**A**

**MINI PROJECT REPORT**

**ON**

**HEART DISEASE IDENTIFICATION METHOD USING MACHINE LEARNING CLASSIFICATION IN E-HEALTHCARE**

*A project report submitted to the*

***Jawaharlal Nehru Technological University***

*In partial fulfillment for the award*

***Bachelor of Technology***

***In***

**COMPUTER SCIENCE AND ENGINEERING**

***Submitted by***

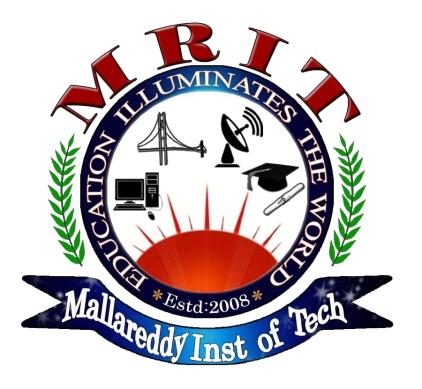
**VADTHYA HARIKA – 21RJ1A05Q9**

**PENALA NIKESH – 22RJ5A0519**

**JATOTH SATHISH – 22RJ5A0512**

**ULINDALA MANIDEEP REDDY – 21RJ1A05Q5**

***Under the esteemed guidance of***

**Mrs.S.Rajeshwari**

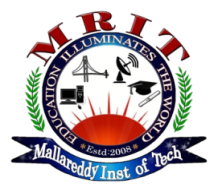
**MALLA REDDY INSTITUTE OF TECHNOLOGY**

**(Affiliated to JNTU, Hyderabad | Approved by AICTE, New Delhi)**

**Accredited by NBA, Certificated by ISO 9001:2015**

**Maisammaguda, Dhullapally, Via: Kompally, Hyderabad - 500100**

**2021 - 2025**

**MALLA REDDY INSTITUTE OF TECHNOLOGY**

**(Sponsored by Malla Reddy Educational Society)**

**Accredited by NBA, Certificated by ISO 9001:2015**

**Approved by AICTE & Affiliated to JNTU, Hyderabad**

Maisammaguda, Dhullapally Post, (Via: Kompally), Secunderabad - 500100.

**CERTIFICATE**

This is to certify that major project work entitled **“HEART DISEASE IDENTIFICATION METHOD USING MACHINE LEARNING IN E-HEALTHCARE”** is a bonafide work carried by **VADTHYA HARIKA (21RJ1A05Q9), PENALA NIKESH (22RJ5A0519), JATOTH SATHISH (22RJ5A0512), ULINDALA MANIDEEP REDDY (21RJ1A05Q5)** of **COMPUTER SCIENCE AND ENGINEERING DEPARTMENT** in **MALLA REDDY INSTITUTE OF TECHNOLOGY** and submitted to **JNT UNIVERSITY, Hyderabad** in the partial fulfillment of the requirement for the award of **BACHELOR OF TECHNOLOGY**.

**Project Guide Project Coordinator Head of Department**

**Mrs.S.Rajeshwari Mr.V.Manikyala Rao Mrs.K.Bhavani**

**External Examiner**

**DECLARATION**

We hereby declare that the project entitled **“HEART DISEASE IDENTIFICATION METHOD USING MACHINE LEARNING IN E-HEALTHCARE”** submitted to **Malla Reddy Institute of Technology**, affiliated to Jawaharlal Nehru Technological University Hyderabad (**JNTUH**) for the award of the degree of **Bachelor of Technology** in **Computer Science and Engineering** is a result of work done by us.

It is further declared that the project report or any part thereof has not been previously submitted to any University or Institute for the award of degree.

**Vadthya Harika – 21RJ1A05Q9**

**Penala Nikesh – 22RJ5A0519**

**Jatoth Sathish – 22RJ5A0512**

**Ulindala Manideep Reddy – 21RJ1A05Q5**

**ACKNOWLEDGEMENT**

We are happy to express our deep sense of gratitude to the **Principal** of the college **Dr. K. Srinivasa Rao, Professor, Malla Reddy Institute of Technology** for having provided us the adequate facilities to pursue our project.

We would like to thank **Mrs.K.Bhavani, Associate Professor and Head, Department of Computer Science and Engineering, Malla Reddy Institute of Technology,** for having provided the freedom to use all the facilities available in the department, especially the laboratories and library.

We also grateful to our project coordinator, **Mr.V.Manikyala Rao**, **Assistant professor**, **Department of Computer Science and Engineering**, **Malla Reddy Institute of Technology**, for extending his support and assisting us throughout our project work.

We very grateful to our project guide, **Mrs.S.Rajeshwari**, **Assistant Professor, Department of Computer science and engineering**, **Malla Reddy Institute of Technology**, for his extensive patience and guidance throughout our project work.

We sincerely thank all the **Teaching and Non-teaching staff** of the department of **Computer Science and Engineering**, also like to thank our **Classmates** for their timely suggestions, healthy criticism and motivation during the course of our project work. We thank our **parents** who were the backbone behind our deeds.

Finally, we express our immense gratitude with pleasure to all **individuals** who have either directly or indirectly contributed to our need at right time for the development and success of our project work.

## INDEX

**TITLES PAGE NO**

1. **INTRODUCTION 1**
2. **LITERATURE SURVEY 3**
   1. 2.1 KEY-DEDUPLICATION WITH IBBE 3
   2. 2.2 SERVER LESS DISTRIBUTED FILE SYSTEM 5
   3. 2.3 VIRTUAL MACHINE CLUSTER MOBILITY IN INTERCLOUD 6
   4. 2.4 A USER CENTRIC DECISION MAKING 7
   5. 2.5 SOFTWARE ENVIRONMENT 8
   6. 2.6 WHY CHOOSE PYTHON 9
3. **SYSTEM ANALYSIS 14**
   1. 3.1 EXISTING SYSTEM 14
   2. 3.2 PROPOSED SYSTEM 17
4. **FEASIBILITY STUDY 18**
   1. 4.1 ECONOMICAL FEASIBILITY 18
   2. 4.2 TECHNICAL FEASIBILITY 19
   3. 4.3 SOCIAL FEASIBILITY 19
5. **SYSTEM REQUIREMENTS 20**
6. [**SYSTEM DESIGN 23**](#_TOC_250015)
   1. 6.1 [SYSTEM ARCHITECTURE 23](#_TOC_250014)
   2. 6.2 [DATA FLOW DIAGRAM 24](#_TOC_250013)
   3. 6.3 [UML DIAGRAMS 26](#_TOC_250012)
7. [**IMPLEMENTATION 32**](#_TOC_250011)
   1. 7.1 [MODULES 32](#_TOC_250010)
   2. 7.2 [SAMPLE CODE 34](#_TOC_250009)
8. [**SYSTEM TESTING 35**](#_TOC_250008)
   1. 8.1 UNIT TESTING 38
   2. 8.2 [INTEGRATION TESTING 39](#_TOC_250007)
   3. 8.3 [ACCEPTANCE TESTING 40](#_TOC_250006)
9. [**INPUT DESIGN AND OUTPUT DESIGN 41**](#_TOC_250005)
   1. 9.1 [INPUT DESIGN 41](#_TOC_250004)
   2. 9.2 [OUTPUT DESIGN 43](#_TOC_250003)
10. [**SCREENSHOTS 44**](#_TOC_250002)
11. [**FUTURE ENHANCEMENT 58**](#_TOC_250001)
12. [**CONCLUSION 59**](#_TOC_250000)
13. **BIBILOGRAPHY 60**

## LIST OF FIGURES

**NAME OF THE FIGURE PAGE NO**

##### FIG-6.1 ARCHITECTURE DESIGN 23

FIG-6.2 DATA FLOW DIAGRAM FOR USER 25

##### FIG-6.3.1 USE CASE DIAGRAM 27

FIG-6.3.2 CLASS DIAGRAM 28

##### FIG-6.3.3 SEQUENCE DIAGRAM 30

FIG-6.3.4 ACTIVITY DIAGRAM 31

## LIST OF SCREEN SHOTS

***NAME OF THE SCREEN SHOT PAGE NO***

##### FIG-10.1 COLUD SERVER LOGIN 43

##### FIG-10.2 ADMIN LOGIN SCREEN 44

##### FIG-10.3 ADMIN HOME SCREEN 44

##### FIG-10.4 FBS LOGIN PAGE 45

##### FIG-10.5 FBS HOME SCREEN 45

##### FIG-10.6 LOGIN FOR USER AND END USER 46

##### FIG-10.7 USER RESGISTRATION SCREEN 46

##### FIG-10.8 VIEW USER DETAILS 47

##### FIG-10.9 UPLOAD THE FILE 48

##### FIG-10.10 UPLOAD THE FILES 48

##### FIG-10.11 FILES BEEN UPLOADED 49

##### FIG-10.12 CHOOSE COULD SERVER 49

##### FIG-10.13 RATING FOR CSP 50

##### FIG-10.14 FEED BACK DETAILS 50

##### FIG-10.15 RATING FOR COLUD STORAGE 51

##### FIG-10.16 SEARCHING THE FILES 51

##### FIG-10.17 TRANSACTION DETAILS 52

##### FIG-10.18 SENDER KEY TO THE USER 52

##### FIG-10.19 KEY VERIFICATION PROCESS 53

##### FIG-10.20 ENTERING THE KEY 53

##### FIG-10.21 REQUESTED THE DETAILS 54

##### FIG-10.22 RECOMMEND COULD STORAGE 54

##### FIG-10.23 BOOTSTRAP EXAMPLE-UPLOAD THE FILE HERE 55

##### FIG-10.24 THIRD PARTY CANNOT ACCESS THE FILE 55

**HEART DISEASE IDENTIFICATION METHOD USING MACHINE LEARNING IN E-HEALTHCARE**

**Abstract** **:**

A stroke is a medical condition in which poor blood flow to the brain results in cell death. It is now a day a leading cause of death all over the world. Several risk factors believe to be related to the cause of stroke has been found by inspecting the affected individuals. Using these risk factors, a number of works have been carried out for predicting and classifying stroke diseases. Most of the models are based on data mining and machine learning algorithms. In this work, we have used four machine learning algorithms to detect the type of stroke that can possibly occur or occurred form a person’s physical state and medical report data. We have collected a good number of entries from the hospitals and use them to solve our problem. The classification result shows that the result is satisfactory and can be used in real time medical report. We believe that machine learning algorithms can help better understanding of diseases and can be a good healthcare companion. Index Terms—Stroke, machine learning, WEKA, Naive Bayes, J48, k-NN, Random Forest.

