

Statistical Machine Learning Approaches for Liver Disease Prediction

Team ID: PNT2022TMID00067

Team Members:

Haris Murugan K

Hudson G

Yeshwaanath K S

Aravind J

Project Report Format

1. INTRODUCTION

1.1 Project Overview

1.2 Purpose

2. LITERATURE SURVEY

2.1 Existing problem

2.2 References

2.3 Problem Statement Definition

3. IDEATION & PROPOSED SOLUTION

3.1 Empathy Map Canvas

3.2 Ideation & Brainstorming

3.3 Proposed Solution

3.4 Problem Solution fit

4. REQUIREMENT ANALYSIS

4.1 Functional requirement

4.2 Non-Functional requirements

5. PROJECT DESIGN

5.1 Data Flow Diagrams

5.2 Solution & Technical Architecture

5.3 User Stories

6. PROJECT PLANNING & SCHEDULING

6.1 Sprint Planning & Estimation

6.2 Sprint Delivery Schedule

6.3 Reports from JIRA

7. CODING & SOLUTIONING (Explain the features added in the project along with code)

7.1 Feature 1

7.2 Feature 2

7.3 Database Schema (if Applicable)

8. TESTING

8.1 Test Cases

8.2 User Acceptance Testing

9. RESULTS

9.1 Performance Metrics

10. ADVANTAGES & DISADVANTAGES

11. CONCLUSION

12. FUTURE SCOPE 13. APPENDIX

Source Code

1 INTRODUCTION

1.1 Project Overview

Given a dataset containing biological and diagnostic data of patients, this project aims to identify a suitable machine learning algorithm which is capable of identifying whether a person has liver disease or not. This is a binary classification problem to be solved using supervised learning. We have ten features for each data point and a label which identifies whether a patient is suffering from liver disease or not. In order to arrive at the solution, our aim should be to train various supervised learning models on this dataset so that we have a well performing model which is able to classify any new data as a positive or negative with a reasonable degree of accuracy and perform better than benchmarks!

1.2 Purpose

Discovering the existence of liver disease at an early stage is a complex task for the doctors. Early prediction of liver disease using classification algorithms is an efficacious task that can help the doctors to diagnose the disease within a short duration of time. The main objective of this project is to analyze the parameters of various classification algorithms and compare their predictive accuracies so as to find out the best classifier for determining the liver disease. Here we are building a model by applying various machine learning algorithms find the best accurate model. The improvement of patient care, research, and policy is significantly impacted by medical diagnoses. Medical practitioners employ a variety of pathological techniques to make diagnoses based on medical records and the conditions of the patients. Disease identification has been significantly enhanced by the application of artificial intelligence and machine learning in conjunction with clinical data. Data driven, machine learning (ML) techniques can be used to test current approaches and support researchers in potentially innovative judgments. So, our purpose is to create a Machine Learning application that attempts to accurate prediction of liver disease.

2 Literature Survey

2.1 Existing Problem

2.1 References

Sl no	Authors	Year	Disease	Machine learning algorithm	Dataset input	Remarks	Conclusion
1	Bendi Venkata Ramana et al. [1]	2011	Liver disease	Naïve Bayes, C4.5, Backward propagation, KNN and SVM	AP liver dataset and UCLA liver dataset	Naïve Bayes, C4.5 KNN, Backward propagation and SVM has 95.07, 96.27, 96.93, 97.47, & 97.07% accuracy respectively	KNN, Backward propagation and SVM are giving more better results. AP data set are better than UCLA for all the selected algorithm
2	Bendi Venkata Ramana and M.Surendra Prasad Babu [2]	2012	Liver disease	Modified Rotation Forest	UCI liver dataset and Indian dataset	MLP algorithm with random subset gives better accuracy 74.78% than NN with CFS of accuracy 73.07%	MLP algorithm with UCI liver dataset has better accuracy than NN with Indian liver dataset
3	Yugal KUMA & G. Sahoo [3]	2013	Liver disease	DT, SVM, NB and ANN	north east area of Andhra Pradesh (India) liver dataset	Decision tree(DT) has better accuracy of 98.46%	Rule based classification with DT algorithm has better accuracy
4	S.Dhamodharan [4]	2014	Liver cancer, Cirrhosis and Hepatitis	Naïve-Bayes, FT Tree	WEKA (Waikato Environment for Knowledge and Analysis) dataset	Naïve Bayes is 75.54% accuracy and FT Tree is 72.6624% accuracy	Naïve Bayes algorithm has better compare to other algorithms
5	Heba Ayeldeen et al. [5]	2015	Liver fibrosis	Decision tree	department of Medical Biochemistry and Molecular Biology, Faculty of Medicine, Cairo University.		decision tree classifier accuracy is 93.7%
6	D Sindhuja & R jemina Priyadarsini [6]	2016	Liver disease disorder	C4.5,Naïve Bayes, SVM, BPNN ,Regression and DT Data	AP has better dataset result than UCLA	Survey paper suggest C4.5 has better results than others	C4.5 has better accuracy result than other algorithms
7	Somaya Hashem et al [8]	2016	Liver fibrosis	PSO, GA, MReg & ADT	Egyptian national committee for control of viral hepatitis database	PSO, GA, MReg & ADT are 66.4, 69.6.69.1, & 84.4% accuracy respectively	ADT has more accuracy result than other algorithms

8	Sumedh Sontakke et al	2017	Liver disease	SVM & Backpropagation	(UCI)Machine Learning Repository	SVM (accuracy 71%))& Backpropagation(accuracy 73.2%)	More accuracy result in Back propagation
9	Han ma et al	2018	Nonalcoholic fatty liver disease	Using 11 classification algorithms	First Affiliated Hospital, Zhejiang University China, College of medicine First Affiliated	Bayesian network accuracy 83%	Concluded Bayesian network has best performance than other algorithms
10	Joel Jacob et al [10]	2018	Liver disease	Logistic regression, K-NN, SVM,&ANN	Indian Liver Patient Dataset comprised of 10 different attributes of 583 patients.	Logistic regression, K-NN, SVM,& ANN has 73.23, 72.05, 75.04 & 92.8% accuracy respectively	ANN has higher accuracy than others
11	Sivakumar D et al [11]	2019	Liver disease	K-means & C4.5 algorithms	UCI Repository	C4.5 algorithm has 94.36% precision.	C4.5 has better accuracy than K-means algorithms
12	Mehtaj Banu H [12]	2019	Liver disease	Supervised ,unsupervised & reinforcement	UCI repository databases.	Note: Only explaining not implementing practically	KNN and AVM has improved prediction performance accuracy
13	Vasan Durai et al [13]	2019	Liver disease	SVM,NB & J48	UCI repository	J48 algorithm has better feature selection with 95.04% accuracy	J48 algorithm is accuracy rate of 95.04%.

2.3 Problem Statement Definition

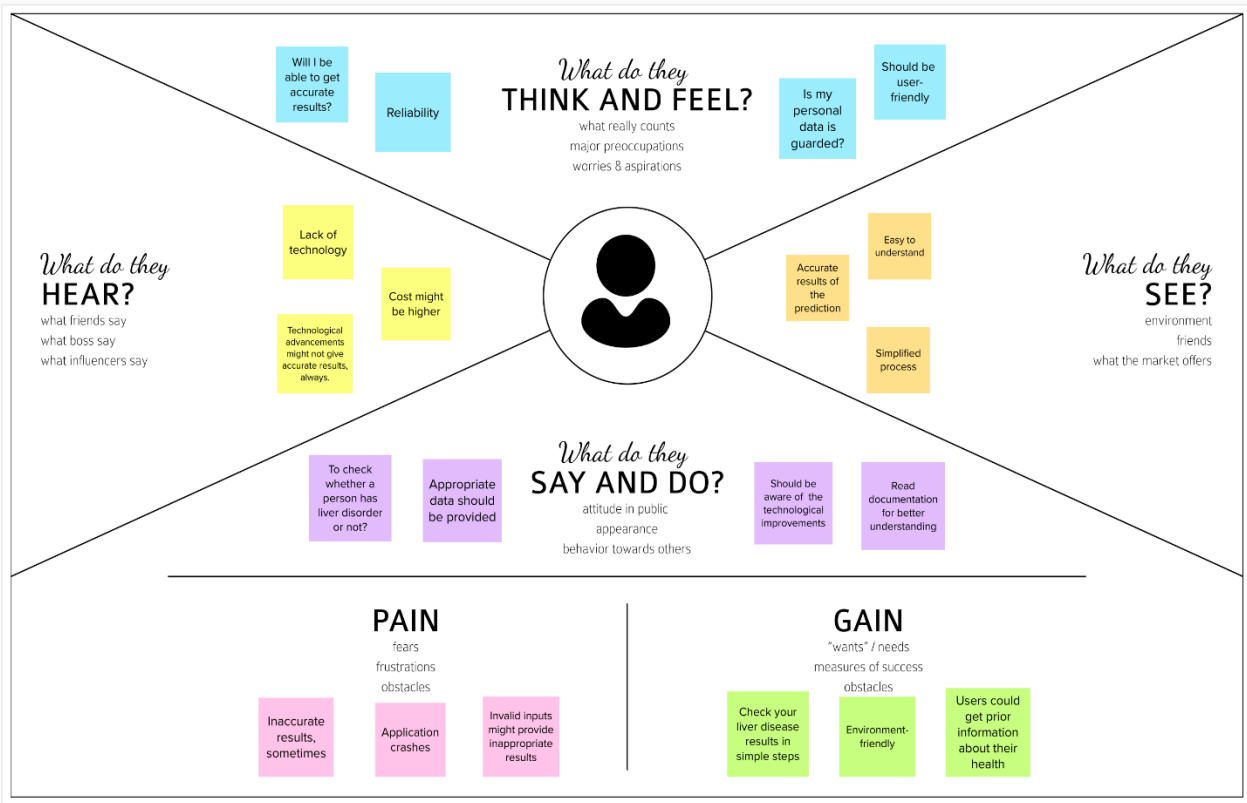
The main objective of this project is to analyze the parameters of various classification algorithms and compare their predictive accuracies so as to find out the best classifier for determining the liver disease. Here we are building a model by applying various machine learning algorithms find the best accurate model. And integrate to flask based web application. User can predict the disease by entering parameters in the web application. The presented problem wants us to identify whether a patient has liver disease or not with statistical machine learning approaches

