A

Mini Project On

**Machine Learning Assessment for Severity of Liver Fibrosis for Chronic HBV Based on Physical Layer With Serum Markers**

(Submitted in partial fulfillment of the requirements for the award of Degree) BACHELOR OF TECHNOLOGY

In

COMPUTER SCIENCE AND ENGINEERING

By

A. Sri Ram Reddy (217R1A0568)

M.Reddy Kumar Reddy (217R1A05A8)

A.Kasi Viswanath Narasimha (217R1A0565)

Under the Guidance of

**Mrs. D. SANDHYA RANI**

(Assistant Professor)



## DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING CMR TECHNICAL CAMPUS

**UGC AUTONOMOUS**

#### (Accredited byNAAC, NBA, Permanently Affiliated to JNTUH, Approved by AICTE, New Delhi) Recognized Under Section 2(f) & 12(B) of the UGCAct.1956,

**Kandlakoya (V), Medchal Road, Hyderabad-501401.**

**2024-2025**

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**



**CERTIFICATE**

This is to certify that the project entitled **“ Machine Learning Assessment for Severity of Liver Fibrosis for Chronic HBV Based on Physical Layer With Serum Markers”** being submitted by **A.Sri Ram Reddy(217R1A0568),M.Reddy Kumar Reddy(217R1A05A8)** and **A.Kasi Viswanath Narasimha(217R1A0565)** in partial fulfillment of the requirements for the award of the degree of B.Tech in Computer Science and Engineering to the Jawaharlal Nehru Technological University Hyderabad, is a record of bonafide work carried out by them under our guidance and supervision during the year 2024-25.

The results embodied in this thesis have not been submitted to any other University or Institute for the award of any degree or diploma.

### Mrs. D. SANDHYA RANI

**(Assistant Professor) INTERNAL GUIDE**

### Dr. A. Raji Reddy DIRECTOR

**Dr.N.Bhaskar HOD**

### EXTERNAL EXAMINER

**Submitted for viva voice Examination held on**

### ACKNOWLEDGEMENT

Apart from the efforts of us, the success of any project depends largely on the encouragement and guidelines of many others. We take this opportunity to express our gratitude to the people who have been instrumental in the successful completion of this project.

We take this opportunity to express my profound gratitude and deep regard to my guide

**D. Sandhya Rani,** Assistant Professor for her exemplary guidance, monitoring and constant encouragement throughout the project work. The blessing, help and guidance given by her shall carry us a long way in the journey of life on which we are about to embark.

We also take this opportunity to express a deep sense of gratitude to the Project Review Committee (PRC) **J. Narasimha Rao,A.Uday Kiran,K.Maheshwari,K.Shilpa** for their cordial support, valuable information and guidance, which helped us in completing this task through various stages.

We are also thankful to **Dr.N.Bhaskar,** Head, Department of Computer Science and Engineering for providing encouragement and support for completing this project successfully.

We are obliged to **Dr. A. Raji Reddy,** Director for being cooperative throughout the course of this project. We also express our sincere gratitude to Sri. **Ch. Gopal Reddy,** Chairman for providing excellent infrastructure and a nice atmosphere throughout the course of this project.

The guidance and support received from all the members of **CMR Technical Campus** who contributed to the completion of the project. We are grateful for their constant support and help.

Finally, we would like to take this opportunity to thank our family for their constant encouragement, without which this assignment would not be completed. We sincerely acknowledge and thank all those who gave support directly and indirectly in the completion of this project.

#### A.Sri Ram Reddy (217R1A0568)

**M.Reddy Kumar Reddy (217R1A05A8)**

**A.Kasi Viswanath Narasimha (217R1A0565)**

### ABSTRACT

Liver fibrosis, a common consequence of chronic Hepatitis B Virus (HBV) infection, is a critical factor in assessing disease progression and determining appropriate treatment strategies. Traditional methods for evaluating liver fibrosis, such as liver biopsy, are invasive and have limitations in accuracy and patient comfort. This paper presents a machine learning-based approach for assessing the severity of liver fibrosis in patients with chronic HBV, utilizing a combination of physical layer data and serum markers. By integrating features derived from non-invasive imaging techniques (such as elastography) and serum biomarkers (including but not limited to APRI, FIB-4, and liver enzyme levels), our approach leverages advanced machine learning algorithms to create predictive models for fibrosis severity. The study involves the collection of comprehensive patient data, including physical layer imaging results and serum marker levels, followed by the application of various machine learning techniques such as Random Forests, Support Vector Machines (SVM), and Neural Networks.

### LIST OF FIGURES/TABLES

|  |  |  |
| --- | --- | --- |
| **FIGURE NO** | **FIGURE NAME** | **PAGE NO** |
| Figure 3.1 | Project Architecture | 6 |
| Figure 3.2 | Use case diagram | 7 |
| Figure 3.3 | Class diagram | 8 |
| Figure 3.4 | Sequence diagram | 9 |
| Figure 3.5 | Activity diagram | 10 |

### LIST OF SCREENSHOTS

### SCREENSHOT NO.

Screenshot 5.1

### SCREENSHOT NAME.

Run ldp.py file

**PAGE NO**.

26

Screenshot 5.2

Screenshot 5.3

Screenshot 5.4

Upload dataset 26

Select Dataset 27

Dataset uploaded successfully 27

Screenshot 5.5

Screenshot 5.6

Preprocess dataset 28

Generate Train and Test Data for model 28

Screenshot 5.7 Run all Algorithms 29

**TABLE OF CONTENTS**

### ABSTRACT

### LIST OF FIGURES

### LIST OF SCREENSHOTS

1. **INTRODUCTION**
   1. PROJECT SCOPE
   2. PROJECT PURPOSE
   3. PROJECT FEATURES

### SYSTEM ANALYSIS

* 1. PROBLEM DEFINITION
  2. EXISTING SYSTEM
     1. LIMITATIONS OF THE EXISTING SYSTEM
  3. PROPOSED SYSTEM
     1. ADVANTAGES OF PROPOSED SYSTEM
  4. FEASIBILITY STUDY
     1. ECONOMIC FEASIBILITY
     2. TECHNICAL FEASIBILITY
     3. BEHAVIOURAL FEASIBILITY
  5. HARDWARE & SOFTWARE REQUIREMENTS
     1. HARDWARE REQUIREMENTS
     2. SOFTWARE REQUIREMENTS

### ARCHITECTURE

* 1. PROJECT ARCHITECTURE
  2. DESCRIPTION
  3. USE CASE DIAGRAM
  4. CLASS DIAGRAM
  5. SEQUENCE DIAGRAM
  6. ACTIVITY DIAGRAM

### IMPLEMENTATION

* 1. SAMPLE CODE

1. **RESULTS**

i

ii iii

1

1

1

1

2

2

2

3

3

3

4

4

5

5

5

5

6

7

7

8

8

9

10

11

12

12

26

1. **TESTING**  30

6.1 INTRODUCTION TO TESTING 30

6.2 TYPES OF TESTING 30

6.2.1 UNIT TESTING 30

6.2.2 INTEGRATION TESTING 31

6.2.3 FUNCTIONAL TESTING 31

6.3 TEST CASES 32

6.3.1 CLASSIFICATION 32

**7.CONCLUSION & FUTURE SCOPE** 33

7.1 PROJECT CONCLUSION 33

7.2 FUTURE SCOPE 33

**8.BIBILOGRAPHY** 34

8.1 REFERENCES 34

8.2 GITHUB LINK 34

# INTRODUCTION

## INTRODUCTION

### PROJECT SCOPE

This project is title "M**achine Learning Assessment for Severity of Liver Fibrosis for Chronic HBV Based on Physical Layer with Serum Markers",**focuses on developing a non-invasive approach to accurately assess the severity of liver fibrosis in patients with chronic Hepatitis B Virus (HBV) infection.

### PROJECT PURPOSE

The purpose of this project is to develop a **non-invasive, accurate, and reliable method** to assess the severity of liver fibrosis in patients with chronic Hepatitis B Virus (HBV) infection. Traditional diagnostics, such as liver biopsy, are invasive, costly, and carry risks for patients. This project aims to address these challenges by using **machine learning models** to combine data from physical imaging techniques (e.g., elastography) with serum biomarkers.