**1. INTRODUCTION**

A stroke occurs due to poor blood flow to the brain which results in cell death. Two main types of stroke are ischemic stroke and hemorrhagic stroke. Ischemic stroke occurs due to lack of blood flow and hemorrhagic stroke occurs due to bleeding [1]. Another type of stroke is transient ischemic attack. Ischemic stroke has two categories- embolic stroke and thrombotic stroke. An embolic stroke occurs by forming a clot in any part of the body and moves toward the brain and blocks blood flow. A thrombotic stroke caused by a clot that weakens blood flow in an artery. Hemorrhagic stroke is classified into two types- subarachnoid hemorrhage and intracerebral hemorrhage. Transient ischemic attack is also known as ”ministroke”.

A large number of people lose their life due to stroke and it is increasing in developing countries [3]. There are several stroke risk factors that regulate different types of stroke. Predictive algorithms help to understand the relation between these risk factors to types of strokes. The machine learning algorithm can improve patients’ health through early detection and treatment. We have used several machine learning algorithms to detect the type of stroke that can occur in a patient or already occurred from their clinical report and statistical data. We have built a stroke dataset by collecting data from various sources validated by medical experts. Then the dataset was processed to be used with the machine learning algorithms. We have built several models of classification. The result of the models is satisfactory and can be used in a realtime patient’s stroke classification.

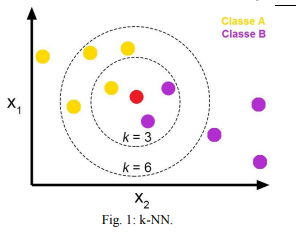
In recent years, there were published different works based on Machine Learning algorithms. Some of them are discussed in here: Govindarajan et al. used Artificial Neural Networks (ANN), Support Vector Machine (SVM), Decision Tree, Logistic Regression, and ensemble methods (Bagging and Boosting) to classify the stroke disease [2]. They have collected the data from Sugam Multispeciality Hospital, India which contains information about 507 stroke patients ranging from 35 to 90 years of age. The novelty of their work is in the data processing phase, where an algorithm called novel stemmer was used to attain the dataset. In their collected dataset, 91.52% of patients were affected by ischemic stroke and only 8.48% of patients were affected by hemorrhagic stroke. Among the mentioned algorithms, Artificial Neural Networks with stochastic gradient descent learning algorithm have the highest accuracy with 95.3% for classifying stroke. Jeena and Kumar proposed a model based on Support Vector Machine for stroke prediction [4]. they have collected data from International Stroke trial Database [5]. The dataset contains 12 risks factors (attributes). They have used 350 samples for their work. For training purpose 300 samples and for testing 50 samples were used. Different kernel functions like polynomial, quadratic, radial basis function and linear functions were applied. The highest accuracy of 91% was found with the linear kernel which gives the balance measure F1-score F-measure 91.7.

Singh and Choudhary developed a model with Artificial Neural Network (ANN) for stroke prediction [6]. They have collected datasets from the Cardiovascular Health Study (CHS) database. Three datasets were constructed which contains 212 strokes (all three) and 52, 69, 79 nonstroke respectively. The final dataset contains 357 attributes and 1824 entities with 212 occurrences of stroke. During feature selection, the C4.5 decision tree algorithm was used and Principle Component Analysis (PCA) for dimension reduction. In ANN implementation they have used Back Propagation learning method. They have got the accuracy as 95%, 95.2% and 97.7% for the three datasets respectively.

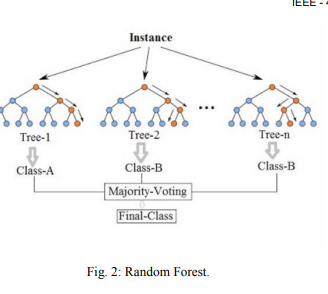
Adam et al. have been developed a classification model for ischemic stroke using decision tree algorithm and knearest neighbor (k-NN)[7]. Their dataset was collected from several hospitals and medical centers in Sudan which is the first dataset for ischemic disease in Sudan. It contains 15 features and information about 400 patients. The results of the experiment show that the performance of decision tree classification is higher than the performance of k-NN algorithm.

Sudha et al. used the Decision Tree, Bayesian Classifier, and Neural Network for stroke classification [8]. Their dataset contains 1000 records. PCA algorithm was used for dimensionality reduction. In ten rounds of each algorithm, they have got the highest accuracy as 92%, 91%, and 94% in Neural Network, Naive Bayes classifier, and Decision tree algorithm respectively. Some of the methods like [4] and [7] use a very small dataset. Govindarajan et al. [2] have predicted only two classes of stroke. Therefore we have proposed a method which uses a large dataset with four classes of stroke.

Our database contains string values which cannot be processed by WEKA. Therefore we had to integer encoding for string values. For example, we have replaced the string ”Male”



with 0 and ”Female” with 1 and so on. Some attributes are missing in the dataset. Some of the attributes do not apply for the individuals i.e. N/A. We replaced them with zero ”0” for avoiding the null value exception. We also removed unnecessary information like ”3 times” used with the frequency of vomiting replaced by only 3 etc. Data preprocessing example is shown in Table II. B. Data mining process Waikato Environment for Knowledge Analysis (WEKA) is a machine learning toolkit, developed and maintained by the University of Waikato, New Zealand [10]. Previous studies show that WEKA is a very reliable suite for machine learning. A large number of similar works has been carried out using

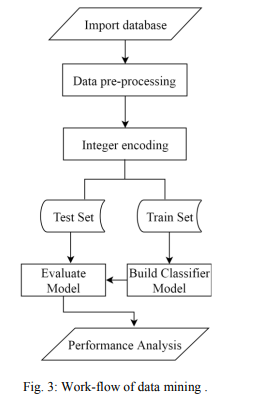


weka and they have found it advantageous [11] [12] [13] [14]. We have used the built-in algorithms in WEKA for stroke disease detection like Naive Bayes, Random Forest, and J48. These algorithms are described previously. First, we import the data from the stroke database. After pre-processing and integer encoding we apply WEKA to classify the strokes. The following steps have been performed for stroke detection in WEKA: • Data pre-processing and visualization .

• Attribute selection

• Test set and train set splitting

• Classification using different algorithms



The work-flow of data mining is given in Fig. 3

**2. LITERATURE SURVEY**

**2.1. Thrombophilia testing in young patients with ischemic stroke :**

The possible significance of thrombophilia in ischemic stroke remains controversial. We aimed to study inherited and acquired thrombophilias 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 prevalences 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 prevalences for the general population were obtained from published data.

**2.2. Classification of stroke disease using machine learning algorithms :**

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 Multispecialty 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.

**2.3**. **Stroke prediction using svm :**

Early diagnosis of stroke is essential for timely prevention and treatment. Investigation shows that measures extracted from various risk parameters carry valuable information for the prediction of stroke. 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). In this work, we have implemented SVM with different kernel functions and found that linear kernel gave an accuracy of 90 %.

**2.4. The international stroke trial database :**

The International Stroke Trial (IST) is one of the largest randomized trials ever conducted on individual patients in acute stroke. The IST dataset includes data on 19 435 patients with acute stroke, with 99% complete follow-up. Over 26.4% patients were aged over 80 years at study entry. Background stroke care was limited and none of the patients received thrombolytic therapy. This clinical trial was conducted between 1991 and 1996 and a pilot phase between 1991 to and 1993. This study is a large, prospective, randomized controlled trial, with 100% complete baseline data and over 99% complete follow-up data. For each randomized patient, data were extracted on the variables assessed at randomization; the early outcome point was 14-days after randomization or prior discharge, and at 6-months and provided as an analyzable database. The aim of the trial was to establish whether early administration of aspirin, heparin, both or neither influenced the clinical course of an acute ischaemic stroke.

**2.5. Effective analysis and predictive model of stroke disease using classification methods :**

In today‟s world data mining plays a vital role for prediction of diseases in medical industry. Stroke is a life-threatening disease that has been ranked third leading cause of death in states and in developing countries. The stroke is a leading cause of serious, long term disability in US. The time taken to recover from stroke disease depends on patients‟ severity. Number of work has been carried out for predicting various diseases by comparing the performance of predictive data mining. Here the classification algorithms like Decision Tree, Naive Bayes and Neural Network is used for predicting the presence of stroke disease with related number of attributes. In our work, principle component analysis algorithm is used for reducing the dimensions and it determines the attributes involving more towards the prediction of stroke disease and predicts whether the patient is suffering from stroke disease or not.

**3. SYSTEM ANALYSIS:**

**3.1. HARDWARE REQUIREMENTS:**

* System : MINIMUM i3.
* Hard Disk : 40 GB.
* Ram : 4 GB.

**3.2. SOFTWARE REQUIREMENTS:**

* **Operating System:** Windows 8
* **Coding Language**: Python 3.7

**4. FEASIBILITY STUDY**

The feasibility of the project is analyzed in this phase and business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company. For feasibility analysis, some understanding of the major requirements for the system is essential.

Three key considerations involved in the feasibility analysis are

* ECONOMICAL FEASIBILITY
* TECHNICAL FEASIBILITY
* SOCIAL FEASIBILITY

**4.1. 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.

### **4.2. TECHNICAL FEASIBILIT**Y

### 

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.

**4.3. 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.

**5. SYSTEM DESIGN**

**5.1 UML DIAGRAMS**

UML stands for Unified Modeling Language. UML is a standardized general-purpose modeling language in the field of object-oriented software engineering. The standard is managed, and was created by, the Object Management Group.

The goal is for UML to become a common language for creating models of object oriented computer software. In its current form UML is comprised of two major components: a Meta-model and a notation. In the future, some form of method or process may also be added to; or associated with, UML.

The Unified Modeling Language is a standard language for specifying, Visualization, Constructing and documenting the artifacts of software system, as well as for business modeling and other non-software systems.

The UML represents a collection of best engineering practices that have proven successful in the modeling of large and complex systems.

The UML is a very important part of developing objects oriented software and the software development process. The UML uses mostly graphical notations to express the design of software projects.

**GOALS:**

The Primary goals in the design of the UML are as follows:

1. Provide users with a ready-to-use, expressive visual modeling Language so that they can develop and exchange meaningful models.
2. Provide extensibility and specialization mechanisms to extend the core concepts.
3. Be independent of particular programming languages and development processes.
4. Provide a formal basis for understanding the modeling language.
5. Encourage the growth of the OO tools market.
6. Support higher level development concepts such as collaborations, frameworks, patterns and components.
7. Integrate best practices.