3 IDEATION & PROPOSED SOLUTION

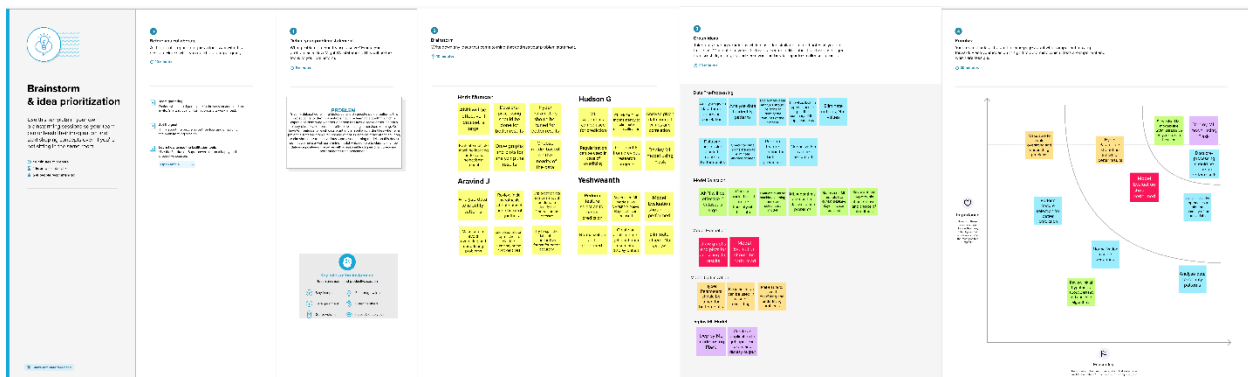
3.1 Empathy Map Canvas

Statistical Machine Learning Approaches To Liver Disease Prediction

IBM-Nalaiyathiran



3.2 Ideation & Brainstorming



3.3 Proposed Solution

S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	Discovering the existence of liver disease at an early stage is a complex task for the doctors. The main objective of the project is to examine data from liver patients concentrating on relationships between a key list of liver enzymes, proteins, age and gender using them to try and predict the likeliness of liver disease.
2.	Idea / Solution description	Our solution is to build a model by applying various machine learning algorithms and find the best accurate model to predict whether as a liver disease or not. We plan to perform data pre-processing and data visualization methods to increase the accuracy of the model. And integrate the chosen model into Flask based web application where the User can predict the disease by entering parameters in the web application.
3.	Novelty / Uniqueness	<ul style="list-style-type: none">• Data pre-processing which includes Data Cleaning, Data transformation, and Data Reduction is performed to increase the accuracy of the model.• Various Machine model is implemented and the highest accurate model is chosen.• Model output is evaluated using Accuracy, confusion matrix and other various metrics.• ROC-AUC considers the rank of the output probabilities and

		<p>intuitively measures the likelihood that model can distinguish between a positive point and a negative point. We will use AUC to select the best model among the various machine learning models.</p> <ul style="list-style-type: none"> • ML Model is deployed in the cloud.
4.	Social Impact / Customer Satisfaction	<ul style="list-style-type: none"> • Since the likeliness of the liver disease is predicted with high accuracy, user can able to take remedial measures • Promotes health care • Accurate prediction of liver disease
5.	Business Model (Revenue Model)	<p>Revenue can be made by collaborating with Hospitals and other health related companies and Integrating subscription services to the application</p>
6.	Scalability of the Solution	<ul style="list-style-type: none"> • Accuracy of the model can be increased by training with large data. The model can be made to learn from the user input. Model is deployed in the web where the public from across the world can use to predict the likeliness of liver disease. • Cloud services guarantee high availability and location independent access to the users.

3.4 Problem Solution Fit

Problem-Solution fit canvas 2.0

Purpose / Vision

Define CS, fit into	1. CUSTOMER SEGMENT(S) CS Who is your customer? 1. Our Project mainly focuses on people who has been suffering from liver disease or symptoms related to liver disorders. 2. People from any age group can use this application.	6. CUSTOMER CC What constraints prevent your customers from taking action or limit their choices of solutions? i.e. spending power, budget, no cash, network connection, available devices. 1. Proper Internet connectivity required. 2. User must enter appropriate details for accurate results. 3. Must read the guidelines for better usage.	5. AVAILABLE SOLUTIONS AS Which solutions are available to the customers when they face the problem or need to get the job done? What have they tried in the past? What pros & cons do these solutions have? i.e. pen and paper is an alternative to digital notetaking 1. If a person has been diagnosed with symptoms related to liver disorders, consult the doctor for better treatment. 2. Appropriate scan/checkups should be performed.	Explore AS,
	2. JOBS-TO-BE-DONE / PROBLEMS J&P Which jobs-to-be-done (or problems) do you address for your customers? There could be more than one; explore different sides. 1. Website crashes should be avoided. 2. Application interface should be user-friendly. 3. Precision of results delivered	9. PROBLEM ROOT CAUSE RC What is the real reason that this problem exists? What is the back story behind the need to do this job? i.e. customers have to do it because of the change in regulations. 1. Expensive consultation fee. 2. Inexperienced professionals. 3. Hospital Accessibility.	7. BEHAVIOUR BE What does your customer do to address the problem and get the job done? i.e. directly related: find the right solar panel installer, calculate usage and benefits; indirectly associated: customers spend free time on volunteering work (i.e. Greenpeace) 1. The customer needs to provide correct measures of the necessary details which includes age, gender and other biological inputs for the accurate prediction of the likeliness of the liver disease.	
Identify strong TR & EM	3. TRIGGERS TR What triggers customers to act? i.e. seeing their neighbour installing solar panels, reading about a more efficient solution in the news. 1. Cost Effective. 2. Early detection can avoid serious problems.	10. YOUR SOLUTION SL If you are working on an existing business, write down your current solution first, fill in the canvas, and check how much it fits reality. If you are working on a new business proposition, then keep it blank until you fill in the canvas and come up with a solution that fits within customer limitations, solves a problem and matches customer behaviour. 1. Our solution is to build a model by applying various machine learning algorithms and find the suitable model for accurate prediction. 2. We plan to perform data-pre-processing and data visualization methods to increase the accuracy of the model. And integrate the chosen model into Flask based web application where the User can predict the disease by entering parameters in the web application.	8. CHANNELS of BEHAVIOUR CH 8.1 ONLINE What kind of actions do customers take online? Extract online channels from #7 1. Searching online for symptoms. 8.2 OFFLINE What kind of actions do customers take offline? Extract offline channels from #7 and use them for customer development. 1. Booking appointments in hospitals.	Extract online & offline CH of BE
	4. EMOTIONS: BEFORE / AFTER EM How do customers feel when they face a problem or a job and afterwards? 1. Fear, sad, anxiety, insecurity, trust.			



Problem-Solution fit canvas is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 license Created by Daria Nepriakhina / Amaltama.com



4 REQUIREMENT ANALYSIS

4.1 Functional Requirements

Following are the functional requirements of the proposed solution.

FR No.	Functional Requirement (Epic)	Sub Requirement (Story/Sub-Task)
FR-1	User Registration	<ul style="list-style-type: none">• Registration through website form and email.
FR-2	User Confirmation	<ul style="list-style-type: none">• Confirmation via EmailConfirmation via OTP
FR-3	User Input	Get necessary details for prediction
FR-4	Data Pre-processing	Data Cleaning Data Scaling Feature Selection
FR-5	Prediction Result	Display result whether the user has liver disease or not.

