CHAPTER 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 India according to the Centers for Disease Control and Prevention (CDC). Stroke is a noncommunicable disease that kills approximately 11% of the population. In the India, approximately 795,000 people suffer from the disabling effects of strokes on a regular basis. It is India's fourth leading cause of death. Strokes are classified as ischemic or haemorrhagic. In a chemical stroke, clots obstruct the drainage; in a haemorrhagic 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 unhealthy behaviour, such as smoking and drinking, keeping a healthy body mass index (BMI) and an average glucose level, and maintaining an 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.

CHAPTER 2

LITERATURE SURVEY

They proposed a model for acute stroke. They have taken a large database and used many algorithms to design a supervised machine. But the accuracy showed only 60% of the predictive result.

They conducted a retrospective study of a prospectively collected database of acute ischemic stroke treated by cardiovascular intervention. Using SPSS, MATLAB, and Rapid Miner, classical statistics as well as artificial neural network and support vector algorithms were applied to design a supervised machine capable of classifying these predictors into potential good and poor outcomes. These algorithms were trained, validated and tested using randomly divided data.[1]

They proposed a model that can assist the doctors in clinical trials. In this paper, they haven't mentioned clearly about the dataset they had used for prediction model and also about the method they have used.

The project will have web application through which the doctor or even the patient can fill the details. Details will consist of patient's characteristics which includes person's age, sex, weight, height etc. they will have the training model that helps to compare the newly feed data with the surveyed one and will generate the report on the basis of the comparison. This report will be sent to the person email id.[2]

Modelling results have shown that there are strong risk factors for ischemic stroke by applying the DATAMINING technique (Regression).

It was observed that the model of logistic regression in their case study witnesses, allowed us to analyse the correlation between the occurrence of an ischemic stroke and its risk factors (genetic and clinical). The computer tool and its applications have enabled us to achieve more easily this analysis. However, in the confusion matrix, we concluded, the prediction model achieves 28 + 37 = 65 bad predictions. The error rate is 65/330 = 19.7%.[3]

The possible significance of thrombophilia in ischemic stroke remains controversial. We aimed to study inherited and acquired thrombophilia's as risk factors for ischemic stroke, transient ischemic attack (TIA) and amaurosis fugax in young patients.

We included patients aged 18 to 50 years with ischemic stroke, TIA or amaurosis fugax referred to thrombophilia investigation at Aarhus University Hospital, Denmark from 1 January 2004 to 31 December 2012 (N = 685). Clinical information was obtained from the Danish Stroke Registry and medical records. Thrombophilia investigation results were obtained from the laboratory information system. Absolute thrombophilia prevalence's and associated odds ratios (OR) with 95% confidence intervals (95% CI) were reported for ischemic stroke (N = 377) and TIA or amaurosis fugax (N = 308). Thrombophilia prevalence's for the general population were obtained from published data.[4]

This paper presents a prototype to classify stroke that combines text mining tools and machine learning algorithms. Machine learning can be portrayed as a significant tracker in areas like surveillance, medicine, data management with the aid of suitably trained machine learning algorithms. Data mining techniques applied in this work give an overall review about the tracking of information with respect to semantic as well as syntactic perspectives.

The proposed idea is to mine patients' symptoms from the case sheets and train the system with the acquired data. In the data collection phase, the case sheets of 507 patients were collected from Sugam Multi specialty Hospital, Kumbakonam, Tamil Nadu, India. Next, the case sheets were mined using tagging and maximum entropy methodologies, and the proposed stemmer extracts the common and unique set of attributes to classify the strokes. Then, the processed data were fed into various machine learning algorithms such as artificial neural networks, support vector machine, boosting and bagging and random forests. Among these algorithms, artificial neural networks trained with a stochastic gradient descent algorithm outperformed the other algorithms with a higher classification accuracy of 95% and a smaller standard deviation of 14.69.[5]

CHAPTER 3

SYSTEM STUDY

3.1 EXISTING SYSTEM

They have collected entries and prepared a data-set. Cleaned and prepared the data for using in WEKA.A pre-processed data-set and machine learning techniques are needed for the detection of stroke. They have predefined four different models (Ex. Naive Bayes, J48, K-NN, Random Forest). After creating four alternative models, the accuracy measures, namely accuracy score, precision score, recall score, and F1 score are used to compare them. This accuracy is done based the attributes like hypertension, body mass index level, brain disease, average glucose level, smoking status, previous stroke and age.

Disadvantages

- But they give us the less classification accuracy.

3.2 PROPOSED SYSTEM

After Using Artificial Neural Networks (ANN)- Sequential model to classify the stroke disease. Data is pre-processed by using this model. We are implementing this model by using python libraries (i.e., Tensor Flow, K eras).

Advantages

- By this model we can get the best accuracy of stroke as compare to existing system.

3.3 FEASIBILITY STUDY

Preliminary investigation examines project feasibility, the likelihood of the system will be useful to the organization. The main objective of the feasibility study is to test the Technical, Operational and Economical feasibility for adding new modules and debugging old running system. All systems are feasible if they are given unlimited resources and infinite time. There are aspects in the feasibility study portion of the preliminary investigation:

- Technical Feasibility
- Social Feasibility
- Economic Feasibility

3.3.1 TECHNICAL FEASIBILITY

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. This will lead to high demands on the available technical resources. This will lead to high demands being placed on the client. The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system.

3.3.2 ECONOMICAL FEASIBILITY

This study is carried out to check the economic impact that the system will have on the organization. The amount of fund that the company can pour into the research and development of the system is limited. The expenditures must be justified. Thus the developed system as well within the budget and this was achieved because most of the technologies used are freely available. Only the customized products had to be purchased.

3.3.3 SOCIAL FEASIBILITY

The aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity. The level of acceptance by the users solely depends on the methods that are employed to educate the user about the system and to make him familiar with it. His level of confidence must be raised so that he is also able to make some constructive criticism, which is welcomed, as he is the final user of the system.

3.4 ABOUT DATASET

The data utilized to support this research findings are accessible online at https://www.kaggle.com/fedesoriano/stoke-prediction-dataset.

We have constructed a dataset by collecting stroke data from various sources. Our dataset contains total 1058 individual patient's information of which 412 and are male and 646 are female patient. The type of strokes as: 437 are from ischemic stroke class, 302 from haemorrhagic stroke, 142 of mini-stroke, and 177 reports as brain stem stroke class. Although the dataset is not perfectly symmetrically distributed over all the classes, it has a good ratio to the others. Practically it is very difficult to gather symmetric dataset. The dataset contains total of 28 features.

3.5 ALGORITHMS

3.5.1 NAÏVE BAYES CLASSIFIER

- Naïve Bayes algorithm is a supervised learning algorithm, which is based on Bayes theorem and used for solving classification problems.
- It is mainly used in *text classification* that includes a high-dimensional training dataset.

- Naïve Bayes Classifier is one of the simple and most effective Classification algorithms which helps in building the fast machine learning models that can make quick predictions.
- It is a probabilistic classifier, which means it predicts on the basis of the probability of an object.
- Bayes' theorem is also known as Bayes' Rule or Bayes' law, which is used to determine the probability of a hypothesis with prior knowledge. It depends on the conditional probability.
- The formula for Bayes' theorem is given as:

$$P(A/B) = \frac{P\left(\frac{B}{A}\right) * P(A)}{P(B)}$$

3.5.2 DECISION TREE

Decision Tree is a Supervised learning technique that can be used for both classification and Regression problems, but mostly it is preferred for solving Classification problems. It is a tree-structured classifier, where internal nodes represent the features of a dataset, branches represent the decision rules and each leaf node represents the outcome

J48 algorithm is one of the most widely used machine learning algorithms to examine the data categorically and continuously. The C4.5 algorithm (J48) is mostly used among many fields for classifying data for example interpreting the clinical data for the diagnosis of coronary heart disease, classifying E-governance data, and many more.