**USE CASE DIAGRAM:**

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



**CLASS DIAGRAM:**

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



**SEQUENCE DIAGRAM:**

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



**COLLABORATION DIAGRAM**



**6. IMPLEMENTATION:**

**6.1.MODULES:**

To implement this project we have designed following modules

1. Upload Stroke Dataset:

using this module we will upload dataset to application

1. Dataset Preprocessing & Features Selection:

using this module we will clean dataset by replacing missing values with 0 and then apply label encoding algorithm to convert non-numeric values to numeric values and then select features from dataset and then split dataset into train and test where application used 80% data for training and 20% for testing

1. Train Naive Bayes Algorithm:

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

1. Train J48 Algorithm:

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

1. Train KNN Algorithm:

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

1. Train Random Forest Algorithm:

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

1. Train ANN Algorithm:

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

1. Comparison Graph:

using this module we will plot accuracy comparison graph between all algorithms.

**6.2.SAMPLE CODE**

from tkinter import \*

import tkinter

from tkinter import filedialog

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 calculateMetrics(predict, testY, algorithm):

    p = precision\_score(testY, predict,average='macro') \* 100

    r = recall\_score(testY, predict,average='macro') \* 100

    f = f1\_score(testY, predict,average='macro') \* 100

    a = accuracy\_score(testY,predict)\*100

    text.insert(END,algorithm+' Accuracy  : '+str(a)+"\n")

    text.insert(END,algorithm+' Precision : '+str(p)+"\n")

    text.insert(END,algorithm+' Recall    : '+str(r)+"\n")

    text.insert(END,algorithm+' FScore    : '+str(f)+"\n\n")

    accuracy.append(a)

    precision.append(p)

    recall.append(r)

    fscore.append(f)

    text.update\_idletasks()

    LABELS = ['Normal','Stroke']

    conf\_matrix = confusion\_matrix(testY, predict)

    plt.figure(figsize =(6, 6))

    ax = sns.heatmap(conf\_matrix, xticklabels = LABELS, yticklabels = LABELS, annot = True, cmap="viridis" ,fmt ="g");

    ax.set\_ylim([0,2])

    plt.title(algorithm+" Confusion matrix")

    plt.ylabel('True class')

    plt.xlabel('Predicted class')

    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 trainRanfomForest():

    global X\_train, X\_test, y\_train, y\_test, rf

    cls = RandomForestClassifier()

    cls.fit(X\_train, y\_train)

    rf = cls

    predict = cls.predict(X\_test)

    calculateMetrics(predict, y\_test, "Random Forest")

def graph():

    df = pd.DataFrame([['Naive Bayes','Precision',precision[0]],['Naive Bayes','Recall',recall[0]],['Naive Bayes','F1 Score',fscore[0]],['Naive Bayes','Accuracy',accuracy[0]],

                       ['J48','Precision',precision[1]],['J48','Recall',recall[1]],['J48','F1 Score',fscore[1]],['J48','Accuracy',accuracy[1]],

                       ['KNN','Precision',precision[2]],['KNN','Recall',recall[2]],['KNN','F1 Score',fscore[2]],['KNN','Accuracy',accuracy[2]],

                       ['Random Forest','Precision',precision[3]],['Random Forest','Recall',recall[3]],['Random Forest','F1 Score',fscore[3]],['Random Forest','Accuracy',accuracy[3]],

                       ['ANN','Precision',precision[4]],['ANN','Recall',recall[4]],['ANN','F1 Score',fscore[4]],['ANN','Accuracy',accuracy[4]],

                      ],columns=['Parameters','Algorithms','Value'])

    df.pivot("Parameters", "Algorithms", "Value").plot(kind='bar')

    plt.show()

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")

font = ('times', 15, 'bold')

title = Label(main, text='Detection of Stroke Disease using Machine Learning Algorithms', justify=LEFT)

title.config(bg='lavender blush', fg='DarkOrchid1')

title.config(font=font)

title.config(height=3, width=120)

title.place(x=100,y=5)

title.pack()

font1 = ('times', 12, 'bold')

loadButton = Button(main, text="Upload Stroke Dataset", command=loadDataset)

loadButton.place(x=10,y=100)

loadButton.config(font=font1)

preprocessButton = Button(main, text="Dataset Preprocessing & Features Selection", command=preprocessDataset)

preprocessButton.place(x=300,y=100)

preprocessButton.config(font=font1)

nbButton = Button(main, text="Train Naive Bayes Algorithm", command=trainNaiveBayes)

nbButton.place(x=730,y=100)

nbButton.config(font=font1)

dtButton = Button(main, text="Train J48 Algorithm", command=trainDT)

dtButton.place(x=10,y=150)

dtButton.config(font=font1)

knnButton = Button(main, text="Train KNN Algorithm", command=trainKNN)

knnButton.place(x=300,y=150)

knnButton.config(font=font1)

rfButton = Button(main, text="Train Random Forest Algorithm", command=trainRanfomForest)

rfButton.place(x=730,y=150)

rfButton.config(font=font1)

annButton = Button(main, text="Train ANN Algorithm", command=trainANN)

annButton.place(x=10,y=200)

annButton.config(font=font1)

graphButton = Button(main, text="Comparison Graph", command=graph)

graphButton.place(x=300,y=200)

graphButton.config(font=font1)

predictButton = Button(main, text="Predict Disease on Test Data", command=predict)

predictButton.place(x=730,y=200)

predictButton.config(font=font1)

font1 = ('times', 12, 'bold')

text=Text(main,height=20,width=160)

scroll=Scrollbar(text)

text.configure(yscrollcommand=scroll.set)

text.place(x=10,y=250)

text.config(font=font1)

main.config(bg='light coral')

main.mainloop()

**7. SOFTWARE ENVIRONMENT**

# What is Python :-

Below are some facts about Python.

Python is currently the most widely used multi-purpose, high-level programming language.

Python allows programming in Object-Oriented and Procedural paradigms. Python programs generally are smaller than other programming languages like Java.

Programmers have to type relatively less and indentation requirement of the language, makes them readable all the time.

Python language is being used by almost all tech-giant companies like – Google, Amazon, Facebook, Instagram, Dropbox, Uber… etc.

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

* + [Machine Learning](https://www.geeksforgeeks.org/machine-learning/)
  + GUI Applications (like Kivy, Tkinter, PyQt etc. )
  + Web frameworks like Django (used by YouTube, Instagram, Dropbox)
  + Image processing (like Opencv, Pillow)
  + Web scraping (like Scrapy, BeautifulSoup, Selenium)
  + Test frameworks
  + Multimedia

### Advantages of Python :-

Let’s see how Python dominates over other languages.

#### 1. Extensive Libraries

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

#### 2. Extensible

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

#### 3. Embeddable

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

#### 4. Improved Productivity

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

#### 5. IOT Opportunities

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

#### 6. Simple and Easy

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

#### 7. Readable

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

#### 8. Object-Oriented

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

#### 9. Free and Open-Source

Like we said earlier, Python is **freely available.** But not only can you[**download Python**](https://data-flair.training/blogs/install-python-windows/) for free, but you can also download its source code, make changes to it, and even distribute it. It downloads with an extensive collection of libraries to help you with your tasks.

#### 10. Portable

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

#### 11. Interpreted

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

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

### **Advantages of Python Over Other Languages**

#### 1. Less Coding

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

#### 2. Affordable

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

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

#### 3. Python is for Everyone

Python code can run on any machine whether it is Linux, Mac or Windows. Programmers need to learn different languages for different jobs but with Python, you can professionally build web apps, perform data analysis and [**machine learning**](https://data-flair.training/blogs/machine-learning-tutorials-home/), automate things, do web scraping and also build games and powerful visualizations. It is an all-rounder programming language.

### **Disadvantages of Python**

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

#### 1. Speed Limitations

We have seen that Python code is executed line by line. But since [Python](https://www.python.org/) is interpreted, it often results in **slow execution**. This, however, isn’t a problem unless speed is a focal point for the project. In other words, unless high speed is a requirement, the benefits offered by Python are enough to distract us from its speed limitations.

#### 2. Weak in Mobile Computing and Browsers

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

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

#### 3. Design Restrictions

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

#### 4. Underdeveloped Database Access Layers

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

#### 5. Simple

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

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

**History of Python : -**

What do the alphabet and the programming language Python have in common? Right, both start with ABC. If we are talking about ABC in the Python context, it's clear that the programming language ABC is meant. ABC is a general-purpose programming language and programming environment, which had been developed in the Netherlands, Amsterdam, at the CWI (Centrum Wiskunde &Informatica). The greatest achievement of ABC was to influence the design of Python.Python was conceptualized in the late 1980s. Guido van Rossum worked that time in a project at the CWI, called Amoeba, a distributed operating system. In an interview with Bill Venners1, Guido van Rossum said: "In the early 1980s, I worked as an implementer on a team building a language called ABC at Centrum voor Wiskunde en Informatica (CWI). I don't know how well people know ABC's influence on Python. I try to mention ABC's influence because I'm indebted to everything I learned during that project and to the people who worked on it."Later on in the same Interview, Guido van Rossum continued: "I remembered all my experience and some of my frustration with ABC. I decided to try to design a simple scripting language that possessed some of ABC's better properties, but without its problems. So I started typing. I created a simple virtual machine, a simple parser, and a simple runtime. I made my own version of the various ABC parts that I liked. I created a basic syntax, used indentation for statement grouping instead of curly braces or begin-end blocks, and developed a small number of powerful data types: a hash table (or dictionary, as we call it), a list, strings, and numbers."

**What is Machine Learning : -**

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

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

**Categories Of Machine Leaning :-**

At the most fundamental level, machine learning can be categorized into two main types: supervised learning and unsupervised learning.

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

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

## Need for Machine Learning

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

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

## Challenges in Machines Learning :-

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

**Quality of data** − Having good-quality data for ML algorithms is one of the biggest challenges. Use of low-quality data leads to the problems related to data preprocessing and feature extraction.

**Time-Consuming task** − Another challenge faced by ML models is the consumption of time especially for data acquisition, feature extraction and retrieval.

**Lack of specialist persons** − As ML technology is still in its infancy stage, availability of expert resources is a tough job.

**No clear objective for formulating business problems** − Having no clear objective and well-defined goal for business problems is another key challenge for ML because this technology is not that mature yet.