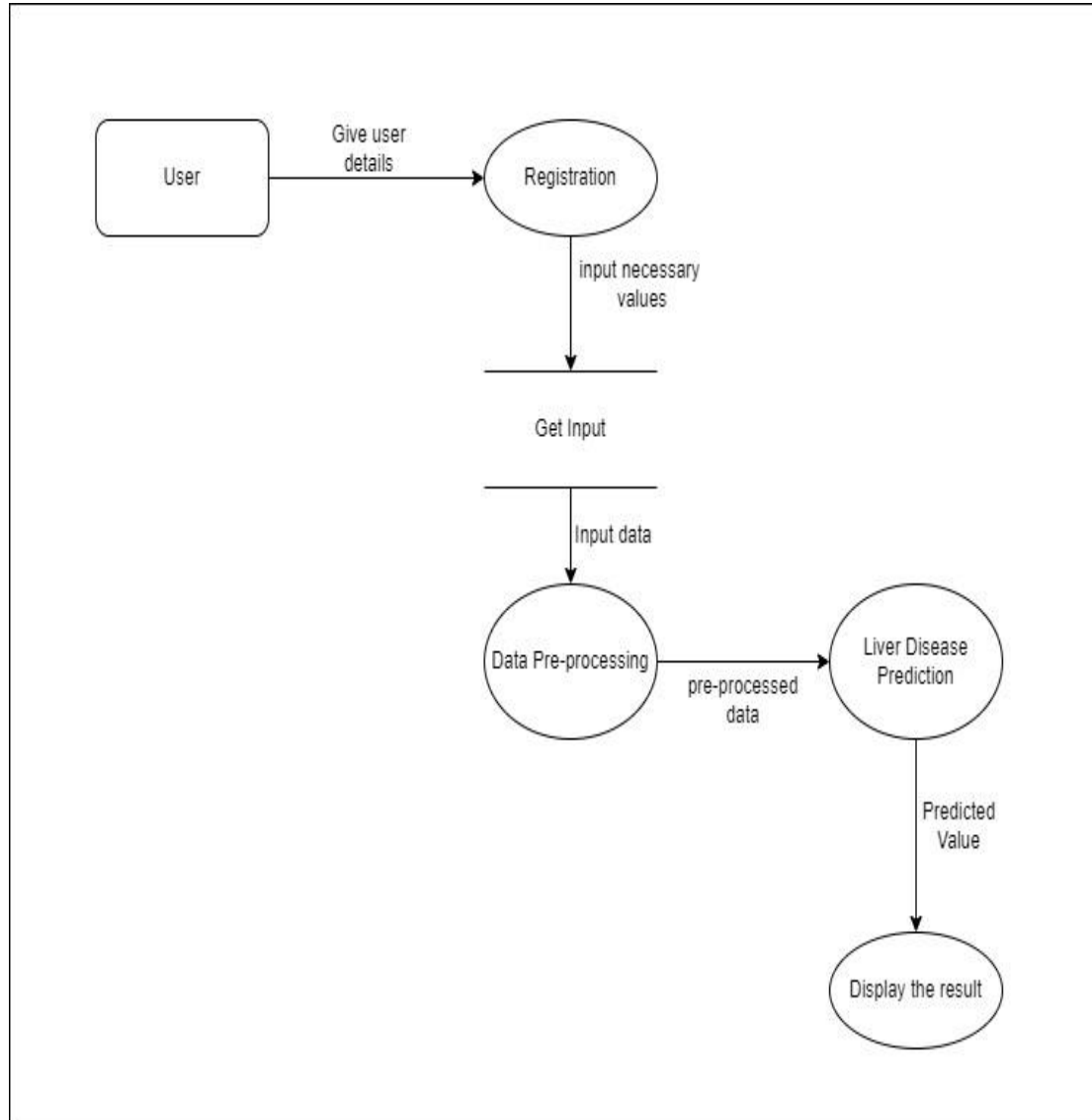
4.2 Non - Functional Requirements

Following are the Non-functional requirements of the proposed solution.

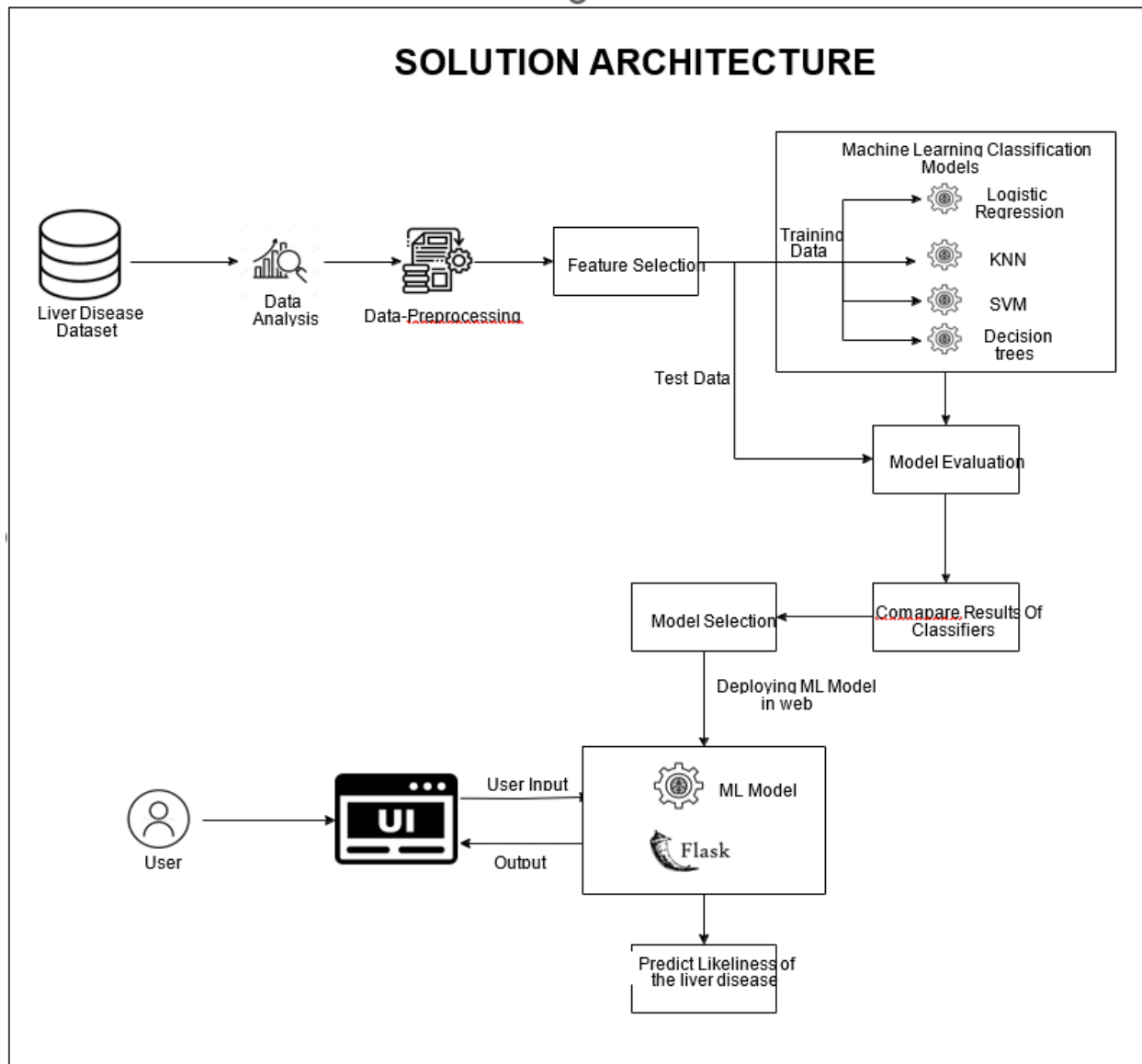
FR No.	Non-Functional Requirement (Epic)	Sub Requirement (Story/Sub-Task)
NFR-1	Usability	Accurate prediction of likeliness of the liver disease for the input details given by the user.
NFR-2	Security	Authentication through sign in process
NFR-3	Reliability	Ensure the model is reliable
NFR-4	Availability	Service should be readily available for 100% of the users
NFR-5	Performance	Use efficient ML techniques for better accuracy
NFR-5	Scalability	Handling more number of users.

5 PROJECT DESIGN

5.1 Data Flow Diagrams



5.2 Solution Technical Architecture



Technical Architecture

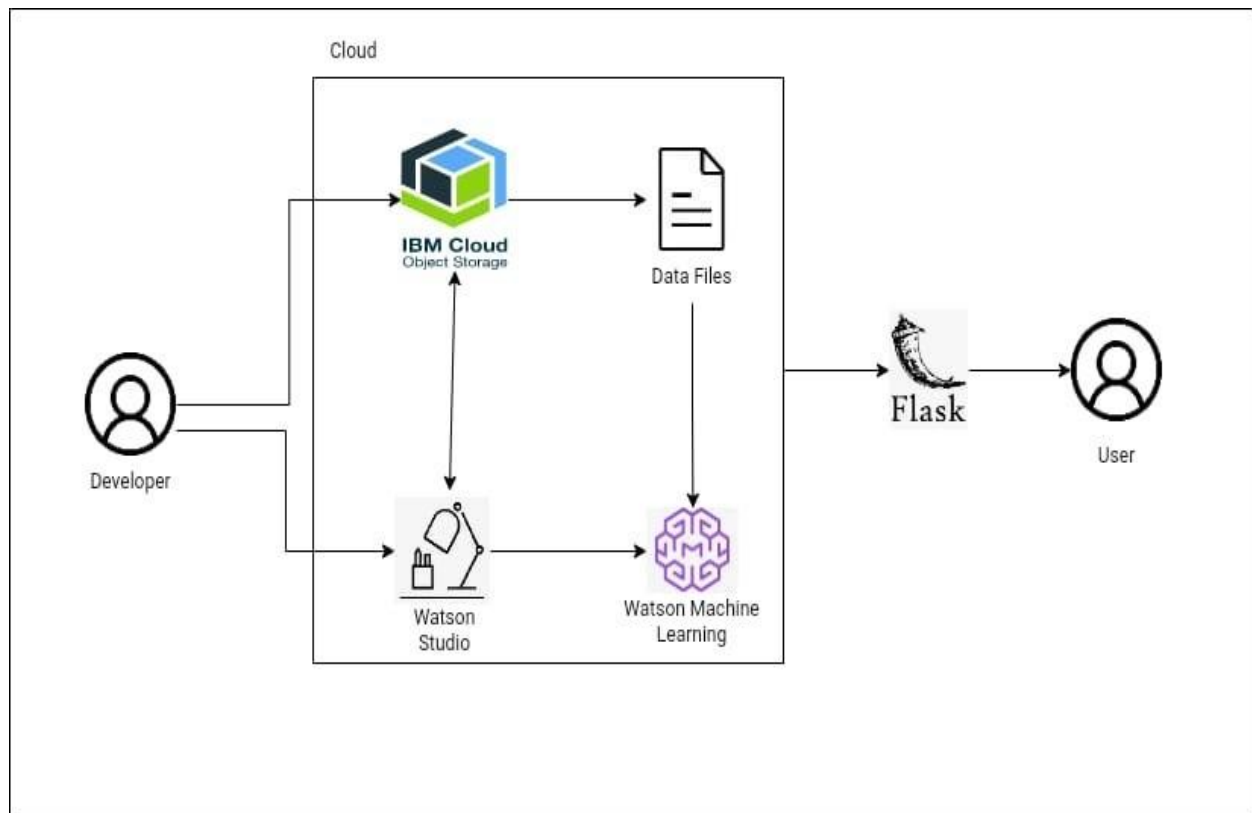


Table-1: Components & Technologies:

S.No	Component	Description	Technology
1.	User Interface	How user interacts with application e.g. Web UI, Mobile App, Chatbot etc.	HTML, CSS, JavaScript
2.	Application Logic-1	Logic for a process in the application	Python
3.	Application Logic-2	Logic for a process in the application	IBM Watson Studio
4.	Application Logic-3	Logic for a process in the application	IBM Watson Machine Learning
5.	Database	Data Type, Configurations etc.	IBM Cloud Object storage
6.	Cloud Database	Database Service on Cloud	IBM DB2, IBM Cloudant
7.	File Storage	File storage requirements	IBM Block Storage or Other Storage Service or Local Filesystem
8.	Machine Learning Model	Purpose of Machine Learning Model	Classification Model
9.	Infrastructure (Server / Cloud)	Application Deployment on Local System / Cloud Local Server Configuration: Cloud Server Configuration :	Local, Cloud Foundry

Table-2: Application Characteristics:

S.No	Characteristics	Description	Technology
1.	Open-Source Frameworks	List the open-source frameworks used	Flask
2.	Security Implementations	List all the security / access controls implemented, use of firewalls etc.	IAM Controls
3.	Scalable Architecture	Justify the scalability of architecture (3 – tier, Microservices)	Cloud Foundry, IBM Cloudant
4.	Availability	Justify the availability of application (e.g. use of load balancers, distributed servers etc.)	Cloud Foundry
5.	Performance	Design consideration for the performance of the application (number of requests per sec, use of Cache, use of CDN's) etc.	Cloud Foundry

5.3 User Stories

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer (Mobile user)	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming my password.	I can access my account / dashboard	High	Sprint-1
		USN-2	As a user, I will receive confirmation email once I have registered for the application	I can receive confirmation email & click confirm	High	Sprint-1
	Login	USN-3	As a user, I can log into the application by entering email & password	I can access my account	High	Sprint-1
Customer (Web user)	Details	USN-4	Should enter necessary details	Can check the entered details	High	Sprint-1
Customer Care Executive	Query	USN-5	As a user, I can give customer feedback	I can get a proper information	Medium	Sprint-2
	Help	USN-6	As a user, I can ask how to use the application	I can clarify regarding the usage of the app	High	Sprint-1
Administrator	Customization	USN-7	As a user, I can change the settings	I can change theme of the app	Low	Sprint-1

6 PROJECT PLANNING SCHEDULING

6.1 Sprint Planning & Estimation

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming my password.	5	High	Haris Murugan Hudson Yeshwanth Aravind
Sprint-1		USN-2	As a user, I will receive confirmation email once I have registered for the application	5	High	Haris Murugan Hudson Yeshwanth Aravind
Sprint-1	Login	USN-3	As a user, I can log into the application by entering email & password	10	High	Haris Murugan Hudson Yeshwanth Aravind
Sprint-2	Input Necessary Details	USN-4	As a user, I can give Input Details to Predict Likelihood of Liver Disease.	15	High	Haris Murugan Hudson Yeshwanth Aravind
Sprint-2	Data Pre-Processing	USN-5	Transform raw data into suitable format for prediction.	5	High	Haris Murugan Hudson Yeshwanth Aravind
Sprint-3	Prediction of Liver Disease	USN-6	As a user, I can predict Liver Disease using machine learning model.	10	High	Haris Murugan Hudson Yeshwanth Aravind
Sprint-3		USN-7	As a user, I can get accurate prediction of liver disease.	5	Medium	Haris Murugan Hudson Yeshwanth Aravind
Sprint-3	Model Deployment	USN-8	Deploy model into flask	5	high	Haris Murugan Hudson Yeshwanth Aravind
Sprint-4	Deployment	USN-8	As a user, I can see the previous history of predictions.	5	High	Haris Murugan Hudson Yeshwanth Aravind
Sprint-4	Deployment	USN-9	Deploy Website into real world	10	High	Haris Murugan Hudson Yeshwanth Aravind
Sprint-4	Deployment	USN-8	As a user, I can give feedback of the application.	5	High	Haris Murugan Hudson Yeshwanth Aravind

6.2 Sprint Delivery Schedule

Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date (Planned)	Story Points Completed (as on Planned End Date)	Sprint Release Date (Actual)
Sprint-1	20	6 Days	24 Oct 2022	29 Oct 2022	20	29 Oct 2022
Sprint-2	20	6 Days	31 Oct 2022	05 Nov 2022	20	05 Nov 2022
Sprint-3	20	6 Days	07 Nov 2022	12 Nov 2022	20	12 Nov 2022
Sprint-4	20	6 Days	14 Nov 2022	19 Nov 2022	20	19 Nov 2022

6.3 Reports from JIRA

LD Sprint 1Add dates (3 issues)

000Start sprint...

LD-7As a user, I can register for the application by entering my email, password, and confirming my password.

DONE▼

LD-8As a user, I can log into the application by entering email & password

DONE▼

LD-9As a user, I can give Input Details to Predict Likeliness of Liver Disease

DONE▼

+ Create issue

LD Sprint 2Add dates (3 issues)

000Start sprint...

LD-10Data Analysis

DONE▼

LD-11Data Pre-processing

DONE▼

LD-12Model Building

DONE▼

+ Create issue

LD Sprint 3Add dates (3 issues)

000Start sprint...

LD-13Deploy model using Flask

DONE▼

LD-14As a user, I can predict Liver Disease using machine learning model

DONE▼

LD-15As a user, I can get accurate prediction of liver disease.

DONE▼

+ Create issue

LD Sprint 4Add dates (2 issues)

000Start sprint...

LD-16As a user, I can check previous history of predictions.

