Statistical Machine Learning Approaches to Liver Disease Prediction

IBM-Project-1940-1658420948

NALAIYA THIRAN PROJECT BASED LEARNING ON PROFESSIONAL READLINESS FOR INNOVATION, EMPLOYNMENT AND ENTERPRENEURSHIP

PROJECT REPORT

Submitted By

THANUJA VARSHINI R [2019506105]

VIJAYASHREE S [2019506113]

EZHIL MUKHI S [2019506029]

PRANUSHA BAVAN S [2019506064]

KAVIPRIYA C [2019506041]

TEAM ID : PNT2022TMID36043

INDUSTRY MENTOR : Mrs. Nidhi

FACULTY MENTOR : Mrs. Eliza Femi Sherley S , Teaching Fellow

EVALUATOR : Dr. P Kola Sujatha, Assistant Professor

MADRAS INSTITUTE OF TECHNOLOGY

CHENNAI - 600044

CONTENTS

1. INTRODUCTION

a. Project Overview

With a growing trend of sedentary life which promotes lack of physical activities, diseases related to the liver have become a common encounter nowadays. Liver diseases have caused millions of deaths every year. There are about 100 different types of liver infections. Liver diseases are not easily discovered in an early stage as even after being affected and undergoing partial damage, it will be functioning normally. Liver failures are at a high rate of risk among Indians. India is expected to become the World Capital for Liver Diseases by 2025. The widespread occurrence of liver diseases is contributed due to deskbound lifestyle, increased alcohol consumption and smoking.

With such complications, it is necessary to have a concern towards tackling these liver-based diseases. An early diagnosis of liver diseases will definitely increase patients' survival rate. Afterall, we cannot expect a developed and prosperous nation, with an unhealthy population.

In this project, we aim to examine data from liver patients concentrating on relationships between a list of liver enzymes, proteins, age, and gender, using them to try and predict the likelihood of the occurrence of a liver disease. The main objective of this project is to analyse the parameters of various classification algorithms and compare their predictive accuracies to find the best classifier for determining liver disease.

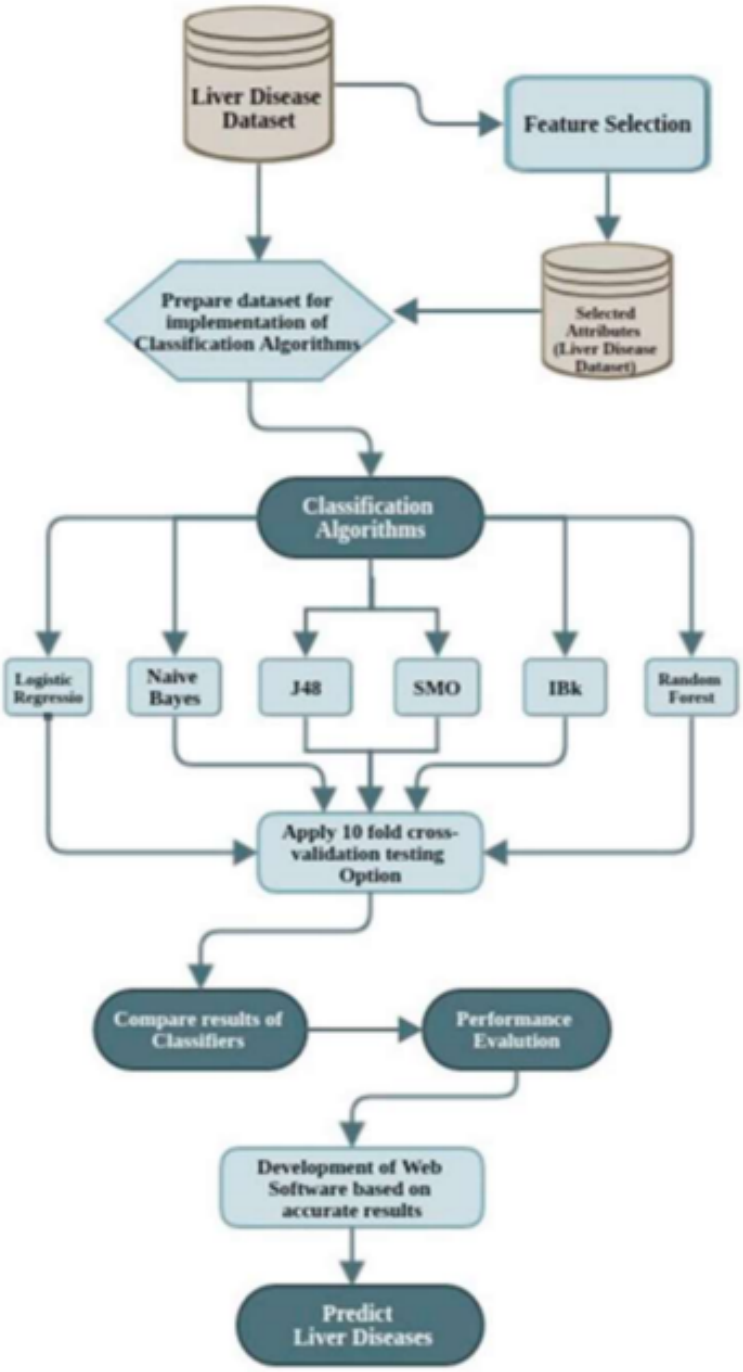
b. Problem Statement Definition:

Who does the problem affect?	Persons with liverdisease symptoms caused by drinking and other conditions
What are the boundaries of theproblem?	People who are indicative of liverdisease symptoms
What is the issue?	With a growing trend of sedentary life which promotes lack of physical activities, diseases related to the liver have become a common encounter nowadays. Liver diseases have caused millions of deaths every year. There are about100 different typesof liver infections. Liver diseases are not easily discovered in an early stage as even after beingaffected and undergoing partial damage, it will be functioning normally.

When does the issue occur?	The liver mainly getsaffected due to intake of alcohol. Intake of pain killer tablets and unusual food habits etc also contribute to liver damage.
Where does the issue occur?	Liver diseases disturb the normalfunctioning of the liver. Most urbancity dwellers face thisproblem.

Why is it important that we fix the problem?	In human beings, the liver is one of the most important parts of the body that performs many functions including the production of bile, excretion of bilirubin, metabolism of proteins and carbohydrates,activation of enzymes, storing vitamins, glycogen, andminerals etc.
What solution to solve this issue?	Early prediction of liver disease using classification algorithms is a beneficial task that can help the doctors to diagnose the disease within a short period of time. This method is cost-effective and saves time by predicting at early stages, preventing further liver damage.
What methodology used to solve the issue?	Machine Learning techniques are used to identify liver diseases and suggest precautions that can be taken for prevention and further treatment for the same.

c. Architecture:



2. LITERATURE SURVEY

1) Liver Disease Prediction Using Machine LearningAlgorithms

The liver disease is categorized by using feature selection and fuzzy K-means classification methods. To classify the liver disease, Euclidean distance method wasused to calculate the distance between each data and assuming k value from 0 to 3. Also, Decisiontree, Adaptive Neuro-Fuzzy Inference System (ANFIS)was also usedfor prediction. Higher accuracy of 98% is obtained from ANFIS.

Advantages: By comparing the parameter values,the ANFIS classification techniqueis more effective than the other and obtains the highest accuracy.

Disadvantages: Here the model is trained with a smallernumber of parameters.

Algorithms used: K-means clustering, Decision tree, ANFIS.

2) Prognosis of Liver Disease:Using Machine Learning Algorithms

Various parameters are studied from Liver FunctionTest (LFT) and they are analysedand used for the prediction of the liver disease. The data is transformed into scaled values so that it can fit with the minimum range. Algorithms like SVM, Logistic Regression, Decision Tree and Linear Discriminant analysis were also implemented and the highestaccuracy of 95.8% is obtained with Logistic Regression.

Advantages: Techniques like redundancy elimination, Integration helped in better and fast prediction

Disadvantages: Only a very few parameters were considered which is not

sufficient for accurate predictions.

Algorithms used: Decision Tree, LinearDiscriminant, SVM Fine Gaussian, LogisticRegression.

3) Evaluation based Approaches for Liver Disease Prediction using Machine Learning Algorithms

Initially after pre-processing the dataset, feature selection is performed which extracts the most sensitive and high impact feature from the dataset. The popular SVM classifieris used with maximised geometric margin and minimised error classification and accuracy of 75.04% is obtained. Also, Logistic Regression method is also used for prediction which acquired the accuracyof 73.23%.

Advantages: Feature selection extracts important feature that helps in efficient modeltraining.

Disadvantages: The accuracy of the model is low.

Algorithms used: Logistic Regression and SVM

4) Diagnosing for Liver Disease Prediction in Patients Using Combined Machine Learning Models

Three models such as KNN, Decision tree and Artificial Neural Network (ANN) were used for prediction. Also, a combined model is proposed by combining the above-mentioned models. Higher accuracyof about 96% is obtained by the combined model.

Advantages: The combined model makes the prediction easier and accurate.

Disadvantages: More number of parameters can be considered for prediction.

Algorithms used: KNN, Decision Tree, Artificial Neural Network(ANN)

4) Performance Analysis of Machine Learning Algorithms for Prediction of Liver Disease

Most common algorithms like Random Forest, Boosting technique like XGBoost, Logistic Regression, Gaussian Naïve Bayes, KNN, Support Vector Machine, GradientBoosting, CatBoost, AdaBoost, LightGBM, and Decision Tree were used for training and testing of the datasetand results of the model were considered.