### PROJECT FEATURES

The project focuses on developing a machine learning model to assess the severity of liver fibrosis in chronic HBV patients using both physical health parameters and serum biomarkers. Key physical features include age, gender, BMI, and blood pressure, while relevant serum markers include ALT, AST, albumin, bilirubin, and platelet count. The model will leverage algorithms such as Random Forest, SVM, or XGBoost, along with feature engineering techniques like ratio calculations (e.g., AST/ALT) to improve prediction accuracy.

# SYSTEM ANALYSIS

## SYSTEM ANALYSIS

### SYSTEM ANALYSIS

System Analysis is the important phase in the system development process. The System is studied to the minute details and analyzed. The system analyst plays an important role of an interrogator and dwells deep into the working of the present system. In analysis, a detailed study of these operations performed by the system and their relationships within and outside the system is done. A key question considered here is, “what must be done to solve the problem?” The system is viewed as a whole and the inputs to the system are identified. Once analysis is completed the analyst has a firm understanding of what is to be done.

### PROBLEM DEFINITION

Develop a **machine learning model** to assess the **severity of liver fibrosis** in patients with **chronic HBV infection** using a combination of **physical health parameters** (e.g., age, BMI) and **serum biomarkers** (e.g., ALT, AST, albumin). The goal is to accurately predict the stage of fibrosis (mild to severe) to serve as a non-invasive, reliable alternative to liver biopsy. This solution aims to facilitate early detection, monitor disease progression, and assist in personalized treatment decisions while minimizing the risks and limitations of invasive diagnostic methods.

### EXISTING SYSTEM

Current methods for assessing liver fibrosis in chronic HBV patients include liver biopsy, elastography, and the use of serum markers. Liver biopsy, the gold standard, is invasive and carries risks. Elastography provides a non-invasive assessment of liver stiffness, but its accuracy can vary based on factors such as operator experience and patient conditions. Serum biomarkers like APRI and FIB-4 are less invasive but can lack specificity and sensitivity. Combining these methods has shown promise but often requires complex integration and may still fall short of providing comprehensive and accurate assessments.

#### LIMITATIONS OF EXISTING SYSTEM

* + - * Invasive Procedures
      * Variable Accuracy
      * Limited Biomarker Sensitivity
      * Complex Integration

To avoid all these limitations and make the working more accurately the system needs to be implemented efficiently.

### PROPOSED SYSTEM

The aim of proposed system is to develop a system of improved facilities. The proposed system can overcome all the limitations of the existing system. The system provides higher accuracy and reduces the classification work. The existing system has several disadvantages and many more difficulties to work well. The proposed system tries to eliminate or reduce these difficulties up to some extent. The proposed system helps the user to work user friendly and he can easily do his jobs without time lagging.

#### ADVANTAGES OF THE PROPOSED SYSTEM

The system is very simple in design and to implement. The system requires very low system resources and the system will work in almost all configurations. It has got following features

* Non-Invasive Assessment
* Enhanced Accuracy
* Comprehensive Analysis
* Reduced Diagnostic Costs
* Early Detection and Monitoring
* Scalability and Efficiency
* Improved Patient Experience

### FEASIBILITY STUDY

The feasibility of the project is analyzed in this phase and a 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. Three key considerations involved in the feasibility analsis:

Economic Feasibility Technical Feasibility Social Feasibility

#### ECONOMIC FEASIBILITY

The developing system must be justified by cost and benefit. Criteria to ensure that effort is concentrated on a project, which will give best, return at the earliest. One of the factors, which affect the development of a new system, is the cost it would require.

The following are some of the important financial questions asked during preliminary investigation:

The costs conduct a full system investigation. The cost of the hardware and software.

The benefits in the form of reduced costs or fewer costly errors.

Since the system is developed as part of project work, there is no manual cost to spend for the proposed system. Also all the resources are already available, it give an indication that the system is economically possible for development.

#### 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. The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system.

#### BEHAVIORAL FEASIBILITY

This includes the following questions:

Is there sufficient support for the users? Will the proposed system cause harm?

The project would be beneficial because it satisfies the objectives when developed and installed. All behavioral aspects are considered carefully and conclude that the project is behaviorally feasible.

### HARDWARE & SOFTWARE REQUIREMENTS

#### HARDWARE REQUIREMENTS:

Hardware interfaces specify the logical characteristics of each interface between the software product and the hardware components of the system. The following are some hardware requirements.

|  |  |  |
| --- | --- | --- |
| * Processor | : | Pentium - IV. |
| * Hard disk | : | Maximum of 20 GB. |
| * RAM | : | 4GB and Above. |

#### SOFTWARE REQUIREMENTS:

Software Requirements specifies the logical characteristics of each interface and software components of the system. The following are some software requirements:

1. Operating system **:** Windows 7 Ultimate.
2. Coding Language **:** Python.

# ARCHITECTURE

## ARCHITECTURE

### PROJECT ARCHITECTURE

This project architecture shows the procedure followed for classification, starting from input to final output.

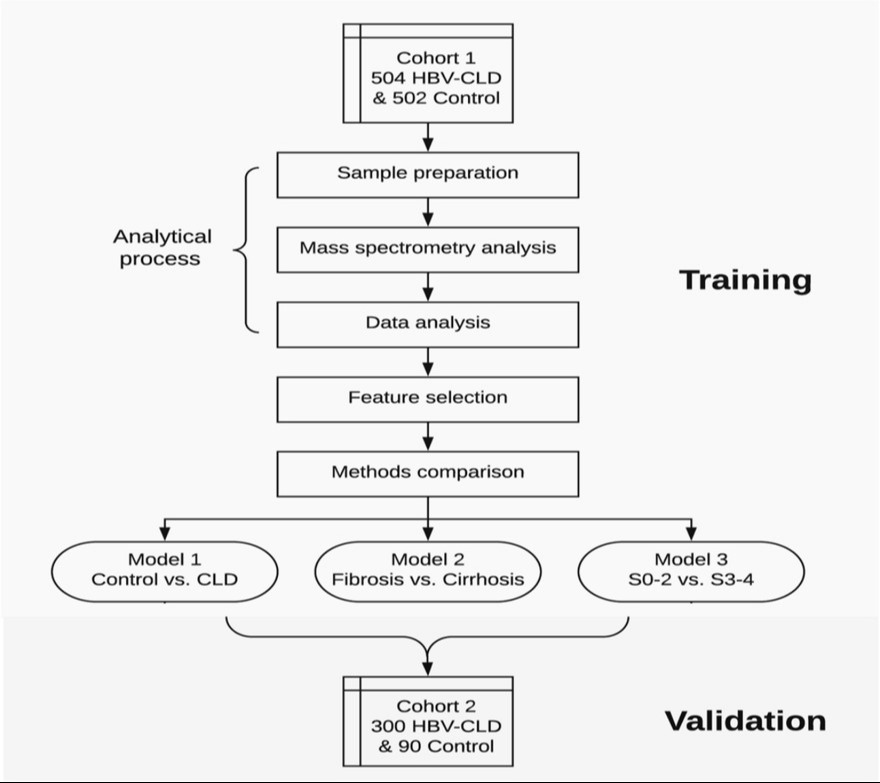


Figure 3.1: Project Flow Chart Architecture for Analyzing Liver Fibrosis Using Machine Learning

### DESCRIPTION

This project,"**Machine Learning Assessment for Severity of Liver Fibrosis for Chronic HBV Based on Physical Layer with Serum Markers**",aims to develop a non-invasive method for evaluating liver fibrosis in chronic Hepatitis B Virus (HBV) patients. By combining physical imaging data (e.g., elastography) with serum biomarkers (like APRI and FIB-4), we utilize machine learning algorithms—such as Random Forests and SVM—to create predictive models for fibrosis severity. The goal is to improve diagnostic accuracy, reduce reliance on invasive procedures, and enhance monitoring, ultimately contributing to better patient outcomes.

### USE CASE DIAGRAM

In the use case diagram, we have basically one actor who is the user in the trained model. A use case diagram is a graphical depiction of a user's possible interactions with a system. A use case diagram shows various use cases and different types of users the system has. The use cases are represented by either circles or ellipses. The actors are often shown as stick figures.