3.5.3 RANDOM FOREST

Random forest is a supervised learning algorithm which is used for both classification as well as regression. But however, it is mainly used for classification problems. As we know that a forest is made up of trees and more trees means more robust forest. Similarly, random forest algorithm creates decision trees on data samples and then gets the prediction from each of them and finally selects the best

solution by means of voting. It is an ensemble method which is better than a single decision tree because it reduces the over-fitting by averaging the result.

3.5.4 K-NEAREST NEIGHBOUR (KNN)

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. 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.

3.5.5 ANN-SEQUENTIAL MODELS

We can start building the neural network using sequential models. This top-down approach helps build a Neural net architecture and play with the shape and layers. The first layer will have the number of features which can be fixed using input dim.

Creating Neural Networks is not a very easy process. There are many trials and errors that take place before a good model is built. We will build a Fully Connected network structure using the Dense class in keras. The Neuron counts as the first argument to be provided to the dense layer.

CHAPTER 4

REQUIREMENT SPECIFICATION

4.1 SOFTWARE REQUIREMENT

- **Operating System** : Windows 8
- **Coding Language** : Python 3.7, Tensor Flow, K eras

4.2 HARDWARE REQUIRMENET

- **System** : MINIMUM i3.
- **GPU** : 4GB GEFORCE GTX.
- **Hard Disk** : 40 GB.
- **Ram** : 4 GB.

CHAPTER 5

SOFTWARE REQIREMENT ANALYSIS

5.1 FUNCTIONAL REQUIREMENTS

1.Data Collection

2.Data Pre-processing

3.Modelling

4 Training and Testing

5.Predicting

5.2 NON-FUNCTIONAL REQUIREMENTS

NON-FUNCTIONAL REQUIREMENT (NFR) specifies the quality attribute of a software system. They judge the software system based on Responsiveness, Usability, Security, Portability and other non-functional standards that are critical to the success of the software system. Example of nonfunctional requirement, “how fast does the website load?” Failing to meet non-functional requirements can result in systems that fail to satisfy user needs. Non- functional Requirements allows you to impose constraints or restrictions on the design of the system across the various agile backlogs. Example, the site should load in 3 seconds when the number of simultaneous users are > 10000. Description of non-functional requirements is just as critical as a functional requirement.

- Usability requirement
- Serviceability requirement
- Manageability requirement
- Recoverability requirement

5.3 PYTHON LIBRARIES

TENSORFLOW

TensorFlow is a free and open-source software library for dataflow and differentiable programming across a range of tasks. It is a symbolic math library, and is also used for machine learning applications such as neural networks. It is used for both research and production at Google.

TensorFlow was developed by the Google Brain team for internal Google use. It was released under the Apache 2.0 open-source license on November 9, 2015.

NUMPY

Numpy is a general-purpose array-processing package. It provides a high-performance multidimensional array object, and tools for working with these arrays.

It is the fundamental package for scientific computing with Python. It contains various features including these important ones: A powerful N-dimensional array object. Sophisticated (broadcasting) functions. Useful linear algebra, Fourier transform, and random number capabilities.

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

PANDAS

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

Pandas is used in a wide range of fields including academic and commercial domains including finance, economics, Statistics, analytic, etc.

MATPLOTLIB

Matplotlib is a Python 2D plotting library which produces publication quality figures in a variety of hard copy formats and interactive environments across platforms. Matplotlib can be used in Python scripts, the Python and IPython shells, the Jupyter Notebook, web application servers, and four graphical user interface toolkit. Matplotlib tries to make easy things easy and hard things possible. You can generate plots, histograms, power spectra, bar charts, error charts, scatter plots, etc., with just a few lines of code. For examples, see the sample plots and thumbnail gallery.

For simple plotting the **pyplot** module provides a MATLAB-like interface, particularly when combined with I Python. For the power user, you have full control of line styles, font properties, axes properties, etc, via an object oriented interface or via a set of functions familiar to MATLAB users.

SCIKIT – LEARN

Scikit-learn provides a range of supervised and unsupervised learning algorithms via a consistent interface in Python. It is licensed under a permissive simplified BSD license and is distributed under many Linux distributions, encouraging academic and commercial use.

CHAPTER 6

SYSTEM DESIGN

6.1ARCHITECTURAL DESIGN

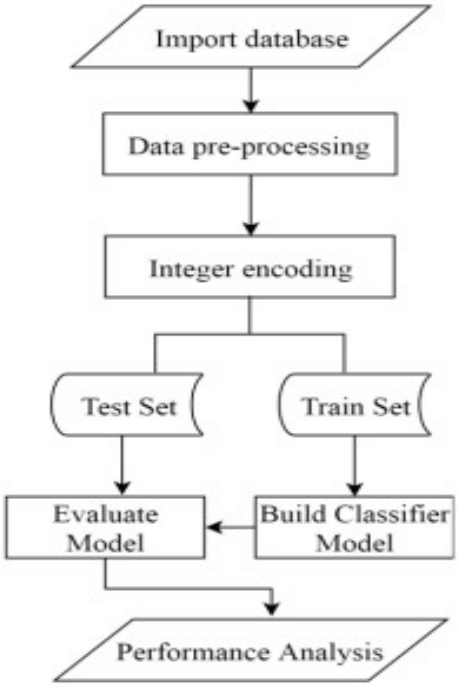

Fig 6.1: System Architecture

6.2 MODULES AND IMPLEMENTATION

To implement this project, we have designed following modules

Upload Stroke Dataset:

Dataset is a file which contains data by using which we train the model, after training the model. The model will be able to predict the required with the help of dataset. Using this module, we will upload dataset to application.

Datasets Pre-processing & Features Selection

Before building a model, data pre processing is required to remove unwanted noise and outliers from the dataset that could lead the model to depart from its intended training. This stage addresses everything that prevents the model from functioning more efficiently. Following the collection of the relevant dataset, the data must be cleaned and prepared for model development. As stated before, the dataset used has twelve characteristics. To begin with, the column id is omitted since its presence has no bearing on model construction. The dataset is then inspected for null values and filled if any are detected. The null values in the column BMI are filled using the data column's mean in this case.

Label encoding converts the dataset's string literals to integer values that the computer can comprehend. As the computer is frequently trained on numbers, the strings must be converted to integers. The gathered dataset has five columns of the data type string. All strings are encoded during label encoding, and the whole dataset is transformed into a collection of numbers. The dataset used for stroke prediction is very imbalanced. The dataset has a total of 5110 rows, with 249 rows indicating the possibility of a stroke and 4861 rows confirming the lack of a stroke. While using such data to train a machine-level model may result in accuracy, other accuracy measures such as precision and recall are inadequate. If such an unbalanced data is not dealt with properly, the findings will be inaccurate, and the forecast will be ineffective. As a result, to obtain an efficient model, this unbalanced data must be dealt with first.

Train Naive Bayes Algorithm

Naïve bayes is a prediction algorithm in machine learning. Above training data will be input to Naïve Bayes algorithm to train a model. and this model will be applied on test data to calculate accuracy, precision, f-score and recall.

Train J48 Algorithm

J48 is a prediction algorithm in machine learning. Above training data will be input to J48 algorithm to train a model and this model will be applied on test data to calculate accuracy, precision, f-score and recall.

Train KNN Algorithm

KNN is a prediction algorithm in machine learning. Above training data will be input to KNN algorithm to train a model and this model will be applied on test data to calculate accuracy, precision, f-score and recall.

Train Random Forest Algorithm

Random Forest is a prediction algorithm in machine learning. Above training data will be input to Random Forest algorithm to train a model and this model will be applied on test data to calculate accuracy, precision, f-score and recall.

Train ANN Algorithm

ANN sequential model is Neural Network algorithm in machine learning. Above training data will be input to ANN algorithm to train a model and this model will be applied on test data to calculate accuracy, precision, f-score and recall.

Prediction Disease on test data

By uploading test data to the application, the user will get prediction result whether the patient will get the stroke or not.