Highest accuracyof 88% is obtained by Random ForestClassifier.

Advantages: The model predicted the result with very a smaller number of time and performance of different algorithms were analysed.

Disadvantages: Though different models were considered; the accuracy is low.

Algorithms used: Random Forest, Boosting technique like XGBoost, Logistic Regression, Gaussian Naïve Bayes, KNN, Support Vector Machine, Gradient Boosting,CatBoost, AdaBoost, LightGBM,and Decision Tree.

5) Prediction of Liver Disease using Gradient Boost Machine Learning Techniques with Feature Scaling

A feature reduction technique is offered in this study that uses recursive feature elimination and machine learning boosting methods. With fewer characteristics, Logistic regression and Multi-Layer Perceptron achieved higher prediction accuracy when basic machine learning models were applied to the dataset.On the dataset, boosting algorithms like CatBoost, LGBM Classifier, XGBoost, and Gradient Boost were used. The effectsof feature reductionon the gradient

boosting machinelearningmethods were examined.

Advantages:

Logistic Regression and Multi-Layer Perceptron performed efficiently comparedto theother basic machinelearning models. Recursive FeatureElimination technique effectively optimized the prediction accuracy of the Gradient Boosting algorithms to 94%.

Disadvantages:

The prediction accuracy of the fundamental models has not been increased by feature selection.

6) Diagnosis of LiverDisease using MachineLearning Models

In order to forecast liver illness more accurately, precisely, and consistently, the approaches of Support Vector Machines (SVM),Decision Trees (DT), and RandomForests (RF) are proposedin this study.

Advantage: Better accuracy precision and reliability

Disadvantage: The model couldn't diagnosethe presence of liver disease in people with subtle symptoms

7) Prediction of Liver Disorders using Machine Learning Algorithms: A Comparative Study

In this study, we examine four different machine learning (ML) techniques for classifying the Indian Liver Patient Dataset, including Logistic Regression, Decision Tree, Random Forest,and Extra Trees (ILPD).To eliminateirrelevant features from the

dataset, feature selection based on Pearson Correlation Coefficient is used.

Advantages:

In the data preprocessing step, to overcomethe issue of imbalanced class distribution ,an oversampling technique is used

Disadvantages:

The obtained results are comparatively inadequate. Robust scaling worked for a specific range only

8) Prediction of LiverDisease using Classification Algorithms

Support vector machines,K-Nearest Neighbor, and logistic regression are the methods utilised for this task. This classification technique is compared usingthe accuracy score and confusion matrix.

Advantages:

Comparing various types of algorithm has been done based on classification accuracy by using confusion Matrix.

Disadvantages:

Even though KNN having high accuracy, we can use only logisticregression for predicting liver disease as it has the highest sensitivity.

9) Computer-aided decision-making for predicting liver disease using PSO-based optimizedSVM with feature selection

Using an extraction, loading,transformation, and analysis(ELTA) approach topredict liver disease, the goal of this study is to choose relevant features for accurate diagnosis. As a result, the ELTA technique is used to analyse various data mining

models, including random forest, Multi-Layer Perceptron (MLP) neural networks, Bayesian networks, Support Vector Machine (SVM), and Particle Swarm Optimization (PSO)-SVM.

Advantages:

The proposed model demonstrated better performance in terms of accuracy, f-measure, precision, sensitivity, specificity, AUC, and FPR criteria.

Disadvantages:

Deep neural networks based on the dataset for liver disease and other datasets can be used to increase accuracy of detectionby choosing more useful and accurate featuresin the use of smart algorithms in the diagnosis and predictionof diseases, in particular liver disease.

REFERENCES:

1. R. Kalaiselvi, K. Meena and V. Vanitha, "Liver Disease Prediction Using Machine Learning Algorithms," 2021 International Conference on Advancements in Electrical, Electronics, Communication, Computing and Automation (ICAECA), 2021, pp. 1-6, doi: 10.1109/ICAECA52838.2021.9675756.
2. V. J. Gogi and V. M.N., "Prognosis of Liver Disease: Using Machine Learning Algorithms," 2018 International Conference on Recent Innovations in Electrical, Electronics& Communication Engineering (ICRIEECE), 2018, pp. 875-879, doi: 10.1109/ICRIEECE44171.2018.9008482.
3. C. Geetha and A. Arunachalam, "Evaluation based Approaches for Liver Disease Prediction using MachineLearning Algorithms," 2021 International Conference on ComputerCommunication and Informatics (ICCCI), 2021, pp. 1-4, doi: 10.1109/ICCCI50826.2021.9402463.
4. C.Anuradha, D. Swapna,B. Thati, V. N. Sree and S. P. Praveen,"Diagnosing for

LiverDisease Prediction in Patients Using Combined Machine Learning Models," 2022 4th International Conference on Smart Systems and Inventive Technology (ICSSIT), 2022, pp.889-896, doi: 10.1109/ICSSIT53264.2022.9716312.

