RISK PREDICTION FOR TRAUMATIC BRAIN INJURY STROKE USING KERNEL EXTREME LEARNING MACHINE AND NEUTROSOPHIC C-MEANS-BASED ATTIRBUTE WEIGHTING ALGORITHM.

A PROJECT REPORT

Submitted by

NIVETHA SANKARASUBRAMANIAN [Reg No:RA2011003010172]

PRAVEEN KRISHNAKANTH S [Reg No: RA2011003010152]

Under the Guidance of

Dr. D. SHINY IRENE

Associate Professor, Department of Computing Technologies

in partial fulfillment of the requirements for the degree of

BACHELOR OF TECHNOLOGY

COMPUTER SCIENCE AND ENGINEERING

DEPARTMENT OF COMPUTING TECHNOLOGIES

COLLEGE OF ENGINEERING AND TECHNOLOGY
SRM INSTITUTE OF SCIENCE AND TECHNOLOGY
KATTANKULATHUR– 603 203

November 2023

SRM INSTITUTE OF SCIENCE AND TECHNOLOGY

KATTANKULATHUR–603 203

BONAFIDE CERTIFICATE

Certified that 18CSP109L / I8CSP111L minor project report titled "**RISK PREDICTION FOR TRAUMATIC BRAIN INJURY STROKE USING KERNEL EXTREME LEARNING MACHINE AND NEUTROSOPHIC C-MEANS-BASED ATTRIBUTE WEIGHTING ALGORITHM**" is the Bonafide work of **NIVETHA SANKARASUBRAMANIAN [RA2011003010172]** and **PRAVEEN KRISHNAKANTH S [RA2011003010152]** who carried out the project work under my supervision. Certified further, that to the best of my knowledge the work reported here in does not form part of any other thesis or dissertation on the basis of which a degree or award was conferred on an earlier occasion for this or any other candidate.

[illegible]

Dr. D. SHINY IRENE
SUPERVISOR
Associate Professor
Department of Computing Technologies

[illegible]

Dr. J. SELVIN PAUL PETER
PANEL HEAD
Associate Professor
Department of Computing Technologies

M. Pushpalatha

Dr. M. PUSHPALATHA
HEAD OF THE DEPARTMENT
Department of Computing Technologies

Department of Computing Technology
Kattankulathur
603 203
* SRM IST *

Department of Computing Technologies

SRM Institute of Science and Technology

Own Work Declaration Form

Degree/Course	: B.Tech in Computer Science and Engineering
Student Names	: Praveen Krishnakanth S, Nivetha Sankarasubramanian
Registration Number	: RA2011003010152, RA2011003010172
Title of Work	: Risk Prediction for Traumatic Brain injury Stroke Using Kernel Extreme Learning Machine and Neutrosophic C-Means-Based Attribute Weighting Algorithm

I/We here by certify that this assessment compiles with the University's Rules and Regulations relating to Academic misconduct and plagiarism, as listed in the University Website, Regulations, and the Education Committee guidelines.

I / We confirm that all the work contained in this assessment is our own except where indicated, and that we have met the following conditions:

- Clearly references / listed all sources as appropriate
- Referenced and put in inverted commas all quoted text(from books, web,etc.)
- Given the sources of all pictures, data etc that are not my own.
- Not made any use of the report(s) or essay(s) of any other student(s)either past or present
- Acknowledged in appropriate places any help that I have received from others (e.g fellow students, technicians, statisticians, external sources)
- Compiled with any other plagiarism criteria specified in the Course hand book / University website

I understand that any false claim for this work will be penalized in accordance with the University policies and regulations.

DECLARATION:

I am aware of and understand the University's policy on Academic misconduct and plagiarism and I certify that this assessment is my / our own work, except where indicated by referring, and that I have followed the good academic practices noted above.

Student 1 Signature: [illegible]

Student 2 Signature: [illegible]

Date: 04|11|2023

If you are working in a group, please write your registration numbers and sign with thedate for every student in your group.

ACKNOWLEDGEMENT

We express our humble gratitude to **Dr. C. Muthamizhchelvan**, Vice-Chancellor, SRM Institute of Science and Technology, for the facilities extended for the project work and his continued support.

We extend our sincere thanks to Dean-CET, SRM Institute of Science and Technology, **Dr. T. V. Gopal**, for his invaluable support.

We wish to thank **Dr. Revathi Venkataraman**, Professor and Chairperson, School of Computing, SRM Institute of Science and Technology, for her support throughout the project work.

We are incredibly grateful to our Head of the Department, **Dr. M. Pushpalatha,** Professor, Department of Computing Technologies, SRM Institute of Science and Technology, for her suggestions and encouragement at all the stages of the project work.

We want to convey our thanks to our Project Coordinators, **Dr. D. Shiny Irene**, **Dr. L. Sathyapriya, Dr. C. Gokulnath and Dr. V.S. Saranya**, Panel Head, **Dr. J. Selvin Paul Peter**, Associate Professor and Panel Members, **Dr. L. Sathyapriya** Assistant Professor**, Dr. C. Gokulnath** Assistant Professor and **Dr. V.S. Saranya** Assistant Professor, Department of Computing Technologies, SRM Institute of Science and Technology, for their inputs during the project reviews and support.

We register our immeasurable thanks to our Faculty Advisor, **Dr. Vinod D** and **Dr. M. Uma Devi** Assistant Professor, Department of Computing Technologies, SRM Institute of Science and Technology, for leading and helping us to complete our course. Our inexpressible respect and thanks to our guide, **Dr. D. Shiny Irene**, Associate Professor, Department of Computing.

Technologies, SRM Institute of Science and Technology, for providing us with an opportunity to pursue our project under his / her mentorship. He / She provided us with the freedom and support to explore the research topics of our interest. His / Her passion for solving problems and making a difference in the world has always been inspiring.

We sincerely thank all the staff and students of Computing Technologies Department, School of Computing, S.R.M Institute of Science and Technology, for their help during our project. Finally, we would like to thank our parents, family members, and friends for their unconditional love, constant support and encouragement.

NIVETHA SANKARASUBRAMANIAN
(RA2011003010172)

PRAVEEN KRISHNAKANTH S
(RA2011003010152)

ABSTRACT

Brain strokes, or cerebrovascular accidents, represent a significant global health challenge due to their debilitating consequences and high mortality rates. Timely detection is paramount for effective intervention and improved patient outcomes. This abstract provides a comprehensive review of the current state of brain stroke detection analysis, encompassing various diagnostic methodologies, technologies, and recent advancements in the field. The review delves into the wide array of diagnostic tools used in brain stroke detection, including clinical assessments, neuroimaging modalities such as magnetic resonance imaging (MRI) and computed tomography (CT), and cutting-edge machine learning and artificial intelligence (AI) techniques. It assesses the strengths and limitations of each method and discusses the potential benefits of combining these approaches to create a more accurate and comprehensive diagnostic framework. The findings suggest that the integration of AI and machine learning with traditional diagnostic methods holds promise for enhancing brain stroke detection accuracy. By incorporating individualized patient data, these technologies offer the potential for early detection, personalized risk assessment, and timely interventions, thereby improving patient outcomes. This review serves as a valuable resource for healthcare professionals, researchers, and policymakers, providing an in-depth understanding of the current landscape of brain stroke detection analysis. It underscores the significance of adopting innovative technologies and data-driven approaches to address this critical healthcare concern, while upholding rigorous ethical and privacy standards. The pursuit of more precise and timely brain stroke detection offers the prospect of reducing the global burden of stroke, ultimately enhancing public health and patient care.

TABLE OF CONTENTS

LIST OF FIGURE

LIST OF TABLE

LIST OF ABBREVIATIONS

CNN	Convolutional Neural Networks
RNN	Recurrent Neural Networks
UML	Unified Modeling Language
SVM	Support Vector Machine

CHAPTER 1

1.INTRODUCTION

1.1 General

At its core, this project represents a comprehensive and ambitious endeavor with the primary aim of revolutionizing the early detection and diagnosis of brain strokes, which stands as a pivotal and urgent goal within modern healthcare. Brain strokes, characterized by their potentially devastating consequences, underscore the critical need for swift and precise identification and intervention. To meet this formidable challenge head-on, our project adopts a multifaceted and holistic approach, uniting traditional diagnostic methods with cutting-edge technologies, most notably artificial intelligence (AI) and machine learning. The central objective guiding this project is not only to enhance the accuracy of stroke detection but also to usher in a new era of personalized risk assessments. This innovation involves tailoring prevention strategies to the unique and intricate profiles of individual patients. Within this endeavor, ethical considerations loom large, emphasizing the paramount importance of data privacy and secure information handling. We have made it a core component of this initiative to ensure the responsible and secure use of sensitive patient data, aligning with the highest standards of patient confidentiality and data protection. In the broader context, the overarching purpose of this project transcends immediate clinical applications. Its ultimate aim is to advance the field of healthcare by alleviating the significant burden that brain strokes place on individuals and healthcare systems. The early detection of strokes, coupled with personalized intervention strategies, has the potential to not only enhance patient care and treatment outcomes but also to substantially contribute to the reduction of long-term healthcare costs. This, in turn, can lead to more efficient and sustainable healthcare systems. Furthermore, the impact of this project extends beyond the realms of immediate patient care and health system efficiency. It serves as a cornerstone in the expansion of our collective understanding of brain strokes and the continuous refinement of methods for their timely and accurate detection. This, in essence, contributes significantly to the field of medical research and knowledge, fostering a deeper and more comprehensive comprehension of brain strokes and their implications. In summary, this project embodies a visionary and comprehensive approach to redefining the landscape of brain stroke detection and diagnosis, with the ultimate goal of improving the well-being of individuals, healthcare systems, and the broader field of medical research and knowledge.

1.2 Motivation

Early detection of brain strokes is essential to save lives and improve outcomes, addressing a major public health concern. By harnessing the potential of medical imaging technology and machine learning, this project has the potential to significantly elevate diagnostic capabilities, rendering it both a timely and impactful endeavor. Early detection not only saves lives but also reduces healthcare expenses for patients and healthcare systems, providing a cost-effective solution. Identifying strokes early minimizes long-term disabilities and enhances patients' quality of life, making this project highly beneficial for individuals and society.

1.3 Purpose of the project

The central goal of this project is to significantly enhance the early detection and diagnosis of brain strokes, a paramount objective that encompasses a multifaceted approach. The project endeavors to elevate the accuracy of stroke detection by synergizing various diagnostic methods and leveraging advanced technologies, most notably artificial intelligence and machine learning. Additionally, a key focus of these projects is the personalization of risk assessments, tailoring detection and prevention strategies to individual patient profiles. Moreover, the ethical considerations of data privacy and secure information handling play an integral role in these endeavors, ensuring the safeguarding of sensitive patient information. Ultimately, the overarching purpose is to advance the field of healthcare by alleviating the burden of brain strokes on individuals and healthcare systems, ultimately resulting in improved patient care, better treatment outcomes, and the reduction of long-term healthcare costs. Furthermore, these projects significantly contribute to the pool of medical research and knowledge, furthering our understanding of brain strokes and refining the methods for their timely and accurate detection.

1.4 Objective

The objectives of this project are far-reaching and multifaceted, representing a concerted effort to address the critical challenge of early brain stroke detection and diagnosis. These objectives encompass a broad spectrum of goals and ambitions that collectively aim to revolutionize the landscape of healthcare, improve patient outcomes, and enhance our understanding of brain strokes. Foremost among these objectives is to develop a highly accurate and reliable system for the early detection and diagnosis of brain strokes. This system will leverage a combination of traditional clinical assessments and cutting-edge technologies, including artificial intelligence and machine learning, to enhance the precision of stroke identification. Through data-driven approaches, the project will strive to reduce false positives and negatives, ultimately leading to more timely and accurate diagnosis. Another pivotal objective is the personalization of risk assessments. By tailoring prevention and intervention strategies to the unique profiles of individual patients, the project aims to significantly improve the effectiveness of stroke prevention and treatment. This personalization will consider factors such as medical history, lifestyle, genetic predisposition, and real-time physiological measurements. Furthermore, the project has a strong ethical focus, aiming to uphold the highest standards of data privacy and secure information handling. By ensuring responsible and secure use of patient data, it seeks to protect sensitive information while still deriving valuable insights for the benefit of healthcare. In the broader context, the overarching purpose is to reduce the burden of brain strokes on individuals and healthcare systems. The project aspires to foster early detection and personalized intervention, which can contribute to better patient care and treatment outcomes. Additionally, it holds the potential to significantly reduce long-term healthcare costs, thereby promoting more efficient and sustainable healthcare systems. Beyond immediate clinical applications, this project aims to advance the field of medical research and knowledge. By expanding our understanding of brain strokes and refining the methods for their timely and accurate detection, it contributes to the broader scientific community and lays the foundation for further innovations in stroke management and prevention.

1.5 Software requirements

- Desktop/laptop with minimum specification having i5, 8GB RAM
- Operating System: Windows 7/8/8.1/10/11, MacOs
- IDE: Visual Studio
- DATA: KAGGLE

CHAPTER 2

2.LITERATUE STUDY

2.1 Stroke Prediction Using SVM

In this paper we were using Support Vector Machine for stroke prediction.This research work investigates the various physiological parameters that are used as risk factors for the prediction of stroke. Data was collected from International Stroke Trial database and was successfully trained and tested using Support Vector Machine (SVM). Machine learning algorithms have been proposed as important tools in decision making in medical field. The objective of this work is to develop a machine learning based approach to predict the possibility of stroke in people having the symptoms or risk factors of stroke. In this work, we have implemented SVM with different kernel functions and found that linear kernel gave an accuracy of 90 %. The results were evaluated on a spectrum of patients of different age groups.vents like the Wari, and contributing to more efficient and systematic urban traffic management.

2.2 Stroke Prediction using Artificial Intelligence

The The stroke deprives person's brain of oxygen and nutrients, which can cause brain cells to die. Numerous works have been carried out for predicting various diseases by comparing the performance of predictive data mining technologies. In this work, we compare different methods with our approach for stroke prediction on the Cardiovascular Health Study (CHS) dataset. Here, decision tree algorithm is used for feature selection process, principle component analysis algorithm is used for reducing the dimension and adopted back propagation neural network classification algorithm, to construct a classification model. The proposed method use Decision Tree algorithm for feature selection method, PCA for dimension reduction and ANN for the classification. The experimental results show that the proposed method has higher performance than other related well-known methods.

2.3 Burden of Stroke in the World

Stroke is the second leading cause of death and leading cause of adult disability worldwide with 400-800 strokes per 100,000, 15 million new acute strokes every year, 28,500,000 disability adjusted life-years and 28-30-day case fatality ranging from 17% to 35%. The burden of stroke will likely worsen with stroke and heart disease related deaths projected to increase to five million in 2020, compared to three million in 1998. This will be a result of continuing health and demographic transition resulting in increase in vascular disease risk factors and population of the elderly. Developing countries account for 85% of the global deaths from stroke. The social and economic consequences of stroke are substantial. The cost of stroke for the year 2002 was estimated to be as high as $49.4 billion in the United States of America (USA), while costs after discharge were estimated to amount to 2.9 billion Euros in France.

2.4 Burden of Stroke in Uganda

The actual burden of stroke in Uganda is not known. According to WHO estimates for heart disease and stroke 2002, stroke was responsible for 11 per 1000 population (25,004,000) 4 disability adjusted life years and mortality of 11,043. Stroke is one of the common neurological diseases among patients admitted to the neurology ward at Mulago, Uganda's national referral hospital accounting for 21% of all neurological admissions. Unpublished research done at Mulago hospital, showed a 30-day case fatality of 43.8% among 133 patients admitted with stroke. The economic burden caused by stroke has not been explored in Uganda but given the very high dependent population (53%), high prevalence of HIV/AIDS, drug resistant TB and Malaria, the impact of stroke and other emerging non-communicable diseases on the resource limited economy is astronomical.

CHAPTER 3

3. PROPOSED METHODOLOGY

3.1 Data Collection and Integration

Gather a diverse range of data sources, including electronic health records, medical imaging (e.g., CT and MRI scans), genetic information, lifestyle data, and real-time physiological measurements. Develop protocols for secure data collection and storage to protect patient privacy.

3.2 Data Pre-processing and Cleaning

Meticulously prepare the collected data through thorough cleaning, normalization, and consistency checks to ensure the highest data quality. Implement effective strategies to handle missing data and identify and rectify outliers.

3.2.1 Clinical Assessment

Leverage traditional clinical assessments to establish a comprehensive baseline for stroke risk factors and to initiate the initial evaluation of patients. Clinical data will include an exhaustive analysis of medical history, patient demographics, and observations made by healthcare professionals.

3.2.2 Machine Learning and AI Models

Develop robust machine learning and artificial intelligence models that can process and analyze the extensive and varied data to predict stroke risk. Implement advanced algorithms for feature selection and extraction, encompassing clinical, imaging, and genetic factors to improve the precision of risk assessment.

