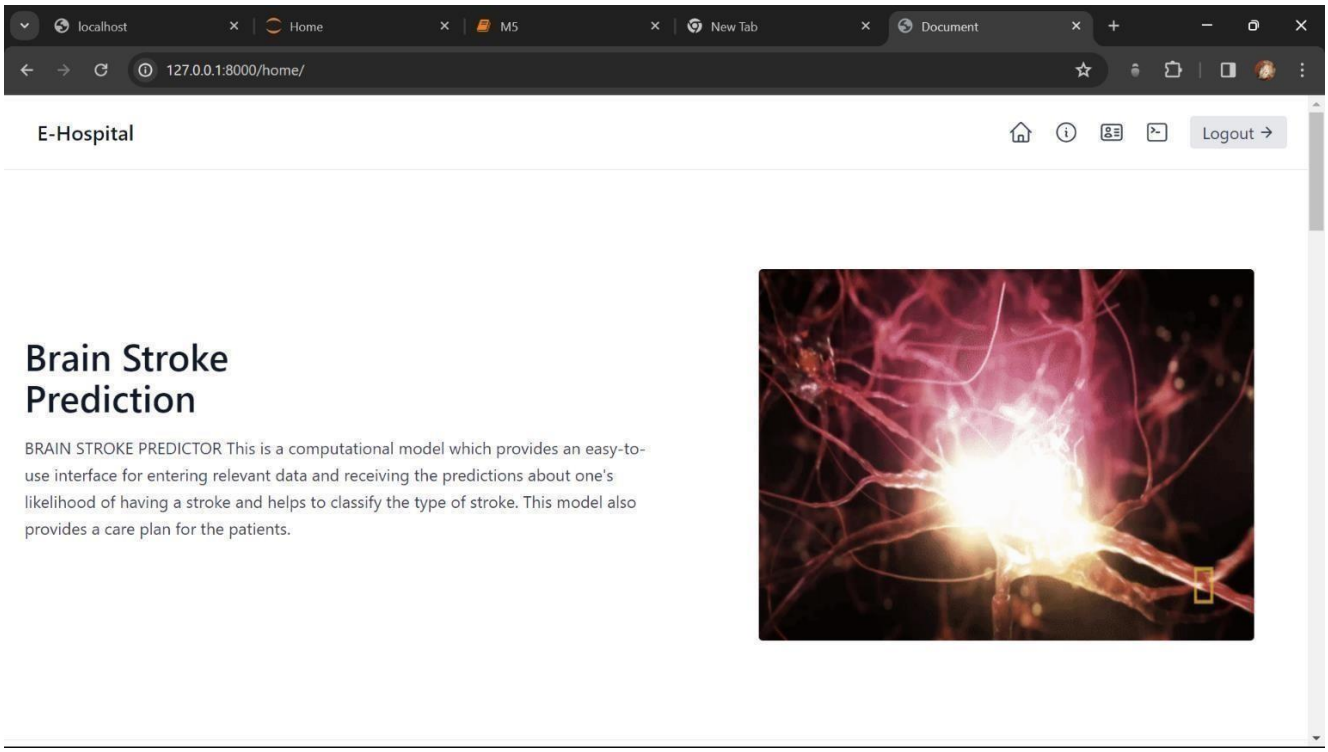
```
    {% extends 'new/base.html'  %}
    {% load static %}
{% block content %}
<body>
    <div>
                <div  style="background-color:  #e9ecef;">
                    <a class="navbar-brand text-dark" style="margin-left: 30px;" href="{% url
%}">All</       'details_all'
a>
                    <a class="navbar-brand text-dark" style="margin-left: 10px;" href="{% url
'details_last' %}">Last</a>
            </div>
        <br>


<center>
```

```
<div class="feed_table">
  <table class="table" border="default" style="table-layout: fixed;">
  <tr>
    <th>Gender</th><th>Age</th><th>Hypertension</th>
    <th>Heartdisease</th><th>avg_glucose_level</th><th>bmi</th>
    <th>smoking_status</th>
    <th>stroke</th>
  </tr>
    <tr>
      <td>{{model.gender}}</td><td>{{model.age}}</td>
      <td>{{model.hypertension}}</td><td>{{model.heart_disease}}</td>
      <td>{{model.avg_glucose_level}}</td>
      <td>{{model.bmi}}</td><td>{{model.smoking_status}}</td>
      <td>{{model.stroke}}</td>
    </tr>
 </table>
</div>
</center>
<br><br>
<div>
<center>
  <form action="{% url 'details_last' %}" method="POST">
    {% csrf_token %}
    {% for f in form %}
    <p>
      {{ f.label_tag }}<br>
      {{ f }}
```