5. V.Singh, M. K. Gourisaria and H. Das, "Performance Analysisof Machine LearningAlgorithms for Prediction of Liver Disease," 2021 IEEE 4th International Conference onComputing, Power and Communication Technologies (GUCON), 2021, pp. 1-7, doi: 10.1109/GUCON50781.2021.9573803.

6. G. Shobana and K. Umamaheswari, "Prediction of Liver Disease using Gradient Boost Machine Learning Techniques with Feature Scaling," 2021 5th International Conference onComputing Methodologies and Communication (ICCMC), 2021, pp. 1223-1229, doi: 10.1109/ICCMC51019.2021.9418333.

7. A.Sivasangari, B. J. Krishna Reddy,A. Kiran and P. Ajitha,"Diagnosis of LiverDisease using Machine Learning Models," 2020 Fourth International Conference on I-SMAC (IoT inSocial,Mobile, Analytics and Cloud) (I-SMAC), 2020, pp. 627-630, doi: 10.1109/I- SMAC49090.2020.9243375.

8. M. F. Rabbi, S. M. Mahedy Hasan, A. I. Champa, M. AsifZaman and M. K. Hasan, "Prediction of Liver Disorders using Machine Learning Algorithms: A Comparative Study," *2020 2nd International Conference on Advanced Information and CommunicationTechnology (ICAICT)*, 2020, pp. 111-116,doi: 10.1109/ICAICT51780.2020.9333528.

9. K. Thirunavukkarasu, A. S. Singh, M. Irfan and A. Chowdhury, "Prediction of Liver Disease using Classification Algorithms," 2018 4th International Conference on ComputingCommunication and Automation (ICCCA),2018, pp. 1-3, doi: 10.1109/CCAA.2018.8777655.

10. Hassannataj Joloudari, Javad & Saadatfar, Hamid & Dehzangi, Iman (Abdollah) & Shamshirband, Shahaboddin. (2019).Computer aided decision-making for predicting liverdisease using PSO-based optimized SVM with feature selection. Informatics in Medicine Unlocked. 17. 100255. 10.1016/j.imu.2019.100255.