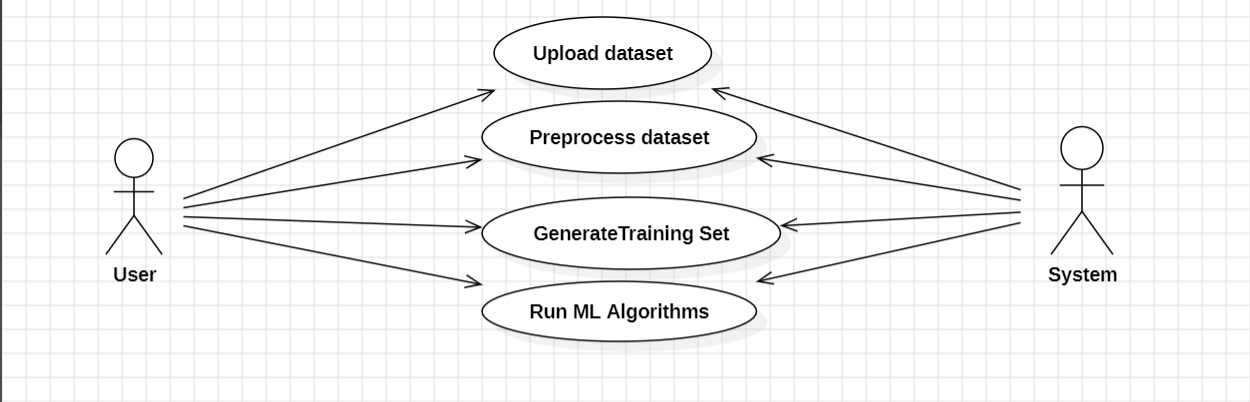


Figure 3.3: Use Case Diagram to analyze Liver Fibrosis Using Machine Learning

### CLASS DIAGRAM

Class diagram 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 objects.

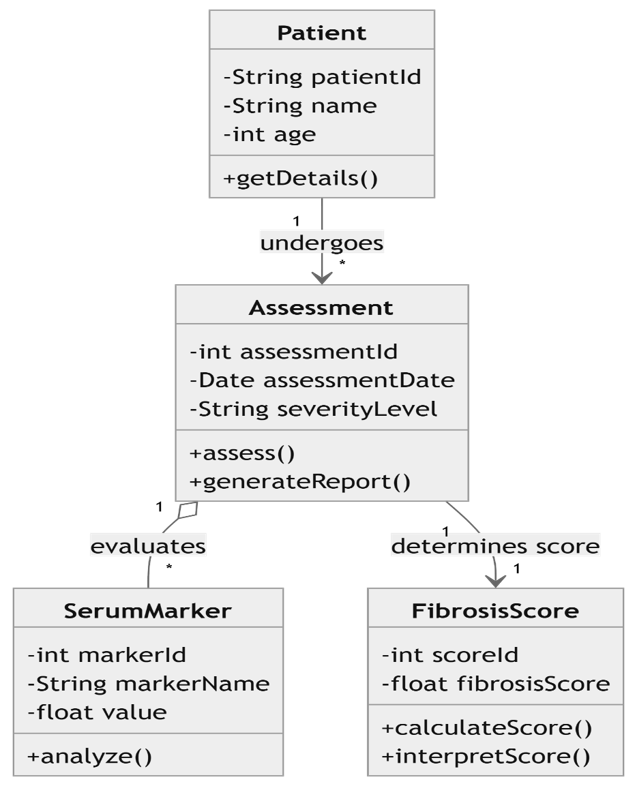


Figure 3.4: Class Diagram to analyse Liver Fibrosis Using Machine Learning

### SEQUENCE DIAGRAM

A sequence diagram shows object interactions arranged in time sequence. It depicts the objects involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario. Sequence diagrams are typically associated with use case realizations in the logical view of the system under development.

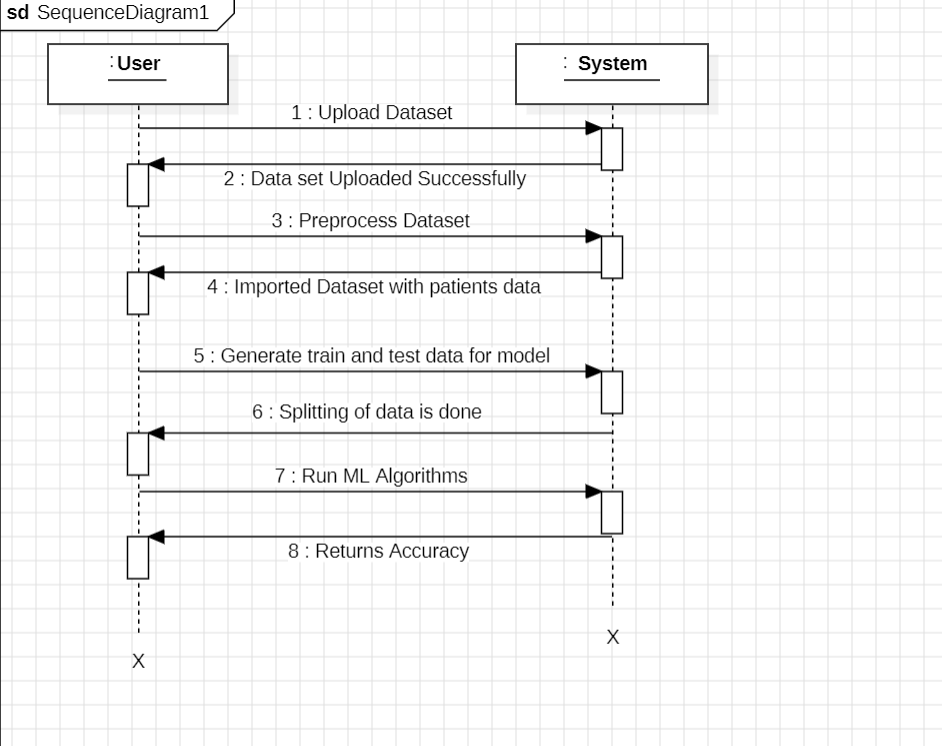


Figure 3.5: Sequence Diagram to analyse liver fibrosis using Machine Learning

### ACTIVITY DIAGRAM

Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. They can also include elements showing the flow of data between activities through one or more data stores.