**Issue of overfitting & underfitting** − If the model is overfitting or underfitting, it cannot be represented well for the problem.

**Curse of dimensionality** − Another challenge ML model faces is too many features of data points. This can be a real hindrance.

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

## Applications of Machines Learning :-

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

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

# How to Start Learning Machine Learning?

Arthur Samuel coined the term **“Machine Learning”** in 1959 and defined it as a **“Field of study that gives computers the capability to learn without being explicitly programmed”.**

And that was the beginning of Machine Learning! In modern times, Machine Learning is one of the most popular (if not the most!) career choices. According to [Indeed](http://blog.indeed.com/2019/03/14/best-jobs-2019/), Machine Learning Engineer Is The Best Job of 2019 with a *344%* growth and an average base salary of **$146,085** per year.

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

### **How to start learning ML?**

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

### Step 1 – Understand the Prerequisites

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

#### (a) Learn Linear Algebra and Multivariate Calculus

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

#### (b) Learn Statistics

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

#### (c) Learn Python

Some people prefer to skip Linear Algebra, Multivariate Calculus and Statistics and learn them as they go along with trial and error. But the one thing that you absolutely cannot skip is [Python](https://www.geeksforgeeks.org/python-programming-language/)! While there are other languages you can use for Machine Learning like R, Scala, etc. Python is currently the most popular language for ML. In fact, there are many Python libraries that are specifically useful for Artificial Intelligence and Machine Learning such as [Keras](https://keras.io/), [TensorFlow](https://www.tensorflow.org/), [Scikit-learn](https://scikit-learn.org/stable/), etc.

So if you want to learn ML, it’s best if you learn Python! You can do that using various online resources and courses such as [**Fork Python**](https://practice.geeksforgeeks.org/courses/fork-python) available Free on GeeksforGeeks.

### **Step 2 – Learn Various ML Concepts**

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

#### (a) Terminologies of Machine Learning

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

#### (b) Types of Machine Learning

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

### **Advantages of Machine learning :-**

#### 1. Easily identifies trends and patterns -

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

#### 2. No human intervention needed (automation)

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

#### 3. Continuous Improvement

As [**ML algorithms**](https://data-flair.training/blogs/machine-learning-algorithms/) gain experience, they keep improving in accuracy and efficiency. This lets them make better decisions. Say you need to make a weather forecast model. As the amount of data you have keeps growing, your algorithms learn to make more accurate predictions faster.

#### 4. Handling multi-dimensional and multi-variety data

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

#### 5. Wide Applications

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

### **Disadvantages of Machine Learning :-**

#### 1. Data Acquisition

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

#### 2. Time and Resources

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

#### 3. Interpretation of Results

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

#### 4. High error-susceptibility

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

**Python Development Steps : -**

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

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

**Purpose :-**

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

**Python**

Python is an interpreted high-level programming language for general-purpose programming. Created by Guido van Rossum and first released in 1991, Python has a design philosophy that emphasizes code readability, notably using significant whitespace.

Python features a dynamic type system and automatic memory management. It supports multiple programming paradigms, including object-oriented, imperative, functional and procedural, and has a large and comprehensive standard library.

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

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

**Modules Used in Project :-**

**Tensorflow**

TensorFlow is a [free](https://en.wikipedia.org/wiki/Free_software) and [open-source](https://en.wikipedia.org/wiki/Open-source_software) [software library for dataflow and differentiable programming](https://en.wikipedia.org/wiki/Library_(computing)) across a range of tasks. It is a symbolic math library, and is also used for [machine learning](https://en.wikipedia.org/wiki/Machine_learning) applications such as [neural networks](https://en.wikipedia.org/wiki/Neural_networks). It is used for both research and production at [Google](https://en.wikipedia.org/wiki/Google).‍

TensorFlow was developed by the [Google Brain](https://en.wikipedia.org/wiki/Google_Brain) team for internal Google use. It was released under the [Apache 2.0](https://en.wikipedia.org/wiki/Apache_License) [open-source license](https://en.wikipedia.org/wiki/Open-source_license) on November 9, 2015.

**Numpy**

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

It is the fundamental package for scientific computing with Python. It contains various features including these important ones:

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

Besides its obvious scientific uses, Numpy can also be used as an efficient multi-dimensional container of generic data. Arbitrary 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, analytics, etc.

**Matplotlib**

Matplotlib is a Python 2D plotting library which produces publication quality figures in a variety of hardcopy formats and interactive environments across platforms. Matplotlib can be used in Python scripts, the Python and [IPython](http://ipython.org/) shells, the [Jupyter](http://jupyter.org/) Notebook, web application servers, and four graphical user interface toolkits. Matplotlib tries to make easy things easy and hard things possible. You can generate plots, histograms, power spectra, bar charts, error charts, scatter plots, etc., with just a few lines of code. For examples, see the [sample plots](https://matplotlib.org/tutorials/introductory/sample_plots.html) and [thumbnail gallery](https://matplotlib.org/gallery/index.html).

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

**Scikit – learn**

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

Python is an interpreted high-level programming language for general-purpose programming. Created by Guido van Rossum and first released in 1991, Python has a design philosophy that emphasizes code readability, notably using significant whitespace.

Python features a dynamic type system and automatic memory management. It supports multiple programming paradigms, including object-oriented, imperative, functional and procedural, and has a large and comprehensive standard library.

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

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

**Install Python Step-by-Step in Windows and Mac :**

Python a versatile programming language doesn’t come pre-installed on your computer devices. Python was first released in the year 1991 and until today it is a very popular high-level programming language. Its style philosophy emphasizes code readability with its notable use of great whitespace.

The object-oriented approach and language construct provided by Python enables programmers to write both clear and logical code for projects. This software does not come pre-packaged with Windows.

## How to Install Python on Windows and Mac :

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

**Note:** The python version 3.7.4 cannot be used on Windows XP or earlier devices.

Before you start with the installation process of Python. First, you need to know about your **System Requirements**. Based on your system type i.e. operating system and based processor, you must download the python version. My system type is a **Windows 64-bit operating system**. So the steps below are to install python version 3.7.4 on Windows 7 device or to install Python 3. [Download the Python Cheatsheet here.](https://myelearninghub.com/python-cheat-sheet/)The steps on how to install Python on Windows 10, 8 and 7 are **divided into 4 parts** to help understand better.

### Download the Correct version into the system

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



Now, check for the latest and the correct version for your operating system.

**Step 2:** Click on the Download Tab.

****

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

****

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

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



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

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

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

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

### Installation of Python

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



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



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



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

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

### Verify the Python Installation

**Step 1:** Click on Start

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



**Step 3:** Open the Command prompt option.

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



**Step 5:** You will get the answer as 3.7.4

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

### Check how the Python IDLE works

**Step 1:** Click on Start

**Step 2:** In the Windows Run command, type “python idle”.



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

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



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

**Step 6:** Now for e.g. **enter print**

**8. SYSTEM TEST**

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub assemblies, assemblies and/or a finished product It is the process of exercising software with the intent of ensuring that the Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

### **TYPES OF TESTS**

**Unit testing**

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

**Integration testing**

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfaction, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

**Functional test**

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centered on the following items:

Valid Input : identified classes of valid input must be accepted.

Invalid Input : identified classes of invalid input must be rejected.

Functions : identified functions must be exercised.

Output : identified classes of application outputs must be exercised.

Systems/Procedures : interfacing systems or procedures must be invoked.

Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined.

**System Test**

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

**White Box Testing**

White Box Testing is a testing in which in which the software tester has knowledge of the inner workings, structure and language of the software, or at least its purpose. It is purpose. It is used to test areas that cannot be reached from a black box level.

**Black Box Testing**

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or requirements document, such as specification or requirements document. It is a testing in which the software under test is treated, as a black box .you cannot “see” into it. The test provides inputs and responds to outputs without considering how the software works.

**Unit Testing**

Unit testing is usually conducted as part of a combined code and unit test phase of the software lifecycle, although it is not uncommon for coding and unit testing to be conducted as two distinct phases.

**Test strategy and approach**

Field testing will be performed manually and functional tests will be written in detail.

**Test objectives**

* All field entries must work properly.
* Pages must be activated from the identified link.
* The entry screen, messages and responses must not be delayed.

**Features to be tested**

* Verify that the entries are of the correct format
* No duplicate entries should be allowed
* All links should take the user to the correct page.

# Integration Testing

Software integration testing is the incremental integration testing of two or more integrated software components on a single platform to produce failures caused by interface defects.

The task of the integration test is to check that components or software applications, e.g. components in a software system or – one step up – software applications at the company level – interact without error.

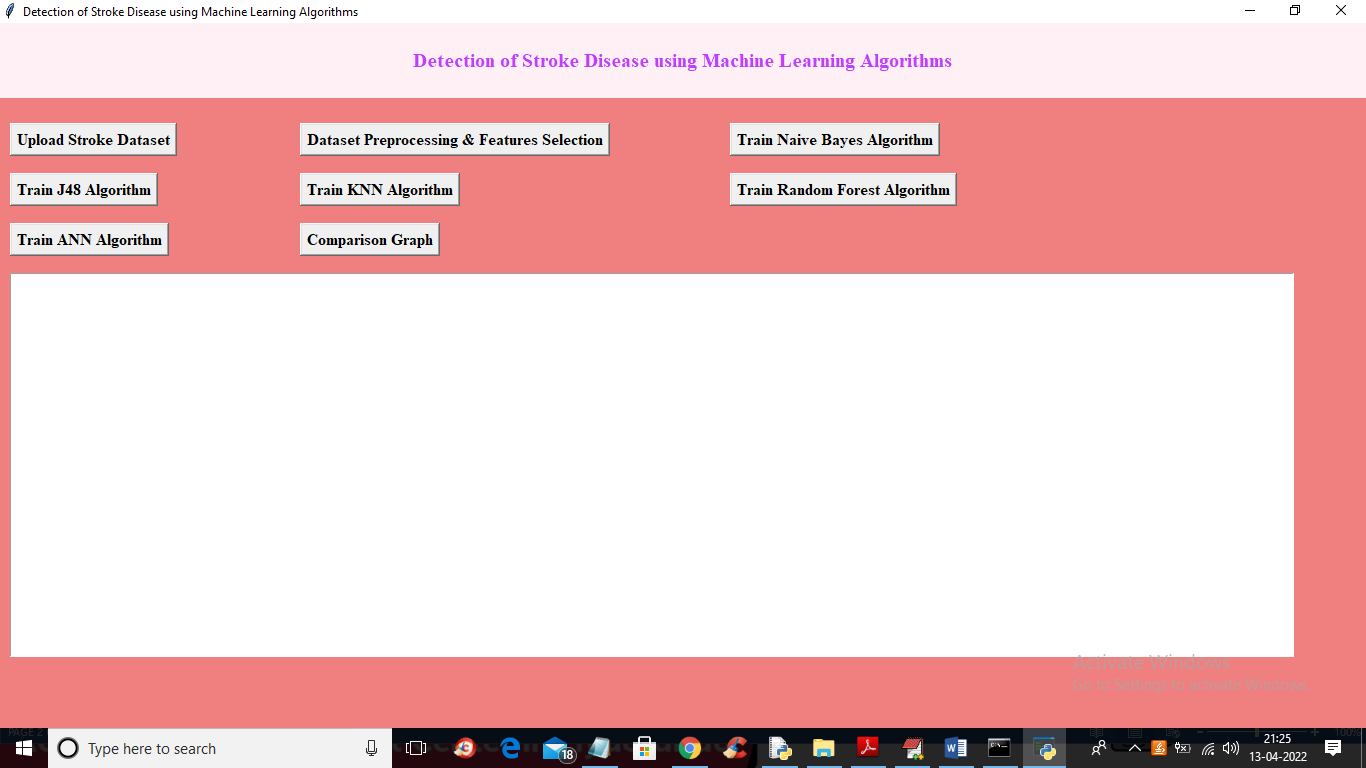
**Test Results:** All the test cases mentioned above passed successfully. No defects encountered.