3. IDEATION & PROPOSED SOLUTION

a. Empathy Map Canvas

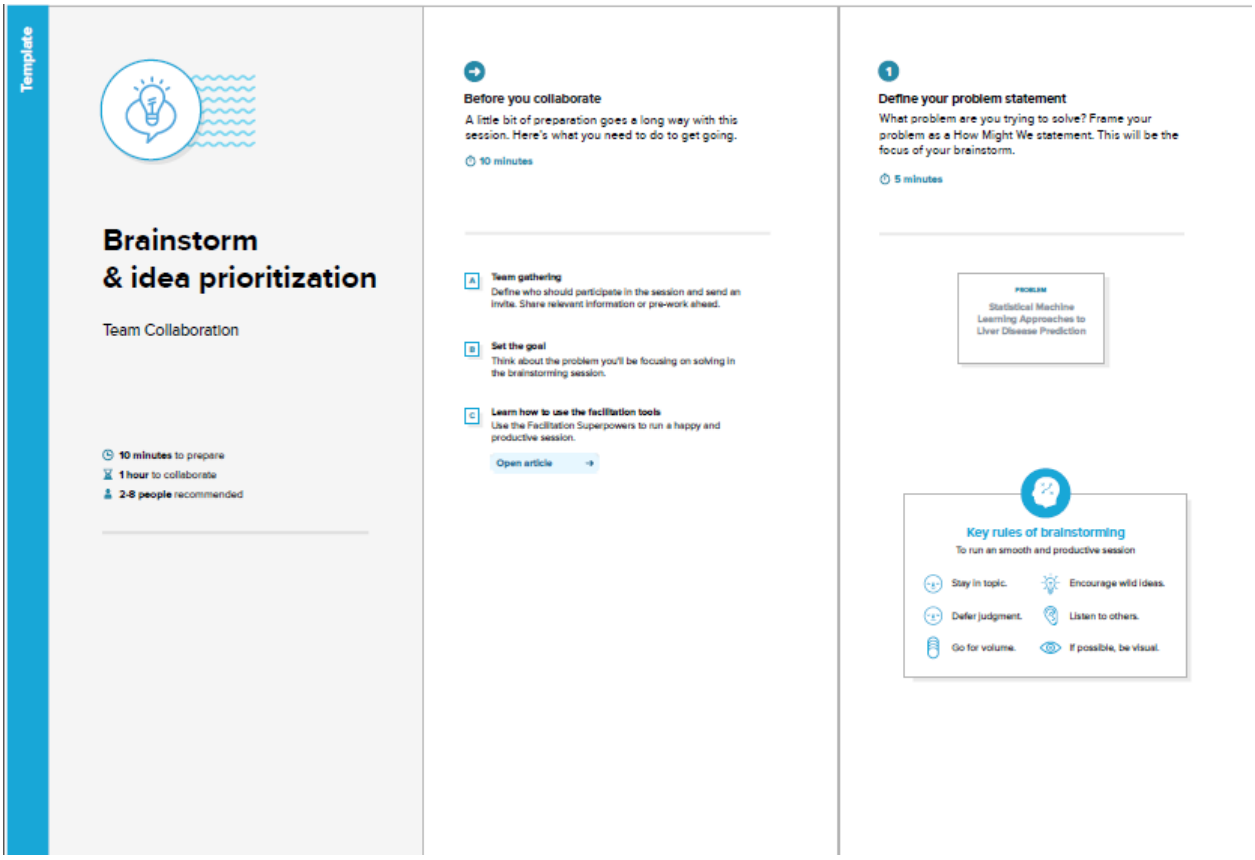
Liver Disease Prediction

Prediction of liver diseases at earlier stages can help in the early onset of treatment. An application that assists in the same can enable patients to test for themselves in case of doubt and eliminates the need for immediate need of a doctor or cost.

1

Build empathy and keep your focus on the user by putting yourself in their shoes.

b. Ideation & Brainstorming



2

Brainstorm

Write down any ideas that come to mind that address your problem statement.

10 minutes

TIP

You can select a sticky note and hit the pencil [switch to sketch] icon to start drawing!

Thanuja Varshini R

- Enhanced UI
- Instant Diagnosis
- Using Feature Selection
- Reduce Cost
- Easy access
- Privacy Protection
- Predict emergency admission risk
- Analyse intensity
- Travel expense reduction

Vijayashree S

- Advanced ML
- Reliability
- Identify major contributors
- Cost reduction
- No prerequisite knowledge
- User friendly
- Accurate results
- Privacy
- Fast diagnosis

Ezhil Mukhi S

- Survival rate
- Access easily
- UI improvement
- Cost effective
- Privacy
- Less complex
- Trustworthy
- Speed
- Accurate

Pranusha Bavan S

- Faster prediction
- ML techniques
- Intensity focused
- Availability
- Risk Alert
- User friendly
- Reduce expense
- Liver State

Kavipriya C

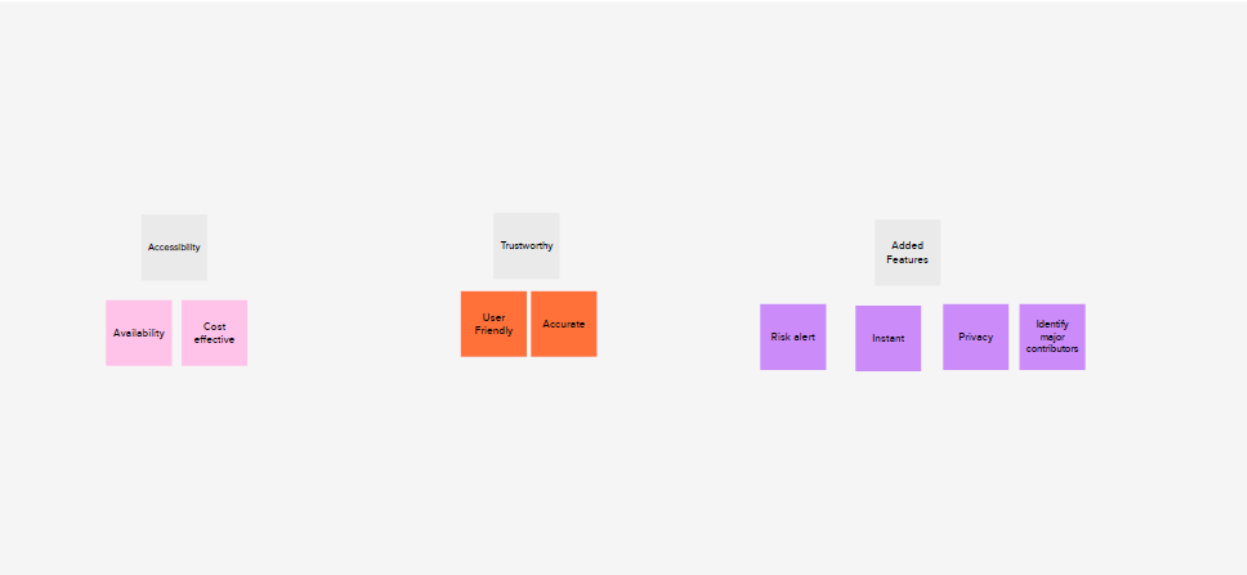
- Reduce diagnosis time
- Instant
- Less complex
- Privacy
- Risk specific
- User friendly
- Reduce expense
- Availability
- Accurate

3

Group ideas

Take turns sharing your ideas while clustering similar or related notes as you go. In the last 10 minutes, give each cluster a sentence-like label. If a cluster is bigger than six sticky notes, try and see if you and break it up into smaller sub-groups.