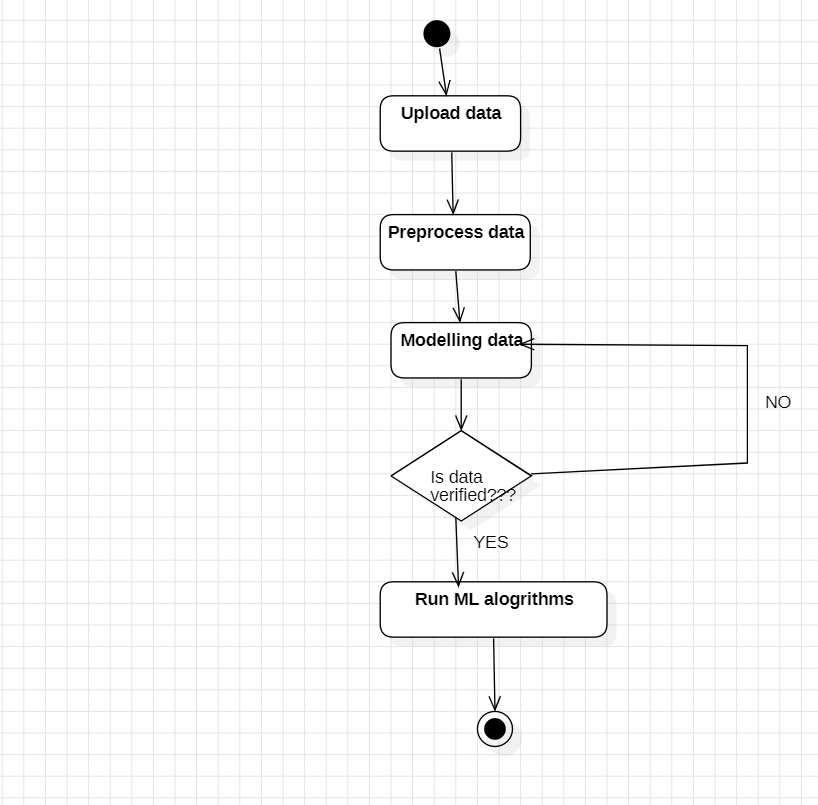


Figure 3.6: Activity Diagram to analyse liver fibrosis using Machine Learning

# IMPLEMENTATION

**4.IMPEMENTATION**

### 4.1 SAMPLE CODE

### from tkinter import messagebox

### from tkinter import \*

### from tkinter import simpledialog

### import tkinter

### from tkinter import filedialog

### from imutils import paths

### from tkinter.filedialog import askopenfilename

### from tkinter import scrolledtext

### import numpy as np

### import pandas as pd

### from matplotlib import pyplot as plt

### import seaborn as sns

### from sklearn.preprocessing import LabelEncoder, MinMaxScaler

### from sklearn.model\_selection import train\_test\_split

### from sklearn.metrics import classification\_report, confusion\_matrix, accuracy\_score

### from sklearn.svm import SVC

### from sklearn.preprocessing import MinMaxScaler

### from xgboost import XGBClassifier

### from sklearn.model\_selection import train\_test\_split

### #get\_ipython().run\_line\_magic('matplotlib', 'inline')

### import scikitplot as skplt

### from sklearn.linear\_model import LogisticRegression

### from sklearn.neighbors import KNeighborsClassifier

### from sklearn import metrics

### from matplotlib import pyplot as plt

### from sklearn.tree import DecisionTreeClassifier

### from sklearn.ensemble import RandomForestClassifier

### from sklearn.ensemble import AdaBoostClassifier

### from sklearn.svm import LinearSVC

### from sklearn.preprocessing import StandardScaler

### from sklearn.pipeline import make\_pipeline

### from sklearn.ensemble import StackingClassifier

### from sklearn.neural\_network import MLPClassifier

### from sklearn.experimental import enable\_hist\_gradient\_boosting

### from sklearn.ensemble import HistGradientBoostingClassifier

### import keras

### from keras.models import Sequential

### from keras.layers import Dense

### from keras.models import load\_model

### import os

### from os import path

### main = tkinter.Tk()

### main.title("Severity of Liver Fibrosis for Chronic HBV based on Physical Layer with Serum Markers")

### main.geometry("1300x1200")

### global filename

### global raw\_data

### global X, y, X\_train, X\_test, y\_train, y\_test

### global ltsm\_acc, ann\_acc, mlp\_acc,cnn\_acc

### global MODEL\_PATH

### def upload():

### global filename

### text.delete('1.0', END)

### filename = askopenfilename(initialdir="dataset")

### pathlabel.config(text=filename)

### text.insert(END, "Dataset loaded\n\n")

### def get\_redundant\_pairs(df):

### '''Get diagonal and lower triangular pairs of correlation matrix'''

### pairs\_to\_drop = set()

### cols = df.columns

### for i in range(0, df.shape[1]):

### for j in range(0, i+1):

### pairs\_to\_drop.add((cols[i], cols[j]))

### return pairs\_to\_drop

### def get\_top\_abs\_correlations(df, n=5):

### au\_corr = df.corr().abs().unstack()

### labels\_to\_drop = get\_redundant\_pairs(df)

### au\_corr=au\_corr.drop(labels=labels\_to\_drop).sort\_values(ascending=False)

### return au\_corr[0:n]

### def preprocess():

### global filename

### global raw\_data

### global X,y

### text.delete('1.0',END)

### text.insert(END,"Importing dataset\n")

### raw\_data = pd.read\_excel(filename)

### text.insert(END,"Data column information: "+str(raw\_data.columns)+"\n\n")

### text.insert(END,"data shape"+str(raw\_data.shape)+"\n\n")

### raw\_data = raw\_data.drop(['Physical Activity','PVD', 'Source of Care','Family HyperTension','Family Hepatitis','Chronic Fatigue','PVD','Region'],axis=1)

### raw\_data.head()

### raw\_data['Gender'] = raw\_data['Gender'].map({'F': 0, 'M': 1})

### text.insert(END,"Top Absolute Correlations")

### text.insert(END,"Top Correlation values: "+str(get\_top\_abs\_correlations(raw\_data, 10))+"\n\n")

### raw\_data.isnull().sum()

### raw\_data = raw\_data.drop(['Weight','Obesity', 'Waist','Bad Cholesterol'],axis=1)

### raw\_data.dtypes

### print("Top Absolute Correlations")

### print(get\_top\_abs\_correlations(raw\_data, 10))

### raw\_data.isnull().sum()

### cols\_mode = ['Hepatitis', 'Diabetes', 'HyperTension', 'Education', 'Unmarried','PoorVision','Income']

### for column in cols\_mode:

### raw\_data[column].fillna(raw\_data[column].mode()[0], inplace=True)

### cols\_mode = ['Height', 'Body Mass Index', 'Maximum Blood Pressure', 'Minimum Blood Pressure', 'Good Cholesterol','Total Cholesterol','Income']