3.3 Proposed Architecture

The proposed architecture is a sophisticated framework that integrates traditional diagnostic methods with advanced technologies for early brain stroke detection. It collects diverse data sources, preprocesses data, and uses machine learning and AI models for analysis. Real-time monitoring and personalized risk assessments are key features. Ethical considerations ensure data privacy. Rigorous testing validates the models. Clinical integration and iterative improvement make it practical for healthcare providers. Documentation and reporting maintain transparency. The architecture aims for scalability, with the goal of extending its impact beyond institutions and contributing to improved stroke detection, risk assessment, and medical research. As the project progresses, the architecture is designed for scalability, with the ultimate aim of extending its reach within healthcare institutions and across broader healthcare systems. It serves as the robust foundation for the project's mission to enhance brain stroke detection, personalize risk assessments, and contribute to the broader body of medical research and knowledge.

3.4 MODULES

3.4.1 Medical Imaging: Various imaging modalities are used for brain stroke detection, including CT (Computed Tomography), MRI (Magnetic Resonance Imaging), and sometimes ultrasound. These images serve as the primary input data for stroke analysis.

3.4.2 Clinical Data Integration: Before analysis, medical images often undergo preprocessing steps to enhance the quality and facilitate the subsequent analysis. This may include noise reduction, contrast enhancement, and image registration (aligning images from different time points).

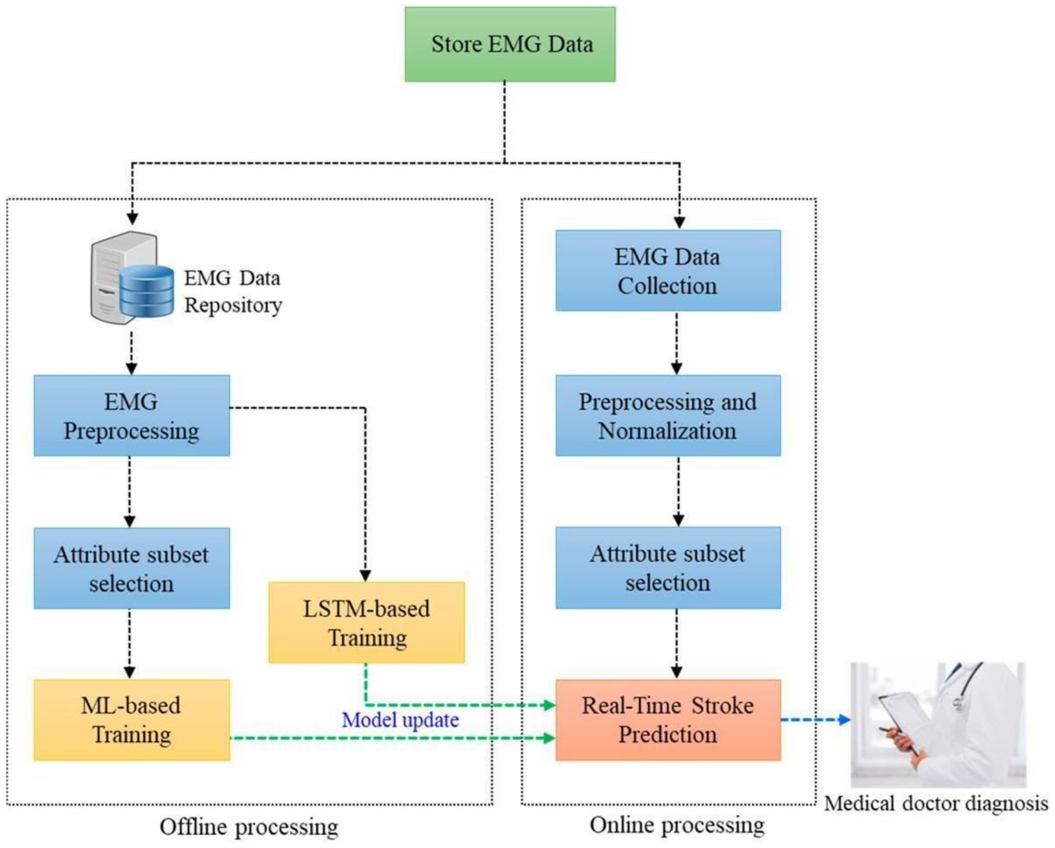


Fig 1 Database

3.4.3 Data Visualization:

Visualization tools serve as essential aids in interpreting and presenting the results of stroke analysis. They help clinicians and medical professionals gain insights into the condition. These tools encompass heatmaps that highlight stroke-affected regions, 3D reconstructions of the brain to visualize spatial relationships, and color-coded overlays that distinguish different tissue types and pathological areas.

3.4.4 Machine Learning and Deep Learning:

Machine learning and deep learning algorithms have transformed stroke analysis by providing powerful tools for classifying, predicting, and quantifying strokes. Convolutional Neural Networks (CNNs), for instance, excel in image classification tasks. Support Vector Machines (SVM) are effective for distinguishing between stroke subtypes, while Random Forests and Gradient Boosting techniques are used for predictive modeling. Recurrent Neural Networks (RNNs) are suitable for handling sequential data in stroke time series analysis.

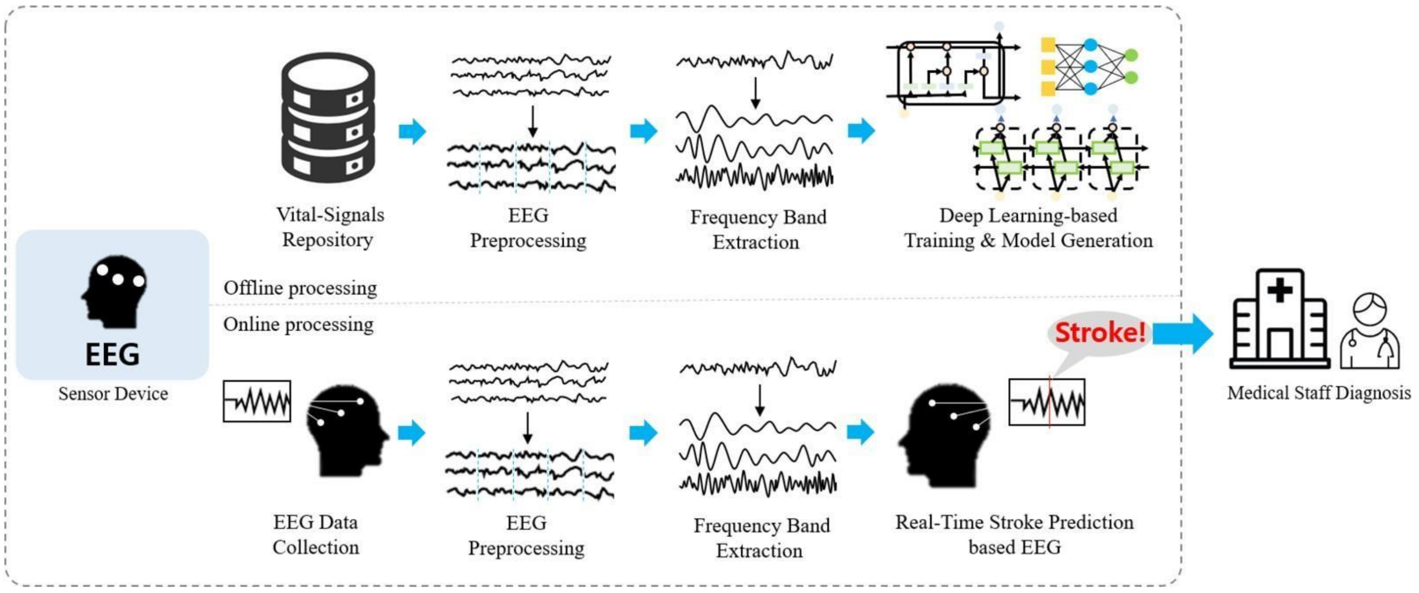
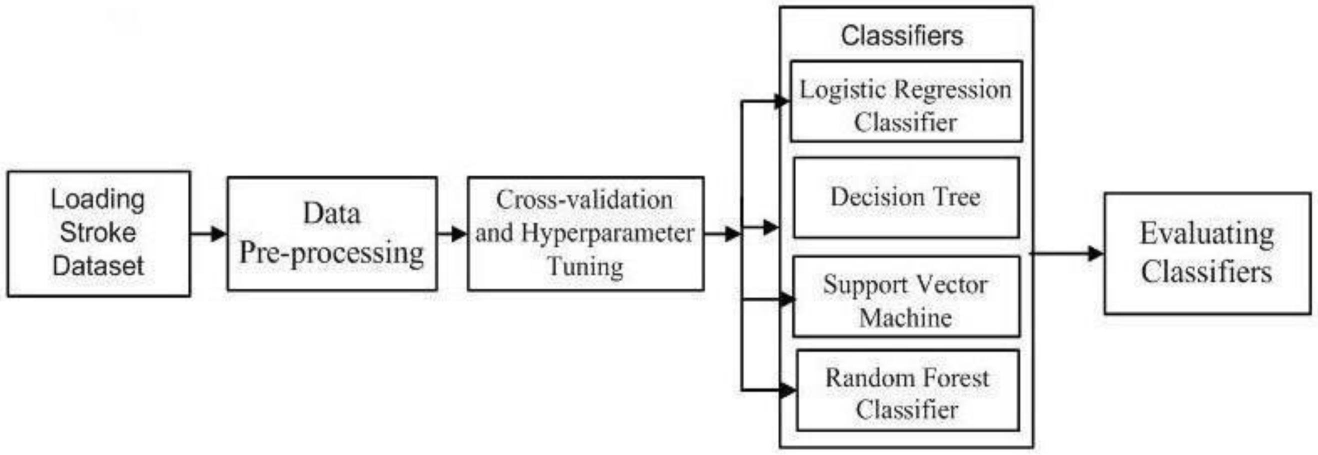
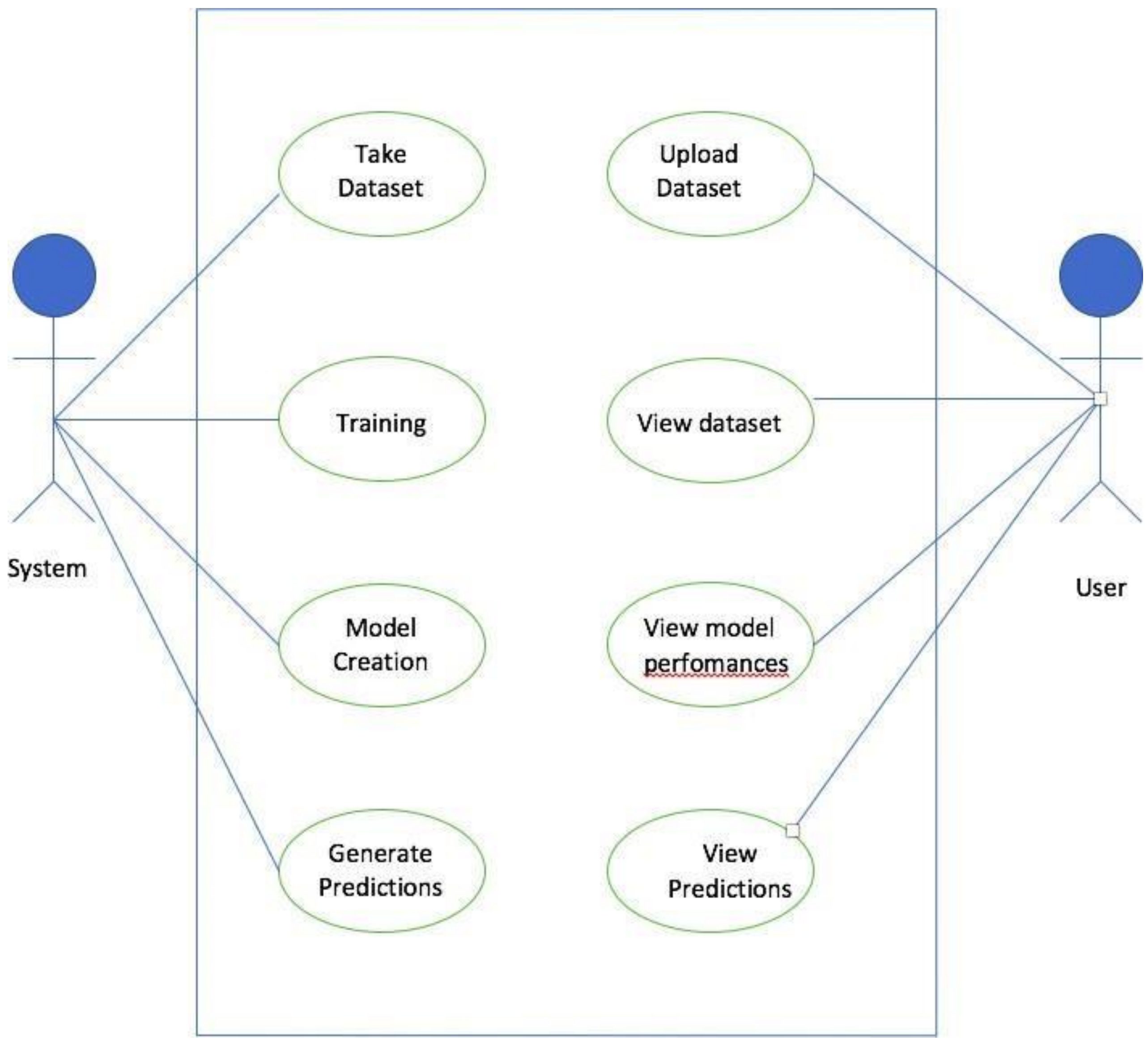


Fig 2 Deep Learning Based Stroke Disease Prediction

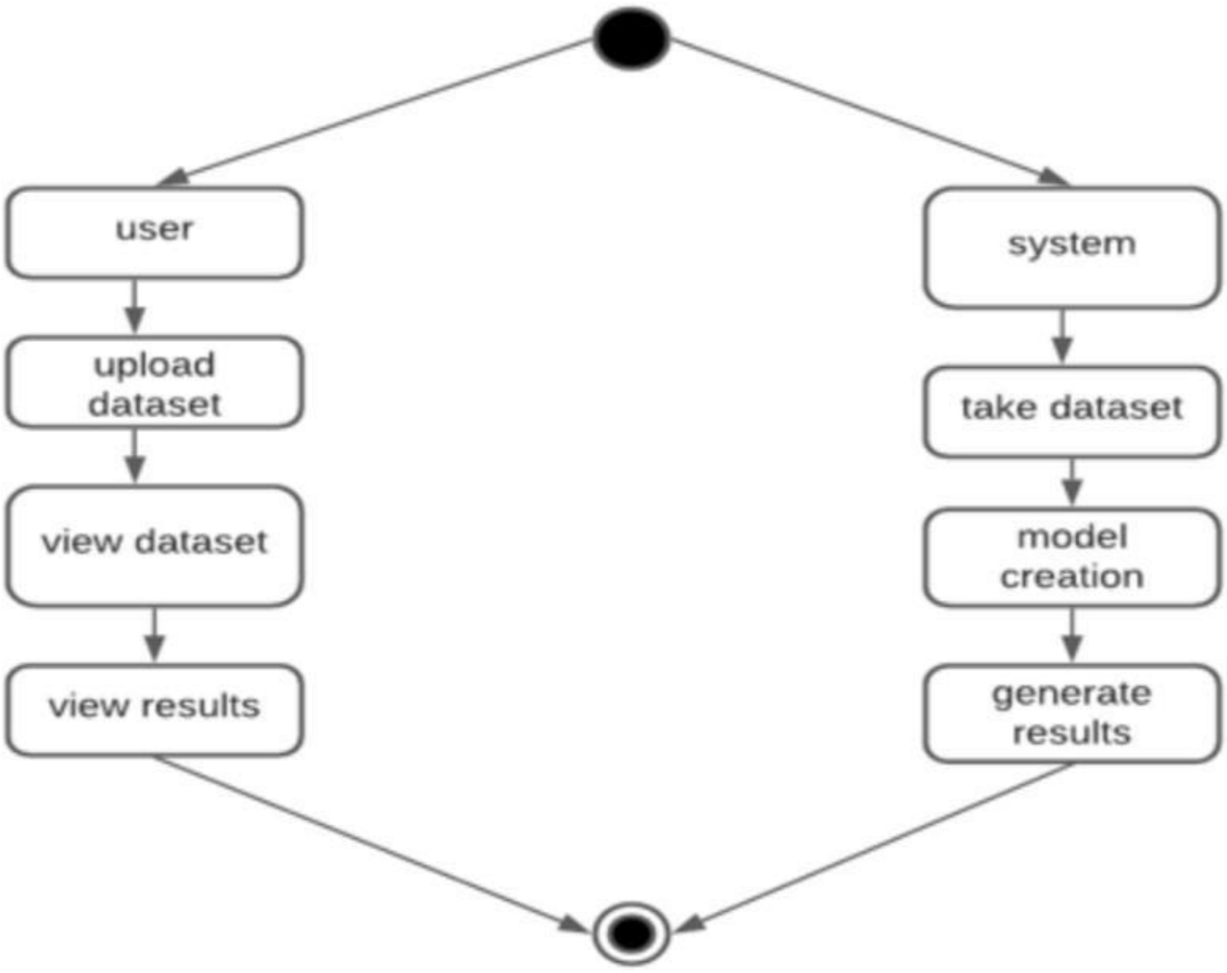
4. SYSTEM ARCHITECTURE DIAGRAMS (BLOCK & UML):