**Acceptance Testing**

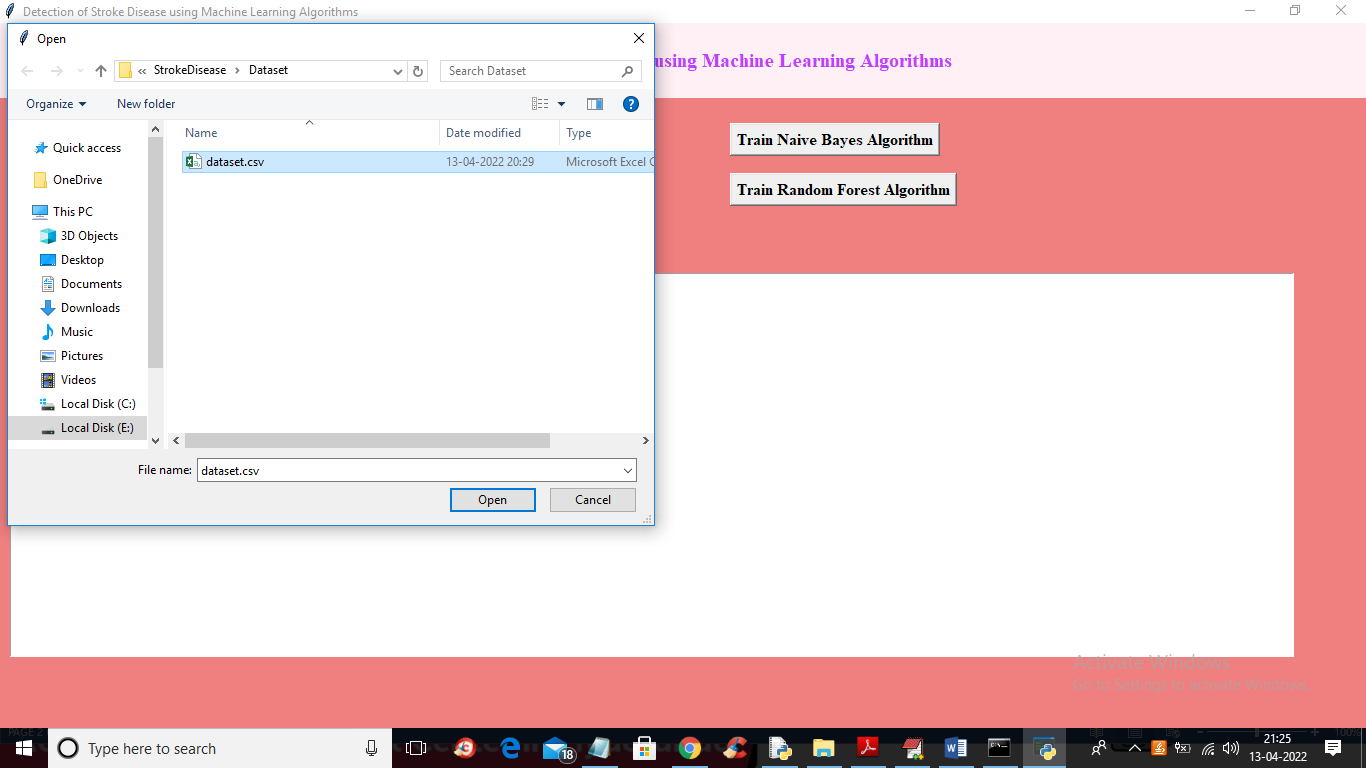
User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements.

**Test Results:** All the test cases mentioned above passed successfully. No defects encountered.

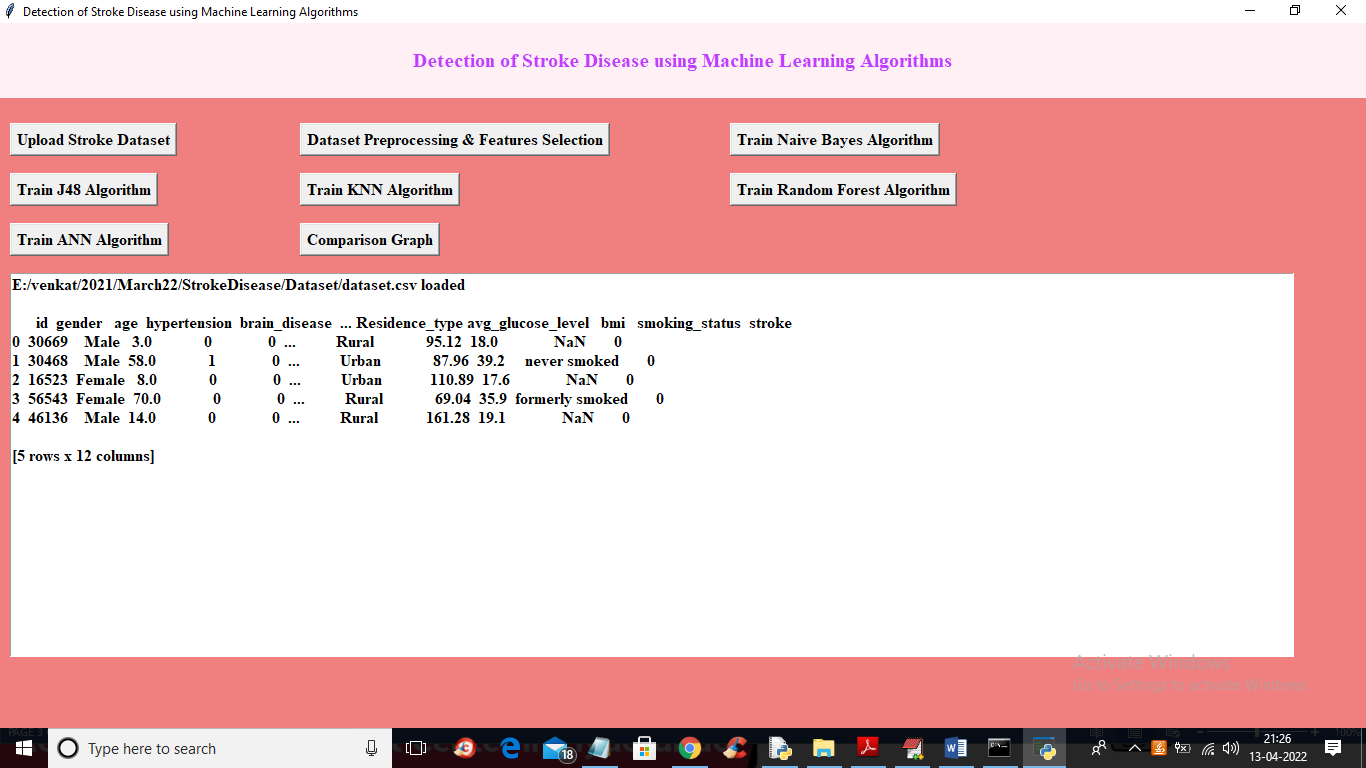
**9.SCREENSHOTS**



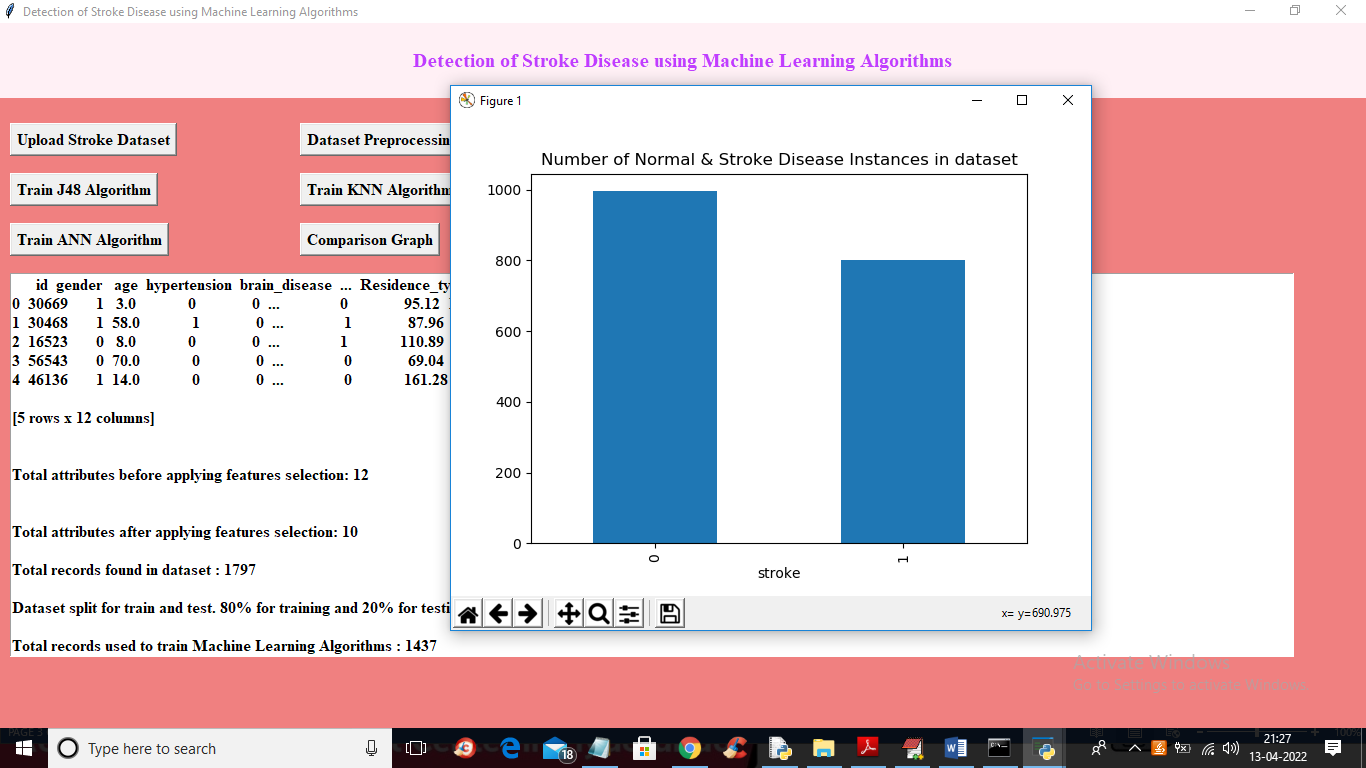
In above screen click on ‘Upload Stroke Dataset’ button to upload dataset