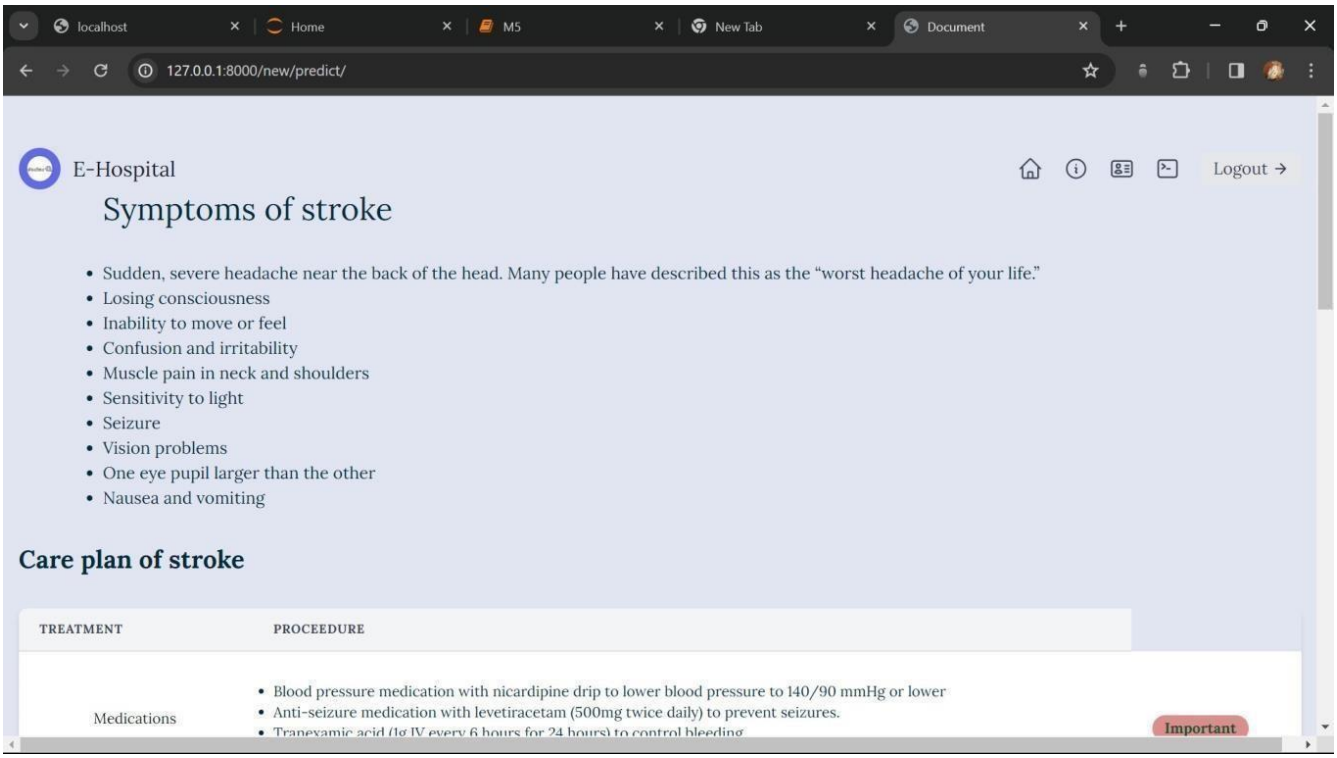
```
            </p>
            {% endfor %}
            <br>


          {% if not msg %}
             <input type="submit">
           {% else %}
              {{msg}}
           {% endif %}
        </form>
     </center>
      </div>
{% endblock content %}
</body>
</html>
```