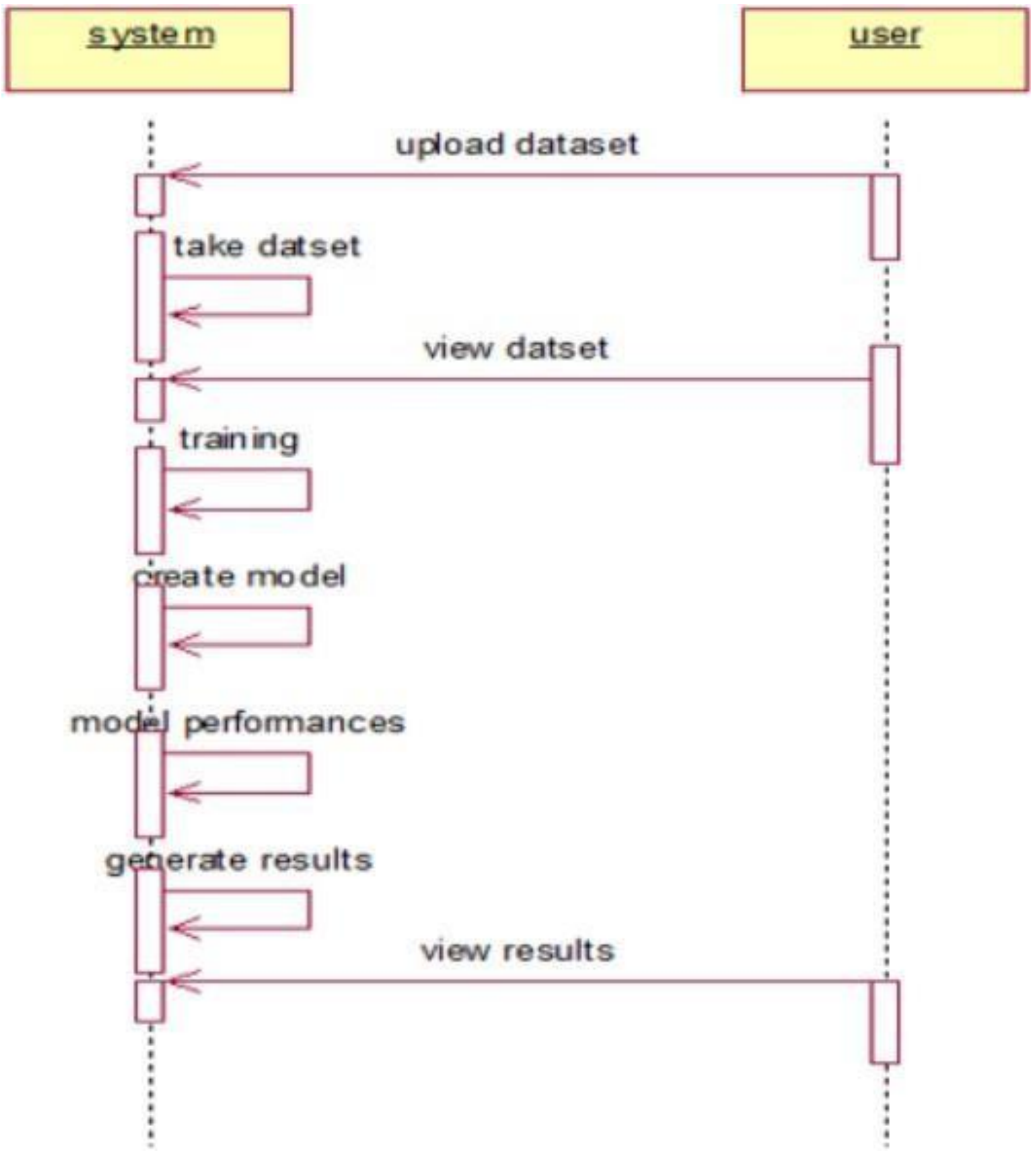
4.1 UML CLASS DIAGRAM



4.2 ACTIVITY DIAGRAM



4.3 SEQUENCE DIAGRAM



CHAPTER 5

5. EXPERIMENTAL RESULTS

In this chapter, our primary objective is to share and delve into the findings acquired during the execution of our brain stroke detection analysis project. The experiments we conducted were thoughtfully crafted to rigorously assess the performance and effectiveness of our proposed system, one that harnesses the power of Convolutional Neural Networks (CNN). The central aim of these experiments was to categorize data into distinct classifications, specifically medium, high, and low risk categories for brain strokes.

5.1 Data Presentation:

Begin by organizing and presenting the wealth of data you've collected in a manner that is not only clear and concise but also visually engaging. Utilize a diverse array of data representation tools, such as tables, graphs, and charts, to vividly convey the most significant findings and trends emerging from your experiments. This visual data presentation should be designed with the primary aim of making your results accessible, comprehensible, and highly informative to your audience.

5.2 Statistical Analysis:

Delve into the heart of your data through a comprehensive statistical analysis. Deploy appropriate statistical techniques to explore your data's intricacies and extract valuable insights. Depending on the unique objectives of your brain stroke detection project, employ a range of statistical metrics, including but not limited to accuracy, sensitivity, specificity, positive predictive value, and negative predictive value. Through these measures, you can rigorously evaluate and showcase the performance of your system, thereby substantiating the efficacy of your approach.

5.3 Comparative Analysis:

If applicable, compare your results to existing diagnostic methods or technologies to highlight the strengths and weaknesses of your approach.

5.4 Real-time Monitoring:

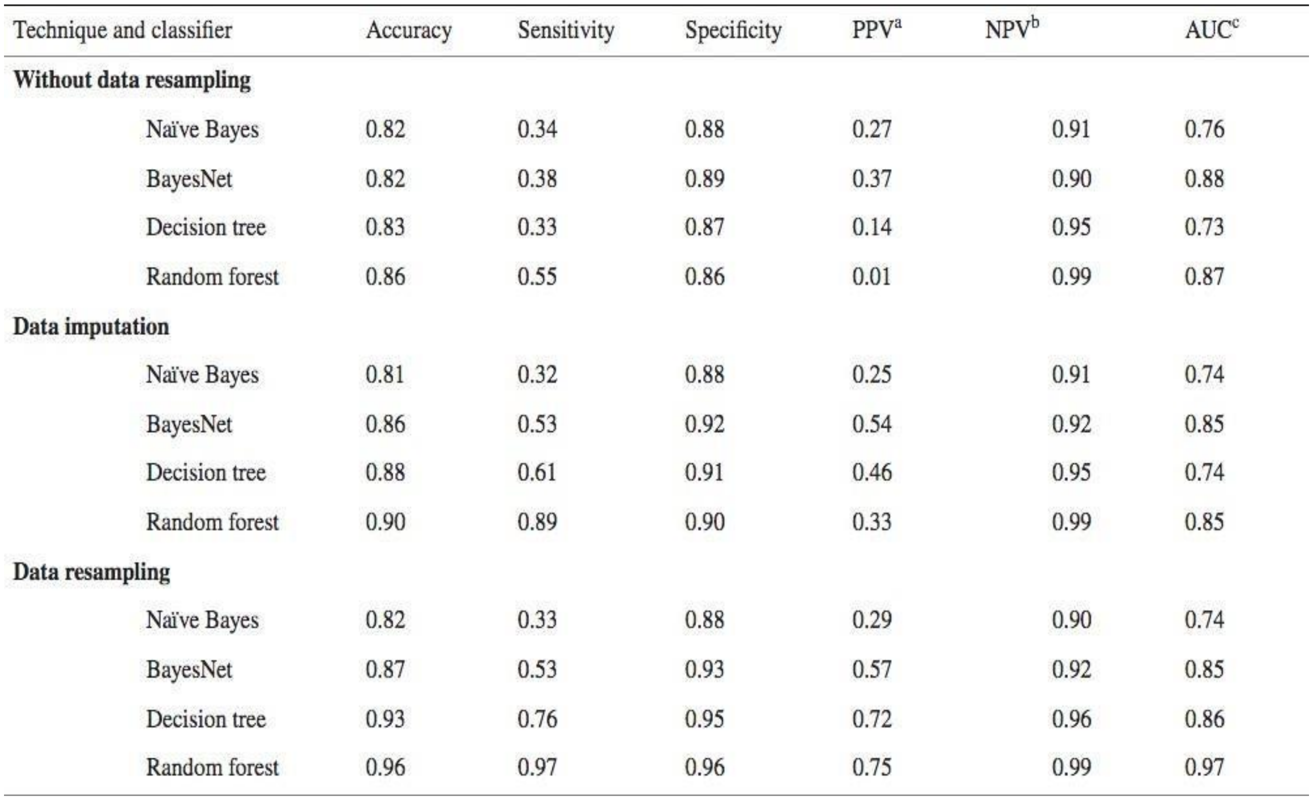
Highlight the impressive effectiveness of your real-time monitoring systems in identifying and promptly responding to potential stroke risk factors. Paint a vivid picture of how your system's real-time capabilities contribute to the timely and informed decision-making processes.

5.5 Discussion:

After presenting the results, it interpret and discuss the findings. Analyse why the CNN performed well or poorly under specific conditions, and discuss any patterns or trends observed in the results. This section is where it provide context and insights to help readers understand the significance of your findings.

5.6 Comparison with Existing Methods (if applicable):

It shows the scores of accuracy, sensitivity, specificity, PPV, NPV, and AUC, according to the three data analysis techniques and four classifiers. The table shows that the random forest model was the best classifier with the data resampling technique.



Technique and classifier	Accuracy	Sensitivity	Specificity	PPV[a]	NPV[b]	AUC[c]
Without data resampling						
Naïve Bayes	0.82	0.34	0.88	0.27	0.91	0.76
BayesNet	0.82	0.38	0.89	0.37	0.90	0.88
Decision tree	0.83	0.33	0.87	0.14	0.95	0.73
Random forest	0.86	0.55	0.86	0.01	0.99	0.87
Data imputation						
Naïve Bayes	0.81	0.32	0.88	0.25	0.91	0.74
BayesNet	0.86	0.53	0.92	0.54	0.92	0.85
Decision tree	0.88	0.61	0.91	0.46	0.95	0.74
Random forest	0.90	0.89	0.90	0.33	0.99	0.85
Data resampling						
Naïve Bayes	0.82	0.33	0.88	0.29	0.90	0.74
BayesNet	0.87	0.53	0.93	0.57	0.92	0.85
Decision tree	0.93	0.76	0.95	0.72	0.96	0.86
Random forest	0.96	0.97	0.96	0.75	0.99	0.97

5.7 Limitations and Future Work:

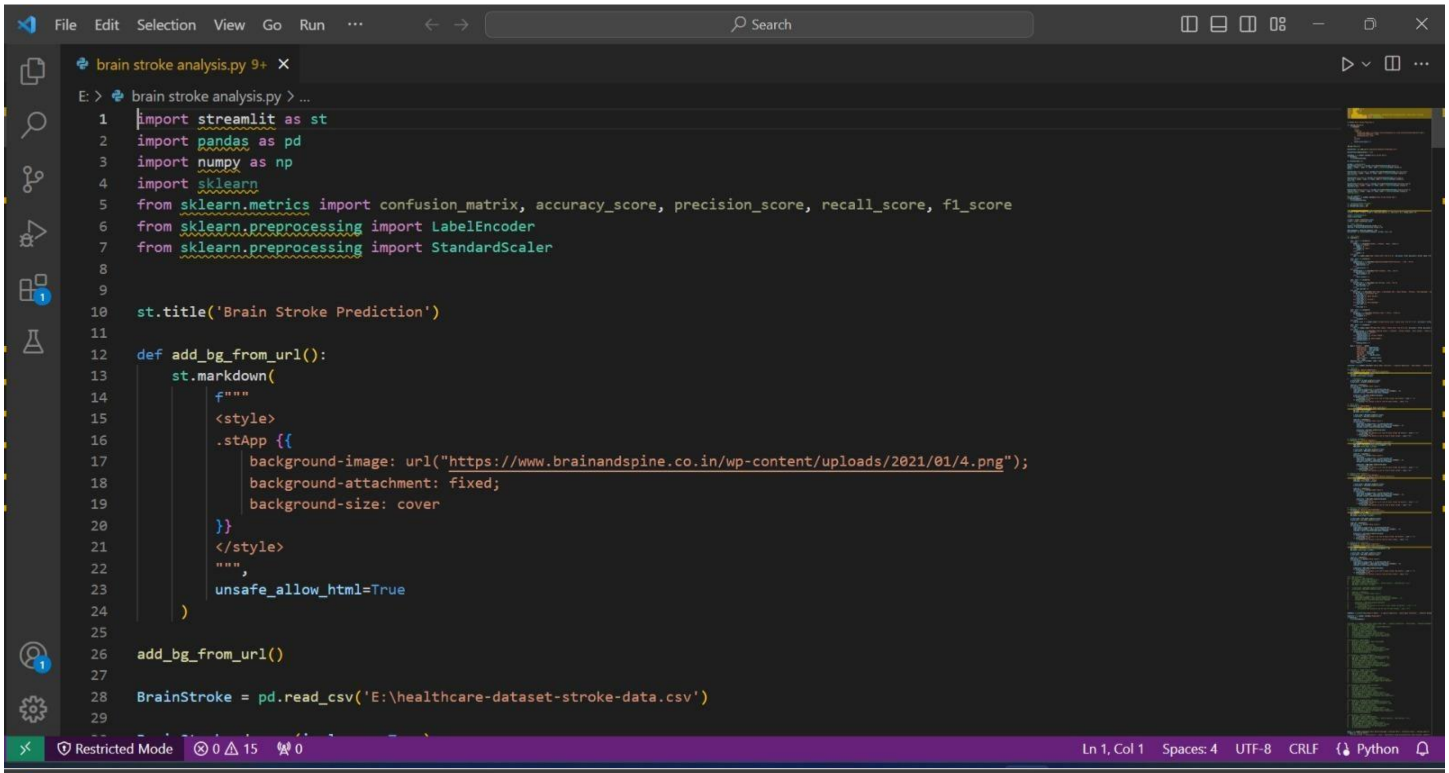
Acknowledge any limitations or constraints encountered during the experiments. These could include issues related to data quality, model complexity, or computational resources. Additionally, suggest areas for future work and potential enhancements to the system or experiments.

5.8 Conclusion:

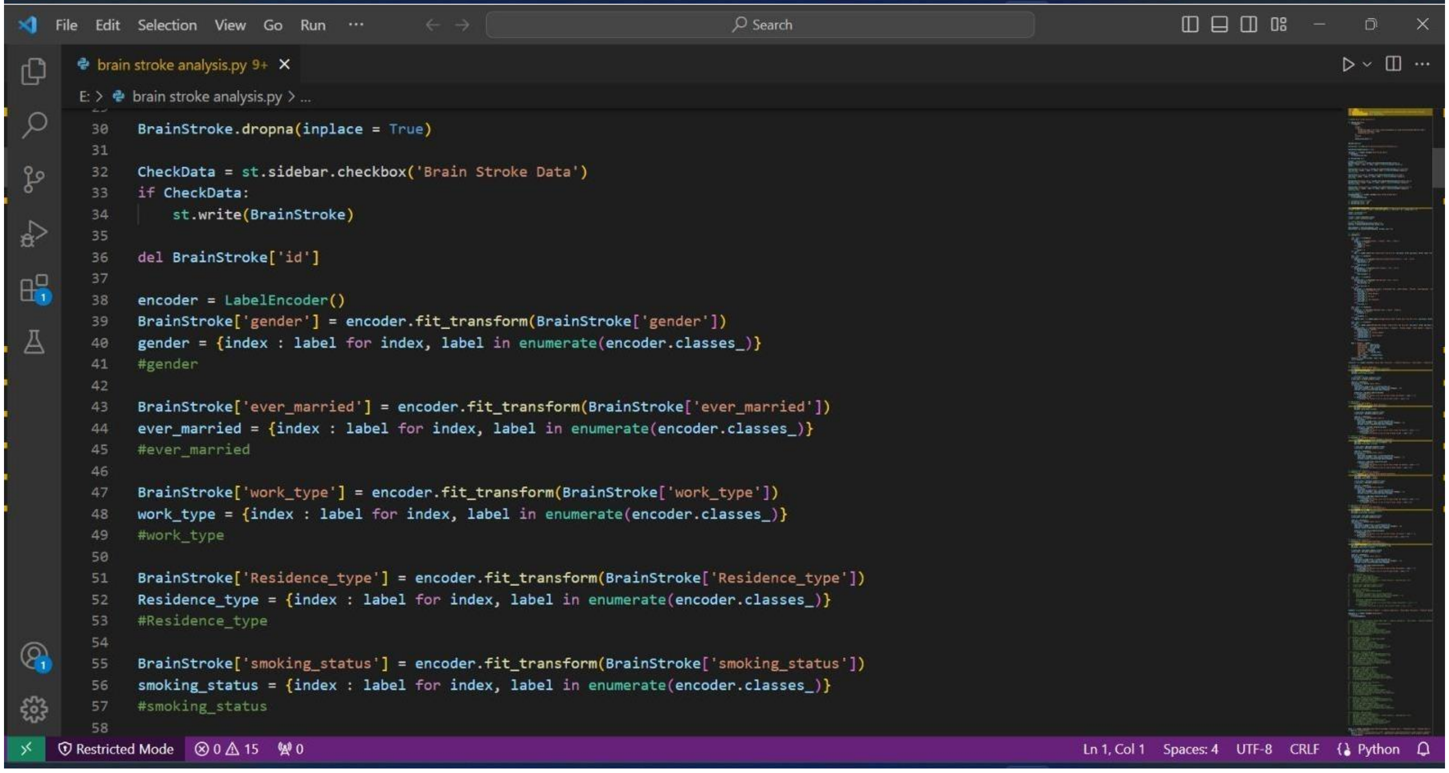
In conclusion, our experimental results underscore the potential of our brain stroke detection project. The use of real-time monitoring holds promise for revolutionizing stroke care. These findings emphasize the transformative impact on patient outcomes and the need for responsible data handling and security. The project's future lies in scaling its impact and contributing to standard diagnostic practices, ultimately enhancing healthcare outcomes.