20 minutes



4

Prioritize

Your team should all be on the same page about what's important moving forward. Place your ideas on this grid to determine which ideas are important and which are feasible.

20 minutes

c. Proposed Solution

Overview:

With a growing trend of sedentary life which promotes lack of physical activities, diseases related to the liver have become a common encounter nowadays. Liver diseases have caused millions of deaths every year. There are about 100 different types of liver infections. Liver diseases are not easily discovered in an early stage as even after being affected and undergoing partial damage, it will be functioning normally. Liver failures are at a high rate of risk among Indians. India is expected to become the World Capital for Liver Diseases by 2025. The widespread occurrence of liver diseases is contributed due to deskbound lifestyle, increased alcohol consumption and smoking.

With such complications, it is necessary to have a concern towards tackling these liver-based diseases. An early diagnosis of liver diseases will definitely increase patients' survival rate. Afterall, we cannot expect a developed and prosperous nation, with an unhealthy population.

In this project, we aim to examine data from liver patients concentrating on relationships between a list of liver enzymes, proteins, age, and gender, using them to try and predict the likelihood of the occurrence of a liver disease. The main objective of this project is to analyse the parameters of various classification algorithms and compare their predictive accuracies to find the best classifier for determining liver disease.

Novelty

In this project to find the best classifier for determining the presence of a liver disease, we need to analyse the parameters of various classification algorithms and compare their predictive accuracies.

In spite of using the well-known classification algorithms of Machine Learning alone, we would also be fine tuning the models by continuously tuning the parameters of the machine learning models and ensuring that the right model and parameter values are chosen, and use these fine-tuned models along with the well-known classification algorithms compare their predictive accuracies to find the best classifier for determining the presence of a liver disease.

Feasibility

In this project, we will be implementing the prediction part by training the dataset using fine-tuned as well as the well-known classification algorithms of Machine Learning. Then after comparing their predictive accuracies, the best classifier for determining the presence of a liver disease is chosen. This best trained model is integrated to a flask based web application enabling the user to predict the disease by entering parameters in the web application.

This project proves it's feasibility as it is capable of achieving the following:

- Users can interact with the User Interface built using flask, to upload the input features
- Uploaded features/input can be analysed by the best trained model which is integrated
- After the model analyses the uploaded inputs, the prediction can be showcased on the User Interface

Business Model

This project would be one of the most useful for the doctors as well as the patients due to the following reasons:

- Liver disease, in the early stage, is hard to discover through traditional tests and

by the time in which the disease is diagnosed, the liver would get partially damaged. Our project would diagnose early, therefore protecting the liver from further damage and also protecting the patient's life.

- The time taken to perform traditional tests for liver disease diagnosis is quite a lot. On the other hand, the well-trained accurate model of ours can diagnose in no time thereby saving a lot of time.
- The traditional tests for liver disease diagnosis are very much expensive, making it infeasible for the patients who cannot afford them.
- Even after the traditional tests are performed, discovering the existence of a liver disease is very complex for the doctors. On the other hand, our model would be able to discover liver disease at ease.

Social Impact

In human beings, the liver is one of the most important parts of the body that performs many functions including the production of bile, excretion of bilirubin, metabolism of proteins and carbohydrates, activation of enzymes, storing vitamins, glycogen, and minerals etc. The liver mainly gets affected due to intake of alcohol. Intake of pain killer tablets and unusual food habits etc also contribute to liver damage. Liver diseases disturb the normal functioning of the liver.

Currently, liver related diseases are identified by analysing liver function blood test reports and scan reports. It takes more time to perform these tests and they are expensive as well. Discovering the existence of liver disease at a very early stage is a tedious task for the doctors.

Early prediction of liver disease using classification algorithms is a beneficial task that can help the doctors to diagnose the disease within a short period of time. This method is not just cost-effective but also saves a lot of time by predicting liver diseases even at early stages, preventing further liver damage.