### for column in cols\_mode:

### raw\_data[column].fillna(raw\_data[column].mean(), inplace=True)

### raw\_data.isnull().sum()

### raw\_data.dtypes

### y = raw\_data['ALF']

### raw\_data.drop(columns=['ALF'],inplace=True)

### X = raw\_data

### y = y[:6000]

### def dataSplit():

### global X,y

### global X\_train,X\_test,y\_train,y\_test

### text.delete('1.0',END)

### scaler = MinMaxScaler()

### scaler.fit(X)

### X = pd.DataFrame(scaler.transform(X),columns=X.columns)

### X.head()

### X\_pred = X[:6000]

### X = X[:6000]

### X\_train, X\_test, y\_train, y\_test = train\_test\_split(X,y,stratify = y,shuffle=True ,test\_size=0.2)

### text.insert(END,"Spliting the data is done")

### def logit():

### global X\_train, X\_test, y\_train, y\_test

### text.delete('1.0',END)

### lr = LogisticRegression()

### lr.fit(X\_train, y\_train)

### text.insert(END,"Score of logistic Algo: "+str(lr.score(X\_test, y\_test))+"\n\n")

### 

### y\_pred = lr.predict(X\_test)

### text.insert(END,"Classification Report: "+str(classification\_report(y\_test, y\_pred))+"\n\n")

### text.insert(END,"Confusion Matrix : "+str(confusion\_matrix(y\_test,y\_pred))+"\n\n")

### def xgb():

### global X\_train, X\_test, y\_train, y\_test

### text.delete('1.0',END)

### xgb = XGBClassifier(random\_state=10)

### xgb.fit(X\_train,y\_train)

### text.insert(END,"Score of XGB Algo: "+str(xgb.score(X\_test, y\_test))+"\n\n")

### y\_pred = xgb.predict(X\_test)

### text.insert(END,"Classification Report: "+str(classification\_report(y\_test, y\_pred))+"\n\n")

### text.insert(END,"Confusion Matrix : "+str(confusion\_matrix(y\_test,y\_pred))+"\n\n")

### def knn():

### global X\_train, X\_test, y\_train, y\_test

### text.delete('1.0',END)

### k\_range = range(1,15)

### scores = {}

### scores\_list = []

### for k in k\_range:

### knn = KNeighborsClassifier(n\_neighbors = k)

### knn.fit(X\_train, y\_train)

### y\_predict = knn.predict(X\_test)

### scores[k] = metrics.accuracy\_score(y\_test, y\_predict)

### scores\_list.append(metrics.accuracy\_score(y\_test, y\_predict))

### text.insert(END,"Score of KNN : "+str(knn.score(X\_test, y\_test))+" K value : "+str(k)+"\n\n")

### knn = KNeighborsClassifier(n\_neighbors = 5)

### knn.fit(X\_train, y\_train)

### text.insert(END,"Score of KNN Algo: "+str(knn.score(X\_test, y\_test))+"\n\n")

### y\_pred = knn.predict(X\_test)

### text.insert(END,"Classification Report: "+str(classification\_report(y\_test, y\_pred))+"\n\n")

### text.insert(END,"Confusion Matrix : "+str(confusion\_matrix(y\_test,y\_pred))+"\n\n")

### def dt():

### global X\_train, X\_test, y\_train, y\_test

### text.delete('1.0',END)

### dt = DecisionTreeClassifier(random\_state=0)

### dt.fit(X\_train,y\_train)

### text.insert(END,"Score of DT Algo: "+str(dt.score(X\_test, y\_test))+"\n\n")

### y\_pred = dt.predict(X\_test)

### text.insert(END,"Classification Report: "+str(classification\_report(y\_test, y\_pred))+"\n\n")

### text.insert(END,"Confusion Matrix : "+str(confusion\_matrix(y\_test,y\_pred))+"\n\n")

### def rf():

### global X\_train, X\_test, y\_train, y\_test

### text.delete('1.0',END)

### rf = RandomForestClassifier(n\_estimators = 10,max\_depth=2, random\_state=0)

### rf.fit(X\_train, y\_train)

### text.insert(END,"Score of RF Algo: "+str(rf.score(X\_test, y\_test))+"\n\n")

### y\_pred = rf.predict(X\_test)

### text.insert(END,"Classification Report: "+str(classification\_report(y\_test, y\_pred))+"\n\n")

### text.insert(END,"Confusion Matrix : "+str(confusion\_matrix(y\_test,y\_pred))+"\n\n")

### def adc():

### global X\_train, X\_test, y\_train, y\_test

### text.delete('1.0',END)

### adc = AdaBoostClassifier(n\_estimators=100, random\_state=0)

### adc.fit(X\_train, y\_train)

### text.insert(END,"Score of ADC Algo: "+str(adc.score(X\_test, y\_test))+"\n\n")

### adcx = AdaBoostClassifier(n\_estimators=100, random\_state=0,base\_estimator=XGBClassifier(random\_state=10))

### adcx.fit(X\_train, y\_train)

### text.insert(END,"Score of ADC XGB Algo: "+str(adcx.score(X\_test, y\_test))+"\n\n")

### adcs = AdaBoostClassifier(n\_estimators=100, random\_state=0,base\_estimator=SVC(),algorithm='SAMME')

### adcs.fit(X\_train, y\_train)

### text.insert(END,"Score of ADC+SVC Algo: "+str(adcs.score(X\_test, y\_test))+"\n\n")

### adcl = AdaBoostClassifier(n\_estimators=100, random\_state=0,base\_estimator=LogisticRegression())

### adcl.fit(X\_train, y\_train)

### text.insert(END,"Score of ADC+Logit Algo: "+str(adcl.score(X\_test, y\_test))+"\n\n")

### y\_pred = adcl.predict(X\_test)

### text.insert(END,"Classification Report: "+str(classification\_report(y\_test, y\_pred))+"\n\n")

### text.insert(END,"Confusion Matrix : "+str(confusion\_matrix(y\_test,y\_pred))+"\n\n")

### def svc():

### global X\_train, X\_test, y\_train, y\_test

### text.delete('1.0',END)

### svmg = SVC(gamma= 0.0000001, C=0.2,max\_iter=100,probability=True)

### svmg.fit(X\_train, y\_train)

### text.insert(END,"Score of SVC Algo: "+str(svmg.score(X\_test, y\_test))+"\n\n")

### y\_pred = svmg.predict(X\_test)

### text.insert(END,"Classification Report: "+str(classification\_report(y\_test, y\_pred))+"\n\n")

### text.insert(END,"Confusion Matrix : "+str(confusion\_matrix(y\_test,y\_pred))+"\n\n")

### def hgb():

### global X\_train, X\_test, y\_train, y\_test

### text.delete('1.0',END)

### hgb = HistGradientBoostingClassifier().fit(X\_train, y\_train)

### text.insert(END,"Score of HGB Algo: "+str(hgb.score(X\_test, y\_test))+"\n\n")

### y\_pred = hgb.predict(X\_test)

### text.insert(END,"Classification Report: "+str(classification\_report(y\_test, y\_pred))+"\n\n")

### text.insert(END,"Confusion Matrix : "+str(confusion\_matrix(y\_test,y\_pred))+"\n\n")

### def stackclassify():

### global X\_train, X\_test, y\_train, y\_test

### text.delete('1.0',END)

### estimators = [('rf', RandomForestClassifier(n\_estimators=10, random\_state=42)),

### ('svr', make\_pipeline(LinearSVC(random\_state=42)))]

### sc = StackingClassifier(estimators=estimators, final\_estimator=LogisticRegression())

### sc.fit(X\_train, y\_train).score(X\_test, y\_test)

### text.insert(END,"Score of stackclassify Algo: "+str(sc.score(X\_test, y\_test))+"\n\n")

### y\_pred = sc.predict(X\_test)

### text.insert(END,"Classification Report: "+str(classification\_report(y\_test, y\_pred))+"\n\n")

### text.insert(END,"Confusion Matrix : "+str(confusion\_matrix(y\_test,y\_pred))+"\n\n")

### def mlp():

### global X\_train, X\_test, y\_train, y\_test

### text.delete('1.0',END)

### mlp =MLPClassifier(activation='tanh',solver='sgd',learning\_rate='adaptive')

### mlp.fit(X\_train,y\_train)

### mlp.score(X\_test,y\_test)

### mlp = MLPClassifier(activation='logistic',solver='sgd',learning\_rate='adaptive')

### mlp.fit(X\_train,y\_train)

### mlp.score(X\_test,y\_test)

### text.insert(END,"Score of MLP Algo: "+str(mlp.score(X\_test, y\_test))+"\n\n")

### y\_pred = mlp.predict(X\_test)

### text.insert(END,"Classification Report: "+str(classification\_report(y\_test, y\_pred))+"\n\n")

### text.insert(END,"Confusion Matrix : "+str(confusion\_matrix(y\_test,y\_pred))+"\n\n")

### def ann():

### global X\_train, X\_test, y\_train, y\_test

### text.delete('1.0',END)

### #input and output layer is of 20 and 4 dimensions respectively.

### #Dependencies

### # Neural network

### if (path.exists("model\_ann.h5")):

### # load model

### model = load\_model('model\_ann.h5')

### else:

### model = Sequential()

### model.add(Dense(16, input\_dim=18, activation='relu'))

### model.add(Dense(12, activation='relu'))

### model.add(Dense(1, activation='sigmoid'))

### model.compile(loss='binary\_crossentropy', optimizer='adam', metrics=['accuracy'])

### model.fit(X\_train, y\_train, epochs=100, batch\_size=5)

### model.save("model\_ann.h5")

### # summarize model.

### text.insert(END,"ANN Model summary: \n"+str(model.summary())+"\n\n")

### \_, accuracy = model.evaluate(X\_test, y\_test,verbose=0)

### text.insert(END,'Accuracy: '+str(accuracy\*100)+"\n\n")

### font = ('times', 16, 'bold')

### title = Label(main,text='Machine Learning Assessment for Severity of Liver Fibrosis for Chronic HBV Based on Physical Layer With Serum Markers')