6.3 UML DIAGRAMS

6.3.1 USE CASE DIAGRAM

A use case diagram in the Unified Modelling Language (UML) is a type of behavioral diagram defined by and created from a Use-case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases. The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted.

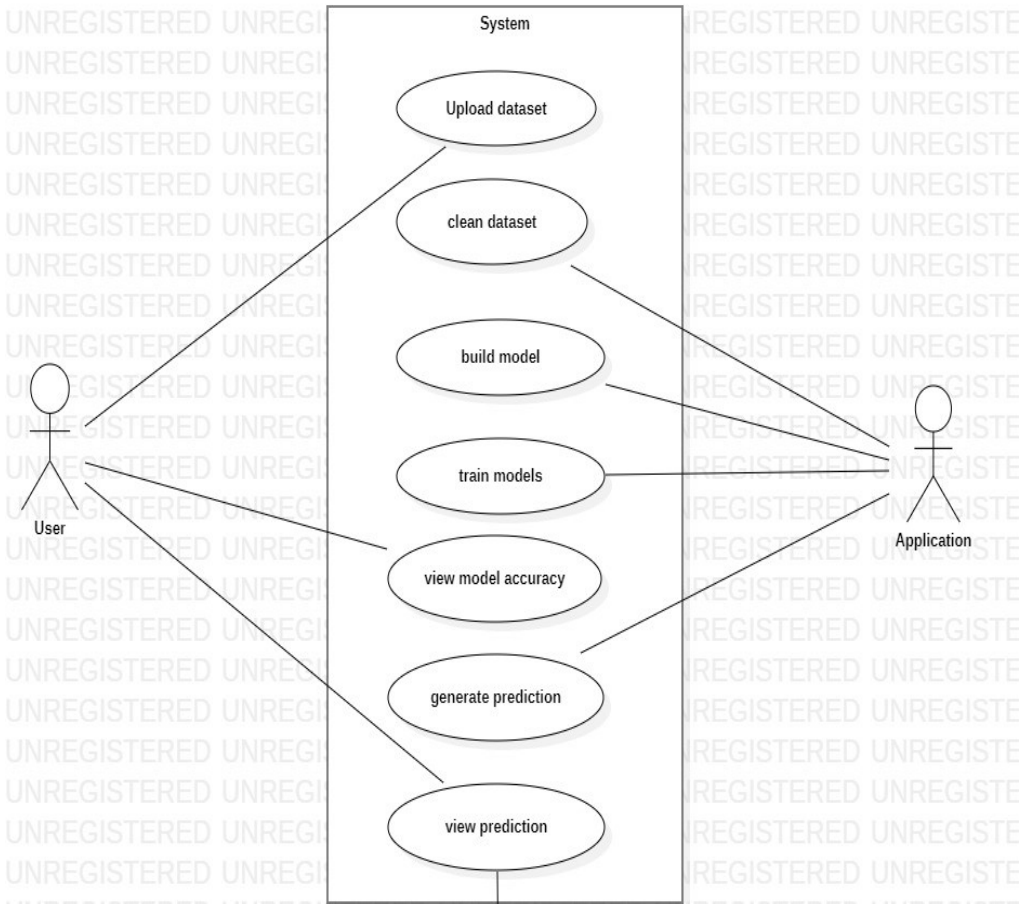

Fig 6.2: Use case diagram

6.3.2 .CLASS DIAGRAM:

In software engineering, a class diagram in the Unified Modelling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among the classes. It explains which class contains information.

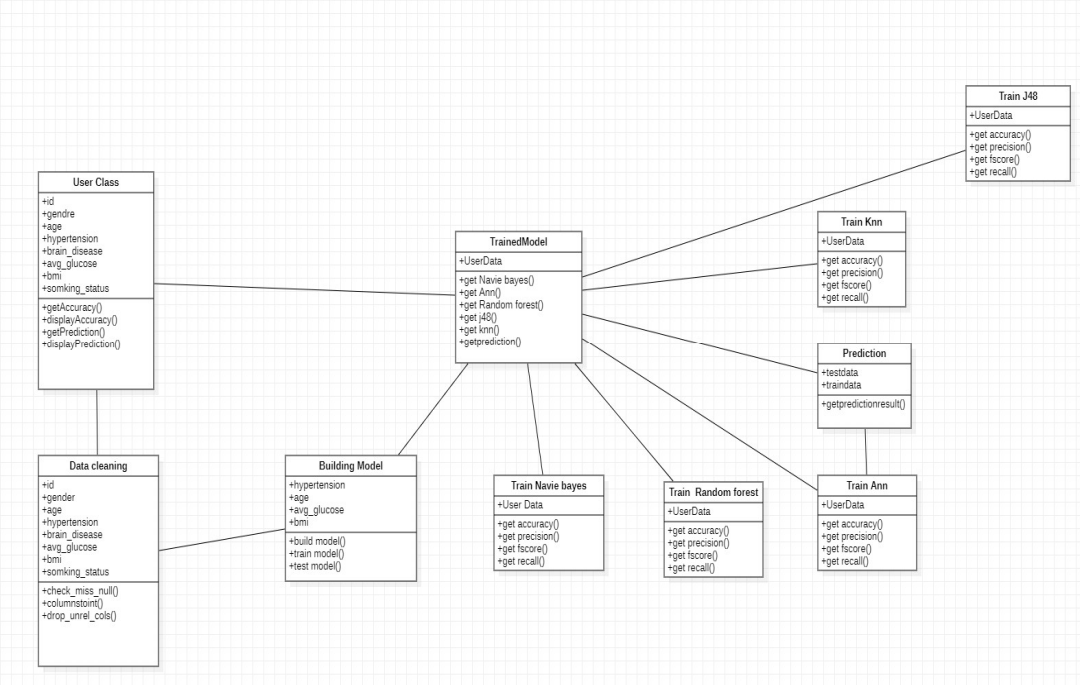

Fig 6.3: Class diagram

6.3.3 SEQUENCE DIAGRAM

A sequence diagram in Unified Modelling Language (UML) is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart. Sequence diagrams are sometimes called event diagrams, event scenarios, and timing diagrams.