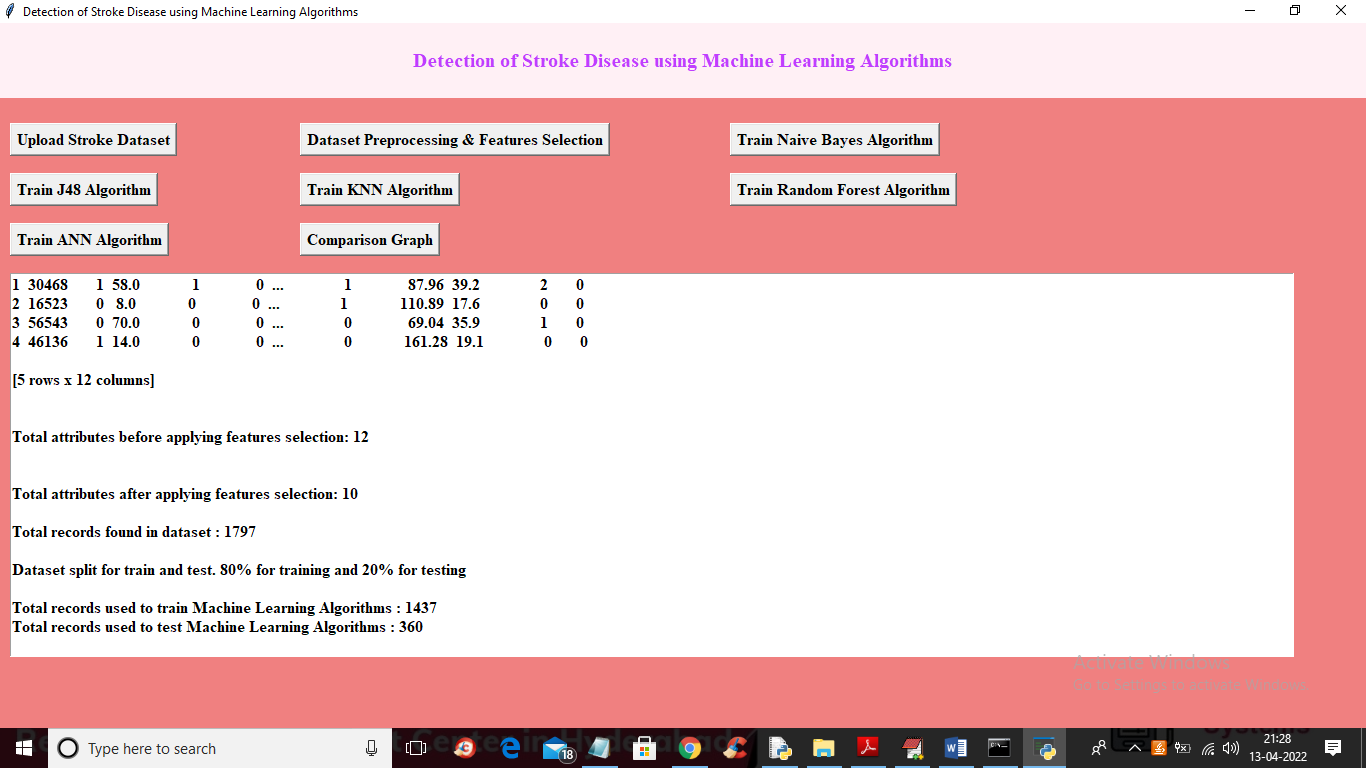
In above screen selecting and uploading dataset.csv file and then click on ‘Open’ button to load dataset and to get below output



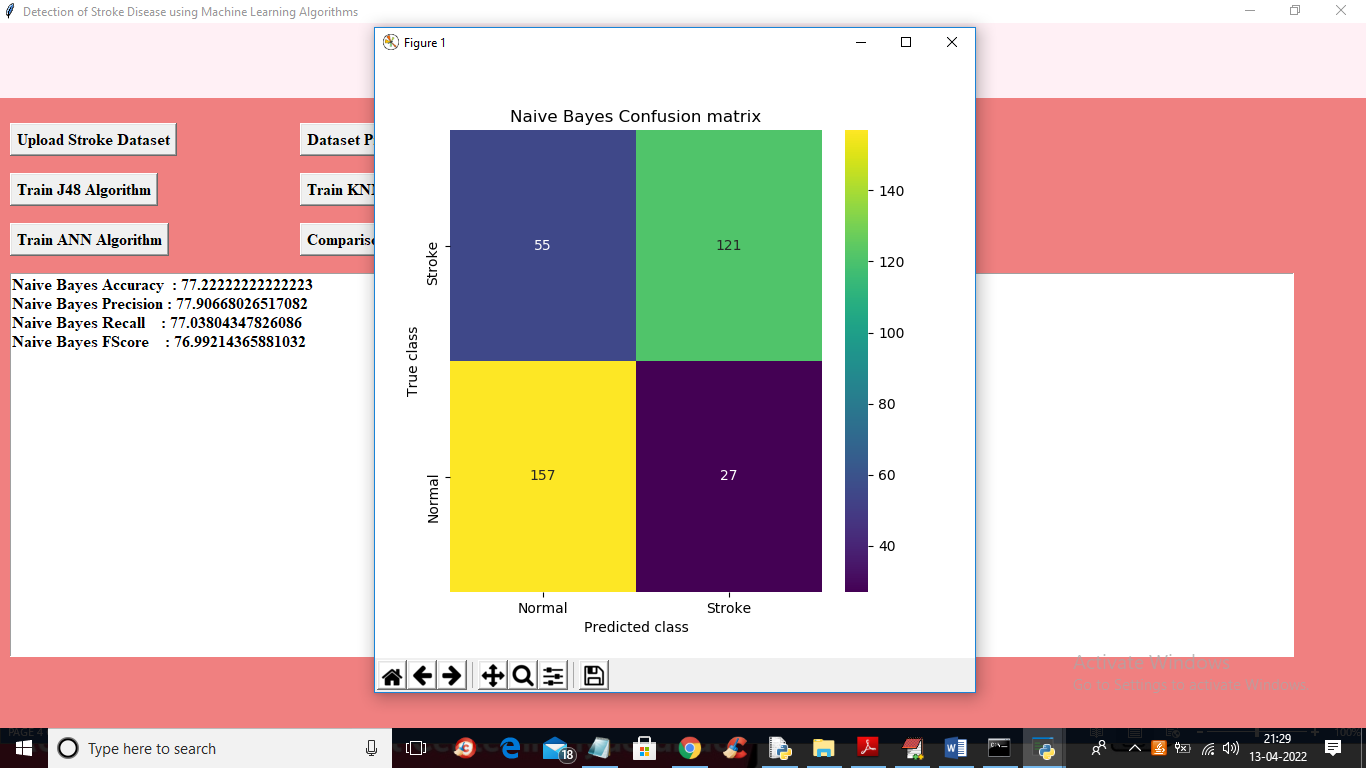
In above screen we can see dataset loaded and dataset contains so many missing and non-numeric data so click on ‘Dataset Preprocessing & Features Selection’ button to process dataset and to get below output



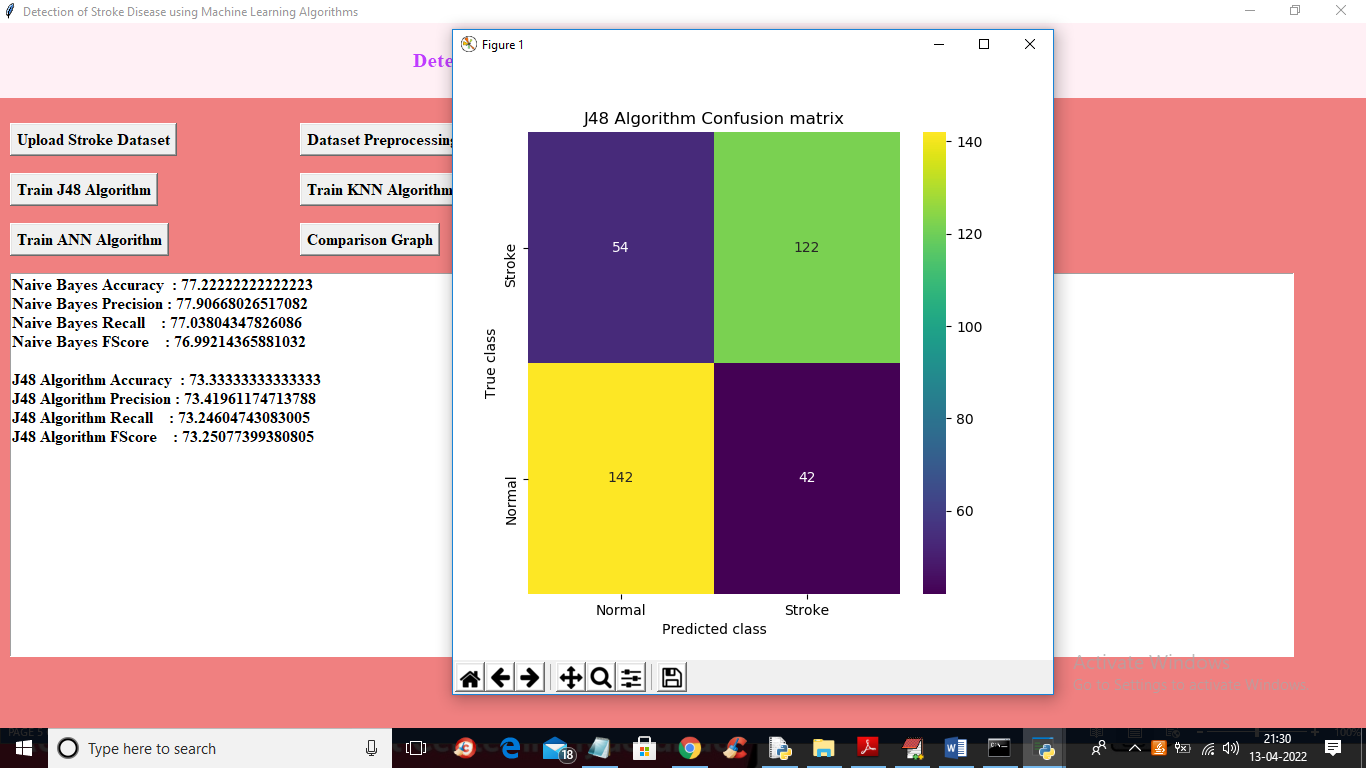
In above graph x-axis represents 0 (normal) and 1 (stroke) and y-axis represents number of instances available in those categories in dataset and now close above graph and see below screen