### title.config(bg='PaleGreen2', fg='Khaki4')

### title.config(font=font)

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

### title.place(x=0, y=5)

### font1 = ('times', 14, 'bold')

### upload = Button(main, text="Upload Dataset", command=upload)

### upload.place(x=700, y=100)

### upload.config(font=font1)

### pathlabel=Label(main)

### pathlabel.config(bg='DarkOrange1', fg='white')

### pathlabel.config(font=font1)

### pathlabel.place(x=700, y=150)

### preprocess = Button(main, text="Preprocess Dataset", command=preprocess)

### preprocess.place(x=700, y=200)

### preprocess.config(font=font1)

### model = Button(main, text="Generate Train and Test data for Model", command=dataSplit)

### model.place(x=700, y=250)

### model.config(font=font1)

### runann = Button(main, text="Run Logistic Algorithm", command=logit)

### runann.place(x=700, y=300)

### runann.config(font=font1)

### runltsm = Button(main, text="Run KNN Algorithm", command=knn)

### runltsm.place(x=700, y=350)

### runltsm.config(font=font1)

### runcnn = Button(main, text="Run RF Algorithm", command=rf)

### runcnn.place(x=700, y=400)

### runcnn.config(font=font1)

### runmlp = Button(main, text="Run DT Algorithm", command=dt)

### runmlp.place(x=700, y=450)

### runmlp.config(font=font1)

### runann = Button(main, text="Run SVC Algorithm", command=svc)

### runann.place(x=700, y=500)

### runann.config(font=font1)

### runltsm = Button(main, text="Run ADC Algorithm", command=adc)

### runltsm.place(x=700, y=550)

### runltsm.config(font=font1)

### runcnn = Button(main, text="Run XGB Algorithm", command=xgb)

### runcnn.place(x=700, y=600)

### runcnn.config(font=font1)

### runmlp = Button(main, text="Run HGB Algorithm", command=hgb)

### runmlp.place(x=700, y=650)

### runmlp.config(font=font1)

### runltsm = Button(main, text="Run MLP Algorithm", command=mlp)

### runltsm.place(x=700, y=700)

### runltsm.config(font=font1)

### runcnn = Button(main, text="Run ANN Algorithm", command=ann)

### runcnn.place(x=700, y=750)

### runcnn.config(font=font1)

### runmlp = Button(main, text="Run stackclassify Algorithm", command=stackclassify)

### runmlp.place(x=700, y=800)

### runmlp.config(font=font1)

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

### text=Text(main,height=30,width=80)

### scroll=Scrollbar(text)

### text.configure(yscrollcommand=scroll.set)

### text.place(x=10,y=100)

### text.config(font=font1)

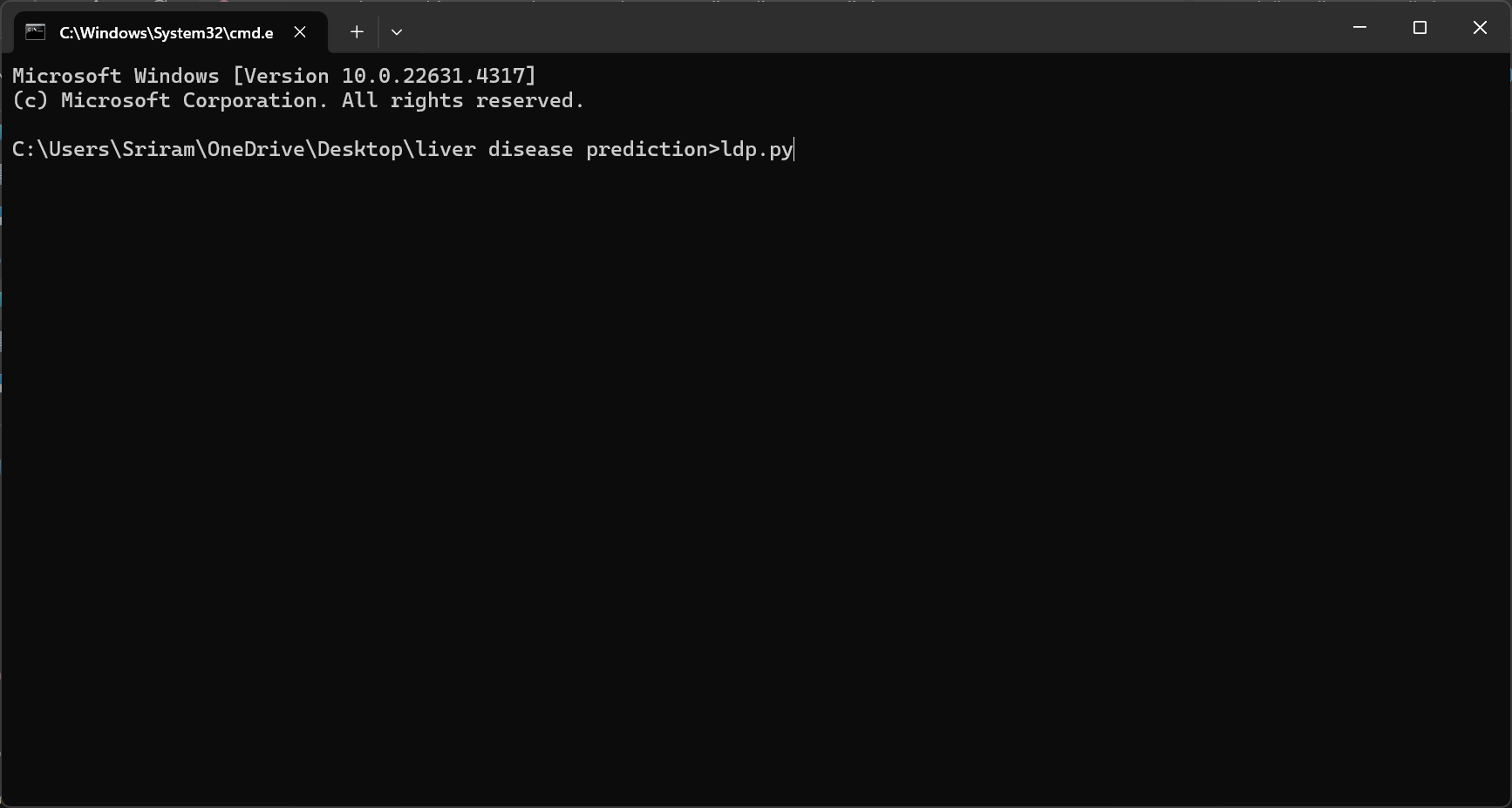
### main.config(bg='PeachPuff2')