6.3.3.1 Sequence diagram for application

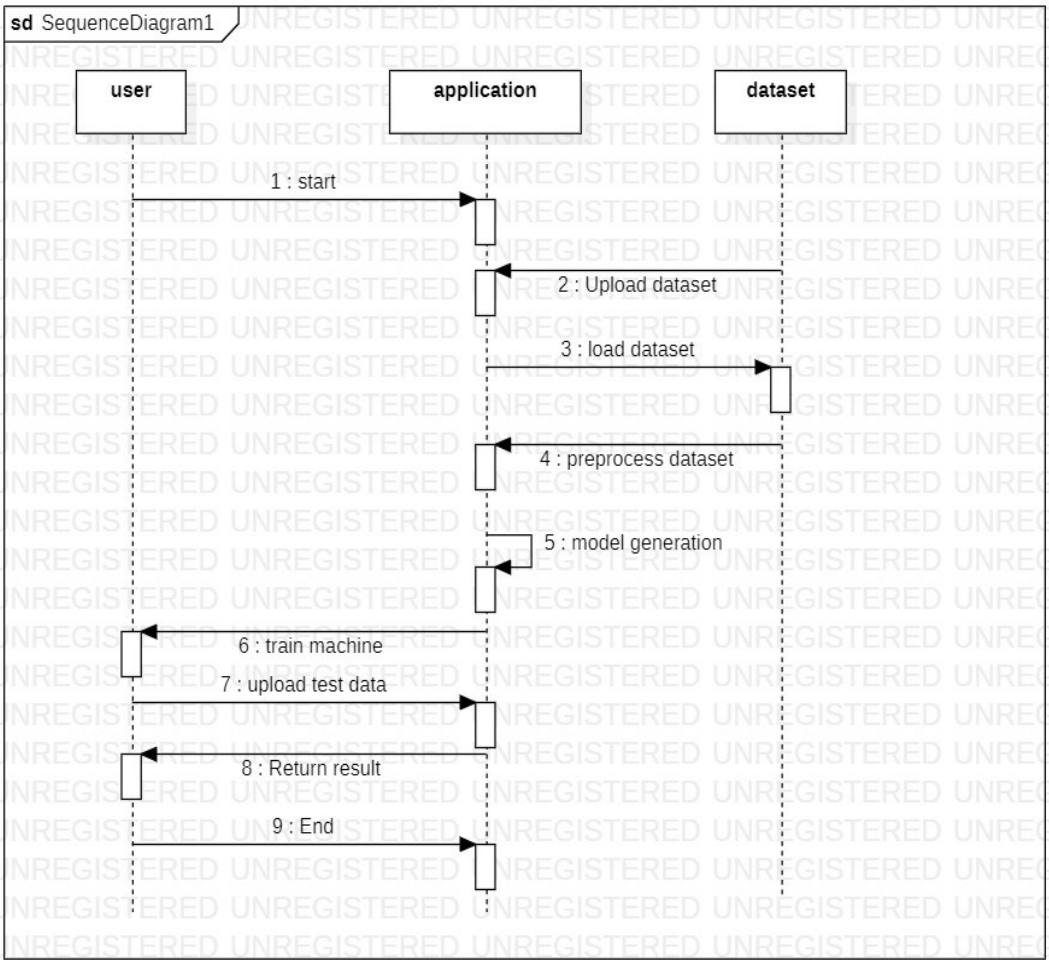

Fig 6.4: Sequence diagram for application

6.3.3.2 Sequence diagram ANN algorithm

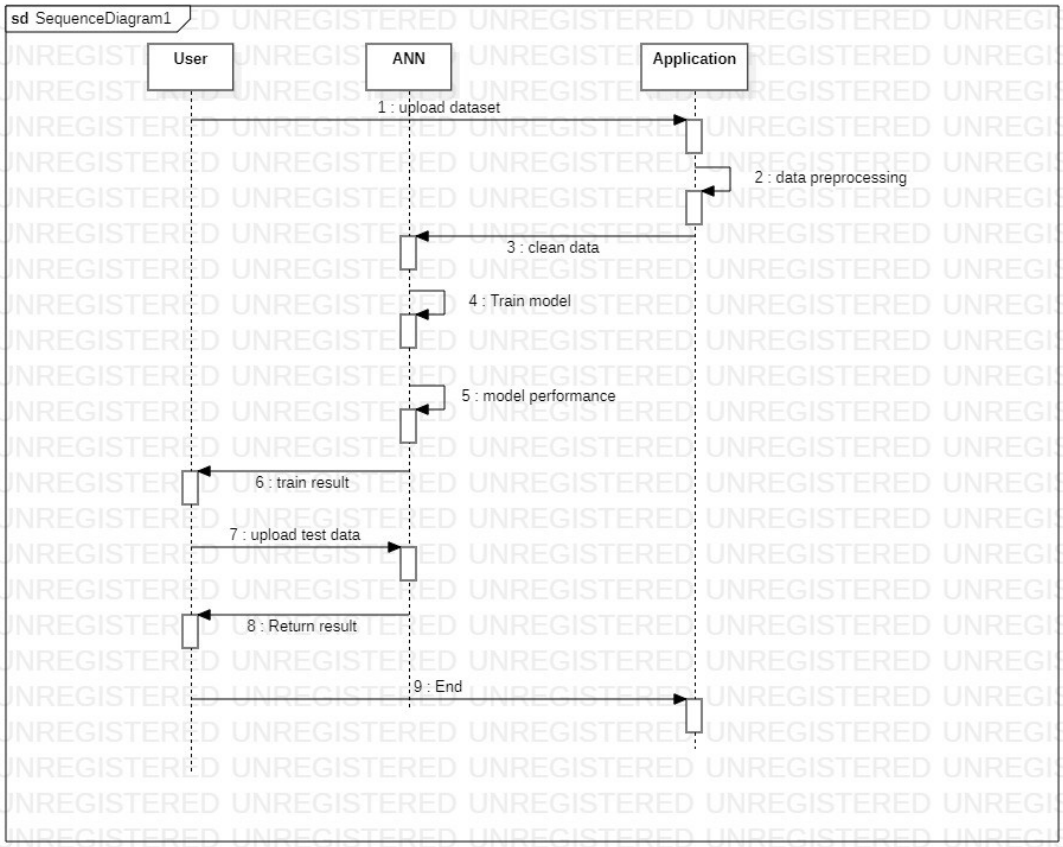

Fig 6.5: Sequence diagram for Ann

CHAPTER 7

SOURCE CODE

```
from tkinter import *
import tkinter
from tkinter import filedialog
from PIL import ImageTk, Image
import numpy as np
from tkinter.filedialog import askopenfilename
from tkinter import simpledialog
#import matplotlib.pyplot as plt
import os
import numpy as np
import pandas as pd
from sklearn.preprocessing import LabelEncoder
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score
from sklearn.metrics import precision_score
from sklearn.metrics import recall_score
from sklearn.metrics import f1_score
import seaborn as sns
```

```
from sklearn.metrics import confusion_matrix
from sklearn.naive_bayes import GaussianNB
from sklearn.tree import DecisionTreeClassifier
from sklearn.neighbors import KNeighborsClassifier
from keras.utils.np_utils import to_categorical
from keras.models import Sequential
from keras.layers import Dense, Dropout, Activation

main = tkinter.Tk()
main.title("Detection of Stroke Disease using Machine Learning Algorithms")
main.geometry("1000x650")

global filename, le1,le2,le3,le4,le5, dataset, rf
global X, Y
global X_train, X_test, y_train, y_test
accuracy = []
precision = []
recall = []
fscore = []
def loadDataset():
    global filename, dataset
    text.delete('1.0', END)
    filename = filedialog.askopenfilename(initialdir="Dataset")
    text.insert(END,str(filename)+" loaded\n\n")
```

```
    dataset = pd.read_csv(filename)
    text.insert(END,str(dataset.head()))


def preprocessDataset():
    text.delete('1.0', END)
    global X, Y
    global X_train, X_test, y_train, y_test
    global dataset, le1,le2,le3,le4,le5
    le1 = LabelEncoder()
    le2 = LabelEncoder()
    le3 = LabelEncoder()
    le4 = LabelEncoder()
    le5 = LabelEncoder()
    dataset.fillna(0, inplace = True)
    dataset['gender'] = pd.Series(le1.fit_transform(dataset['gender'].astype(str)))
    dataset['ever_married'] =
pd.Series(le2.fit_transform(dataset['ever_married'].astype(str)))
    dataset['work_type'] = pd.Series(le3.fit_transform(dataset['work_type'].astype(str)))
    dataset['Residence_type'] =
pd.Series(le4.fit_transform(dataset['Residence_type'].astype(str)))
    dataset['smoking_status'] =
pd.Series(le5.fit_transform(dataset['smoking_status'].astype(str)))
    text.insert(END,str(dataset.head())+"\n\n")
    text.update_idletasks()
    label = dataset.groupby('stroke').size()
```

```
    dataset = dataset.values
    text.insert(END,"\nTotal attributes before applying features selection: "+str(dataset.shape[1])+"\n\n")
    X = dataset[:,1:dataset.shape[1]-1]
    Y = dataset[:,dataset.shape[1]-1]
    indices = np.arange(X.shape[0])
    np.random.shuffle(indices)
    X = X[indices]
    Y = Y[indices]
    text.insert(END,"\nTotal attributes after applying features selection: "+str(X.shape[1])+"\n\n")
    X_train, X_test, y_train, y_test = train_test_split(X, Y, test_size=0.2)
    text.insert(END,"Total records found in dataset : "+str(X.shape[0])+"\n\n")
    text.insert(END,"Dataset split for train and test. 80% for training and 20% for testing\n\n")
    text.insert(END,"Total records used to train Machine Learning Algorithms : "+str(X_train.shape[0])+"\n")
    text.insert(END,"Total records used to test Machine Learning Algorithms : "+str(X_test.shape[0])+"\n")
    label.plot(kind="bar")
    plt.title("Number of Normal & Stroke Disease Instances in dataset")
    plt.show()

def trainNaiveBayes():
    global X_train, X_test, y_train, y_test
```

```
    text.delete('1.0', END)
    cls = GaussianNB()
    cls.fit(X_train, y_train)
    predict = cls.predict(X_test)
    calculateMetrics(predict, y_test, "Naive Bayes")

def trainDT():
    global X_train, X_test, y_train, y_test
    cls = DecisionTreeClassifier()
    cls.fit(X_train, y_train)
    predict = cls.predict(X_test)
    calculateMetrics(predict, y_test, "J48 Algorithm")

def trainKNN():
    global X_train, X_test, y_train, y_test
    cls = KNeighborsClassifier(n_neighbors = 2)
    cls.fit(X_train, y_train)
    predict = cls.predict(X_test)
    calculateMetrics(predict, y_test, "KNN")

def trainANN():