DONE▼

LD-17Deploy website into real world

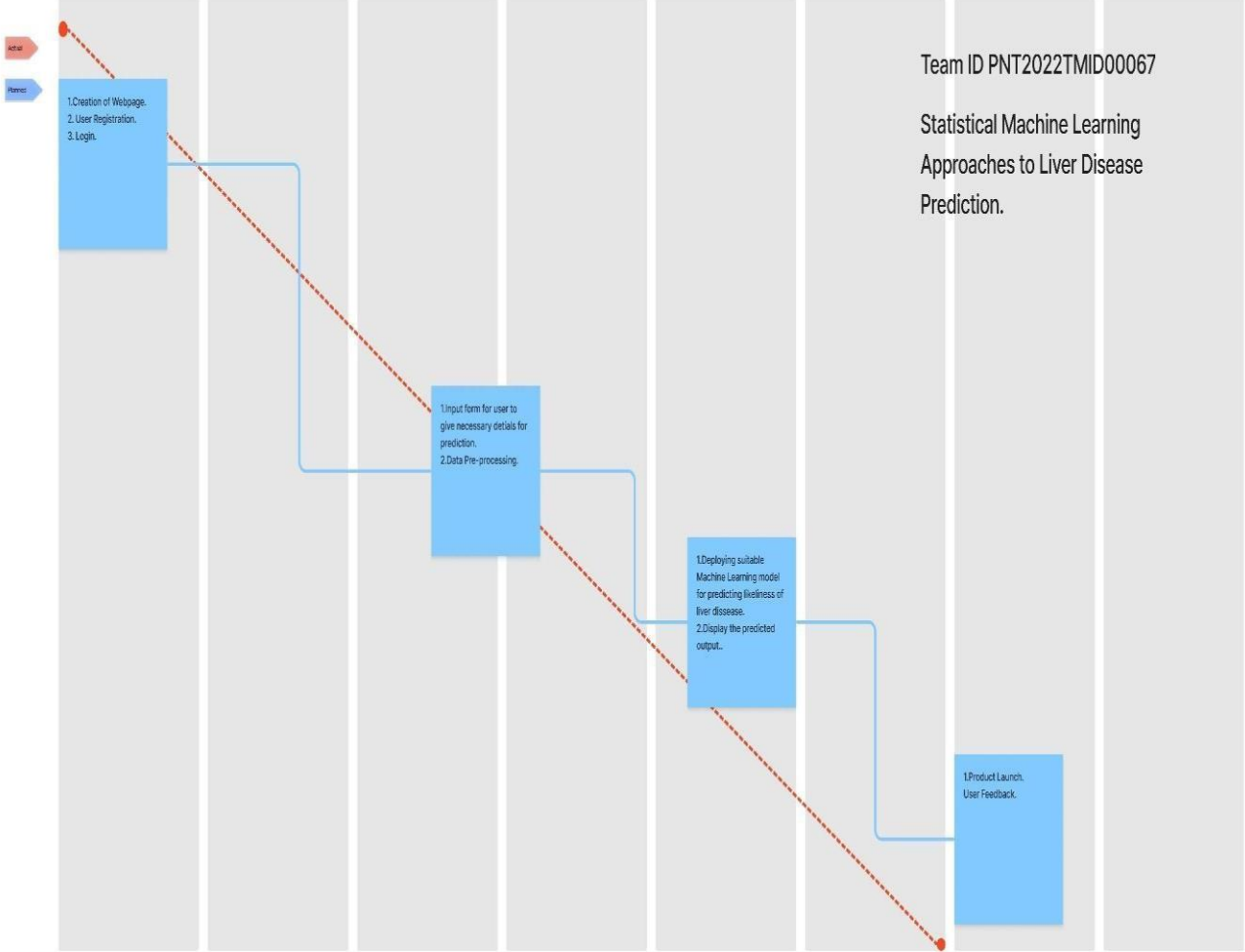
DONE▼

+ Create issue

October / November 2022

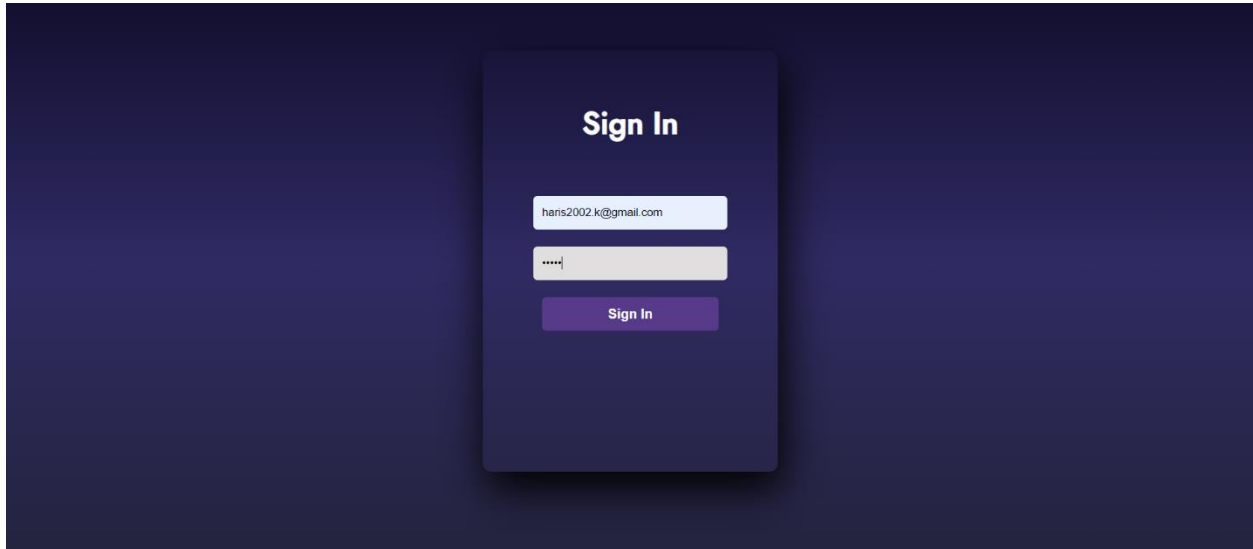
Oct

- 10/23 - 10/28



7 CODING & SOLUTIONING

Login Feature



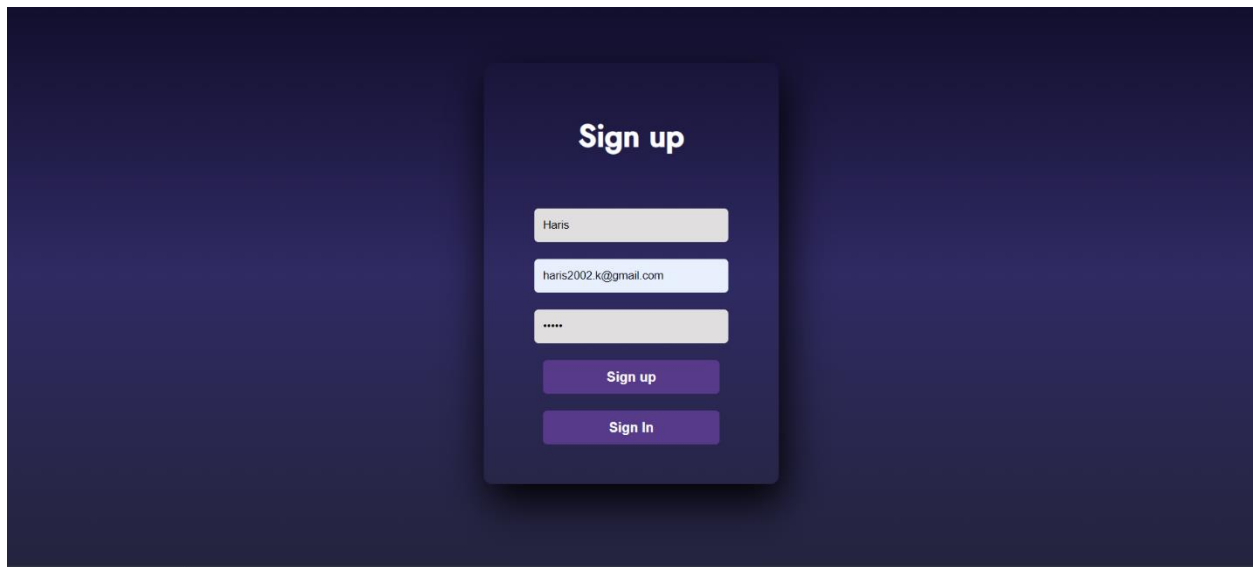
A login form titled "Sign In" centered on a dark blue background. The form is contained within a dark blue rounded rectangle. It features two input fields: the first is labeled with the email "haris2002.k@gmail.com" and the second is masked with "****". Below the inputs is a purple "Sign In" button.

Sign In

haris2002.k@gmail.com

Sign In

Register Feature



A registration form titled "Sign up" centered on a dark blue background. The form is contained within a dark blue rounded rectangle. It features three input fields: the first is labeled "Haris", the second is labeled with the email "haris2002.k@gmail.com", and the third is masked with "****". Below the inputs are two buttons: a purple "Sign up" button and a purple "Sign In" button.

Sign up

Haris

haris2002.k@gmail.com

Sign up

Sign In

Classification Feature



LIVER DISEASE PREDICTION APP



Welcome Haris

Input Data

Age

21

Gender

☒ Male ☐ Female

Total Billirubin

3.9

Direct Billirubin

1.9

Alkaline Phosphate

150

Alamine Aminotransferase

36

Asparatate Aminotransferase

27

Total Proteins

6.8

Albumin

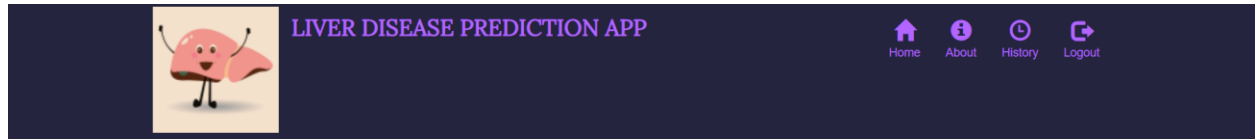
3.9

Albumin and Globulin Ratio

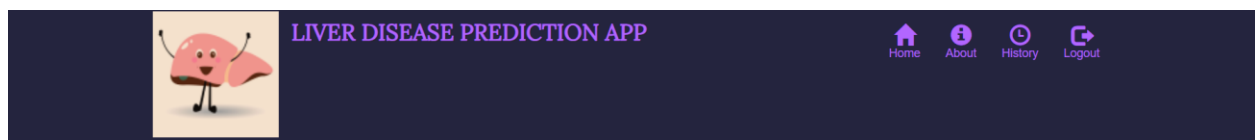
1.34

Submit

Result

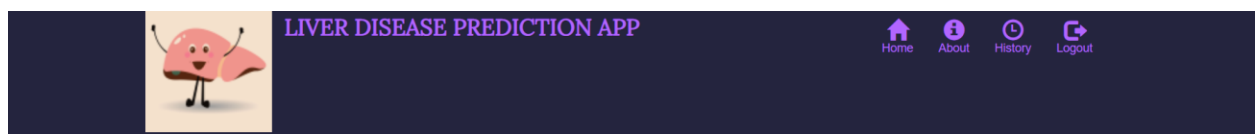


YOU DO NOT HAVE ANY SYMPTOMS RELATED TO LIVER DISEASE! 😊



Predicted Result shows there is a possibility of Liver Disease

Classification History Feature



Age	Total Billirubin	Direct Billirubin	Alkaline Phosphate	Alamine Aminotransferase	Asparatate Aminotransferase	Total proteins	Albumin	Al and Gb Ratio	Prediction
54.0	1.0	0.2	239.0	29.0	23.0	3.0	3.0	1.85	Liver Disease
42.0	0.8	0.2	187.0	16.0	18.0	6.8	3.3	0.9	No Liver Disease
42.0	0.8	0.2	187.0	16.0	18.0	6.8	3.3	0.9	No Liver Disease
54.0	0.7	0.2	198.0	16.0	18.0	6.8	3.3	0.9	No Liver Disease
54.0	0.7	0.2	198.0	16.0	18.0	6.8	3.3	0.9	No Liver Disease
54.0	0.7	0.2	198.0	16.0	18.0	6.8	3.3	0.9	No Liver Disease
54.0	0.7	0.2	187.0	16.0	18.0	6.6	3.3	0.9	No Liver Disease
42.0	0.7	0.2	187.0	16.0	18.0	6.8	3.3	0.9	No Liver Disease
42.0	0.7	0.2	187.0	16.0	18.0	6.8	3.3	0.9	No Liver Disease
42.0	0.7	0.2	187.0	16.0	18.0	6.8	3.3	0.9	No Liver Disease
54.0	0.7	0.2	187.0	16.0	18.0	6.8	3.3	0.9	No Liver Disease
54.0	0.7	0.2	187.0	16.0	18.0	6.8	3.3	0.9	No Liver Disease

8 TESTING

8.1 Test Cases

8.1.1 Functional Testing

Functional test can be defined as testing two or more modules together with the intent of finding defects, demonstrating that defects are not present, verifying that the module performs its intended functions as stated in the specification and establishing confidence that a program does what it is supposed to do.