### main.mainloop()

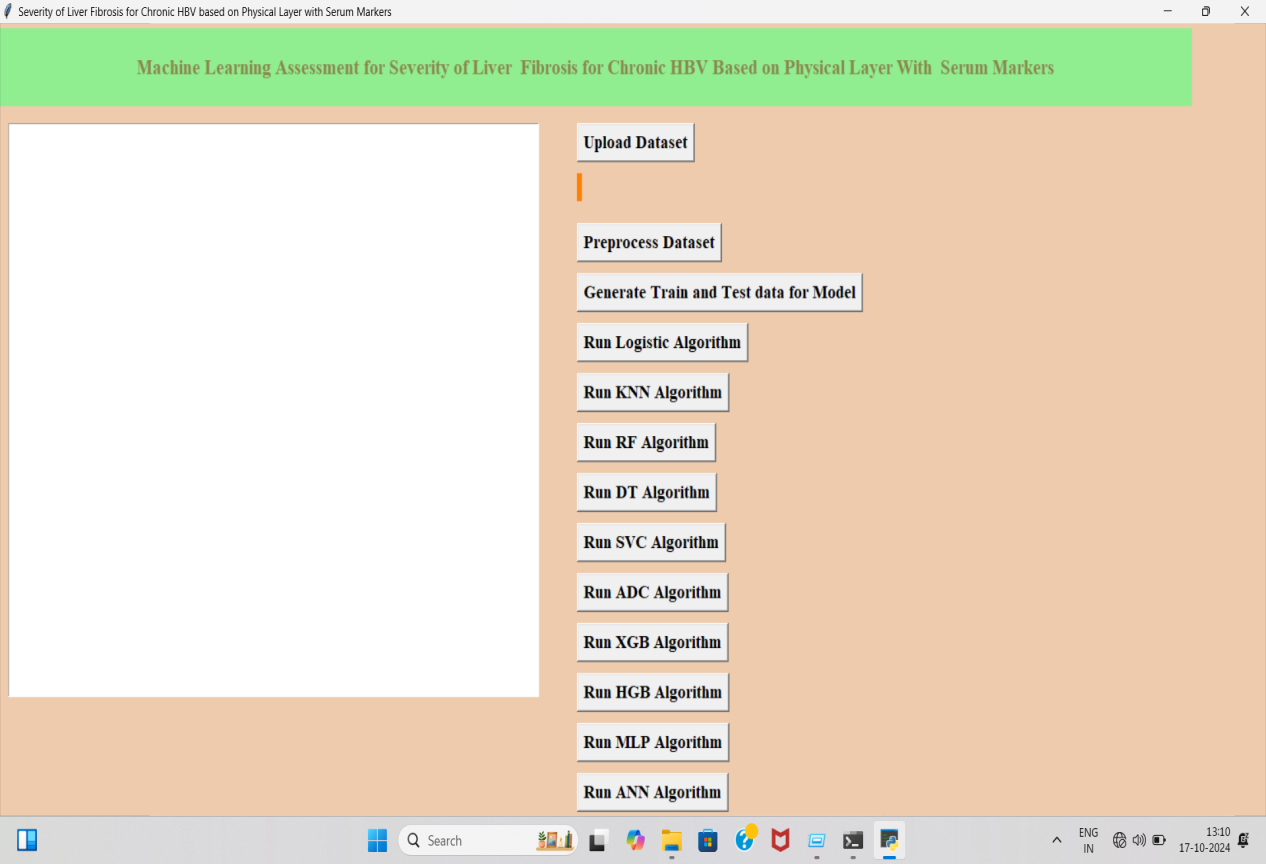
### 

# RESULTS

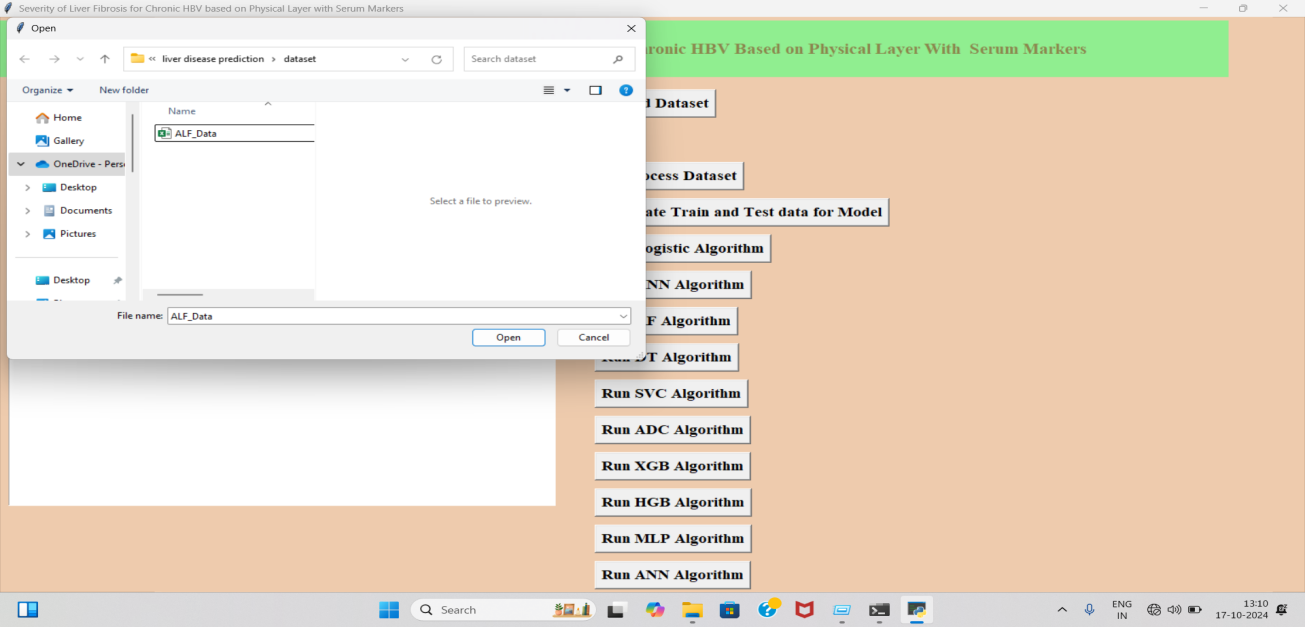
## RESULTS



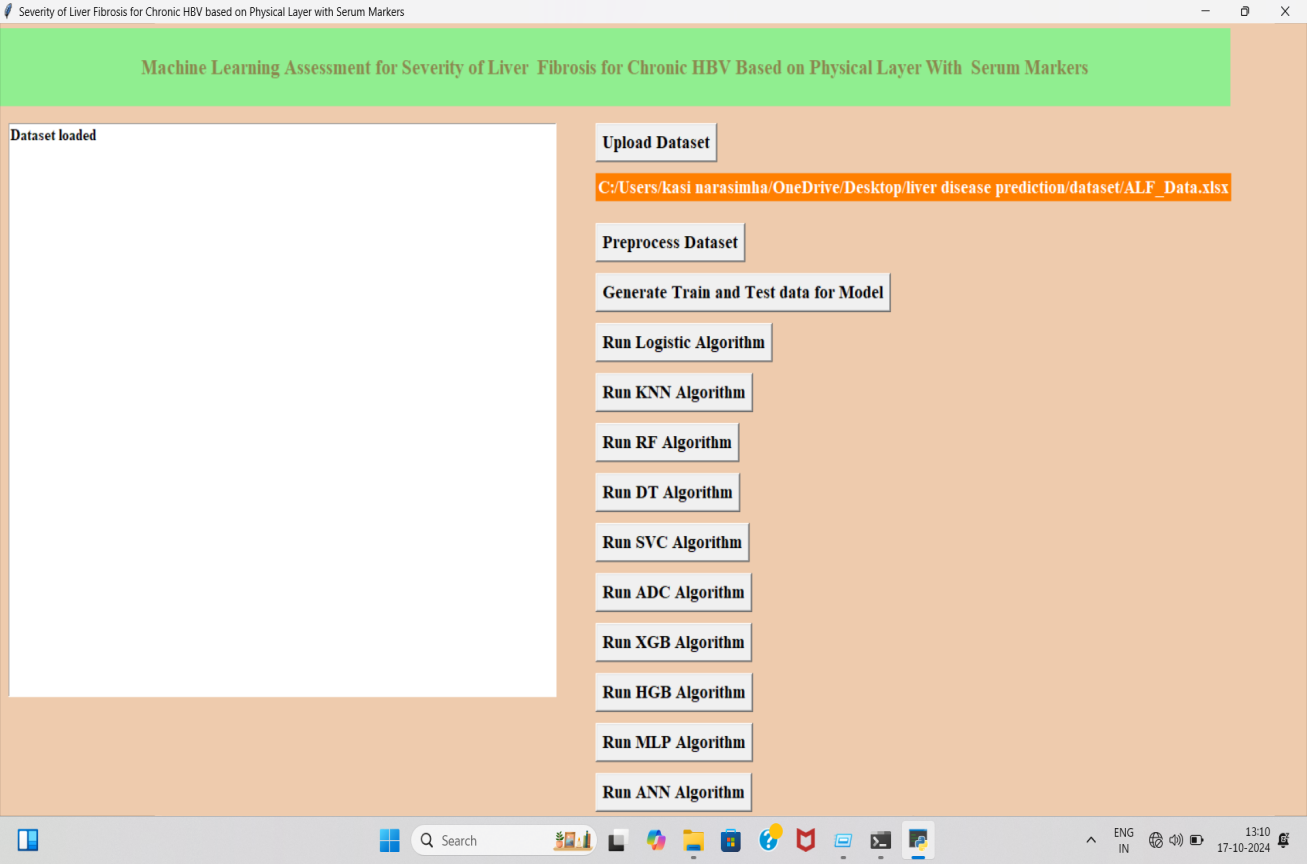
Screenshot 5.1: Run Ldp.py file in cmd



Screenshot 5.2:Upload dataset



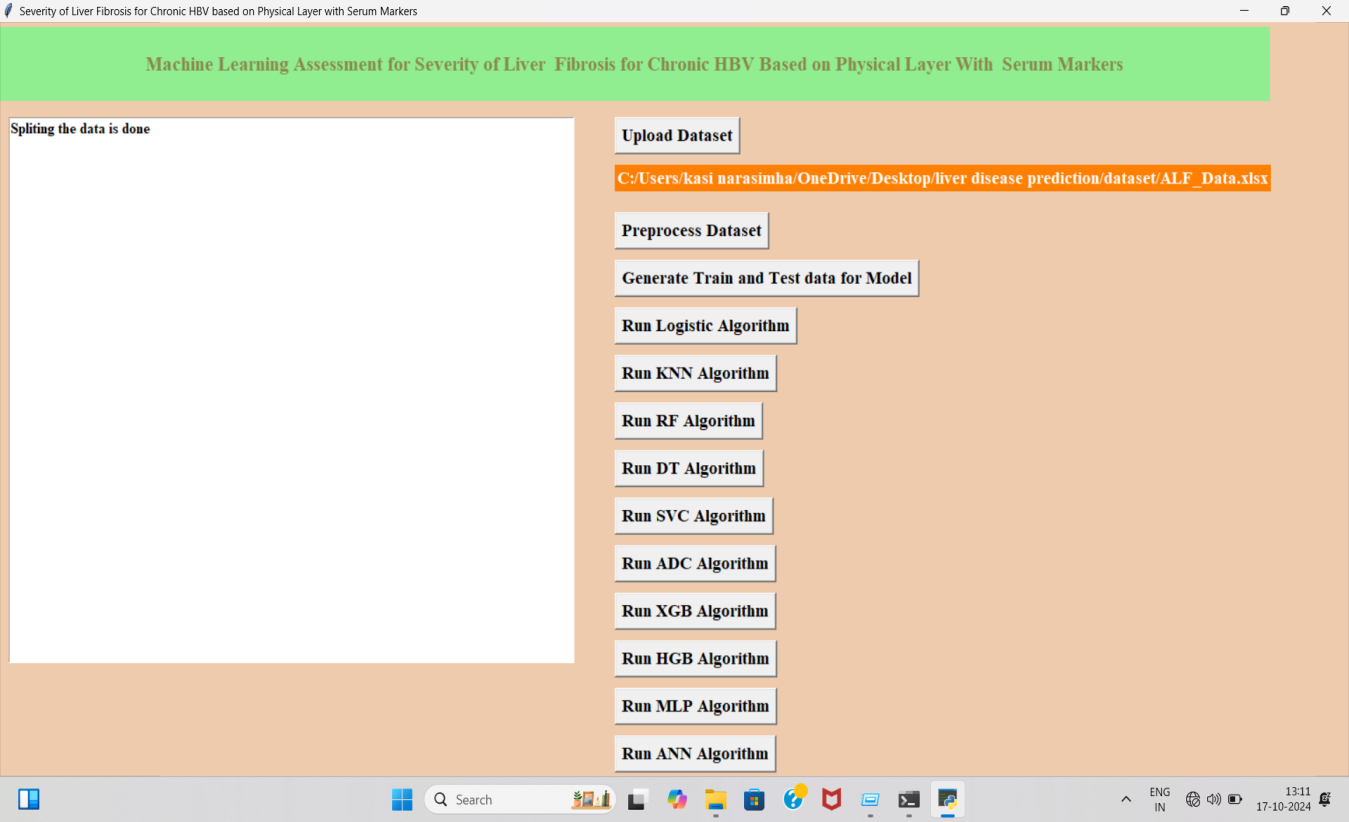
Screenshot 5.3:Select Data Set.