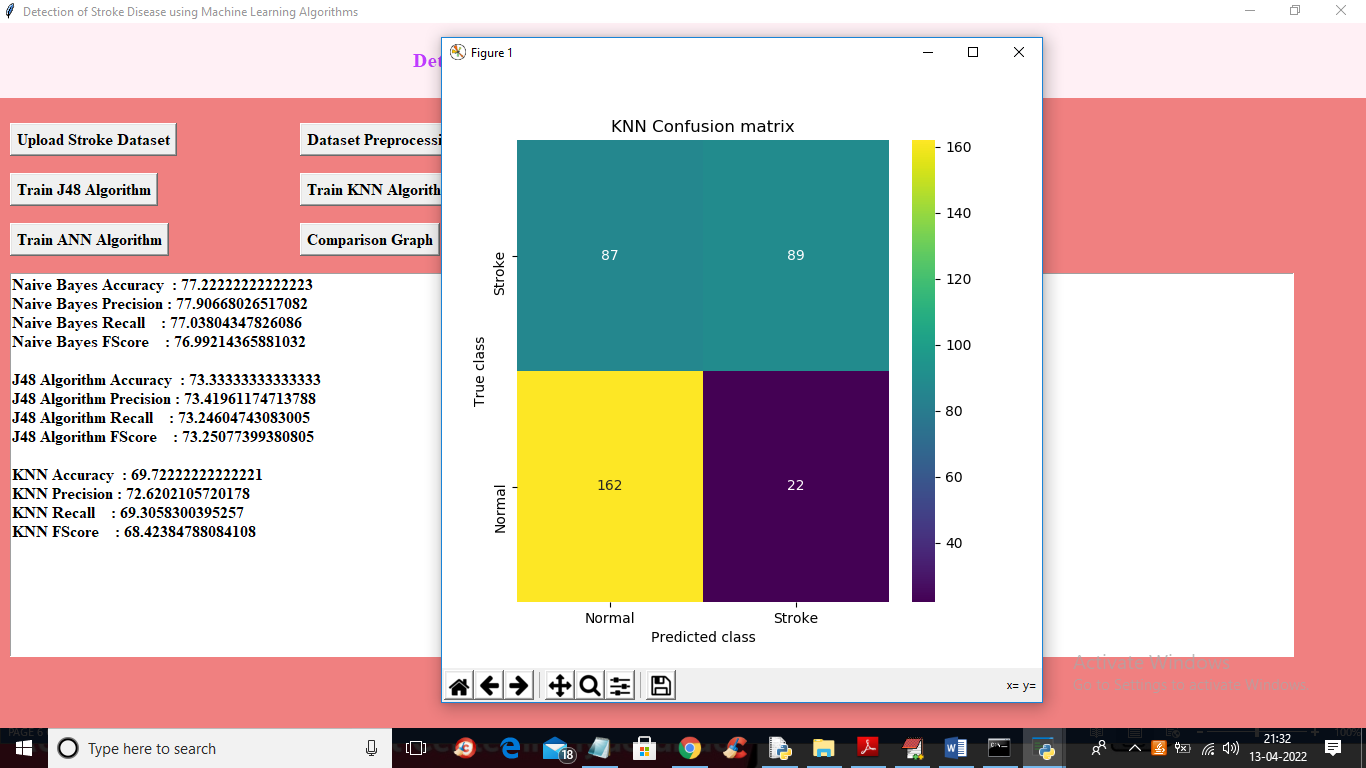
In above screen we can see all dataset converted to numeric format and then split dataset into train and test and now click on ‘Train Naïve Bayes Algorithm’ button to train Naïve Bayes on above dataset and get below output



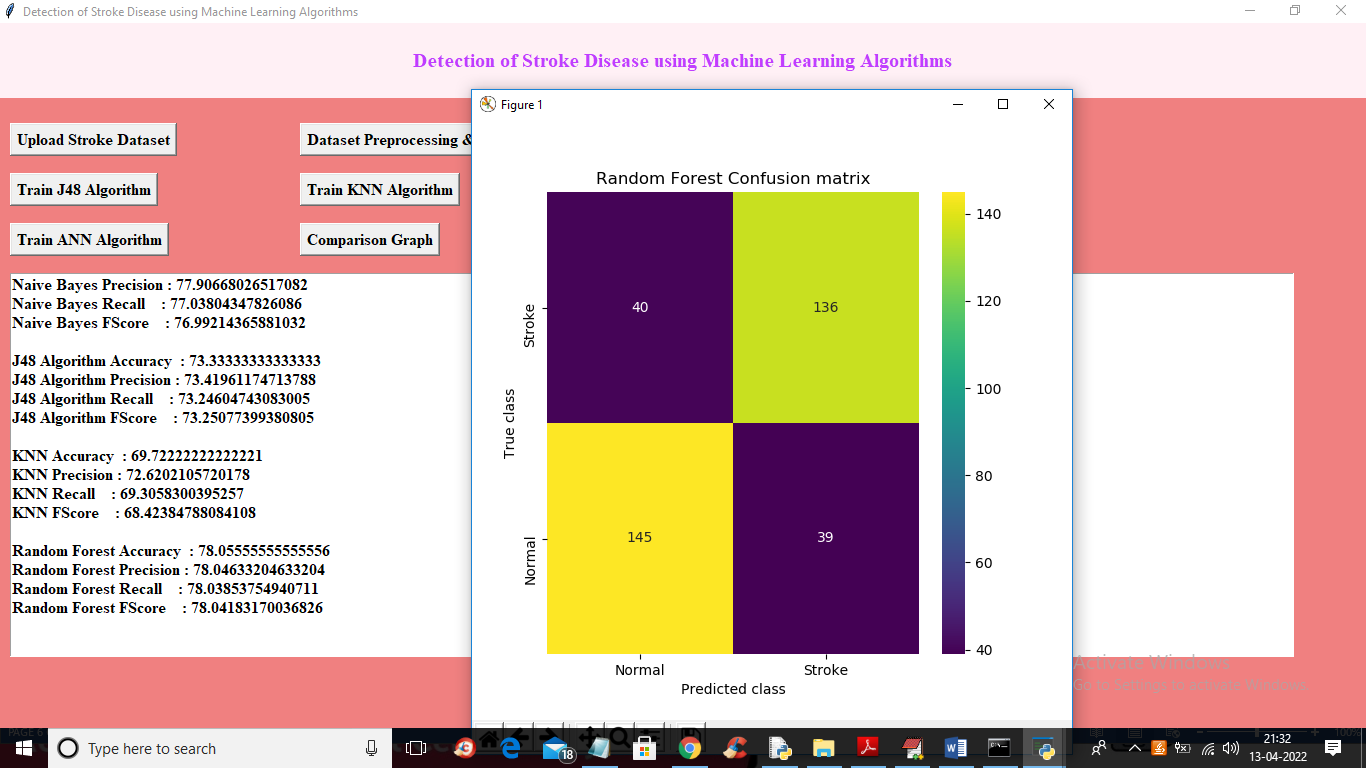
In above screen with Naïve Bayes we got 77% accuracy and in confusion matrix graph we can see number of correct and incorrect prediction by Naïve Bayes. Now click on ‘Train J48 Algorithm’ button to get below output