8.1.2 White Box Testing

Testing based on an analysis of internal workings and structure of a piece of software. This testing can be done using the percentage value of load and energy. The tester should know what exactly is done in the internal program. Includes techniques such as Branch Testing and Path Testing. Also known as Structural Testing and Glass Box Testing.

8.1.3 Black Box Testing

Testing without knowledge of the internal workings of the item being tested. Tests are usually functional. This testing can be done by the user who has no knowledge of how the shortest path is found.

8.2 User Acceptance Testing

Acceptance testing can be defined in many ways, but a simple definition is the succeeds when the software functions in a manner that can be reasonably expected by the customer. After the acceptance test has been conducted, one of the two possible conditions exists. This is to find whether the inputs are accepted by the database or other validations. For example accept only numbers in the numeric field, date format data in the date field. Also the null check for the not null fields. If any error occurs then show the error messages. The function of performance characteristics to specification and is accepted. A deviation from specification is uncovered and a deficiency list is created. 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.

8.3 Test Results


Section	Total Cases	Not Tested	Fail	Pass
Registration	5	0	0	5
Login	5	0	0	5
Security (Log In and Log out)	2	0	0	2
Liver Disease Prediction	10	0	0	10
No Liver Disease Prediction	10	0	0	10
Final Output	10	0	0	10

			Team ID	PNT2022TMD00067							
			Project Name	Statistical Machine Learning Approach							
			Maximum Marks	4 marks							
Test case ID	Feature Type	Component	Test Scenario	Steps To Execute	Test Data	Expected Result	Actual Result	Status	TC for Automation(Y/N)	Executed By	
LoginPage_TC_OO 1	Functional	Login Page	Verify user is able to see the Login/Signup popup when user clicked on Sign in	1.Enter URL and click go 2.Verify login/Signup popup displayed or not		Login/Signup popup should display	Working as expected	Pass	Y	Haris Murugan K	
LoginPage_TC_OO 2	UI	Login Page	Verify the UI elements in Login/Signup popup	1.Enter URL and click go 2.Verify login/Signup popup with below UI elements: a.email text box b.password text box c.Sign in button d. Sign Up button		Application should show below UI elements: a.email text box b.password text box c.Sign up button d.Sign in Button	Working as expected	Pass	Y	Hudson G	
LoginPage_TC_OO 3	Functional	Login page	Verify user is able to log into application with Valid credentials	1.Enter URL 2.Click on My Account dropdown button 3.Enter Valid username/email in Email text box 4.Enter valid password in password text box 5.Click on login button	Username: sample@gmail.com password: Test123	User should navigate to user account homepage	Working as expected	Pass	Y	Aravind J	
LoginPage_TC_OO 4	Functional	Login page	Verify user is able to log into application with Invalid credentials	1.Enter URL 2. Click on My Account dropdown button 3.Enter invalid username/email in Email text box 4.Enter valid password in password text box 5.Click on sign in button	Username: sample1@gmail.com password: 0000	Application should show 'Incorrect email or password' validation message.	Working as expected	Pass	Y	Yeshwanth K S	
LoginPage_TC_OO 4	Functional	Login page	Verify user is able to log into application with Invalid credentials	1.Enter URL 2. Click on My Account dropdown button 3.Enter valid username/email in Email text box 4.Enter invalid password in password text box 5.Click on Sign in button	Username: sample@gmail.com password: Testing	Application should show 'Incorrect email or password' validation message.	Working as expected	Pass	Y	Haris Murugan K	
LoginPage_TC_OO 5	Functional	Login page	Verify user is able to log into application with Invalid credentials	1.Enter URL 2.Click on My Account dropdown button 3.Enter invalid username/email in Email text box 4.Enter invalid password in password text box 5.Click on Sign in button	Username: testing password: Te123	Application should show 'Incorrect email or password' validation message.	Working as expected	pass	Y	Hudson G	
RegisterPage_TC_O 01	Functional	Register Page	Verify User is able to see register Page	1.Enter URL 2.Click Sign Up Button		Register Page should display	Working as expected	Pass	Y	Aravind J	
RegisterPage_TC_O 02	UI	Register Page	Verify the UI elements in Login/Signup popup	1.Enter URL and click go 2.Verify login/Signup popup with below UI elements: a.email text box b.password text box c.username text box d. Sign Up button		Application should show below UI elements: a.email text box b.password text box c.Username button d.Sign up and Sign in Button	Working as expected	Pass	Y	Yeshwanth K S	

4	RegisterPage_TC_003	Functional	Register Page	Verify user is able to register details	1.Enter URL 2.Enter username 3.Enter Valid username/email in Email text box 4.Enter valid password in password text box 5.Click on Sign up button	username: Sample Email: sample@gmail.com Password: 12345	Registration Successful	Working as expected	Pass	Y	Haris Murugan K
5	InputFormPage_TC_001	Functional	Input form page	Verify user is able to see Input form Page	1. Sign In onto the website 2.Click home button		Input form should display	Working as expected	Pass	Y	Hudson G
6	InputFormPage_TC_001	Functional	Input form page	Verify user is able to see all the input fields and able give inputs	1. Sign In onto the website 2.Click home button 3.Verify all the required input field for prediction is visible 4.Give input values in input field 5.click submit	enter values for Age,Gender,Total_Bilirubin, Direct_Bilirubin,Alkaline_P_hosphatase,Alamine_Amino transferase,Aspartate_Amino transferase,Total_Protiens ,Albumin,Albumin_and_Globulin_Ratio Dataset as 65,Female,0.7,0.1,187,16,18,6,8,3,3,0,9,1	Input form should accept valid values for processing and produce error when invalid data is given	Working as expected	Pass	Y	Aravind J
7	ResultPage_TC_001	Functional	Result Page	Verify user is able to see result page after giving inputs	1. Sign In onto the website 2.Click home button 3.Verify all the required input field for prediction is visible 4.Give input values in input field 5.click submit		Result page should be visible	Working as expected	Pass	Y	Yeshwanth K S
8	ResultPage_TC_002	Functional	Result Page	Verify user is able to see the prediction result	1. Sign In onto the website 2.Click home button 3.Verify all the required input field for prediction is visible 4.Give input values in input field 5.click submit 6.Verify if it displays predicted message	enter values for Age,Gender,Total_Bilirubin, Direct_Bilirubin,Alkaline_P_hosphatase,Alamine_Amino transferase,Aspartate_Amino transferase,Total_Protiens ,Albumin,Albumin_and_Globulin_Ratio Dataset as 65,Female,0.7,0.1,187,16,18,6,8,3,3,0,9,1	Predicted Liver Disease Message should be visible.	Working as expected	Pass	Y	Haris Murugan K

9 RESULTS

9.1 Performance Metrics

S No	Parameter	Values	Screenshot
1.	Metrics	Classification Model: Confusion Matrix , Accuracy Score & Classification Report	Logistic Regression   Random Forest Classifier

```
[ ] x_test_predict=rfc.predict(x_test)

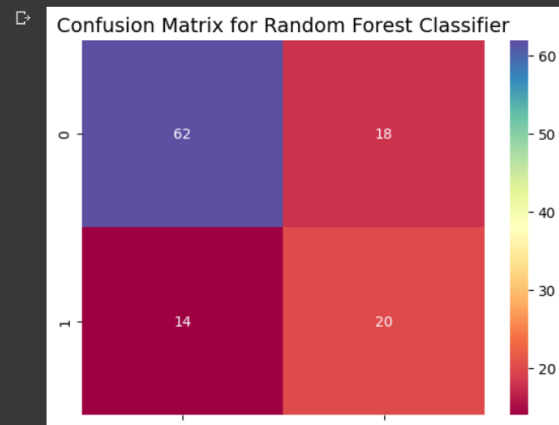
[ ] RandomForestClassifierScore = rfc.score(x_train,y_train)
print("Accuracy obtained by Random Forest Classifier model:",RandomForestClassifierScore*100)
RandomForestClassifierScore = rfc.score(x_test, y_test)
print("Accuracy obtained by Random Forest Classifier model:",RandomForestClassifierScore*100)
```