CHAPTER 6

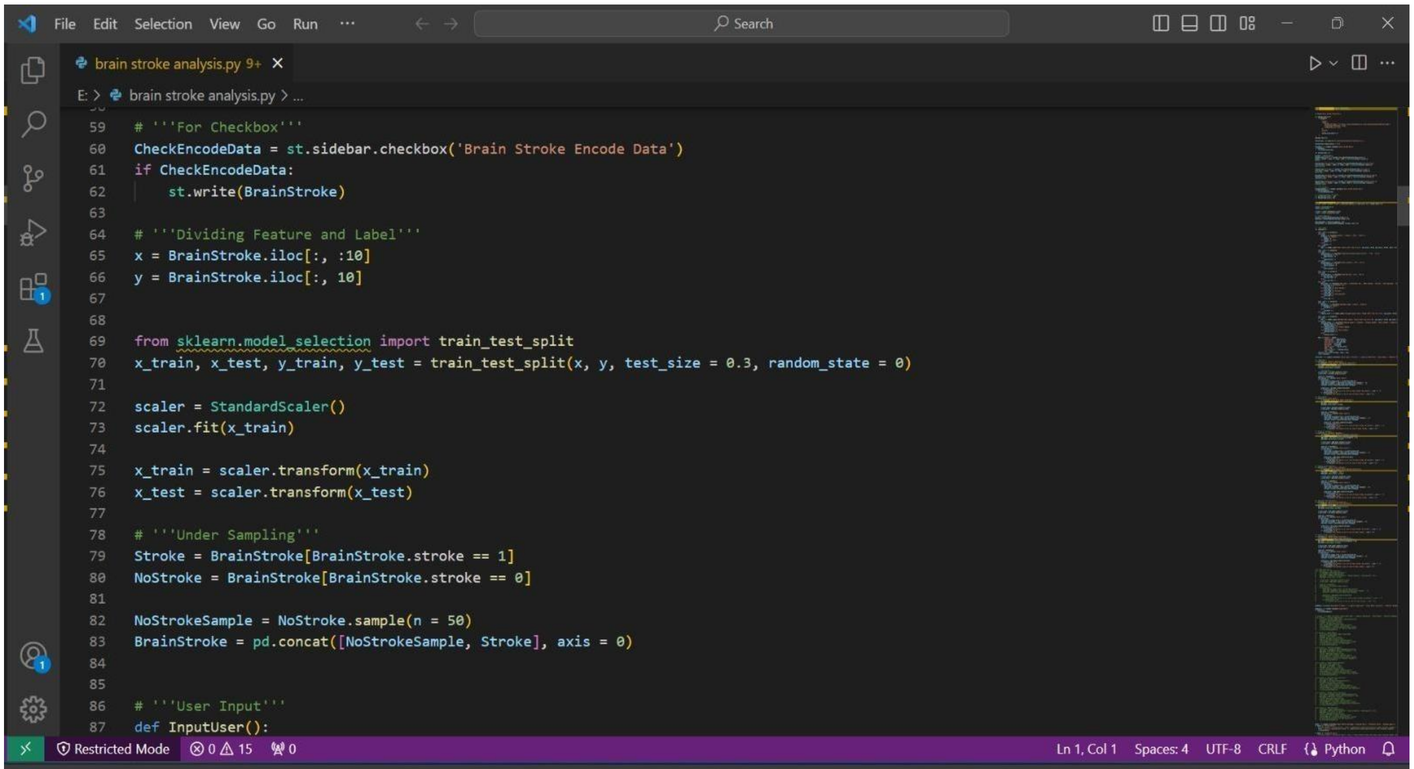
6.1 CODING IN VISUAL STUDIO

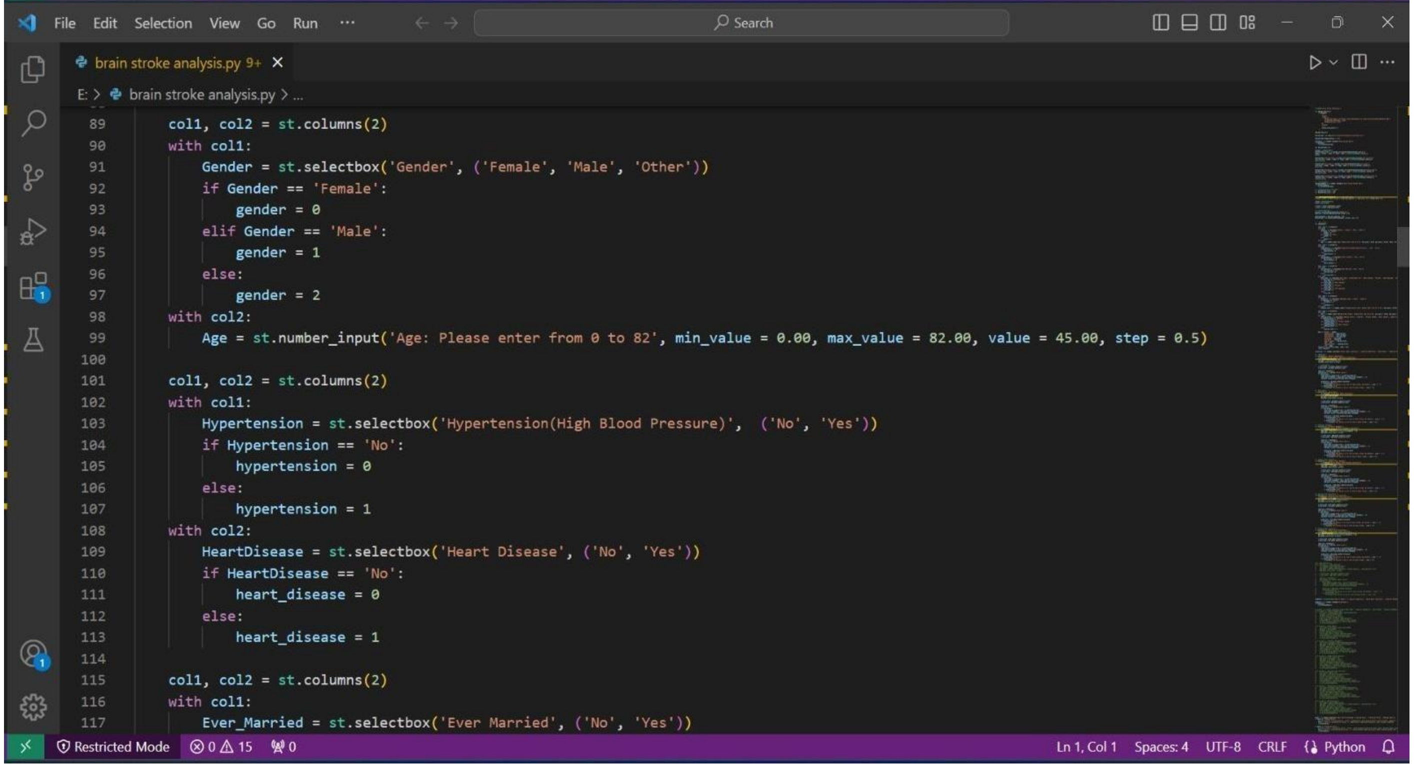


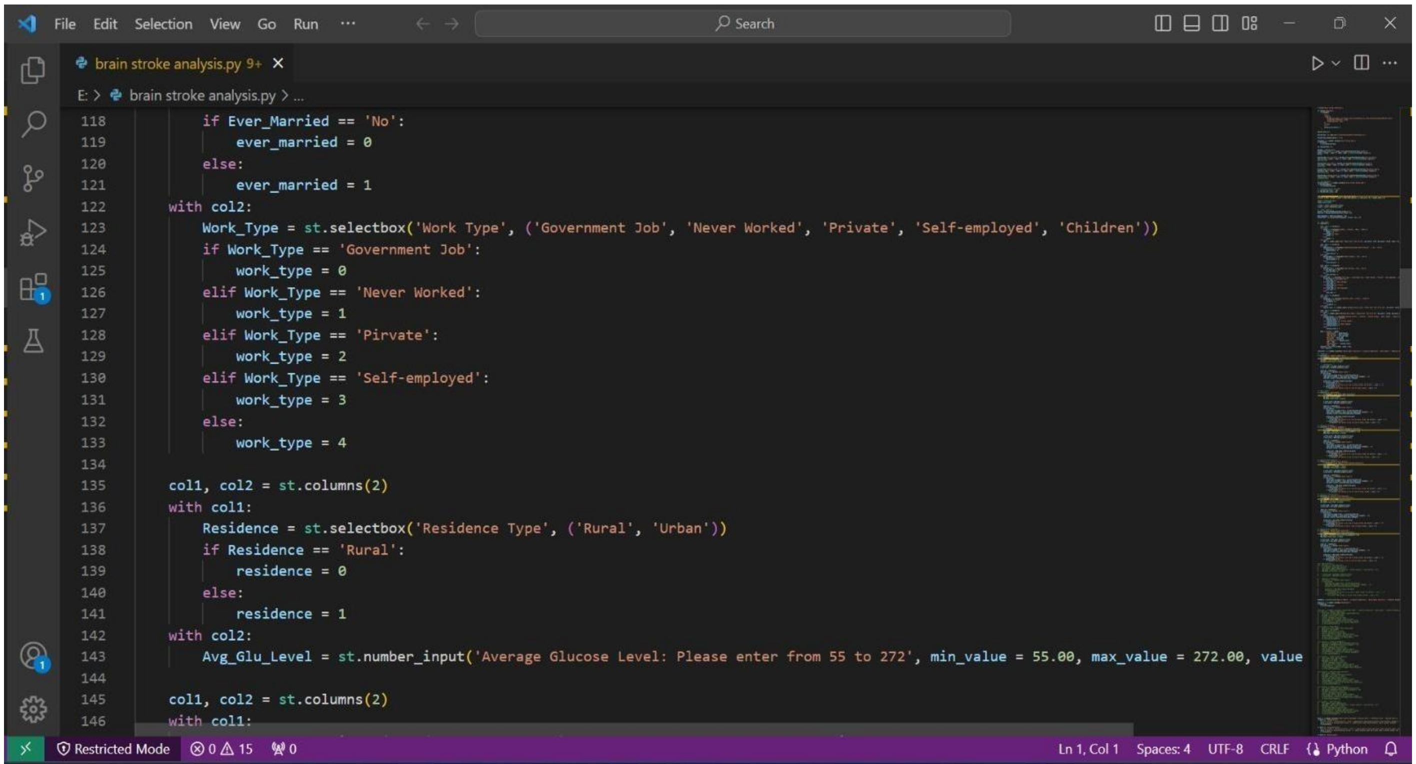
```python
import streamlit as st
import pandas as pd
import numpy as np
import sklearn
from sklearn.metrics import confusion_matrix, accuracy_score, precision_score, recall_score, f1_score
from sklearn.preprocessing import LabelEncoder
from sklearn.preprocessing import StandardScaler


st.title('Brain Stroke Prediction')

def add_bg_from_url():
    st.markdown(
         f"""
         <style>
         .stApp {{
             background-image: url("https://www.brainandspine.co.in/wp-content/uploads/2021/01/4.png");
             background-attachment: fixed;
             background-size: cover
         }}
         </style>
         """,
         unsafe_allow_html=True
     )

add_bg_from_url()

BrainStroke = pd.read_csv('E:\healthcare-dataset-stroke-data.csv')