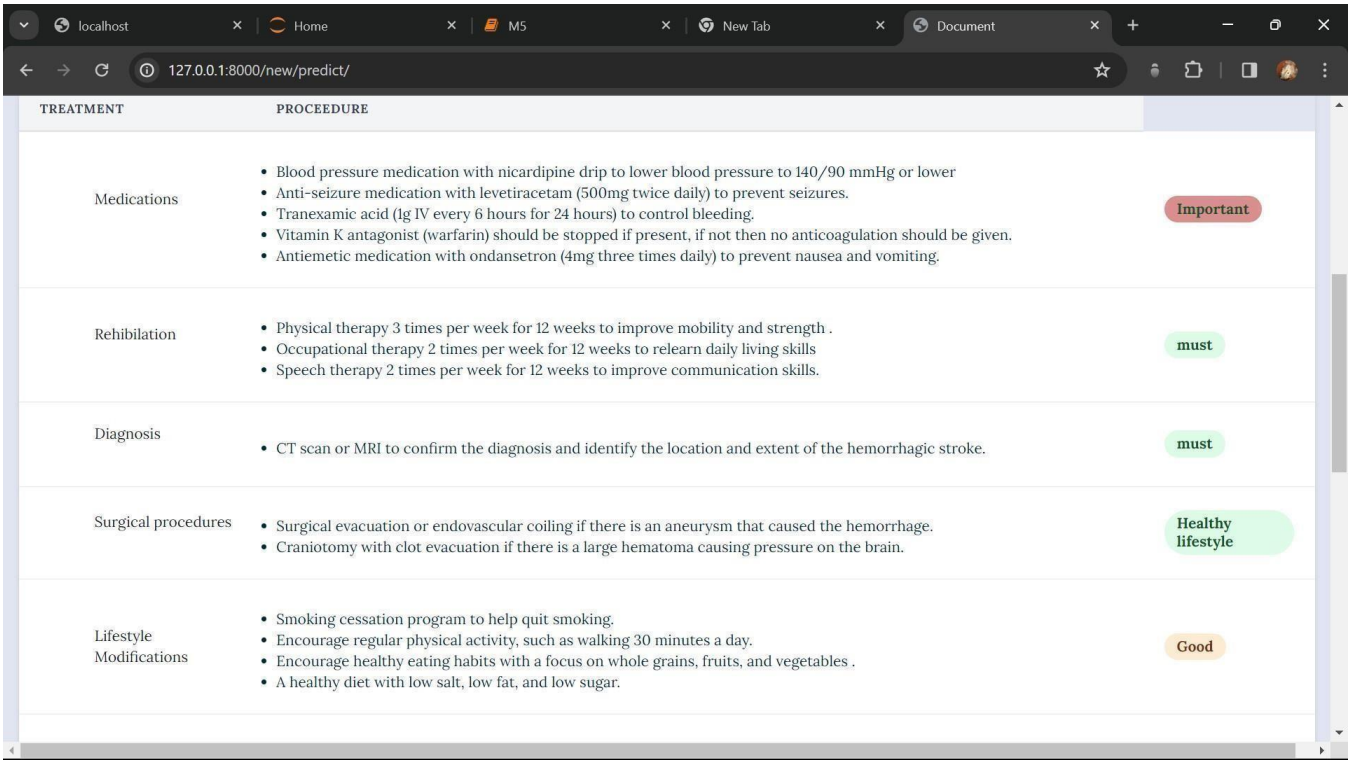
7.2 SCREEN SHOT:



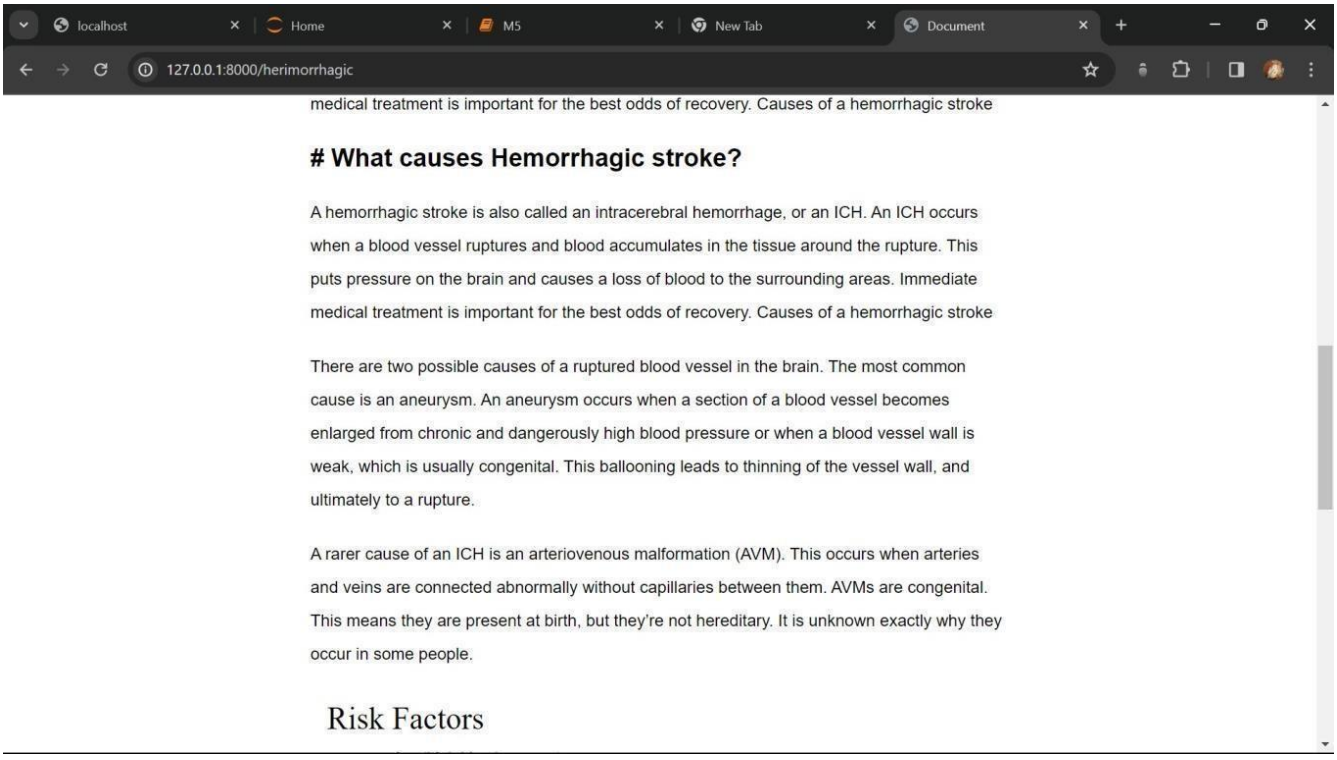



localhost
Home
M5
New Tab
Doc
127.0.0.1:8000/new/predict/
BRAIN STROKE PREDICTION
Enter the Detail
GENDER
Press 1 Male 2 Female
AGE
AGE
HYPERTENSION
HYPERTENSION


localhost
Home
M5
New Tab
Document
127.0.0.1:8000/new/predict/
E-Hospital
Logout
Symptoms of stroke
Sudden, severe headache near the back of the head. Many people have described this as the "worst headache of your life."
Losing consciousness
Inability to move or feel
Confusion and irritability
Muscle pain in neck and shoulders
Sensitivity to light
Seizure
Vision problems
One eye pupil larger than the other
Nausea and vomiting
Care plan of stroke
TREATMENT
PROCEEDURE
Medications
Blood pressure medication with nicardipine drip to lower blood pressure to 140/90 mmHg or lower
Anti-seizure medication with levetiracetam (500mg twice daily) to prevent seizures.
Important