Accuracy obtained by Random Forest Classifier model: 100.0
Accuracy obtained by Random Forest Classifier model: 71.9298245614035

```
[ ] report=classification_report(y_train,x_train_predict)
print(report)
```

	precision	recall	f1-score	support
1	1.00	1.00	1.00	313
2	1.00	1.00	1.00	313
accuracy			1.00	626
macro avg	1.00	1.00	1.00	626
weighted avg	1.00	1.00	1.00	626

```
y_pred_rfc = rfc.predict(x_test)
cf_matrix = confusion_matrix(y_test, x_test_predict)
sns.heatmap(cf_matrix, annot=True, cmap="Spectral")
plt.title("Confusion Matrix for Random Forest Classifier", fontsize=14,y=1.03);
```



KNN

```
[ ] KNeighborsClassifierScore = KNN.score(x_test, y_test)
print("Accuracy obtained by K Neighbors Classifier model:",KNeighborsClassifierScore*100)
```

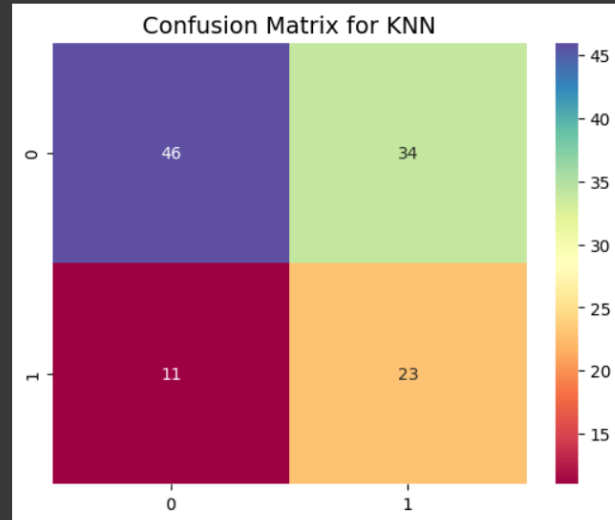
Accuracy obtained by K Neighbors Classifier model: 60.526315789473685

```
[ ] Accuracy_test_knn=accuracy_score(y_test,pred_knn)
```

```
[ ] Accuracy_test_knn
0.6052631578947368
```



```
cf_matrix = confusion_matrix(y_test, pred_KNN)
sns.heatmap(cf_matrix, annot=True, cmap="Spectral")
plt.title("Confusion Matrix for KNN", fontsize=14,y=1.03);
```



Decision Tree Classifier

```
[ ] Accuracy_test_DT=accuracy_score(y_test,x_test_pred_DT)
Accuracy_test_DT
```

0.6754385964912281



```
report_test_DT=classification_report(y_test, x_test_pred_DT)#
print(report_test_DT)
```

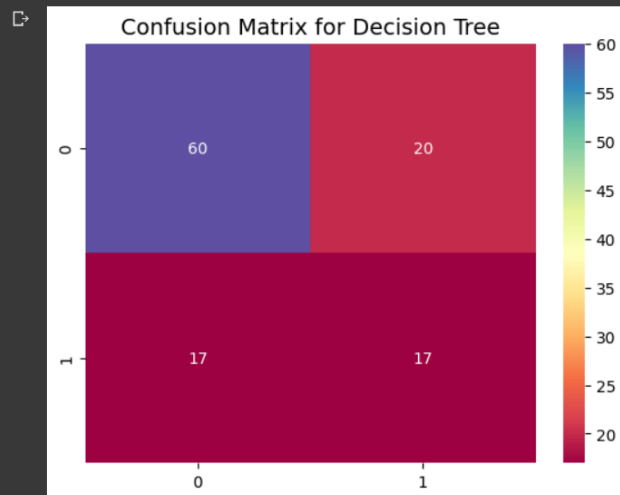


	precision	recall	f1-score	support
1	0.78	0.75	0.76	80
2	0.46	0.50	0.48	34
accuracy			0.68	114
macro avg	0.62	0.62	0.62	114
weighted avg	0.68	0.68	0.68	114

```
[ ] print('Testing score for DT regression is',DT.score(x_test,y_test))
```

Testing score for DT regression is 0.6754385964912281

```
plt.title("Confusion Matrix for Decision Tree", fontsize=14,y=1.03);
```

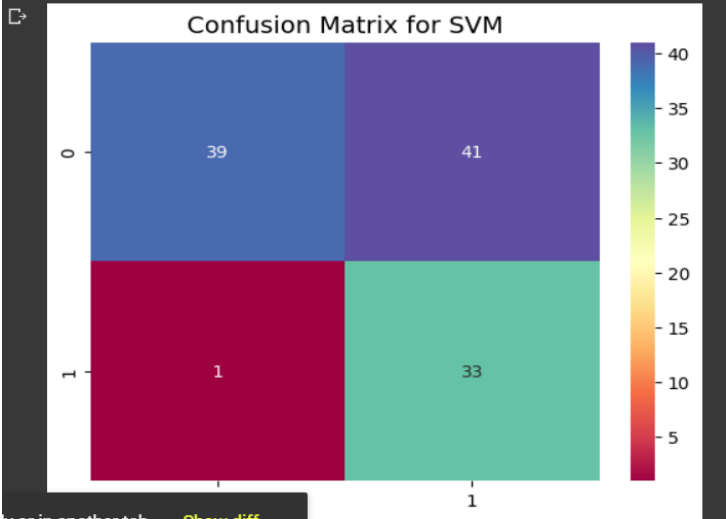


Support Vector Machine

```
[75] x_test_predict=svclassifier.predict(x_test)

[76] report=classification_report(y_test,x_test_predict)
print(report)
```

	precision	recall	f1-score	support
1	0.97	0.49	0.65	80
2	0.45	0.97	0.61	34
accuracy			0.63	114
macro avg	0.71	0.73	0.63	114
weighted avg	0.82	0.63	0.64	114

			<pre>cf_matrix = confusion_matrix(y_test, x_test_predict) sns.heatmap(cf_matrix, annot=True, cmap="Spectral") plt.title("Confusion Matrix for SVM", fontsize=14,y=1.03);</pre>  <p>The heatmap displays the confusion matrix for an SVM model. The x-axis represents the predicted classes (0 and 1) and the y-axis represents the actual classes (0 and 1). The color scale ranges from 5 (dark red) to 40 (dark blue). The counts are: True Positives (TP) = 39, True Negatives (TN) = 41, False Positives (FP) = 1, and False Negatives (FN) = 33.</p>
			<p>Conclusion</p> <p>Random Forest Classifier as the highest accuracy but it is overfitting the training data.</p> <p>Therefore, we used Logistic Regression model which gives second best accuracy.</p>
2.	Tune the Model	Hyperparameter Tuning, Validation Method	<p>Grid Search CV For Logistic Regression</p> <pre>from sklearn.model_selection import GridSearchCV model = LogisticRegression() solvers = ['newton-cg', 'lbfgs', 'liblinear'] penalty = ['l2'] c_values = [100, 10, 1.0, 0.1, 0.01] # define grid search grid = dict(solver=solvers,penalty=penalty,C=c_values) grid_search = GridSearchCV(estimator=model, param_grid=grid, n_jobs=-1, scoring='accuracy',error_score=0) grid_result = grid_search.fit(x_train, y_train) # summarize results print("Best: %f using %s" % (grid_result.best_score_, grid_result.best_params_)) means = grid_result.cv_results_['mean_test_score'] stds = grid_result.cv_results_['std_test_score'] params = grid_result.cv_results_['params'] for mean, stdev, param in zip(means, stds, params): print("%f (%f) with: %s" % (mean, stdev, param))</pre> <p>Best: 0.702933 using {'C': 100, 'penalty': 'l2', 'solver': 'newton-cg'} 0.702933 (0.030790) with: {'C': 100, 'penalty': 'l2', 'solver': 'newton-cg'} 0.702933 (0.030790) with: {'C': 100, 'penalty': 'l2', 'solver': 'lbfgs'} 0.702933 (0.030790) with: {'C': 100, 'penalty': 'l2', 'solver': 'liblinear'} 0.699733 (0.039575) with: {'C': 10, 'penalty': 'l2', 'solver': 'newton-cg'} 0.699733 (0.039575) with: {'C': 10, 'penalty': 'l2', 'solver': 'lbfgs'} 0.696546 (0.040268) with: {'C': 10, 'penalty': 'l2', 'solver': 'liblinear'} 0.693333 (0.015549) with: {'C': 1.0, 'penalty': 'l2', 'solver': 'newton-cg'} 0.693333 (0.015549) with: {'C': 1.0, 'penalty': 'l2', 'solver': 'lbfgs'} 0.690146 (0.018940) with: {'C': 1.0, 'penalty': 'l2', 'solver': 'liblinear'}</p>