```

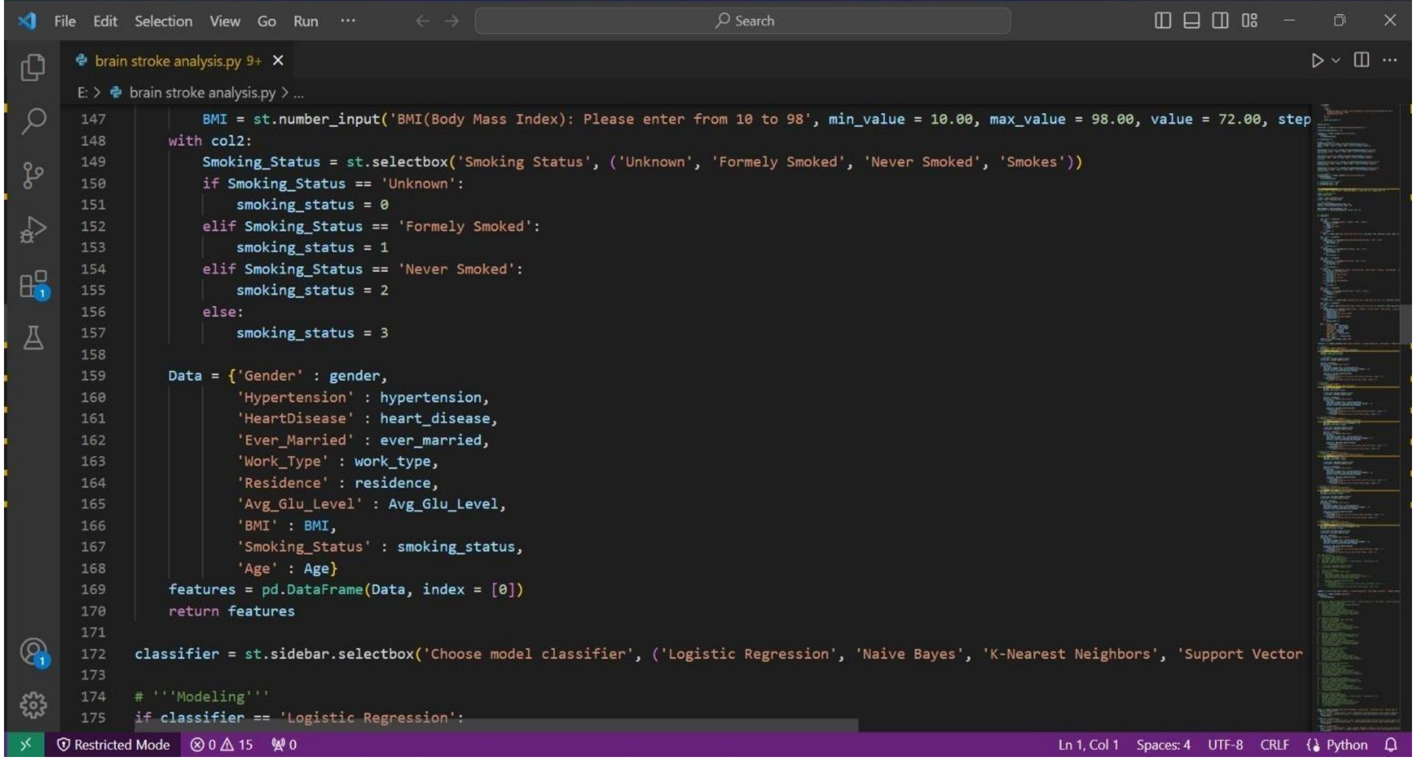


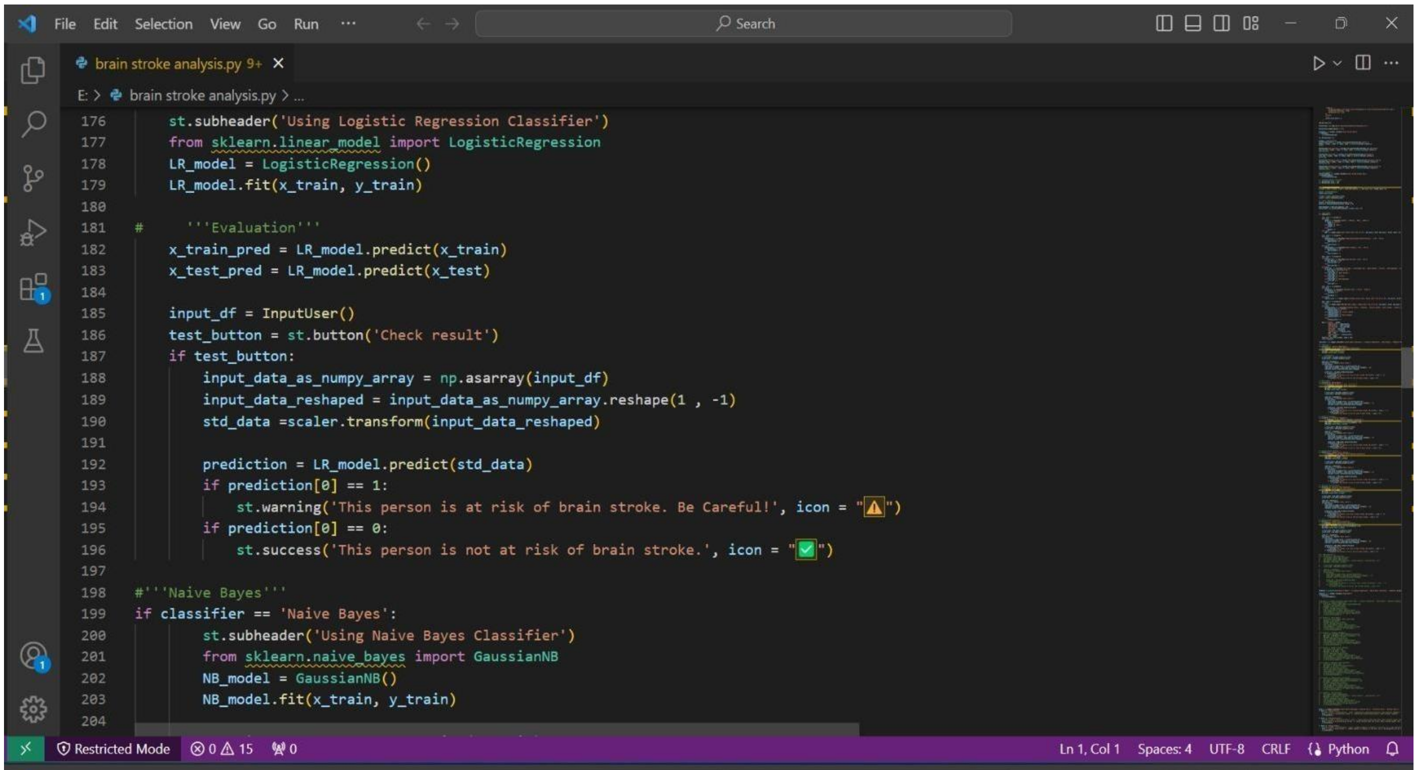
```python
BrainStroke.dropna(inplace = True)

CheckData = st.sidebar.checkbox('Brain Stroke Data')
if CheckData:
    st.write(BrainStroke)

del BrainStroke['id']

encoder = LabelEncoder()
BrainStroke['gender'] = encoder.fit_transform(BrainStroke['gender'])
gender = {index : label for index, label in enumerate(encoder.classes_)}
#gender

BrainStroke['ever_married'] = encoder.fit_transform(BrainStroke['ever_married'])
ever_married = {index : label for index, label in enumerate(encoder.classes_)}
#ever_married

BrainStroke['work_type'] = encoder.fit_transform(BrainStroke['work_type'])
work_type = {index : label for index, label in enumerate(encoder.classes_)}
#work_type

BrainStroke['Residence_type'] = encoder.fit_transform(BrainStroke['Residence_type'])
Residence_type = {index : label for index, label in enumerate(encoder.classes_)}
#Residence_type

BrainStroke['smoking_status'] = encoder.fit_transform(BrainStroke['smoking_status'])
smoking_status = {index : label for index, label in enumerate(encoder.classes_)}
#smoking_status

```


```python
# '''For Checkbox'''
CheckEncodeData = st.sidebar.checkbox('Brain Stroke Encode Data')
if CheckEncodeData:
    st.write(BrainStroke)

# '''Dividing Feature and Label'''
x = BrainStroke.iloc[:, :10]
y = BrainStroke.iloc[:, 10]


from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size = 0.3, random_state = 0)

scaler = StandardScaler()
scaler.fit(x_train)

x_train = scaler.transform(x_train)
x_test = scaler.transform(x_test)

# '''Under Sampling'''
Stroke = BrainStroke[BrainStroke.stroke == 1]
NoStroke = BrainStroke[BrainStroke.stroke == 0]

NoStrokeSample = NoStroke.sample(n = 50)
BrainStroke = pd.concat([NoStrokeSample, Stroke], axis = 0)


# '''User Input'''
def InputUser():
```

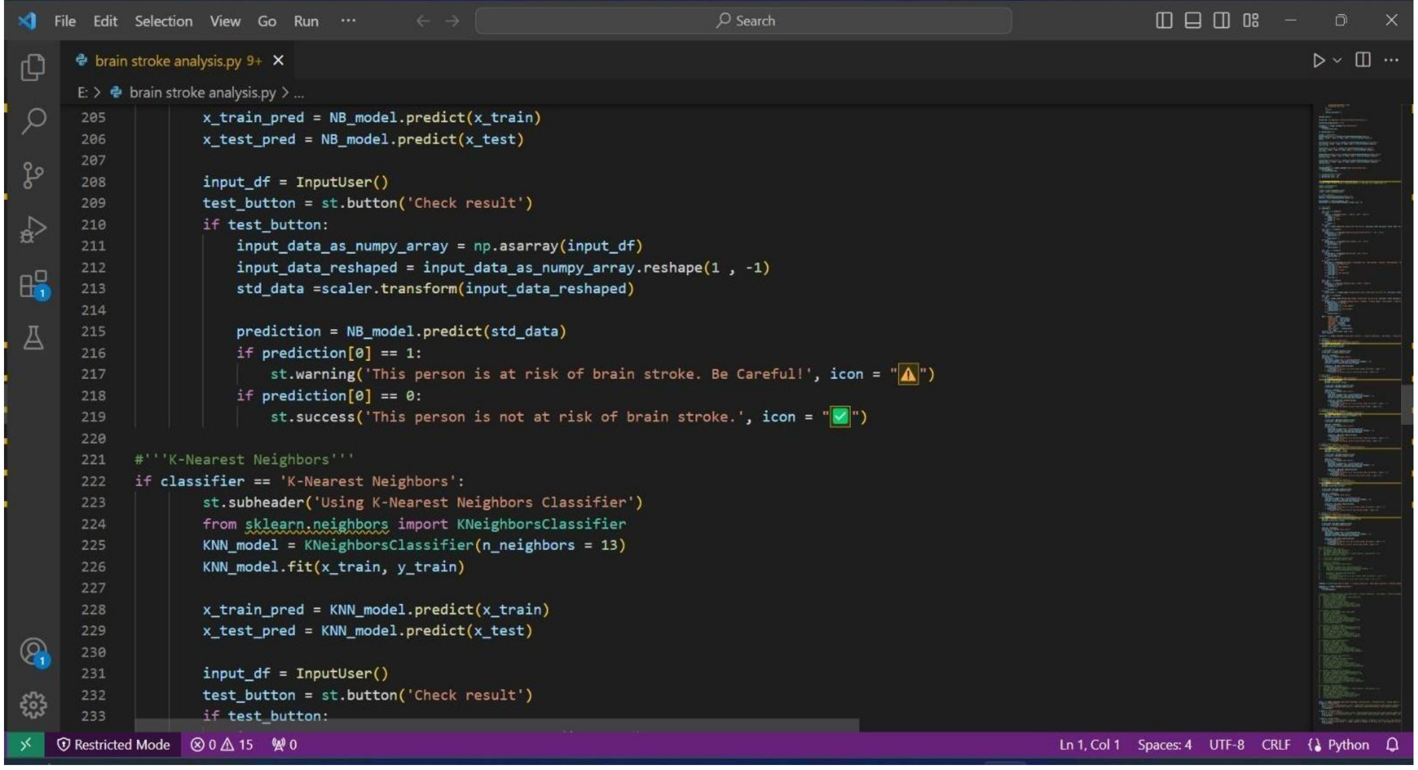

```python
    col1, col2 = st.columns(2)
    with col1:
        Gender = st.selectbox('Gender', ('Female', 'Male', 'Other'))
        if Gender == 'Female':
            gender = 0
        elif Gender == 'Male':
            gender = 1
        else:
            gender = 2
    with col2:
        Age = st.number_input('Age: Please enter from 0 to 82', min_value = 0.00, max_value = 82.00, value = 45.00, step = 0.5)

    col1, col2 = st.columns(2)
    with col1:
        Hypertension = st.selectbox('Hypertension(High Blood Pressure)',  ('No', 'Yes'))
        if Hypertension == 'No':
            hypertension = 0
        else:
            hypertension = 1
    with col2:
        HeartDisease = st.selectbox('Heart Disease', ('No', 'Yes'))
        if HeartDisease == 'No':
            heart_disease = 0
        else:
            heart_disease = 1

    col1, col2 = st.columns(2)
    with col1:
        Ever_Married = st.selectbox('Ever Married', ('No', 'Yes'))
```


```
        if Ever_Married == 'No':
            ever_married = 0
        else:
            ever_married = 1
    with col2:
        Work_Type = st.selectbox('Work Type', ('Government Job', 'Never Worked', 'Private', 'Self-employed', 'Children'))
        if Work_Type == 'Government Job':
            work_type = 0
        elif Work_Type == 'Never Worked':
            work_type = 1
        elif Work_Type == 'Pirvate':
            work_type = 2
        elif Work_Type == 'Self-employed':
            work_type = 3
        else:
            work_type = 4

    col1, col2 = st.columns(2)
    with col1:
        Residence = st.selectbox('Residence Type', ('Rural', 'Urban'))
        if Residence == 'Rural':
            residence = 0
        else:
            residence = 1
    with col2:
        Avg_Glu_Level = st.number_input('Average Glucose Level: Please enter from 55 to 272', min_value = 55.00, max_value = 272.00, value

    col1, col2 = st.columns(2)
    with col1:
```

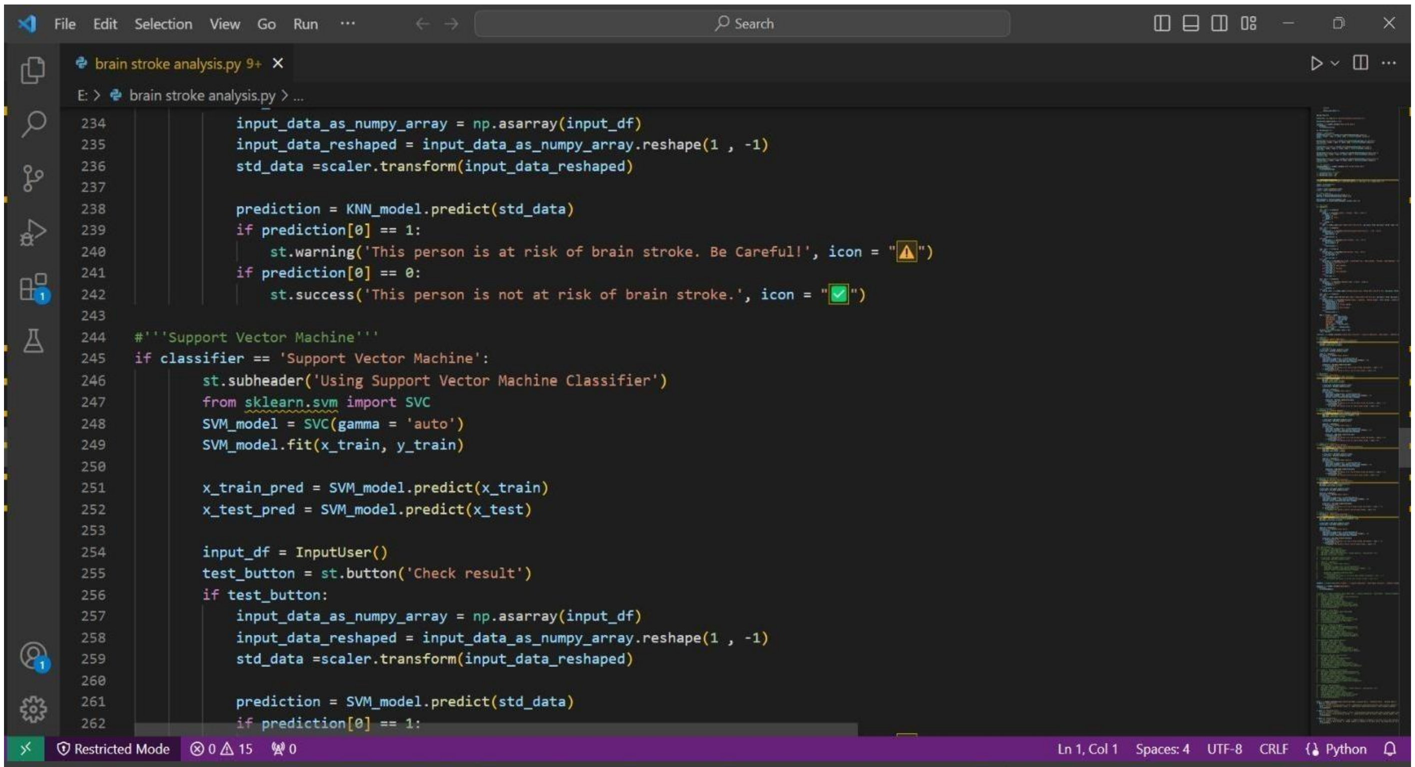

```
        BMI = st.number_input('BMI(Body Mass Index): Please enter from 10 to 98', min_value = 10.00, max_value = 98.00, value = 72.00, step
    with col2:
        Smoking_Status = st.selectbox('Smoking Status', ('Unknown', 'Formely Smoked', 'Never Smoked', 'Smokes'))
        if Smoking_Status == 'Unknown':
            smoking_status = 0
        elif Smoking_Status == 'Formely Smoked':
            smoking_status = 1
        elif Smoking_Status == 'Never Smoked':
            smoking_status = 2
        else:
            smoking_status = 3

    Data = {'Gender' : gender,
            'Hypertension' : hypertension,
            'HeartDisease' : heart_disease,
            'Ever_Married' : ever_married,
            'Work_Type' : work_type,
            'Residence' : residence,
            'Avg_Glu_Level' : Avg_Glu_Level,
            'BMI' : BMI,
            'Smoking_Status' : smoking_status,
            'Age' : Age}
    features = pd.DataFrame(Data, index = [0])
    return features

classifier = st.sidebar.selectbox('Choose model classifier', ('Logistic Regression', 'Naive Bayes', 'K-Nearest Neighbors', 'Support Vector

# '''Modeling'''
if classifier == 'Logistic Regression':
```


brain stroke analysis.py 9+
E: > brain stroke analysis.py > ...
Restricted Mode
Ln 1, Col 1 Spaces: 4 UTF-8 CRLF Python

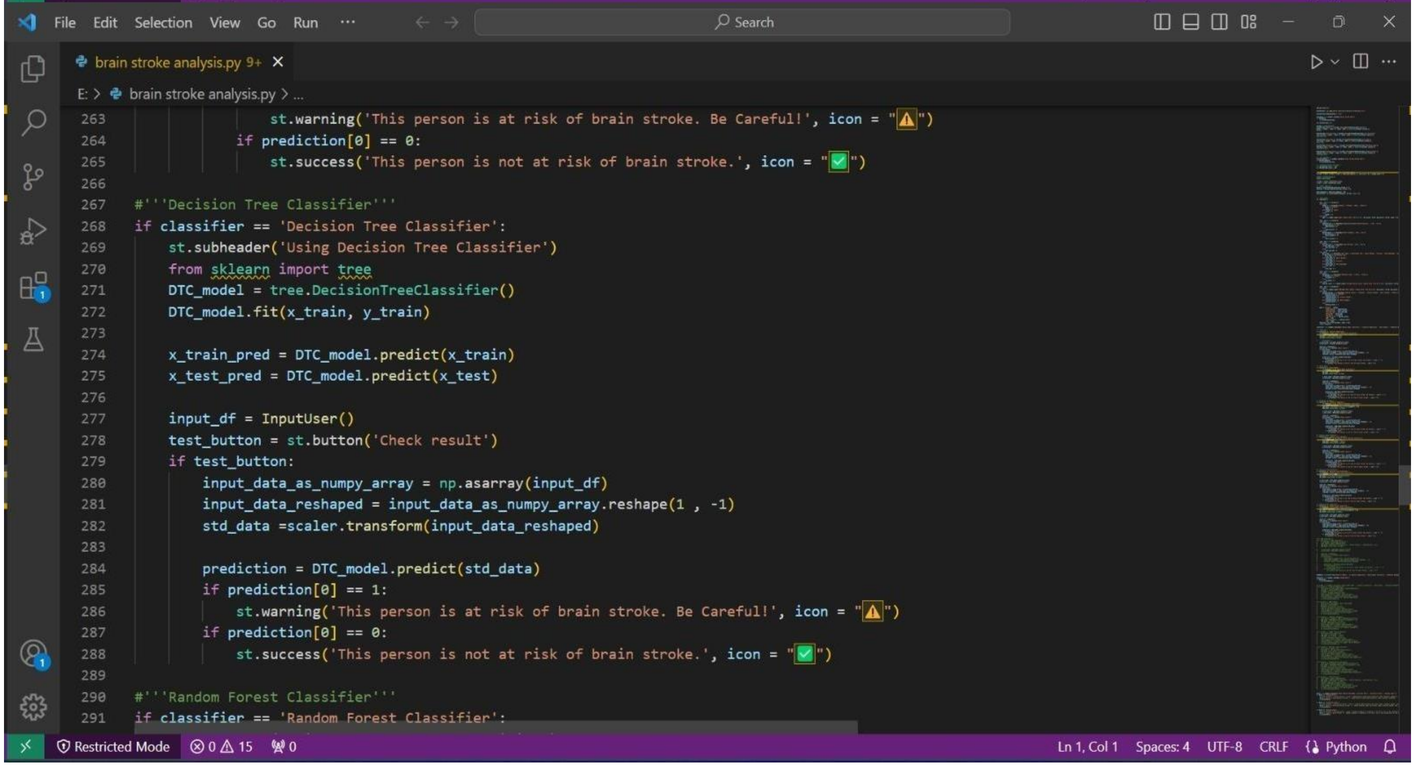
```python
    st.subheader('Using Logistic Regression Classifier')
    from sklearn.linear_model import LogisticRegression
    LR_model = LogisticRegression()
    LR_model.fit(x_train, y_train)

#    '''Evaluation'''
    x_train_pred = LR_model.predict(x_train)
    x_test_pred = LR_model.predict(x_test)

    input_df = InputUser()
    test_button = st.button('Check result')
    if test_button:
        input_data_as_numpy_array = np.asarray(input_df)
        input_data_reshaped = input_data_as_numpy_array.reshape(1 , -1)
        std_data =scaler.transform(input_data_reshaped)

        prediction = LR_model.predict(std_data)
        if prediction[0] == 1:
            st.warning('This person is at risk of brain stroke. Be Careful!', icon = "⚠️")
        if prediction[0] == 0:
            st.success('This person is not at risk of brain stroke.', icon = "✅")

#'''Naive Bayes'''
if classifier == 'Naive Bayes':
        st.subheader('Using Naive Bayes Classifier')
        from sklearn.naive_bayes import GaussianNB
        NB_model = GaussianNB()
        NB_model.fit(x_train, y_train)
```


brain stroke analysis.py 9+
E: > brain stroke analysis.py > ...
Restricted Mode
Ln 1, Col 1 Spaces: 4 UTF-8 CRLF Python