    global X, Y
    Y1 = to_categorical(Y)
    X_train1, X_test1, y_train1, y_test1 = train_test_split(X, Y1, test_size=0.2)
```

```
    ann_model = Sequential()
    ann_model.add(Dense(512, input_shape=(X_train1.shape[1],)))
    ann_model.add(Activation('relu'))
    ann_model.add(Dropout(0.3))
    ann_model.add(Dense(512))
    ann_model.add(Activation('relu'))
    ann_model.add(Dropout(0.3))
    ann_model.add(Dense(2))
    ann_model.add(Activation('softmax'))
    ann_model.compile(loss='categorical_crossentropy', optimizer='adam', metrics=['accuracy'])
    print(ann_model.summary())
    acc_history = ann_model.fit(X, Y1, epochs=200, validation_data=(X_test1, y_test1))
    print(ann_model.summary())
    predict = ann_model.predict(X_test1)
    predict = np.argmax(predict, axis=1)
    testY = np.argmax(y_test1, axis=1)
    calculateMetrics(predict, testY, "ANN")


def predict():
    text.delete('1.0', END)
    global rf, le1,le2,le3,le4,le5
    testfile = filedialog.askopenfilename(initialdir="Dataset")
```

```
    dataset = pd.read_csv(testfile)
    dataset.fillna(0, inplace = True)
    dataset['gender'] = pd.Series(le1.transform(dataset['gender'].astype(str)))
    dataset['ever_married'] = pd.Series(le2.transform(dataset['ever_married'].astype(str)))
    dataset['work_type'] = pd.Series(le3.transform(dataset['work_type'].astype(str)))
    dataset['Residence_type'] = pd.Series(le4.transform(dataset['Residence_type'].astype(str)))
    dataset['smoking_status'] = pd.Series(le5.transform(dataset['smoking_status'].astype(str)))
    dataset = dataset.values

    dataset = dataset[:,1:dataset.shape[1]]
    predict = rf.predict(dataset)
    print(predict)
    for i in range(len(predict)):
        if predict[i] == 0:
            text.insert(END,"Test Data = "+str(dataset[i])+" PREDICTED AS ====> NO STROKE\n\n")
        if predict[i] == 1:
            text.insert(END,"Test Data = "+str(dataset[i])+" PREDICTED AS ====> STROKE\n\n")
nbButton.config(font=font1)
```

CHAPTER 8

SYSTEM TESTING

Testing is a process, which reveals errors in the program. It is the major quality measure employed during software development. During software development. During testing, the program is executed with a set of test cases and the output of the program for the test cases is evaluated to determine if the program is performing as it is expected to perform.

8.1 TESTING CASES

Test case no.	Test case name	Test Description	Expected output	Actual output	Status
1	Upload Dataset	Uploading dataset into the application	Success	Success	Pass

Table 8.1: Test case for upload dataset

Test case no.	Test case name	Test Description	Expected output	Actual output	Status
1	Data Pre-processing	All the missing value, unwanted data, noisy data must be cleaned and data must be divided into train and test data	Success	Success	Pass

Table 8.2: Test case for Data Pre-processing

Test case no.	Test case name	Test Description	Expected output	Actual output	Status
1	Train Naïve Bayes Algo.	Upload dataset must be train to get Accuracy, Recall, Precision, F score	Success	Success	Pass

Table 8.3: Test case for Naïve bayes model

Test case No.	Test case name`	Test Description	Expected output	Actual output	Status
1	Train J48 Algo.	Upload dataset must be train to get Accuracy, Recall, Precision, F score	Success	Success	Pass

Table 8.4: Test case for J48 model

Test Cases no	Test case name	Test Description	Expected output	Actual output	Status
1	Train KNN Algo.	Upload dataset must be train to get Accuracy, Recall, Precision, F score	Success	Success	Pass

Table 8.5: Test case for KNN model

Test Cases no	Test case name	Test Description	Expected output	Actual output	Status
1	Train Random Forest Algo.	Upload dataset must be train to get Accuracy, Recall, Precision, F score	Success	Success	Pass

Table 8.6: Test case for Random Forest model

Test case no	Test case name	Test Description	Expected output	Actual output	Status
1	Train ANN Algo.	Upload dataset must be train to get Accuracy, Recall, Precision, F score	Success	Success	Pass

Table 8.7: Test case for ANN model

Test case no	Test case name	Test Description	Expected output	Actual output	status
1	Predict disease on test data	Uploading test data to the application user will get stroke prediction result	Success	Success	Pass