			<pre>{'C': 100, 'penalty': 'l2', 'solver': 'newton-cg'} model_lor= LogisticRegression(C=0.01, penalty= 'l2', solver= 'liblinear') model_lor.fit(x_train,y_train) LogisticRegression(C=0.01, solver='liblinear') [] x_test_predict_lor=model_lor.predict(x_test) [] accuracy = accuracy_score(y_test,x_test_predict_lor) accuracy 0.6403508771929824</pre> <h3>Grid Search CV For SVM</h3> <pre>[] parameters = {'kernel':['linear','rbf'], 'C':[0.1,0.5,1.0], 'gamma':['scale', 'auto']} [] clf=GridSearchCV(SVC(),param_grid=parameters,verbose=2) [] clf.fit(x_train,y_train) Fitting 5 folds for each of 12 candidates, totalling 60 fits [CV] ENDC=0.1, gamma=scale, kernel=linear; total time= 0.0s [CV] ENDC=0.1, gamma=scale, kernel=linear; total time= 0.0s [CV] ENDC=0.1, gamma=scale, kernel=linear; total time= 0.0s [CV] ENDC=0.1, gamma=scale, kernel=linear; total time= 0.0s [] clf.best_score_ 0.6901079365079363 [] clf.best_params_ {'C': 1.0, 'gamma': 'scale', 'kernel': 'linear'} [] model_svc=SVC(C=1,gamma='scale',kernel='linear') model_svc.fit(x_train,y_train) SVC(C=1, kernel='linear') [] y_=model_svc.predict(x_test) [] accuracy_score(y_test,y_) 0.6578947368421053</pre>
--	--	--	--

			<div><div><div><div><div><div></div><div>pd.crosstab(y_test,y_)</div></div></div><div><div></div><div>col_012</div></div><div>Dataset</div><div><div>14238</div><div>2133</div></div></div></div><div><div>[]</div><div>print(classification_report(y_test,y_))</div></div><div><div>precisionrecallf1-score support</div><div>10.980.530.6880</div><div>20.460.970.6334</div><div>accuracy0.66114</div><div>macro avg0.720.750.66114</div><div>weighted avg0.820.660.67114</div></div></div>
--	--	--	--

10 ADVANTAGES & DISADVANTAGES

Advantages

- Get instantaneous results
- Accurate prediction of whether user as liver disease or not
- User data is secure
- Authentication using login process
- User friendly and simple user interface
- Previous results are added in history

Disadvantages

- Type of liver disease is not handled

11 CONCLUSION

In this project, We have built an effective machine learning model to predict the likeliness of the liver disease. We trained data with various machine learning models for finding the effective model. Our results showed that random forest classifier gives the best accuracy but it overfits the data. Therefore, we have taken decision tree classifier which is the next accurate model. We deployed the model in the IBM cloud with Watson studio and Machine learning service. Then we integrated the model with flask. User can predict the disease by entering parameters in the web application.

The user gives the input through form and user input undergoes scaling process and feed into the machine learning model which gives the predicted result. The predicted message is displayed to the user.

The machine learning algorithms presented in this study can support medical experts but are not the alternative when making decisions from ML classifiers for diagnostic pathways. These methods can reduce many of the limitations that occur in healthcare associated with inaccuracy in diagnoses, missing data, cost, and time. Application of the ML methods can help reduce the total burden of liver disease on public health worldwide by improving recognition of risk factors and diagnostic variables. More importantly, for chronic liver disease, detecting liver disease at earlier stages or in hidden cases by ML could decrease liver-related mortality, transplants, and/or hospitalizations. Early detection improves prognosis, since treatment can be given before progression of the disease to later stages.

12 FUTURE SCOPE

In future work, we can improve the accuracy of the results by training the model against large dataset. We can integrate this design in the hospital with the instruments that is used to detect the measurements of input elements like proteins etc. So that doctors can automate the process of prediction and get an idea about whether the patient has liver disease or not. We can also predict the type of liver disease if liver disease exist.

13 APPENDIX

Source Code

```
import requests
from sklearn.preprocessing import MinMaxScaler
import numpy as np
import flask
from flask import request, render_template, session, redirect, url_for
from flask_cors import CORS
from ibm_db import connect
from ibm_db import fetch_assoc
from ibm_db import exec_immediate
from ibm_db import tables
import joblib
import os

response=flask.Response()

# NOTE: you must manually set API_KEY below using information retrieved from your
IBM Cloud account.
API_KEY = os.environ["api_key"]
token_response = requests.post('https://iam.cloud.ibm.com/identity/token',
data={"apikey": API_KEY, "grant_type": 'urn:ibm:params:oauth:grant-type:apikey'})
mltoken = token_response.json()["access_token"]
header = {'Content-Type': 'application/json', 'Authorization': 'Bearer ' +
mltoken}

app = flask.Flask(__name__, static_url_path='')
CORS(app)
app.secret_key=os.urandom(12)

try:
    print("Connecting")
    conn_string = os.environ["database_url"]
    con=connect(conn_string,"","")
except:
    print("Unable to connect")

def results(command):

    ret = []
    result = fetch_assoc(command)
    while result:
        ret.append(result)
```

```

        result = fetch_assoc(command)
        return ret

@app.route('/', methods=['GET'])
def sendHomePage():
    return render_template('index.html')

@app.route('/register', methods=['GET', 'POST'])
def register():
    msg = ''
    if request.method == 'POST' and 'username' in request.form and 'password' in
request.form and 'email' in request.form:
        username = request.form['username']
        password = request.form['password']
        email = request.form['email']
        sql1="select username from user1 where email='"+email+"'"
        try:
            rows = results(exec_immediate(con, sql1))
            if rows:
                return render_template('index.html',msg="Email Already
Registered")
            else:
                sql2="insert into user1(username,email,password)
values('"+username+"','"+email+"','"+password+"')"
                print("Hello")
                exec_immediate(con,sql2)
                return render_template('login.html')
        except:
            print("Failed")
    return render_template('index.html')

@app.route('/gotologin', methods=['GET', 'POST'])
def gotologin():
    return render_template('login.html')

@app.route('/login', methods=['GET', 'POST'])
def login():
    msg = ''

    if request.method == 'POST' and 'password' in request.form and 'email' in
request.form:

        password = request.form['password']
        email = request.form['email']
        sql1="select * from user1 where email='"+email+"' and
password='"+password+"'"

```

```

        try:
            rows = results(exec_immediate(con, sql1))
            print(rows[0]['USERNAME'])
            if rows:
                ##login_user(rows[0]['UID'])
                session['uid']=email
                session['uname']=str(rows[0]['USERNAME'])
                return redirect('/home')

            else:
                return render_template('login.html',msg='Email or Password is
Incorrect')
        except:
            print("Failed")

        print('uid' in session)
    elif 'uid' not in session:
        return render_template("login.html")

    return render_template('login.html')

@app.route('/home')
def home():
    if 'uid' in session:
        return render_template('dataform.html',msg=session['uname'])
    return render_template('login.html')

@app.route('/logout')
def logout():
    session.pop('uid',None)
    print('uid' in session)
    return redirect(url_for('login'))

@app.after_request
def after_request(response):
    response.headers["Cache-Control"] = "no-cache, no-store, must-revalidate"
    return response

@app.route('/predict', methods=['POST'])
def predictSpecies():
    if 'uid' in session:
        if request.method=="POST":
            email=session['uid']
            age = float(request.form['age'])
            gender = float(request.form['gender'])
            tb = float(request.form['tb'])
            db = float(request.form['db'])

```

```

        ap = float(request.form['ap'])
        aa1 = float(request.form['aa1'])
        aa2 = float(request.form['aa2'])
        tp = float(request.form['tp'])
        a = float(request.form['a'])
        agr = float(request.form['agr'])
        X =
np.array([float(age),float(gender),float(tb),float(db),float(ap),float(aa1),float
(aa2),float(tp),float(a),float(agr))].reshape(1,-1);
        print(X)
        mc=joblib.load('mc1.pkl')
        scaled_x=mc.transform(X)
        X=scaled_x.tolist()
        print(X)
        payload_scoring = {"input_data": [{"field":
[['age','gender','tb','db','ap','aa1','aa2','tp','a','agr']], "values": X}]}
        response_scoring = requests.post('https://us-
south.ml.cloud.ibm.com/ml/v4/deployments/8216d53f-c157-4b9f-89af-
174229f60e03/predictions?version=2022-11-11',
json=payload_scoring,headers={'Authorization': 'Bearer ' + mltoken})
        print(response_scoring)
        predictions = response_scoring.json()
        predict = predictions['predictions'][0]['values'][0][0]
        print("Final prediction :",predict)

        if(predict==1):
            predict1='Liver Disease'
            msg="Predicted Result shows there is a possibility of Liver
Disease"
        else:
            predict1='No Liver Disease'
            msg="Predicted Result Shows No Liver Disease"
            sql2="insert into history
values('"+str(email)+"','"+str(age)+"','"+str(gender)+"','"+str(tb)+"','"+str(db)
+"','"+str(ap)+"','"+str(aa1)+"','"+str(aa2)+"','"+str(tp)+"','"+str(a)+"','"+str
(agr)+"','"+str(predict1)+"')"

            exec_immediate(con,sql2)
            # showing the prediction results in a UI# showing the prediction
results in a UI
            return render_template('predict.html', msg=msg)
        else:
            return render_template('login.html')

@app.route('/history')
def history():
    if 'uid' in session:

```

```
sql1="select * from history where email='"+str(session['uid'])+"'"
##try:
rows = results(exec_immediate(con, sql1))
if rows:
    print(len(rows))
    print(rows[0]['AGE'])
    return render_template('history.html',rows=rows)
else:
    return render_template('history.html')
##except:
    ##print("Failed")
return render_template('login.html')

if __name__ == '__main__':
    app.run(debug= False)
```

Github Link

<https://github.com/IBM-EPBL/IBM-Project-10321-1659165524>

Website Link

<https://liver-disease-predictor-app.herokuapp.com/>

Demo Link

https://drive.google.com/file/d/1r6aVsDf1_eKyhddofW5LscXpbwtQY49g/view?usp=sharing