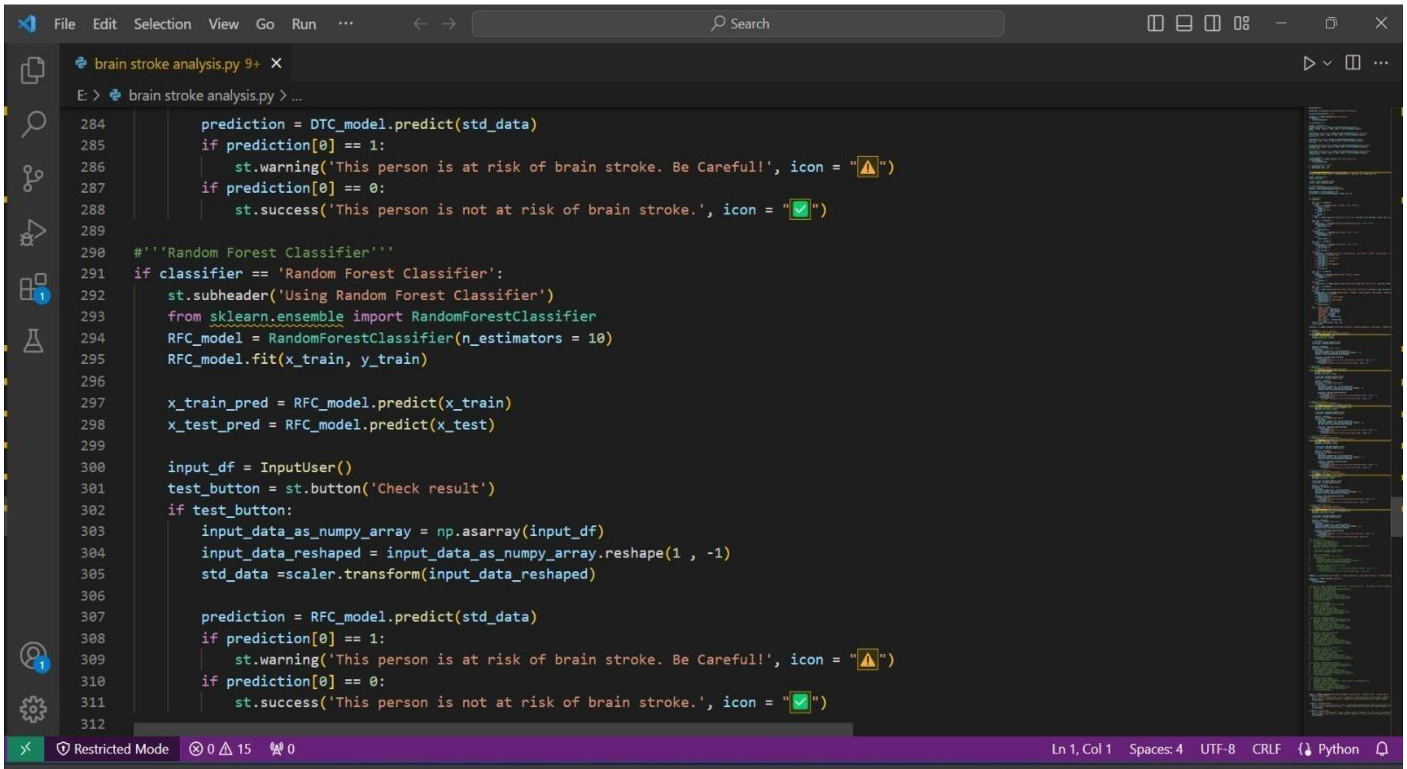
```python
        x_train_pred = NB_model.predict(x_train)
        x_test_pred = NB_model.predict(x_test)

        input_df = InputUser()
        test_button = st.button('Check result')
        if test_button:
            input_data_as_numpy_array = np.asarray(input_df)
            input_data_reshaped = input_data_as_numpy_array.reshape(1 , -1)
            std_data =scaler.transform(input_data_reshaped)

            prediction = NB_model.predict(std_data)
            if prediction[0] == 1:
                st.warning('This person is at risk of brain stroke. Be Careful!', icon = "⚠️")
            if prediction[0] == 0:
                st.success('This person is not at risk of brain stroke.', icon = "✅")

#'''K-Nearest Neighbors'''
if classifier == 'K-Nearest Neighbors':
        st.subheader('Using K-Nearest Neighbors Classifier')
        from sklearn.neighbors import KNeighborsClassifier
        KNN_model = KNeighborsClassifier(n_neighbors = 13)
        KNN_model.fit(x_train, y_train)

        x_train_pred = KNN_model.predict(x_train)
        x_test_pred = KNN_model.predict(x_test)

        input_df = InputUser()
        test_button = st.button('Check result')
        if test_button:
```


brain stroke analysis.py 9+
E: > brain stroke analysis.py > ...
Restricted Mode
Ln 1, Col 1 Spaces: 4 UTF-8 CRLF Python

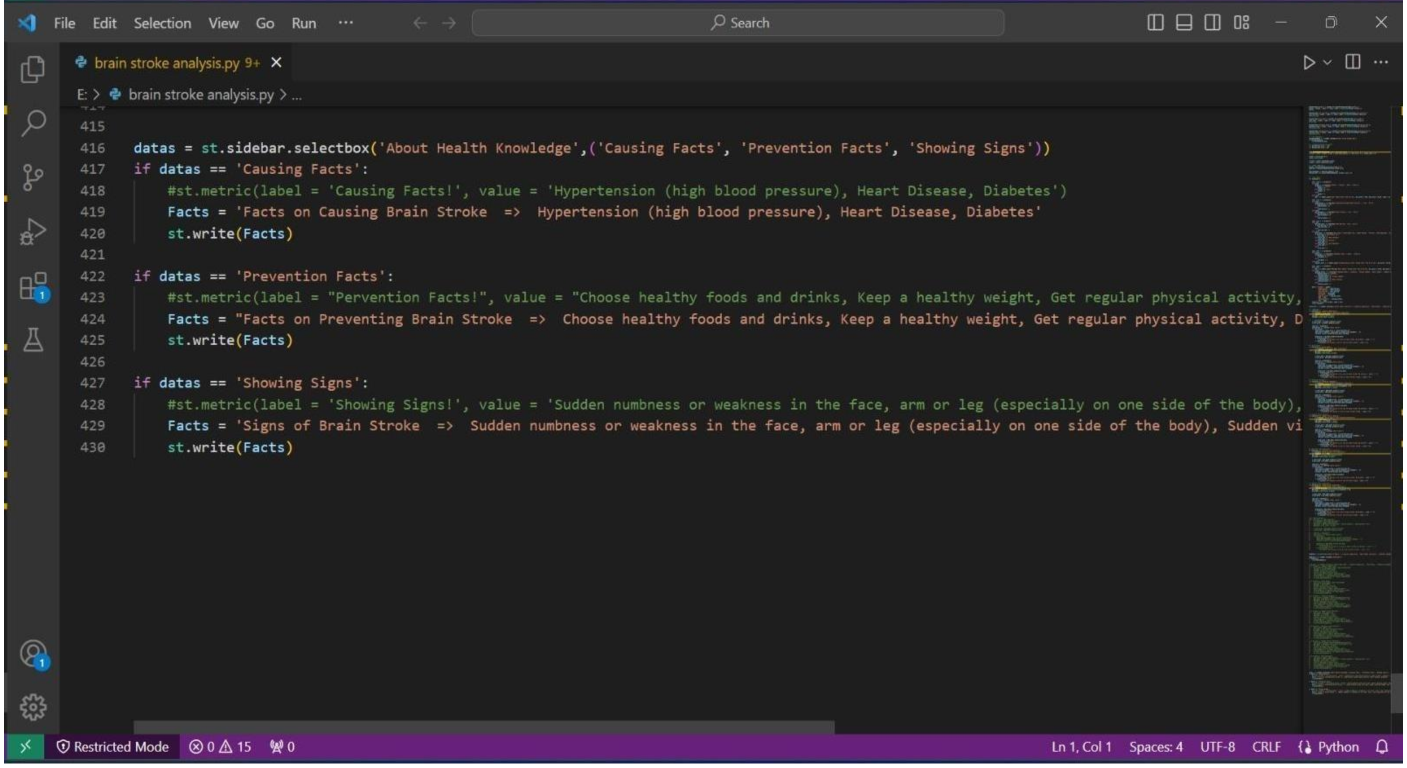
```python
            input_data_as_numpy_array = np.asarray(input_df)
            input_data_reshaped = input_data_as_numpy_array.reshape(1 , -1)
            std_data =scaler.transform(input_data_reshaped)

            prediction = KNN_model.predict(std_data)
            if prediction[0] == 1:
                st.warning('This person is at risk of brain stroke. Be Careful!', icon = "⚠️")
            if prediction[0] == 0:
                st.success('This person is not at risk of brain stroke.', icon = "✅")

#'''Support Vector Machine'''
if classifier == 'Support Vector Machine':
        st.subheader('Using Support Vector Machine Classifier')
        from sklearn.svm import SVC
        SVM_model = SVC(gamma = 'auto')
        SVM_model.fit(x_train, y_train)

        x_train_pred = SVM_model.predict(x_train)
        x_test_pred = SVM_model.predict(x_test)

        input_df = InputUser()
        test_button = st.button('Check result')
        if test_button:
            input_data_as_numpy_array = np.asarray(input_df)
            input_data_reshaped = input_data_as_numpy_array.reshape(1 , -1)
            std_data =scaler.transform(input_data_reshaped)

            prediction = SVM_model.predict(std_data)
            if prediction[0] == 1:
```


brain stroke analysis.py 9+
E: > brain stroke analysis.py > ...
Restricted Mode
Ln 1, Col 1 Spaces: 4 UTF-8 CRLF Python

```python
                st.warning('This person is at risk of brain stroke. Be Careful!', icon = "⚠️")
            if prediction[0] == 0:
                st.success('This person is not at risk of brain stroke.', icon = "✅")

#'''Decision Tree Classifier'''
if classifier == 'Decision Tree Classifier':
    st.subheader('Using Decision Tree Classifier')
    from sklearn import tree
    DTC_model = tree.DecisionTreeClassifier()
    DTC_model.fit(x_train, y_train)

    x_train_pred = DTC_model.predict(x_train)
    x_test_pred = DTC_model.predict(x_test)

    input_df = InputUser()
    test_button = st.button('Check result')
    if test_button:
        input_data_as_numpy_array = np.asarray(input_df)
        input_data_reshaped = input_data_as_numpy_array.reshape(1 , -1)
        std_data =scaler.transform(input_data_reshaped)

        prediction = DTC_model.predict(std_data)
        if prediction[0] == 1:
            st.warning('This person is at risk of brain stroke. Be Careful!', icon = "⚠️")
        if prediction[0] == 0:
            st.success('This person is not at risk of brain stroke.', icon = "✅")

#'''Random Forest Classifier'''
if classifier == 'Random Forest Classifier':
```


brain stroke analysis.py 9+
E: > brain stroke analysis.py > ...
Restricted Mode
Ln 1, Col 1 Spaces: 4 UTF-8 CRLF Python

```python
        prediction = DTC_model.predict(std_data)
        if prediction[0] == 1:
            st.warning('This person is at risk of brain stroke. Be Careful!', icon = "⚠️")
        if prediction[0] == 0:
            st.success('This person is not at risk of brain stroke.', icon = "✅")

#'''Random Forest Classifier'''
if classifier == 'Random Forest Classifier':
    st.subheader('Using Random Forest Classifier')
    from sklearn.ensemble import RandomForestClassifier
    RFC_model = RandomForestClassifier(n_estimators = 10)
    RFC_model.fit(x_train, y_train)

    x_train_pred = RFC_model.predict(x_train)
    x_test_pred = RFC_model.predict(x_test)

    input_df = InputUser()
    test_button = st.button('Check result')
    if test_button:
        input_data_as_numpy_array = np.asarray(input_df)
        input_data_reshaped = input_data_as_numpy_array.reshape(1 , -1)
        std_data =scaler.transform(input_data_reshaped)

        prediction = RFC_model.predict(std_data)
        if prediction[0] == 1:
            st.warning('This person is at risk of brain stroke. Be Careful!', icon = "⚠️")
        if prediction[0] == 0:
            st.success('This person is not at risk of brain stroke.', icon = "✅")
```


brain stroke analysis.py 9+
E: > brain stroke analysis.py > ...
Restricted Mode
Ln 1, Col 1 Spaces: 4 UTF-8 CRLF Python

```python

datas = st.sidebar.selectbox('About Health Knowledge',('Causing Facts', 'Prevention Facts', 'Showing Signs'))
if datas == 'Causing Facts':
    #st.metric(label = 'Causing Facts!', value = 'Hypertension (high blood pressure), Heart Disease, Diabetes')
    Facts = 'Facts on Causing Brain Stroke  =>  Hypertension (high blood pressure), Heart Disease, Diabetes'
    st.write(Facts)

if datas == 'Prevention Facts':
    #st.metric(label = "Pervention Facts!", value = "Choose healthy foods and drinks, Keep a healthy weight, Get regular physical activity,
    Facts = "Facts on Preventing Brain Stroke  =>  Choose healthy foods and drinks, Keep a healthy weight, Get regular physical activity, D
    st.write(Facts)

if datas == 'Showing Signs':
    #st.metric(label = 'Showing Signs!', value = 'Sudden numbness or weakness in the face, arm or leg (especially on one side of the body),
    Facts = 'Signs of Brain Stroke  =>  Sudden numbness or weakness in the face, arm or leg (especially on one side of the body), Sudden vi
    st.write(Facts)
```