Scalability of Solution

In this project, we will be building a model by applying various machine learning algorithms and comparing the models to find the best accurate model. And finally, integrate the best accurate model to a flask based web application. Hence, the users can predict the disease with ease by entering parameters which are well-known to the patient/user in the web application. The result is instantly displayed on the User Interface to the user.

d. Problem Solution fit

Project Title: Statistical Machine Learning Approaches to Liver Disease Prediction

Project Design Phase-I Solution Fit Template

Team ID: PNT2022TMID36043

Define CS, fit into CC

1. CUSTOMER SEGMENT(S) — CS

The target customer of our project is the patients who are suffering from liver disease.
Especially people with the age limit 40 - 60 suffers a lot.

6. CUSTOMER CONSTRAINTS — CC

As we are proposing an application to analyze and predict the the liver disease, people who live in area with low quality network connection becomes unable to access the application and make use of it.
Also people who has no digital/smart devices cannot make use of it.

5. AVAILABLE SOLUTIONS — AS

Many people has suggested many solutions to overcome the problem of predicting the liver disease earlier with higher accuracy. The suggested solutions are:
1. Logistic Regression
2. SVM
3. Decision Tree
4. Linear Regression
and so on..

Explore AS, differentiate

Focus on J&P, tap into

2. JOBS-TO-BE-DONE / PROBLEMS — J&P

The problems encountered while analyzing the solution is as follows:

1. first and the major one that is accuracy - how accurate our model predicts the output, because this ideation deals with human's life so it should be more accurate.

2. The second problem is the parameters consider for the prediction of output as different liver disease has different parameters to be considered.

3. The last problem is that the classification of liver disease as there are numerous different liver diseases.

9. PROBLEM ROOT CAUSE — RC

The root cause of the problem is the dataset because acquiring a proper dataset is a challenge.

Also there are many liver disease and different parameters need to be considered for different liver disease which makes the task difficult.

7. BEHAVIOUR — BE

The problems which were faced by the customer while using a mobile application is the response speed, and inaccessible account and so on.

These problems can be addressed under the queries or Q/A section and it will be addressed as soon as possible to ensure customer satisfaction.

Focus on J&P, tap int C

Identify strong TR & EM

3. TRIGGERS — TR

Email Marketing is the initial way to start triggering people to use our application.

Now-a-days , mobile application are the one which is easily reachable to the people in an efficient way.

4. EMOTIONS: BEFORE / AFTER — EM

Before: Sometimes people find it difficult to go to the hospitals especially the elderly people.

After: Like diabetes diagnosis one can check the health of their from their home itself.

10. YOUR SOLUTION — SL

The solution which we are proposing to overcome the existing problem is that:

1. Acquiring proper dataset for accurate predictions

2. KNN algorithm with fine tuning can be performed to get higher accuracy.

3. And SVM can be used to classify the type of liver disease.

This make the prediction accurate and meet up the customer expectations and overcome the limitations of the previous proposed solutions.

8. CHANNELS of BEHAVIOUR — CH

8.1 ONLINE

In today's life, people expect things that are easily acquirable and accessible. So a mobile application is the one that suits the requirements.

8.2 OFFLINE

The Offline activities that can be provided is that filling out a form in hospitals and predicting the results according to the input given by the users.

Extract online & offline CH of BE

4. REQUIREMENT ANALYSIS

a. Customer Journey Map

b. Functional requirements

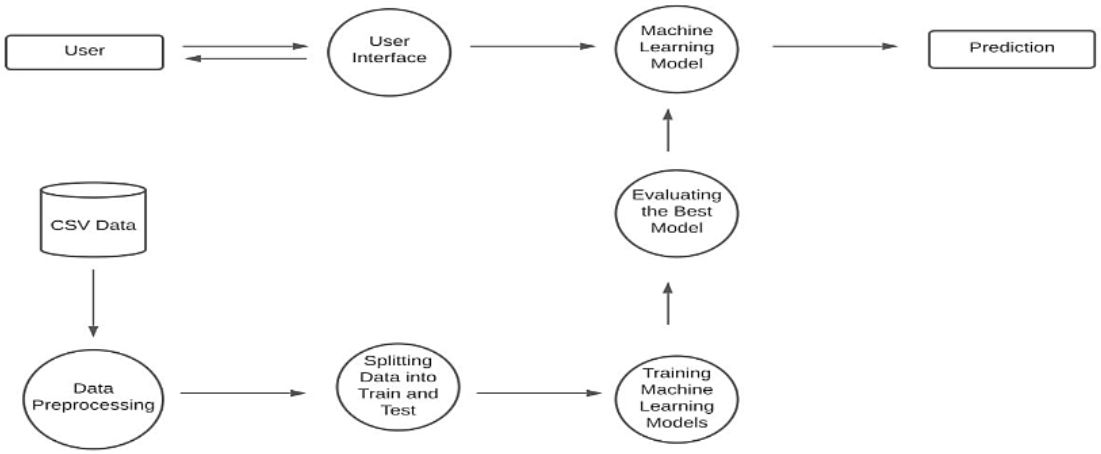
FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	User Registration	Registration through Form Registration through Gmail
FR-2	User Confirmation	Confirmation via Email Confirmation via Password
FR-3	User Input	Get necessary details for prediction
FR-4	Data Processing	Data Cleaning Data Scaling Augmentati on Feature selection
FR-5	Prediction	Predicting whether the user has Liver disease or not and its type