Table 8.8: Test case for stroke prediction

CHAPTER 9

OUTPUT SCREENS

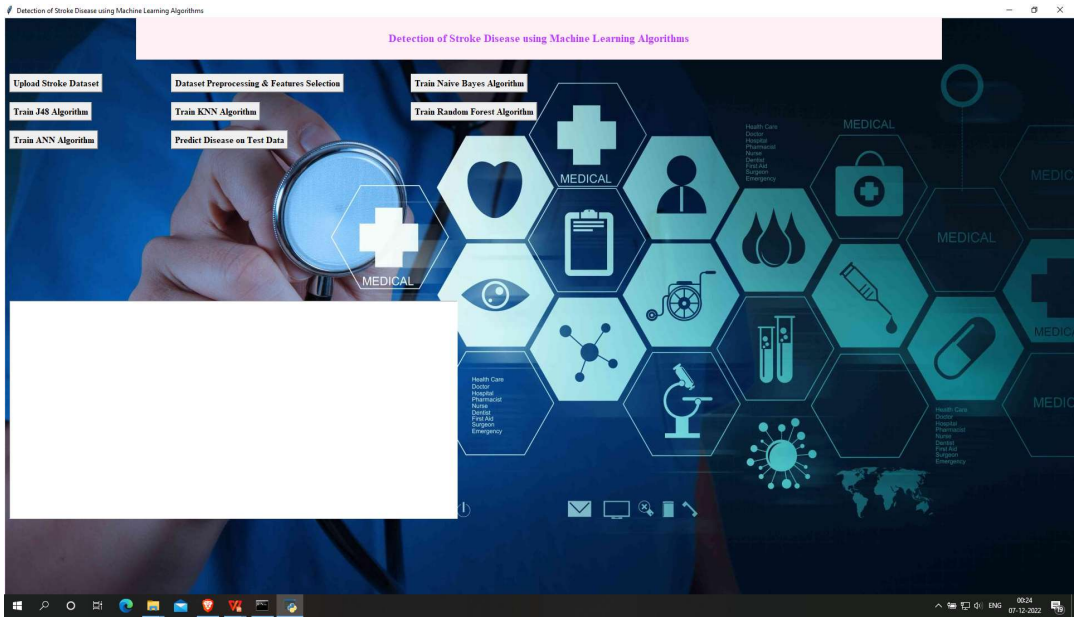

Fig 9.1: Application interface

In above screen click on 'Upload Stroke Dataset' button to upload dataset

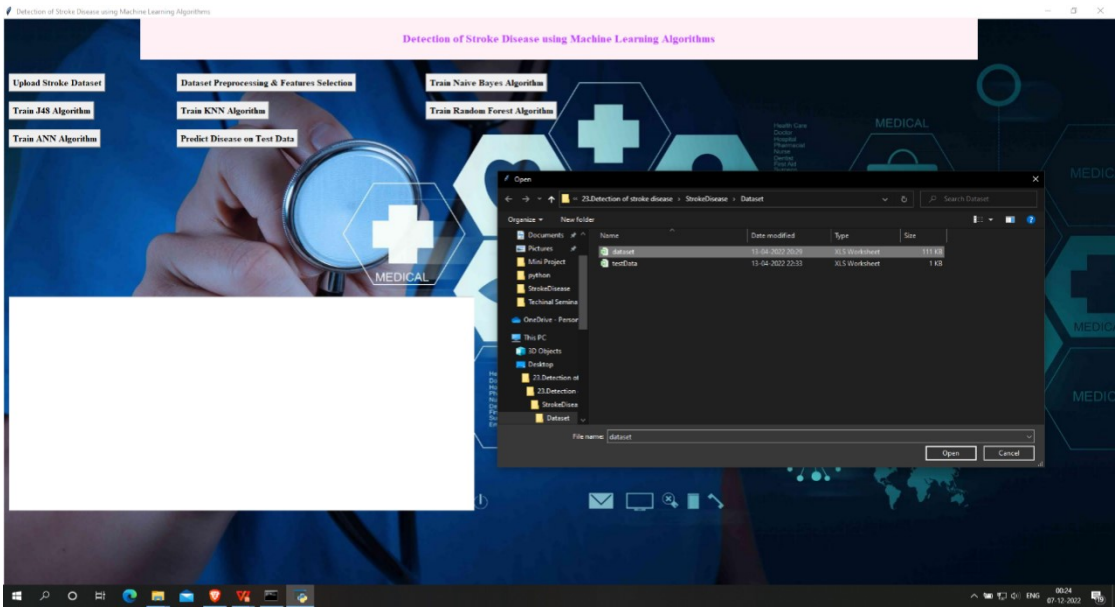

Fig 9.2: Upload dataset

In above screen selecting and uploading dataset.csv file and then click on 'Open' button to load dataset and to get below output

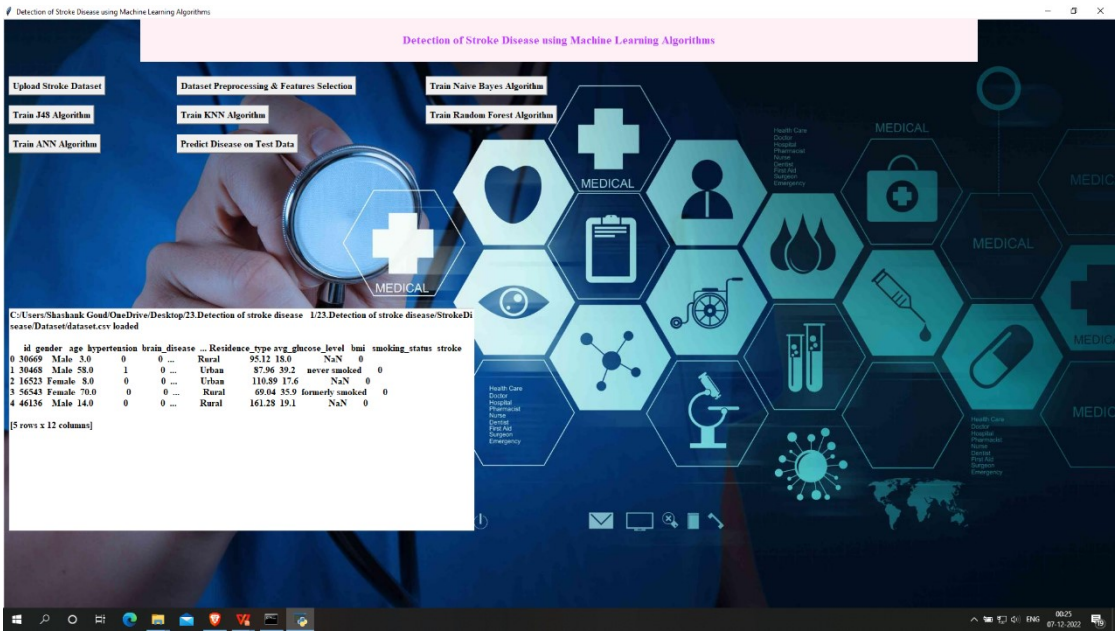

Fig 9.3 Uploaded dataset

In above screen we can see dataset loaded and dataset contains so many missing and non-numeric data so click on 'Dataset Preprocessing & Features Selection' button to process dataset and to get below output

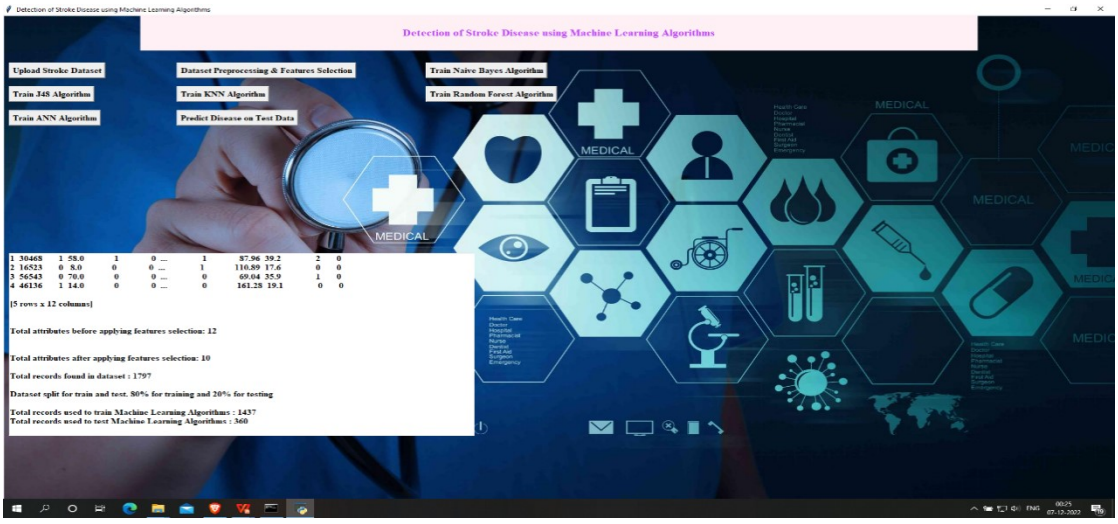

Fig 9.4: Data pre-processing & Features selection

In above screen we can see all dataset converted to numeric format and then split dataset into train and test and now click on 'Train Naïve Bayes Algorithm' button to train Naïve Bayes on above dataset and get below output