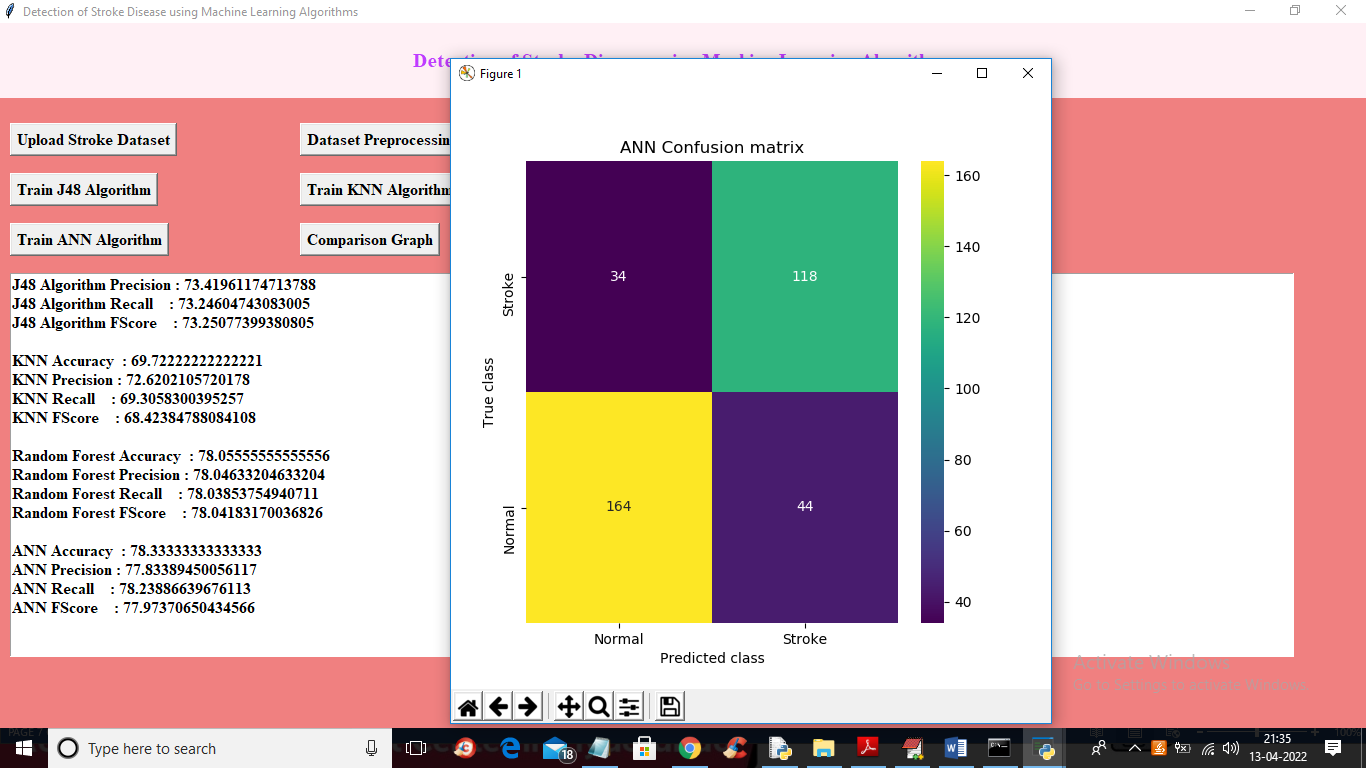
In above screen with J48 we got 73% accuracy and in confusion matrix graph we can see number of correct and incorrect prediction by J48.Now close above Graph and then click on ‘Run KNN Algorithm’ button to get below output



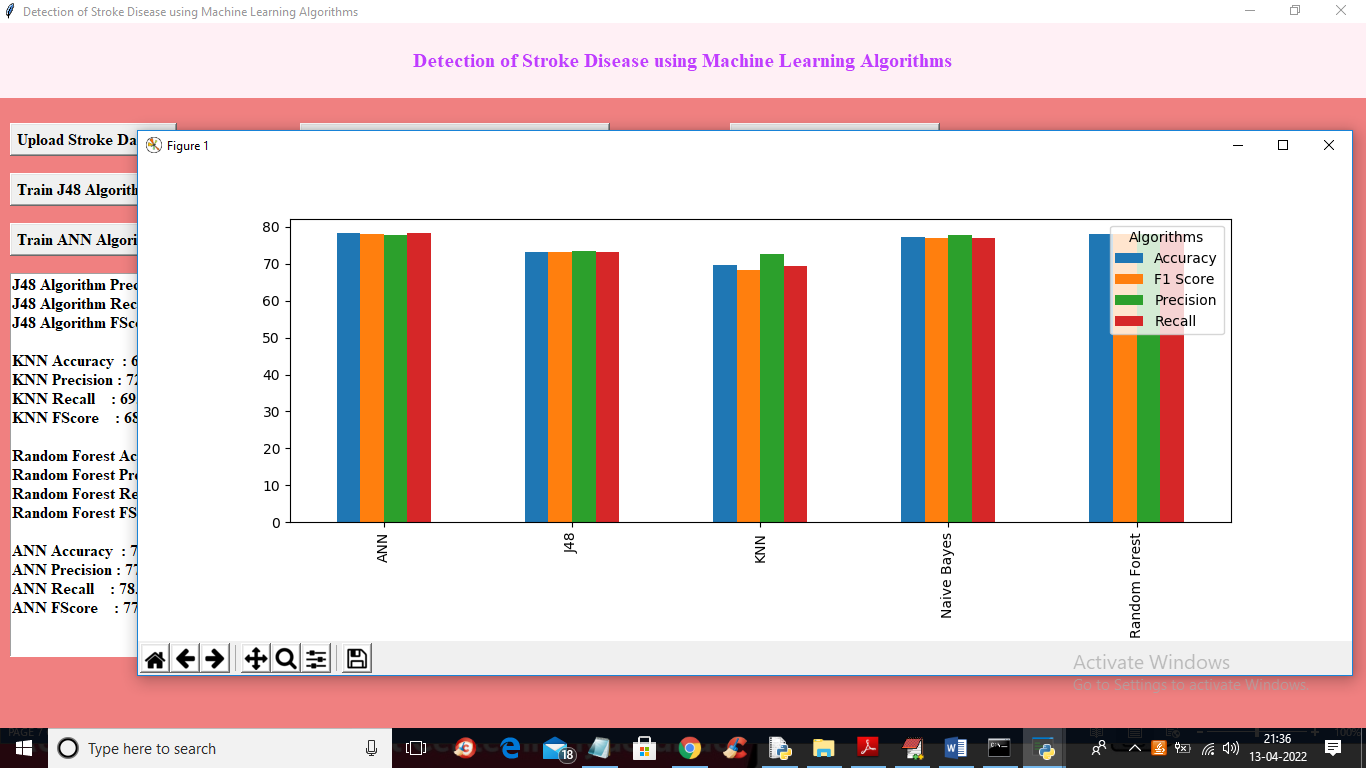
In above screen with KNN we got 69% accuracy and in confusion matrix graph we can see number of correct and incorrect prediction by KNN.Now close above Graph and then click on ‘Run Random Forest Algorithm’ button to get below output



In above screen with Random Forest we got 78% accuracy and in confusion matrix graph we can see number of correct and incorrect prediction by Random Forest. Now close above Graph and then click on ‘Run ANN Algorithm’ button to get below output



In above screen with ANN we got 78.33% accuracy and in confusion matrix graph we can see number of correct and incorrect prediction by ANN and in all algorithm ANN got high accuracy. Now close above Graph and then click on ‘Comparison Graph’ button to get below graph



In above graph x-axis represents algorithm names and y-axis represents accuracy and other metrics like precision, recall etc. different colour bar represents different metrics and in all algorithms ANN got high accuracy.

**10. FUTURE ENHANCEMENTS**

Here are some potential enhancements for the future:

**1. Advanced Machine Learning Models**

* **Deep Learning Techniques:** Use advanced architectures like Convolutional Neural Networks (CNNs) for imaging data or Recurrent Neural Networks (RNNs) for time-series data like ECG signals.
* **Ensemble Models:** Combine multiple machine learning models to improve prediction accuracy by reducing biases and variance.
* **Explainable AI (XAI):** Develop interpretable models that provide insights into how predictions are made, enhancing trust among clinicians.

**2. Integration of Multi-Modal Data**

* **Comprehensive Data Sources:** Integrate data from electronic health records (EHRs), wearable devices, imaging tests (e.g., echocardiograms), and genomic information.
* **Real-Time Monitoring:** Use wearable devices for continuous data collection and real-time disease prediction.
* **Feature Fusion:** Use advanced algorithms to combine diverse features (e.g., lifestyle, environment, and genetics) for holistic analysis.

**3. Personalized Healthcare**

* **Risk Profiling:** Build personalized risk prediction models based on patient demographics, medical history, and lifestyle factors.
* **Adaptive Models:** Develop adaptive learning systems that refine predictions based on the patient’s evolving health data.

**4. Edge Computing and IoT**

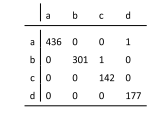
* **Low-Latency Predictions:** Implement ML models on edge devices like smartphones or wearables to provide instant health insights without relying on cloud processing.
* **IoT Integration:** Connect IoT devices to a central platform for centralized monitoring and ML-driven alerts.

**5. Enhanced Data Management**

* **Data Augmentation:** Use synthetic data generation techniques to overcome limited dataset challenges.
* **Federated Learning:** Enable collaborative training of ML models across institutions while maintaining patient data privacy.
* **Data Privacy and Security:** Implement blockchain or advanced encryption techniques to ensure secure and private handling of sensitive health data.

**11. CONCLUSION:**

In this paper, a sufficiently large dataset of stroke attacked patients has been classified accurately. Four classifiers such as TABLE XI: Confusion matrix for Random Forest algorithm.



Naive Bayes, J48, k-NN, and ANN Algorithms were used for detection of stroke disease. From the performance analysis we see that ANN Algorithm performs better than other methods. The novelty and the main contribution of our work are collecting this dataset and preparing them to use with WEKA. The model can help people with a cautionary indication of being affected by stroke. Healthcare industries generate huge amounts of complex data about patients, hospitals resources, disease diagnosis, electronic patient records, medical devices, etc. Which is very difficult to relate to one another even by a field expert. It will help the clinician to better understand the type of disease. The limitations of our method are that the dataset is not perfectly symmetrical. However, it did not affect the predicted accuracy for the other algorithms.

In future work, it is possible to extend the research by using different classification techniques. Moreover, the prediction of stroke can be done by adding some non-stroke data with the existing dataset.

**12. BIBLIOGRAPHY**

[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.

[8] A. Sudha, P. Gayathri, and N. Jaisankar, “Effective analysis and predictive model of stroke disease using classification methods,” International Journal of Computer Applications, vol. 43, no. 14, pp. 26– 31, 2012.

[9] G. Kaur and A. Chhabra, “Improved j48 classification algorithm for the prediction of diabetes,” International Journal of Computer Applications, vol. 98, no. 22, 2014.

[10] I. H. Witten, E. Frank, M. A. Hall, and C. J. Pal, Data Mining: Practical machine learning tools and techniques. Morgan Kaufmann, 2016.

[11] P. Sewaiwar and K. K. Verma, “Comparative study of various decision tree classification algorithm using weka,” International Journal of Emerging Research in Management &Technology, vol. 4, pp. 2278– 9359, 2015.

[12] K. A. Shakil, S. Anis, and M. Alam, “Dengue disease prediction using weka data mining tool,” arXiv preprint arXiv:1502.05167, 2015.

[13] J. A. Alkrimi, H. A. Jalab, L. E. George, A. R. Ahmad, A. Suliman, and K. Al-Jashamy, “Comparative study using weka for red blood cells classification,” International Journal of Medical, Health, Pharmaceutical and Biomedical Engineering, vol. 9, no. 1, pp. 19–22, 2015.

[14] M. S. Siddiqui and A. I. Abidi, “Comparative study of different classification techniques using weka tool,” Global Sci-Tech, vol. 10, no. 4, pp. 200–208, 2018.