6.2 CODE FOR BRAIN STROKE DETECTION

```
import streamlit as st
import pandas as pd
import numpy as np
import sklearn
from sklearn.metrics import confusion_matrix, accuracy_score, precision_score, recall_score, f1_score
from sklearn.preprocessing import LabelEncoder
from sklearn.preprocessing import StandardScaler


st.title('Brain Stroke Prediction')

def add_bg_from_url():
st.markdown(
f"""
<style>
.stApp {{
background-image: url("https://www.brainandspine.co.in/wp-content/uploads/2021/01/4.png");
background-attachment: fixed;
background-size: cover
}}
</style>
""",
unsafe_allow_html=True
)

add_bg_from_url()

BrainStroke = pd.read_csv('E:\healthcare-dataset-stroke-data.csv')

BrainStroke.dropna(inplace = True)

CheckData = st.sidebar.checkbox('Brain Stroke Data')
if CheckData:
st.write(BrainStroke)

del BrainStroke['id']

encoder = LabelEncoder()
```

```
BrainStroke['gender'] = encoder.fit_transform(BrainStroke['gender'])
gender = {index : label for index, label in enumerate(encoder.classes_)}
#gender

BrainStroke['ever_married'] = encoder.fit_transform(BrainStroke['ever_married'])
ever_married = {index : label for index, label in enumerate(encoder.classes_)}
#ever_married

BrainStroke['work_type'] = encoder.fit_transform(BrainStroke['work_type'])
work_type = {index : label for index, label in enumerate(encoder.classes_)}
#work_type

BrainStroke['Residence_type'] = encoder.fit_transform(BrainStroke['Residence_type'])
Residence_type = {index : label for index, label in enumerate(encoder.classes_)}
#Residence_type

BrainStroke['smoking_status'] = encoder.fit_transform(BrainStroke['smoking_status'])
smoking_status = {index : label for index, label in enumerate(encoder.classes_)}
#smoking_status

# '''For Checkbox'''
CheckEncodeData = st.sidebar.checkbox('Brain Stroke Encode Data')
if CheckEncodeData:
st.write(BrainStroke)

# '''Dividing Feature and Label'''
x = BrainStroke.iloc[:, :10]
y = BrainStroke.iloc[:, 10]


from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size = 0.3, random_state = 0)

scaler = StandardScaler()
scaler.fit(x_train)

x_train = scaler.transform(x_train)
x_test = scaler.transform(x_test)

# '''Under Sampling'''
Stroke = BrainStroke[BrainStroke.stroke == 1]
NoStroke = BrainStroke[BrainStroke.stroke == 0]
```

```
NoStrokeSample = NoStroke.sample(n = 50)
BrainStroke = pd.concat([NoStrokeSample, Stroke], axis = 0)


# '''User Input'''
def InputUser():

col1, col2 = st.columns(2)
with col1:
Gender = st.selectbox('Gender', ('Female', 'Male', 'Other'))
if Gender == 'Female':
gender = 0
elif Gender == 'Male':
gender = 1
else:
gender = 2
with col2:
Age = st.number_input('Age: Please enter from 0 to 82', min_value = 0.00, max_value = 82.00, value =
45.00, step = 0.5)

col1, col2 = st.columns(2)
with col1:
Hypertension = st.selectbox('Hypertension(High Blood Pressure)',  ('No', 'Yes'))
if Hypertension == 'No':
hypertension = 0
else:
hypertension = 1
with col2:
HeartDisease = st.selectbox('Heart Disease', ('No', 'Yes'))
if HeartDisease == 'No':
heart_disease = 0
else:
heart_disease = 1

col1, col2 = st.columns(2)
with col1:
Ever_Married = st.selectbox('Ever Married', ('No', 'Yes'))
if Ever_Married == 'No':
ever_married = 0
else:
ever_married = 1
```

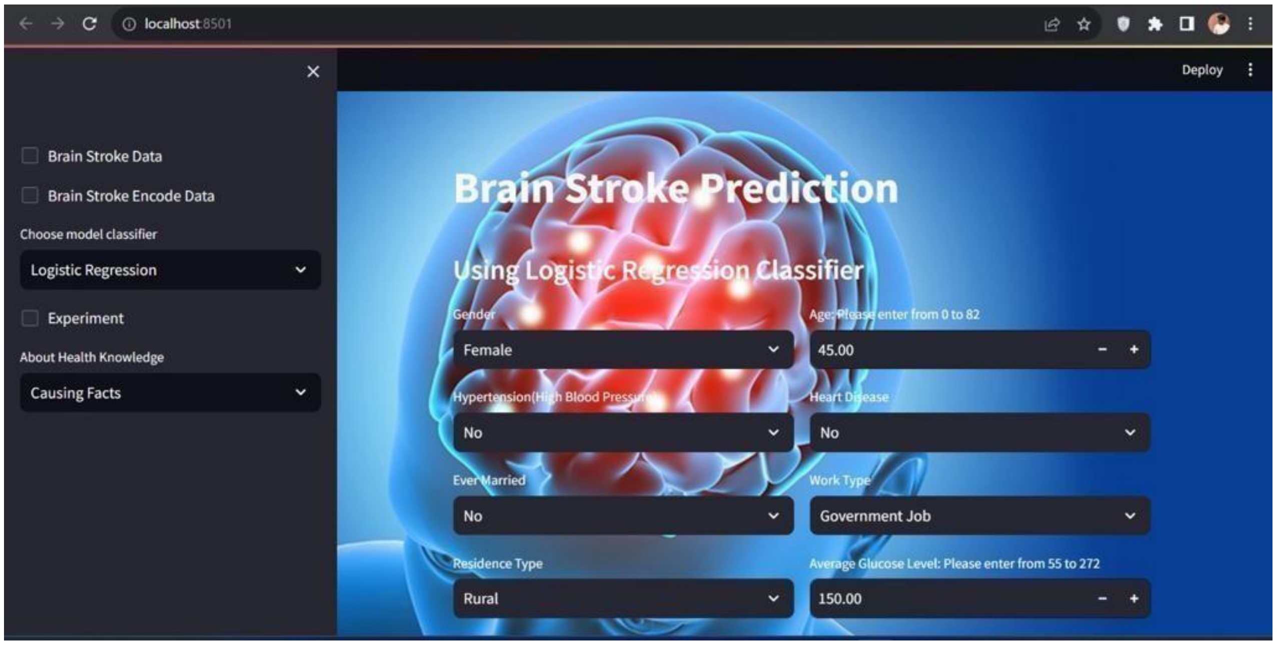
```
with col2:
Work_Type = st.selectbox('Work Type', ('Government Job', 'Never Worked', 'Private', 'Self-employed',
'Children'))
if Work_Type == 'Government Job':
work_type = 0
elif Work_Type == 'Never Worked':
work_type = 1
elif Work_Type == 'Pirvate':
work_type = 2
elif Work_Type == 'Self-employed':
work_type = 3
else:
work_type = 4

col1, col2 = st.columns(2)
with col1:
Residence = st.selectbox('Residence Type', ('Rural', 'Urban'))
if Residence == 'Rural':
residence = 0
else:
residence = 1
with col2:
Avg_Glu_Level = st.number_input('Average Glucose Level: Please enter from 55 to 272', min_value =
55.00, max_value = 272.00, value = 150.00, step = 0.5)

col1, col2 = st.columns(2)
with col1:
BMI = st.number_input('BMI(Body Mass Index): Please enter from 10 to 98', min_value = 10.00,
max_value = 98.00, value = 72.00, step = 0.5)
with col2:
Smoking_Status = st.selectbox('Smoking Status', ('Unknown', 'Formely Smoked', 'Never Smoked',
'Smokes'))
if Smoking_Status == 'Unknown':
smoking_status = 0
elif Smoking_Status == 'Formely Smoked':
smoking_status = 1
elif Smoking_Status == 'Never Smoked':
smoking_status = 2
else:
smoking_status = 3

Data = {'Gender' : gender,
'Hypertension' : hypertension,
'HeartDisease' : heart_disease,
'Ever_Married' : ever_married,
'Work_Type' : work_type,
'Residence' : residence,
'Avg_Glu_Level' : Avg_Glu_Level,
'BMI' : BMI,
'Smoking_Status' : smoking_status,
'Age' : Age}
features = pd.DataFrame(Data, index = [0])
return features

classifier = st.sidebar.selectbox('Choose model classifier', ('Logistic Regression', 'Naive Bayes', 'K-Nearest
Neighbors', 'Support Vector Machine', 'Decision Tree Classifier', 'Random Forest Classifier', 'XGB
Classifier'))

# '''Modeling'''
if classifier == 'Logistic Regression':
st.subheader('Using Logistic Regression Classifier')
from sklearn.linear_model import LogisticRegression
LR_model = LogisticRegression()
LR_model.fit(x_train, y_train)

#     '''Evaluation'''
x_train_pred = LR_model.predict(x_train)
x_test_pred = LR_model.predict(x_test)

input_df = InputUser()
test_button = st.button('Check result')
if test_button:
input_data_as_numpy_array = np.asarray(input_df)
input_data_reshaped = input_data_as_numpy_array.reshape(1 , -1)
std_data =scaler.transform(input_data_reshaped)

prediction = LR_model.predict(std_data)
if prediction[0] == 1:
st.warning('This person is at risk of brain stroke. Be Careful!', icon = "⚠") if
prediction[0] == 0:
st.success('This person is not at risk of brain stroke.', icon = "✅")

#'''Naive Bayes'''
if classifier == 'Naive Bayes':
```

```
st.subheader('Using Naive Bayes Classifier')
from sklearn.naive_bayes import GaussianNB
NB_model = GaussianNB()
NB_model.fit(x_train, y_train)

x_train_pred = NB_model.predict(x_train)
x_test_pred = NB_model.predict(x_test)

input_df = InputUser()
test_button = st.button('Check result')
if test_button:
input_data_as_numpy_array = np.asarray(input_df)
input_data_reshaped = input_data_as_numpy_array.reshape(1 , -1)
std_data =scaler.transform(input_data_reshaped)

prediction = NB_model.predict(std_data)
if prediction[0] == 1:
st.warning('This person is at risk of brain stroke. Be Careful!', icon = "⚠") if
prediction[0] == 0:
st.success('This person is not at risk of brain stroke.', icon = "✅")

#'''K-Nearest Neighbors'''
if classifier == 'K-Nearest Neighbors':
st.subheader('Using K-Nearest Neighbors Classifier')
from sklearn.neighbors import KNeighborsClassifier
KNN_model = KNeighborsClassifier(n_neighbors = 13)
KNN_model.fit(x_train, y_train)

x_train_pred = KNN_model.predict(x_train)
x_test_pred = KNN_model.predict(x_test)

input_df = InputUser()
test_button = st.button('Check result')
if test_button:
input_data_as_numpy_array = np.asarray(input_df)
input_data_reshaped = input_data_as_numpy_array.reshape(1 , -1)
std_data =scaler.transform(input_data_reshaped)

prediction = KNN_model.predict(std_data)
if prediction[0] == 1:
st.warning('This person is at risk of brain stroke. Be Careful!', icon = "⚠")
```

```
if prediction[0] == 0:
st.success('This person is not at risk of brain stroke.', icon = "✅")

#'''Support Vector Machine'''
if classifier == 'Support Vector Machine':
st.subheader('Using Support Vector Machine Classifier')
from sklearn.svm import SVC
SVM_model = SVC(gamma = 'auto')
SVM_model.fit(x_train, y_train)

x_train_pred = SVM_model.predict(x_train)
x_test_pred = SVM_model.predict(x_test)

input_df = InputUser()
test_button = st.button('Check result')
if test_button:
input_data_as_numpy_array = np.asarray(input_df)
input_data_reshaped = input_data_as_numpy_array.reshape(1 , -1)
std_data =scaler.transform(input_data_reshaped)

prediction = SVM_model.predict(std_data)
if prediction[0] == 1:
st.warning('This person is at risk of brain stroke. Be Careful!', icon = "⚠") if
prediction[0] == 0:
st.success('This person is not at risk of brain stroke.', icon = "✅")

#'''Decision Tree Classifier'''
if classifier == 'Decision Tree Classifier':
st.subheader('Using Decision Tree Classifier')
from sklearn import tree
DTC_model = tree.DecisionTreeClassifier()
DTC_model.fit(x_train, y_train)

x_train_pred = DTC_model.predict(x_train)
x_test_pred = DTC_model.predict(x_test)

input_df = InputUser()
test_button = st.button('Check result')
if test_button:
input_data_as_numpy_array = np.asarray(input_df)
input_data_reshaped = input_data_as_numpy_array.reshape(1 , -1)
```

```
std_data =scaler.transform(input_data_reshaped)

prediction = DTC_model.predict(std_data)
if prediction[0] == 1:
st.warning('This person is at risk of brain stroke. Be Careful!', icon = "⚠") if
prediction[0] == 0:
st.success('This person is not at risk of brain stroke.', icon = "✅")

#'''Random Forest Classifier'''
if classifier == 'Random Forest Classifier':
st.subheader('Using Random Forest Classifier')
from sklearn.ensemble import RandomForestClassifier
RFC_model = RandomForestClassifier(n_estimators = 10)
RFC_model.fit(x_train, y_train)

x_train_pred = RFC_model.predict(x_train)
x_test_pred = RFC_model.predict(x_test)

input_df = InputUser()
test_button = st.button('Check result')
if test_button:
input_data_as_numpy_array = np.asarray(input_df)
input_data_reshaped = input_data_as_numpy_array.reshape(1 , -1)
std_data =scaler.transform(input_data_reshaped)

prediction = RFC_model.predict(std_data)
if prediction[0] == 1:
st.warning('This person is at risk of brain stroke. Be Careful!', icon = "⚠") if
prediction[0] == 0:
st.success('This person is not at risk of brain stroke.', icon = "✅")

# #'''XGB Classifier'''
# if classifier == 'XGB Classifier':
#     st.subheader('Using XGB Classifier')
#     from xgboost import XGBClassifier
#     XGB_model = XGBClassifier(objective = 'binary:logistic', learning_rate = 0.2)
#     XGB_model.fit(x_train, y_train)

#     x_train_pred = XGB_model.predict(x_train)
#     x_test_pred = XGB_model.predict(x_test)
#     input_df = InputUser()
#     test_button = st.button('Check result')
#     if test_button:
#         input_data_as_numpy_array = np.asarray(input_df)
#         input_data_reshaped = input_data_as_numpy_array.reshape(1 , -1)
#         std_data =scaler.transform(input_data_reshaped)

#         prediction = XGB_model.predict(std_data)
#         if prediction[0] == 1:
#             st.warning('This person is at risk of brain stroke. Be Careful!', icon = "⚠") # if
          prediction[0] == 0:
#             st.success('This person is not at risk of brain stroke.', icon = "✅")



ExpModels = pd.DataFrame({'Name of Models' : ['Logistic Regression', 'Naive Bayes Classifier', 'K-
Nearest Neighbors', 'Support Vector Machine', 'Decision Tree Classifier', 'Random Forest Classifier', 'XGB
Classifier'], 'Accuracy (%)' : [96.3340, 86.8296, 96.2661, 96.1982, 91.8534, 96.1982, 96.0625], 'Precision
(%)' : [98.1658, 56.5565, 48.1331, 48.1318, 53.2501, 68.1948, 66.4376], 'Recall (%)' : [50.9091, 71.3136,
50.0000, 49.9647, 54.6987, 51.7124, 53.3895], 'F1 (%)' : [50.8515, 58.2078, 49.0488, 49.0311, 53.7471,
52.3631, 55.0537], 'Comfusion Matrix' : [[[1179, 0], [49, 0]], [[1044, 135], [24, 25]], [[1178, 1], [49, 0]],
[[1179, 0], [49, 0]], [[1113, 66], [40, 9]], [[1177, 2], [49, 0]], [[1176, 3], [47, 2]]]})

ExpScores = st.sidebar.checkbox('Experiment')
if ExpScores:
st.write(ExpModels)



# cm_name = st.sidebar.selectbox('Choose Model Name', ('Logistic Regression', 'Naive Bayes', 'K-Nearest
Neighbors', 'Support Vector Machine', 'Decision Tree Classifier', 'Random Forest Classifier', 'XGB
Classifier'))
# if cm_name == 'Logistic Regression':
#     from sklearn.linear_model import LogisticRegression
#     LR_model = LogisticRegression()
#     LR_model.fit(x_train, y_train)
#     y_pred = LR_model.predict(x_test)
#     from sklearn.metrics import confusion_matrix
#     ConfusionMatrix = confusion_matrix(y_test, y_pred)
#     st.text("Confusion Matrix for Logistic Regression")
#     st.write(ConfusionMatrix)

# if cm_name == 'Naive Bayes':
#     from sklearn.naive_bayes import GaussianNB
#     NB_model = GaussianNB()
#     NB_model.fit(x_train, y_train)
#     y_pred = NB_model.predict(x_test)
#     from sklearn.metrics import confusion_matrix
#     ConfusionMatrix = confusion_matrix(y_test, y_pred)
#     st.text("Confusion Matrix for Naive Bayes")
#     st.write(ConfusionMatrix)

# if cm_name == 'K-Nearest Neighbors':
#     from sklearn.neighbors import KNeighborsClassifier
#     KNN_model = KNeighborsClassifier(n_neighbors = 13)
#     KNN_model.fit(x_train, y_train)
#     y_pred = KNN_model.predict(x_test)
#     from sklearn.metrics import confusion_matrix
#    ConfusionMatrix = confusion_matrix(y_test, y_pred)
#    st.text("Confusion Matrix for K-Nearest Neighbors")
# st.write(ConfusionMatrix)

# if cm_name == 'Support Vector Machine':
#     from sklearn.svm import SVC
#     SVM_model = SVC(gamma = 'auto')
#     SVM_model.fit(x_train, y_train)
#     y_pred = SVM_model.predict(x_test)
#     from sklearn.metrics import confusion_matrix
#     ConfusionMatrix = confusion_matrix(y_test, y_pred)
#     st.text("Confusion Matrix for Support Vector Machine")
#     st.write(ConfusionMatrix)

# if cm_name == 'Decision Tree Classifier':
#     from sklearn import tree
#     DTC_model = tree.DecisionTreeClassifier()
#     DTC_model.fit(x_train, y_train)
#     y_pred = DTC_model.predict(x_test)
#     from sklearn.metrics import confusion_matrix
#     ConfusionMatrix = confusion_matrix(y_test, y_pred)
#     st.text("Confusion Matrix for Decision Tree Classifier")
#     st.write(ConfusionMatrix)

# if cm_name == 'Random Forest Classifier':
#     from sklearn.ensemble import RandomForestClassifier
#     RFC_model = RandomForestClassifier(n_estimators = 10)
#     RFC_model.fit(x_train, y_train)
#     y_pred = RFC_model.predict(x_test)
#     from sklearn.metrics import confusion_matrix
#     ConfusionMatrix = confusion_matrix(y_test, y_pred)
#     st.text("Confusion Matrix for Random Forest Classifier")
#     st.write(ConfusionMatrix)

# if cm_name == 'XGB Classifier':
#     from xgboost import XGBClassifier
#     XGB_model = XGBClassifier(objective = 'binary:logistic', learning_rate = 0.2)
#     XGB_model.fit(x_train, y_train)
#     y_pred = XGB_model.predict(x_test)
#     from sklearn.metrics import confusion_matrix
#     ConfusionMatrix = confusion_matrix(y_test, y_pred)
#     st.text("Confusion Matrix for XGB Classifier")
#     st.write(ConfusionMatrix)


datas = st.sidebar.selectbox('About Health Knowledge',('Causing Facts', 'Prevention Facts', 'Showing Signs'))
if datas == 'Causing Facts':
#st.metric(label = 'Causing Facts!', value = 'Hypertension (high blood pressure), Heart Disease, Diabetes')
Facts = 'Facts on Causing Brain Stroke => Hypertension (high blood pressure), Heart Disease, Diabetes'
st.write(Facts)

if datas == 'Prevention Facts':
#st.metric(label = "Pervention Facts!", value = "Choose healthy foods and drinks, Keep a healthy weight, Get regular physical activity, Don't smoke, Limit alcohol, Check cholesterol, Control blood pressure, Control diabetes")
Facts = "Facts on Preventing Brain Stroke  =>  Choose healthy foods and drinks, Keep a healthy weight, Get regular physical activity, Don't smoke, Limit alcohol, Check cholesterol, Control blood pressure, Control diabetes"
st.write(Facts)

if datas == 'Showing Signs':
#st.metric(label = 'Showing Signs!', value = 'Sudden numbness or weakness in the face, arm or leg (especially on one side of the body), Sudden vision problems in one or both eyes, Severe headache with no known cause')
Facts = 'Signs of Brain Stroke  =>  Sudden numbness or weakness in the face, arm or leg (especially on one side of the body), Sudden vision problems in one or both eyes, Severe headache with no known cause'
st.write(Facts)
```