TREATMENT	PROCEEDURE	
Medications	• Blood pressure medication with nicardipine drip to lower blood pressure to 140/90 mmHg or lower • Anti-seizure medication with levetiracetam (500mg twice daily) to prevent seizures. • Tranexamic acid (1g IV every 6 hours for 24 hours) to control bleeding. • Vitamin K antagonist (warfarin) should be stopped if present, if not then no anticoagulation should be given. • Antiemetic medication with ondansetron (4mg three times daily) to prevent nausea and vomiting.	Important
Rehibilation	• Physical therapy 3 times per week for 12 weeks to improve mobility and strength . • Occupational therapy 2 times per week for 12 weeks to relearn daily living skills • Speech therapy 2 times per week for 12 weeks to improve communication skills.	must
Diagnosis	• CT scan or MRI to confirm the diagnosis and identify the location and extent of the hemorrhagic stroke.	must
Surgical procedures	• Surgical evacuation or endovascular coiling if there is an aneurysm that caused the hemorrhage. • Craniotomy with clot evacuation if there is a large hematoma causing pressure on the brain.	Healthy lifestyle
Lifestyle Modifications	• Smoking cessation program to help quit smoking. • Encourage regular physical activity, such as walking 30 minutes a day. • Encourage healthy eating habits with a focus on whole grains, fruits, and vegetables . • A healthy diet with low salt, low fat, and low sugar.	Good



7.3 Plagiarism Report:

1221

ORIGINALITY REPORT

14%	10%	4%	5%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Submitted to Manipal University Student Paper	2%
2	www.hindawi.com Internet Source	1%
3	www.jetir.org Internet Source	1%
4	www.researchgate.net Internet Source	1%
5	doctorpenguin.com Internet Source	1%
6	www.irejournals.com Internet Source	

1%

7	export.arxiv.org Internet Source	1%
8	sist.sathyabama.ac.in Internet Source	1%
9	www.ijritcc.org Internet Source	1%

10	Raja Waseem Anwar, Muhammad Abrar, Faizan Ullah. "Transfer Learning in Brain Tumor Classification: Challenges, Opportunities and Future Prospects"2023 4th International Conference on Information 1 and Communication Technology Convergence (ICTC), 2023 Publication	<1%
		<1%
	ijritcc.org Internet Source	<1%
12	Submitted to Panimalar Engineering College Student Paper	<1%
13	Submitted to Asia Pacific University College of Technology and Innovation (UCTI) Student Paper	
14	www.coursehero.com Internet Source	<1%
	pubmed.ncbi.nlm.nih.gov	<1%

15 Internet Source

---

16 www.ijisr.issr-journals.org
Internet Source

<1%

17 Chidambaram. "Ensemble Classification for Stroke Prediction and Diagnosis: Enhancing Accuracy through Collaborative Algorithms", 2023 International Conference on Research Methodologies in Knowledge Management, Artificial Intelligence and Telecommunication Engineering (RMKMATE), 2023 <1%

Publication

Exclude quotes On Exclude bibliography On

Exclude matches Off

REFERENCE:

Y. Zhang et al., "Automatic Detection of Cerebral Stroke from MRI Images Using Deep Learning," in IEEE Transactions on Medical Imaging, vol. 39, no. 8, pp. 2676-2687, Aug. 2020.

S. Gupta et al., "Real-time Monitoring of Cerebral Stroke Using Wireless Sensor Networks," in IEEE Transactions on Biomedical Engineering, vol. 66, no. 3, pp. 786-794, March 2019.

H. Kim et al., "Detection of Early Signs of Cerebral Stroke Using Wearable EEG Devices," in IEEE Sensors Journal, vol. 20, no. 10, pp. 5496-5504, May 2020.

R. Patel et al., "An IoT-Based Smart System for Remote Monitoring and Diagnosis of Cerebral Stroke," in IEEE Internet of Things Journal, vol. 7, no. 6,pp. 5037-5046, June 2020.

A. Sharma et al., "Deep Reinforcement Learning for Decision Support in Cerebral Stroke Treatment," in IEEE Transactions on Neural Networks and Learning Systems, vol. 32, no. 9, pp. 3821-3832, Sept. 2021.

J. Chen et al., "Electroencephalogram-Based Classification of Cerebral Stroke Severity Using Machine Learning Techniques," in IEEE Access, vol. 9, pp. 41890-41900, March 2021.

T. Li et al., "A Novel Brain-Computer Interface for Communication in Patients with Locked-In Syndrome Following Cerebral Stroke," in IEEE Transactions on Neural Rehabilitation and Engineering, vol. 29, no. 5, pp.1158-1167, May 2021.

75