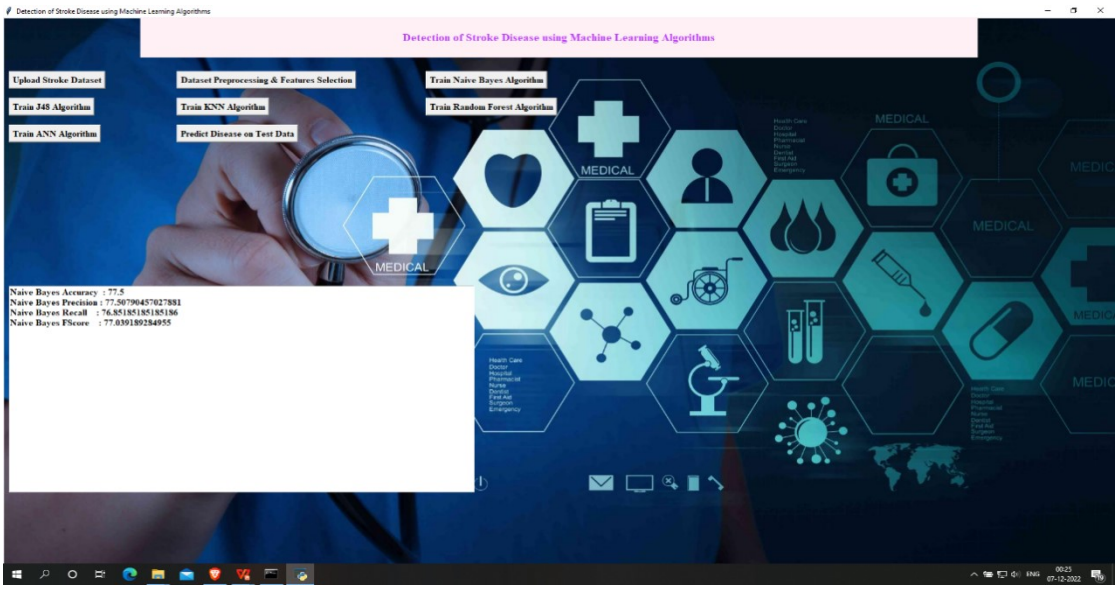

Fig 9.5: Train Naïve Bayes

In above screen with Naïve Bayes, we got 77% accuracy. Now click on 'Train J48 Algorithm' button to get below output

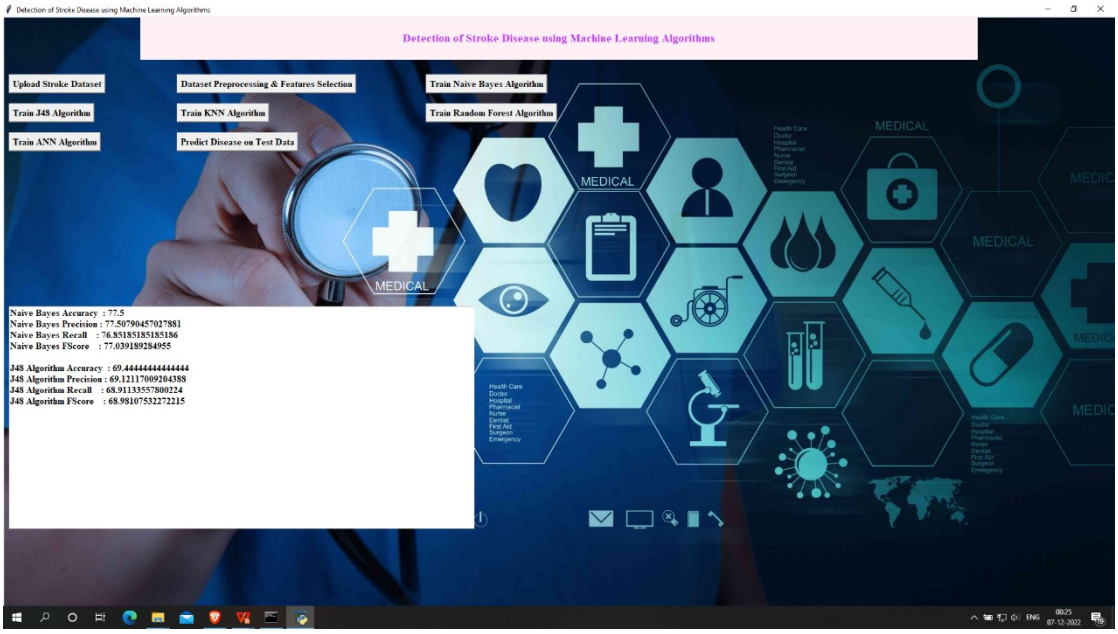

Fig 9.6: Train J48 Algo.

In above screen with J48 we got 69% accuracy and then click on 'Run KNN Algorithm' button to get below output

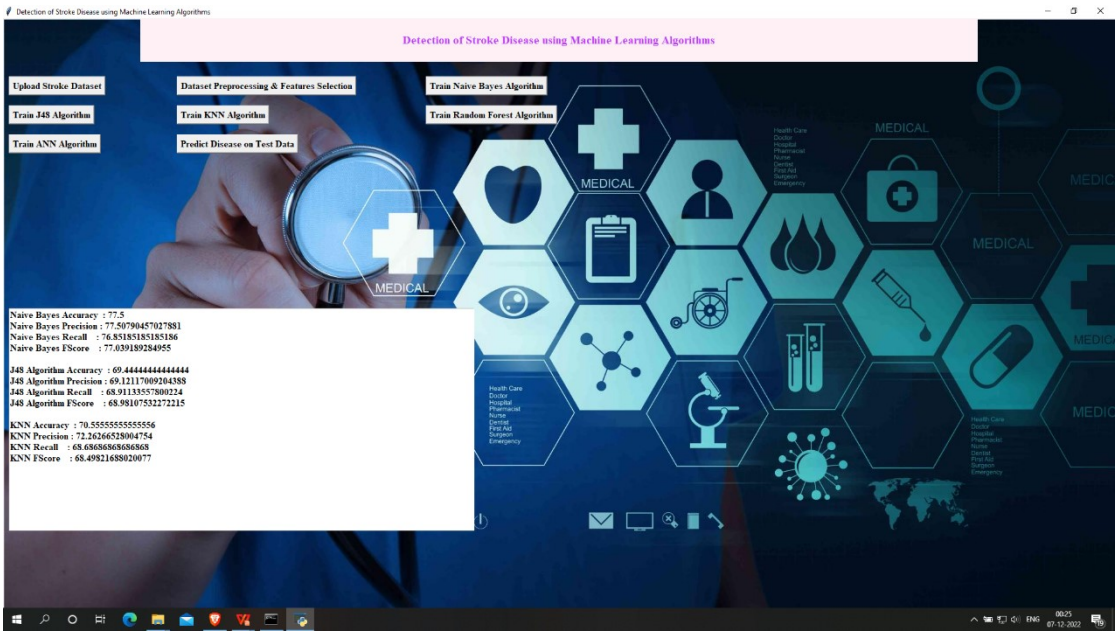

Fig 9.7: Train KNN

In above screen with KNN we got 70% accuracy and then click on 'Run Random Forest Algorithm' button to get below output

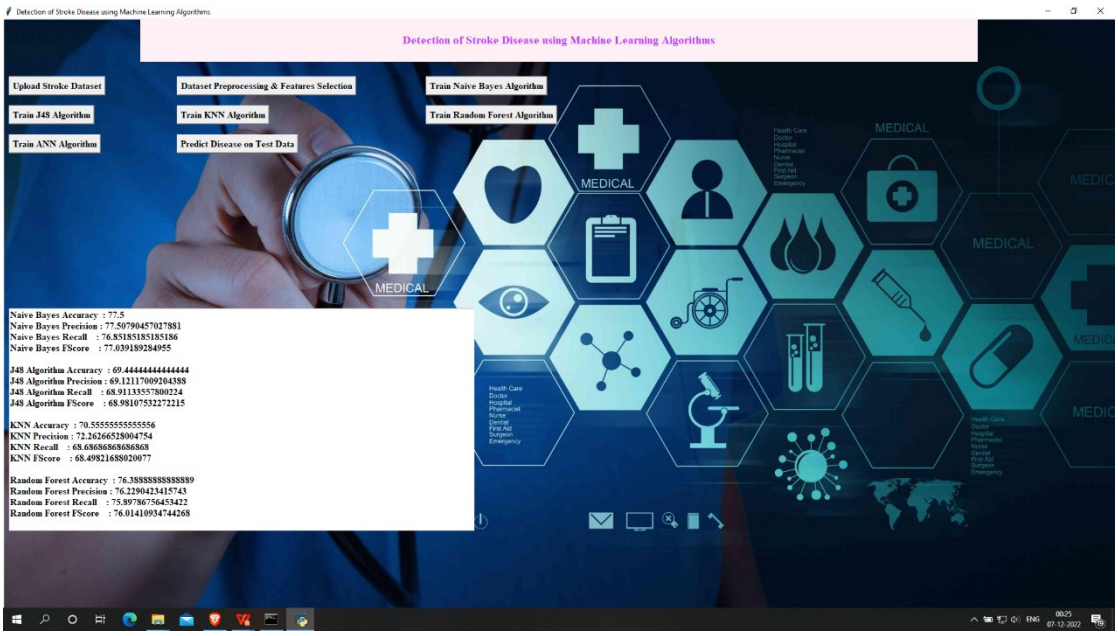

Fig 9.8: Train Random Forest

In above screen with Random Forest we got 76% accuracy and then click on 'Run ANN Algorithm' button to get below output