1. RESULTS AND DISCUSSION



Brain Stroke Data

Brain Stroke Encode Data

Choose model classifier: Logistic Regression

Experiment

About Health Knowledge: Causing Facts

Brain Stroke Prediction

Using Logistic Regression Classifier

Gender: Female

Age: Please enter from 0 to 82: 45.00

Hypertension(High Blood Pressure): No

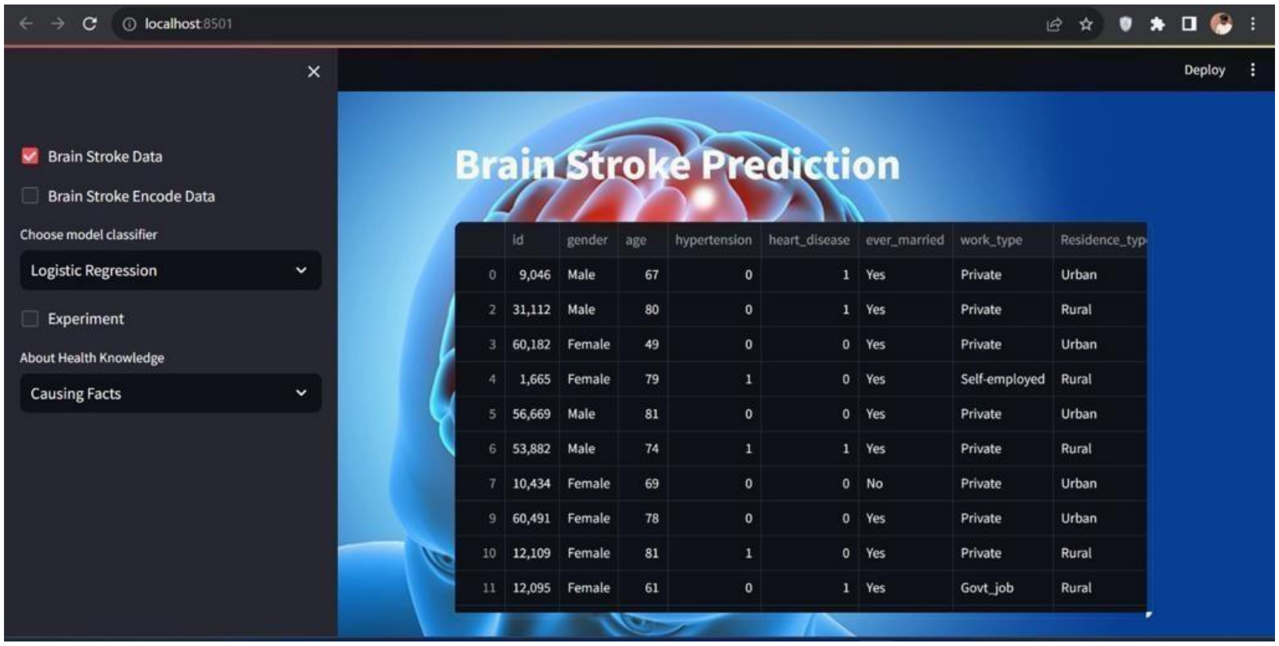
Heart Disease: No

Ever Married: No

Work Type: Government Job

Residence Type: Rural

Average Glucose Level: Please enter from 55 to 272: 150.00



✓ Brain Stroke Data

Brain Stroke Encode Data

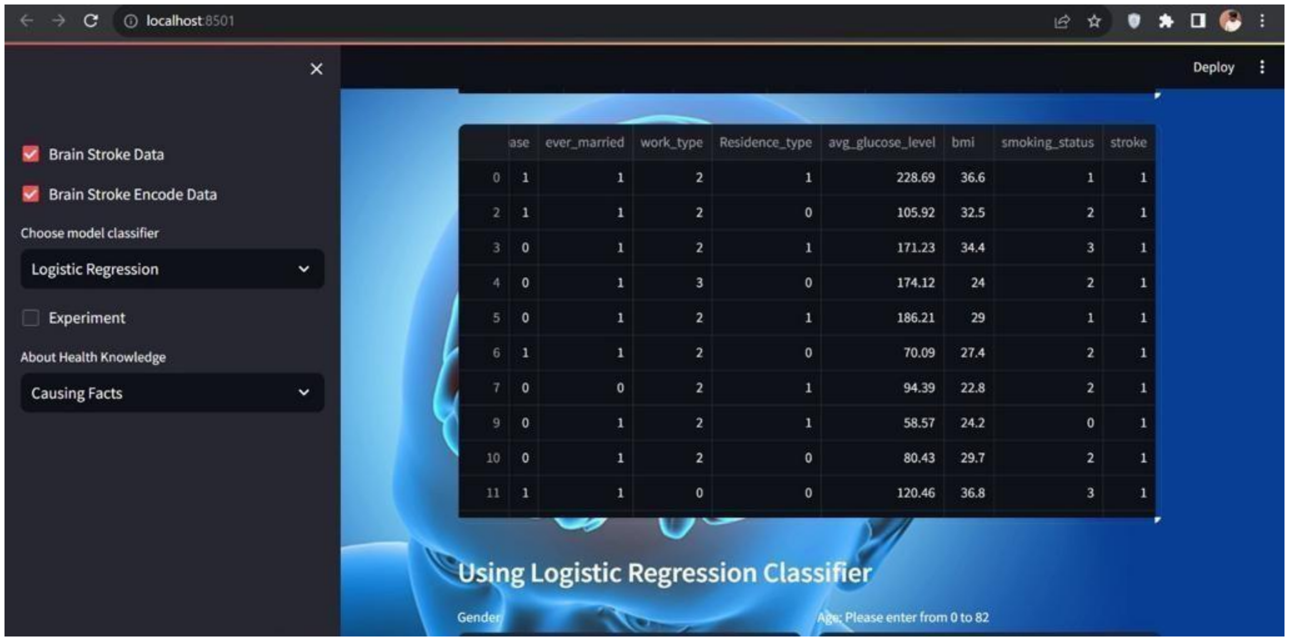
Choose model classifier: Logistic Regression

Experiment

About Health Knowledge: Causing Facts

Brain Stroke Prediction

	id	gender	age	hypertension	heart_disease	ever_married	work_type	Residence_typ
0	9,046	Male	67	0	1	Yes	Private	Urban
2	31,112	Male	80	0	1	Yes	Private	Rural
3	60,182	Female	49	0	0	Yes	Private	Urban
4	1,665	Female	79	1	0	Yes	Self-employed	Rural
5	56,669	Male	81	0	0	Yes	Private	Urban
6	53,882	Male	74	1	1	Yes	Private	Rural
7	10,434	Female	69	0	0	No	Private	Urban
9	60,491	Female	78	0	0	Yes	Private	Urban
10	12,109	Female	81	1	0	Yes	Private	Rural
11	12,095	Female	61	0	1	Yes	Govt_job	Rural

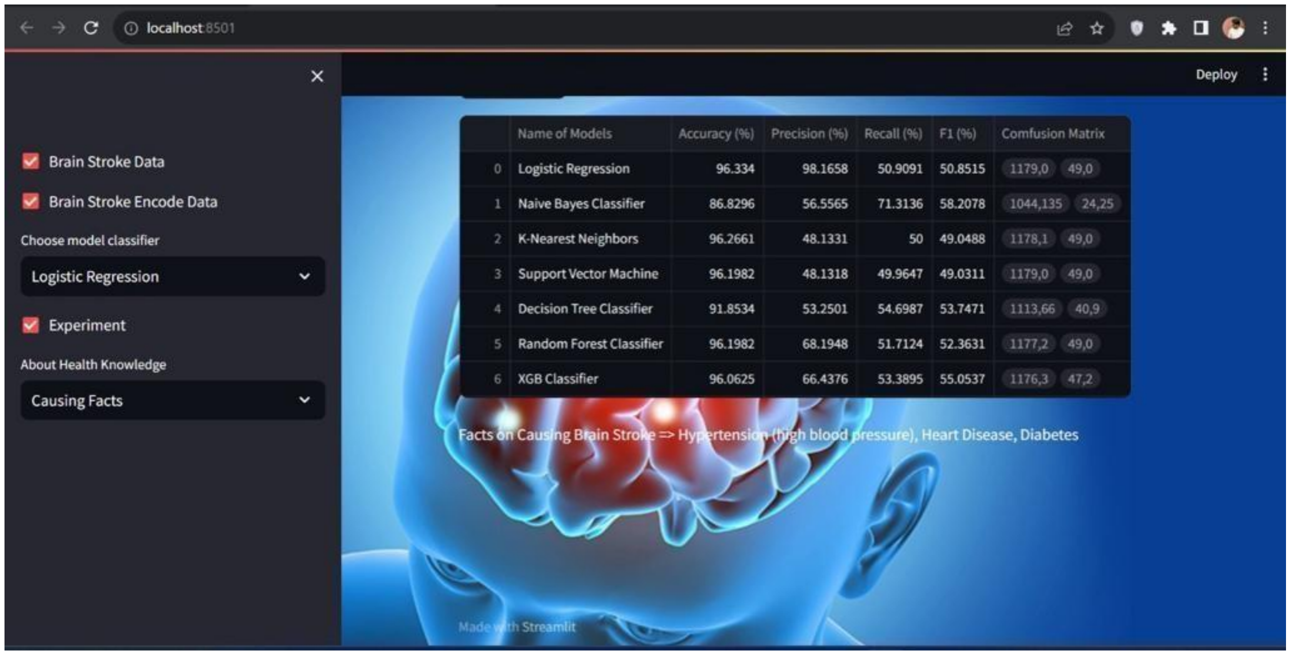


	ase	ever_married	work_type	Residence_type	avg_glucose_level	bmi	smoking_status	stroke
0	1	1	2	1	228.69	36.6	1	1
2	1	1	2	0	105.92	32.5	2	1
3	0	1	2	1	171.23	34.4	3	1
4	0	1	3	0	174.12	24	2	1
5	0	1	2	1	186.21	29	1	1
6	1	1	2	0	70.09	27.4	2	1
7	0	0	2	1	94.39	22.8	2	1
9	0	1	2	1	58.57	24.2	0	1
10	0	1	2	0	80.43	29.7	2	1
11	1	1	0	0	120.46	36.8	3	1

Using Logistic Regression Classifier

Gender

Age: Please enter from 0 to 82



	Name of Models	Accuracy (%)	Precision (%)	Recall (%)	F1 (%)	Comfusion Matrix
0	Logistic Regression	96.334	98.1658	50.9091	50.8515	1179,0 49,0
1	Naive Bayes Classifier	86.8296	56.5565	71.3136	58.2078	1044,135 24,25
2	K-Nearest Neighbors	96.2661	48.1331	50	49.0488	1178,1 49,0
3	Support Vector Machine	96.1982	48.1318	49.9647	49.0311	1179,0 49,0
4	Decision Tree Classifier	91.8534	53.2501	54.6987	53.7471	1113,66 40,9
5	Random Forest Classifier	96.1982	68.1948	51.7124	52.3631	1177,2 49,0
6	XGB Classifier	96.0625	66.4376	53.3895	55.0537	1176,3 47,2

Facts on Causing Brain Stroke => Hypertension (high blood pressure), Heart Disease, Diabetes

CHAPTER 8

CONCLUSION AND FUTURE WORK

In conclusion, this project marks a significant milestone in the healthcare domain, particularly in the early detection and diagnosis of brain strokes. The multifaceted approach, merging traditional diagnostic methods with cutting-edge technologies such as machine learning and artificial intelligence, holds the potential to revolutionize stroke management. The project's strong focus on personalizing risk assessments and upholding ethical considerations ensures that patients receive the best care while safeguarding their privacy and data security. The project's success lies in its ability to substantially enhance the accuracy of stroke detection, improve patient outcomes, and potentially alleviate the long-term burden on healthcare systems. It also represents a testament to the progressive nature of medical research and knowledge, contributing to a deeper comprehension of brain strokes and their timely detection. Looking forward, numerous promising avenues for future work and expansion beckon. These include continuous refinement of machine learning and AI models to enhance their accuracy, real-world clinical deployment, integration of larger-scale data sources, exploration of telemedicine solutions, patient education and engagement strategies, collaboration with medical research institutions, ensuring regulatory compliance, and expanding the project's global reach to benefit a more extensive international community. In essence, the project's future is teeming with opportunities for further advancements in the realm of brain stroke detection. The potential for more accurate diagnoses, personalized care, and innovative research stands to contribute to improved patient outcomes and the ongoing evolution of stroke management and prevention.

REFERENCE

1. **DATA SET USED** : https://www.kaggle.com/datasets/fedesoriano/stroke-prediction-dataset
2. **Stroke Disease Detection and Prediction Using Robust Learning Approaches** https://www.hindawi.com/journals/jhe/2021/7633381/
3. **Brain Stroke Classification via Machine Learning Algorithms Trained with a Linearized Scattering Operator** https://www.ncbi.nlm.nih.gov/pmc/articles/PMC9818173/
4. https://www.researchgate.net/publication/366605948_Brain_Stroke_Prediction_by_Using_Machine_Learning_-_A_Mini_Project.
5. https://griddb.net/en/blog/stroke-prediction-using-machine-learning-python-and-griddb/
6. https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=10130159
7. https://jpinfotech.org/a-machine-learning-model-to-predict-a-diagnosis-of-brain-stroke/
8. https://link.springer.com/chapter/10.1007/978-3-030-74188-4_5
9. https://www.questjournals.org/jecer/papers/vol8-issue4/F08042530.pdf
10. https://www.frontiersin.org/articles/10.3389/fbioe.2023.1257591/full
11. http://203.201.63.46:8080/jspui/bitstream/123456789/6226/1/PR3223%20-%20Prediction_of_Stroke_Report%20-%20SRIKANTH%20S.pdf

Risk Prediction For Traumatic Brain Injury Stroke Using Kernel Extreme Learning Machine and Neutrosophic C-means-based Attribute Weighting Algorithm

ORIGINALITY REPORT

3%	2%	0%	1%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	www.ijraset.com Internet Source	1%
2	www.journal-iiie-india.com Internet Source	<1%
	Submitted to Liverpool John Moores University Student Paper	<1%
4	www.mdpi.com Internet Source	<1%
5	Submitted to University of Bradford Student Paper	<1%
6	www.griet.ac.in Internet Source	<1%
7	www.irjmets.com Internet Source	<1%
8	Submitted to Coventry University Student Paper	<1%
9	D. Shiny Irene, T. Sethukarasi. "Efficient Kernel Extreme Learning Machine and Neutrosophic C-means- based Attribute	<1%

Weighting Method for Medical Data Classification" Journal of Circuits, Systems and Computers, 2020

Publication

10 www.researchgate.net

Internet Source

<1%

Exclude quotes	On	Exclude matches	< 5 words
Exclude bibliography	On		