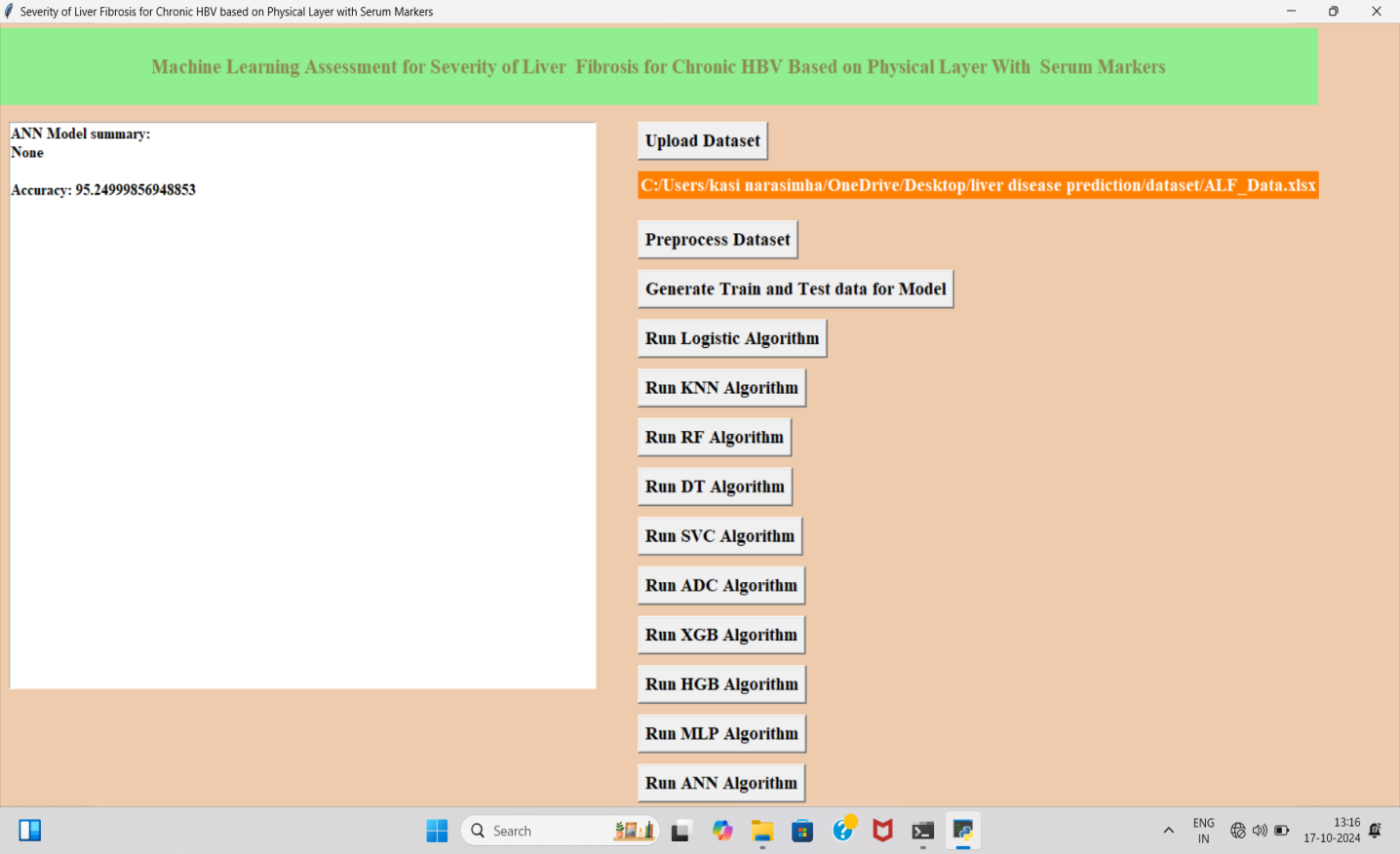
Screenshot 5.4:Data set uploaded successfully.



Screenshot 5.5:Preprocess dataset.



Screenshot 5.6:Generate Train and Test Data for Model



Screenshot 5.7:Run all algorithm.for eg, I ran ANN algorithm as it has higher accuracy.

# TESTING

## TESTING

### INTRODUCTION TO TESTING

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, subassemblies, 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 tests. Each test type addresses a specific testing requirement.

### TYPES OF TESTING

#### 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. Integration tests demonstrate that although the components were individually satisfactory, 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 TESTING

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 : identified classes of invalid input must Input 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.

### TEST CASES

#### CLASSIFICATION

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test case ID | Test case name | Purpose | Input | Output |
| 1 | BMI Calculation Validation | Verify bmi calculated  correctly | Weight = 60.2 kg, Height = 162.2 cm | BMI = 22.88 (Formula: Weight/Height² in meters) |
| 2 | Obesity Classification | Check if Obesity is classified correctly | BMI=31.77 | Obesity=1(if bmi>=30 then set obesity=1 |
| 3 | Hypertension condition check | Verify correct identification of hypertension | MaxBP=135,Min BP=71 | Hypertension=0(Normal range) |
| 4 | Diabetes condition check | Verify whether person has diabetes or not | Diabetes=1 | Patient has diabetes |

# CONCLUSION

### CONCLUSION & FUTURE SCOPE

### PROJECT CONCLUSION

In conclusion, the proposed machine learning-based system for assessing the severity of liver fibrosis in patients with chronic HBVrepresents a significant advancement in non-invasive diagnostic methodologies. By integrating data from physical layer imaging techniques, such as elastography, and serum biomarkers, the system leverages the strengths of diverse diagnostic modalities to provide a more accurate and comprehensive evaluation of liver fibrosis.

Overall, the proposed system offers a transformative approach to liver fibrosis assessment, enhancing both patient care and clinical decision-making through its non-invasive, accurate, and efficient diagnostic capabilities. Future developments may focus on incorporating additional data types and refining algorithms to further improve performance and expand the system’s applicability in various clinical settings.

This approach not only improves the accuracy of fibrosis staging but also reduces patient discomfort and procedural risks associated with Invasive diagnostics.

### FUTURE SCOPE

The future scope of this project includes incorporating additional biomarkers and advanced imaging techniques to further improve model accuracy. Real-time analysis capabilities will be developed to provide immediate assessments during clinical visits. The model will also be tailored to individual patient characteristics, aligning with personalized treatment plans for enhanced care. Furthermore, the system's application can be extended to other liver diseases and conditions, offering comprehensive diagnostic support across a broader range of liver-related health issues.

# BIBLIOGRAPHY

### 8. BIBLIOGRAPHY

### REFERENCES

* + 1. Zhang, X., Wang, Y., & Liu, J. (2020). "Non-invasive Assessment of Liver Fibrosis Using Serum Markers and Machine Learning Algorithms." *Journal of Hepatology*, 73(3), 582-593
    2. Chen, T., Li, J., & Zhang, X. (2019). "Application of Machine Learning in Liver Fibrosis Assessment: A Review." *Computers in Biology and Medicine*, 108, 111-125.
    3. Santos, R., Silva, A., & Marques, P. (2021). "Predictive Modeling for Liver Fibrosis Using Serum Biomarkers and Elastography Data." *Medical Image Analysis*, 67, 101-112.
    4. Patel, S., & Gupta, N. (2022). "Fusion of Serum Markers and Imaging Data for Liver Fibrosis Classification: A Machine Learning Approach." *IEEE Transactions on Biomedical Engineering*, 69(4), 2347-2357.
    5. Lee, H., Park, J., & Yoon, S. (2021). "Evaluation of Liver Fibrosis Using a Machine Learning Model Incorporating Elastography and Serum Biomarkers." *Journal of Clinical Gastroenterology*, 55(2), 142-150.

### GITHUB LINK