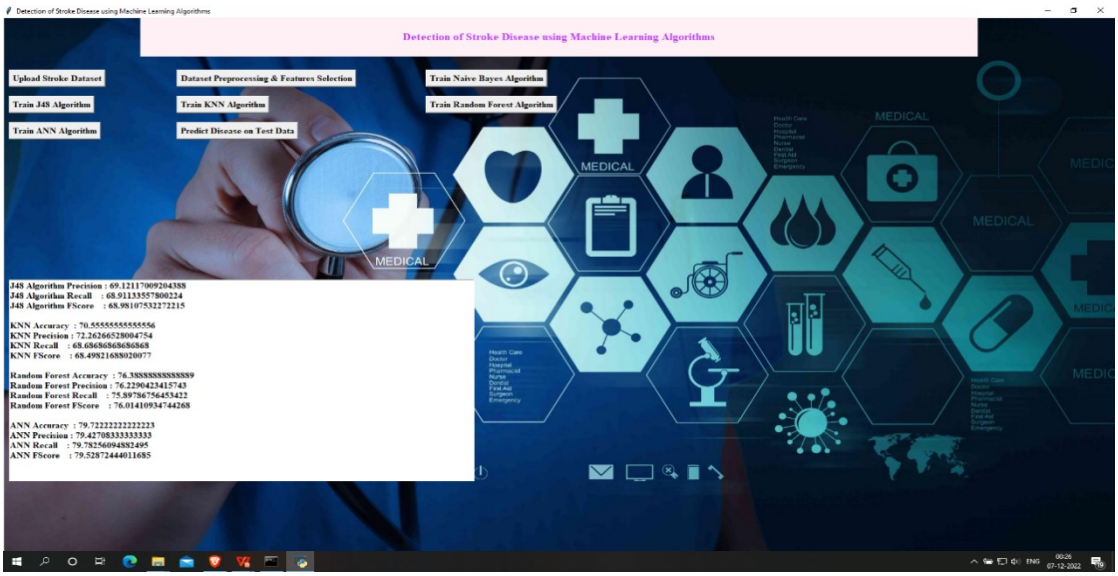

Fig 9.9: Train Ann

In above screen with ANN we got 80% accuracy and in all algorithm ANN got high accuracy.Now, then click on 'Predict Disease on testdata ' button to get prediction

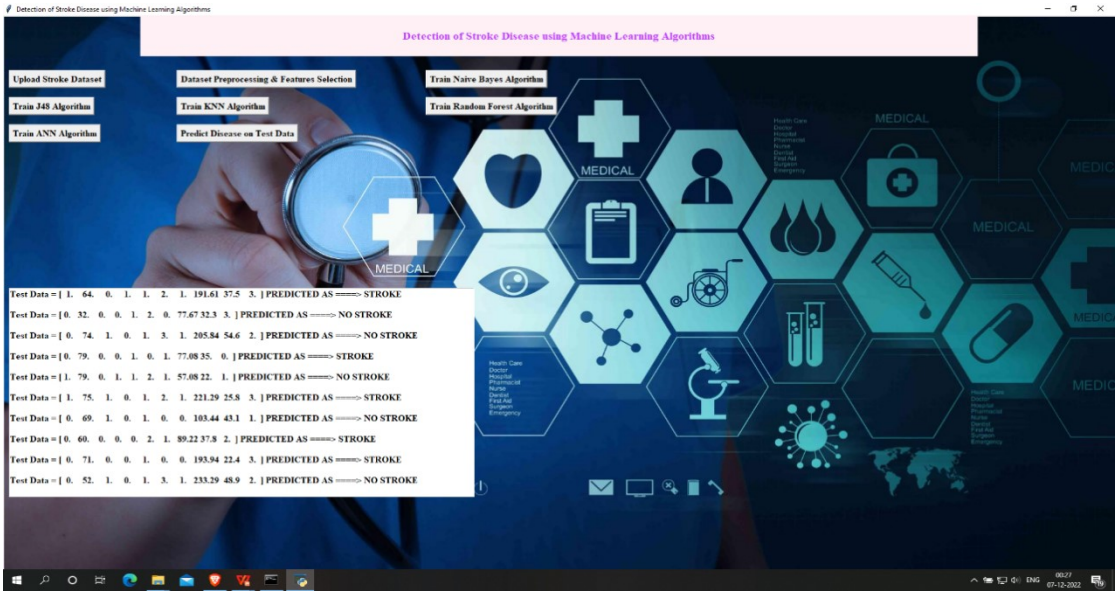

Fig 9.10: Stroke Prediction

In above screen we can see whether the patient is going to get the stroke or not

CHAPTER 10

CONCLUSION

Stroke is a life-threatening medical illness that should be treated as soon as possible to avoid further complications. The development of an ML model could aid in the early detection of stroke and the subsequent mitigation of its severe consequences. The effectiveness of several ML algorithms in properly predicting stroke based on a number of physiological variables is investigated in this study. ANN–Sequential model outperforms the other methods tested with a classification accuracy of 80% percent. According to the research, the ANN–Sequential model outperforms other processes when cross-validation metrics are used in brain stroke forecasting. The future scope of this study is that using a larger dataset and machine learning models, such as AdaBoost, SVM, and Bagging, the framework models may be enhanced. -is will enhance the dependability of the framework and the framework's presentation. In exchange for just providing some basic information, the machine learning architecture may help the general public in determining the likelihood of a stroke occurring in an adult patient. In an ideal world, it would help patients obtain early treatment for strokes and rebuild their lives after the event.

CHAPTER 11

REFERENCES

1. S. H. Pahus, A. T. Hansen, and A.-M. Hvas, “Thrombophilia testing in young patients with ischemic stroke,” Thrombosis research, vol. 137, pp. 108–112, 2016.

2. P. Govindarajan, R. K. Soundarapandian, A. H. Gandomi, R. Patan, P. Jayaraman, and R. Manikandan, “Classification of stroke disease using machine learning algorithms,” Neural Computing and Applications, pp. 1–12.

3. L. T. Kohn, J. Corrigan, M. S. Donaldson, et al., To err is human: building a safer health system, vol. 6. National academy press Washington, DC, 2000.

4. R. Jeena and S. Kumar, “Stroke prediction using svm,” in 2016 International Conference on Control, Instrumentation, Communication and Computational Technologies (ICCICCT), pp. 600–602, IEEE, 2016.

5. P. A. Sandercock, M. Niewada, and A. Członkowska, “The international stroke trial database,” Trials, vol. 13, no. 1, pp. 1–1, 2012.

6. M. S. Singh and P. Choudhary, “Stroke prediction using artificial intelligence,” in 2017 8th Annual Industrial Automation and Electromechanical Engineering Conference (IEMECON), pp. 158–161, IEEE, 2017.

7. S. Y. Adam, A. Yousif, and M. B. Bashir, “Classification of ischemic stroke using machine learning algorithms,” Int J Comput Appl, vol. 149, no. 10, pp. 26–31, 2016.