c. Non-Functional requirements

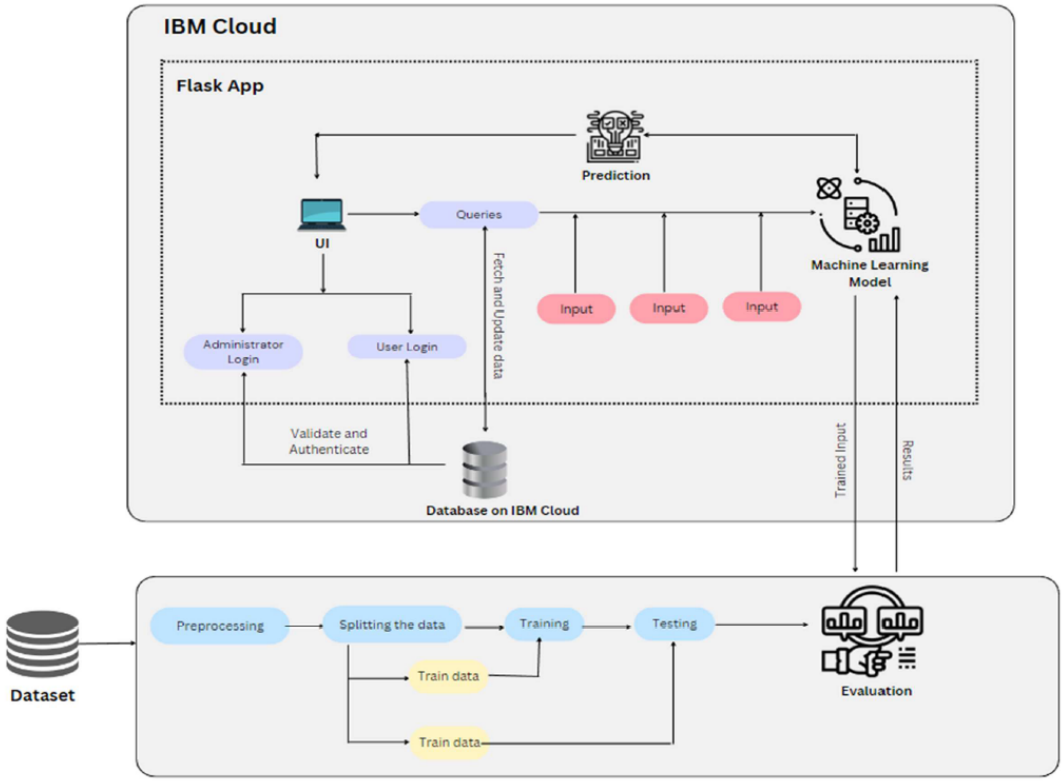
FR No.	Non-Functional Requirement	Description
NFR-1	**Usability**	To check whether the patient has liver disease or not
NFR-2	**Security**	Implement necessary methods to provide security to the user data
NFR-3	**Reliability**	Make ensure that the model's reliablility
NFR-4	**Performance**	Use efficient ML techniques for better accuracy
NFR-5	**Scalability**	Predicts various types of liver disease

5. PROJECT DESIGN

a. Data Flow Diagrams



b. Technology Stack



S.No	Component	Description	Technology
1.	User Interface	User interacts with the system through the developed Web Application	Flask
2.	Building Model	Pre-process the dataset, train the model using the train data and test the model with the test data and user input data as per performance metrics.	Python, Numpy, Scikit-learn, Tensorflow
3.	Fine tuning the model	Model is fine tuned to increase the accuracy of prediction	Optimizer, Tensorflow
4.	Navigation within Web UI	All the available features can be accessed from the dashboard.	Flask
5.	Cloud Database	Database Service on Cloud	IBM DB2
6.	File Storage	File storage requirements	IBM Block Storage
7.	External API	Login/Registration through Google Account	Google API
8.	Machine Learning Model	To detect Liver Disease using Machine Learning	Xception, VGG19
9.	Cloud Infrastructure	Cloud Server Configuration	Cloud Foundry

S.No	Characteristics	Description	Technology
1.	Open-Source Frameworks	Flask micro web framework	Python, Numpy, Tensorflow, Scikit-learn, IBM Watson, Google API, Flask
2.	Security Implementations	With all aspects of the job including detecting malicious attacks, analysing the network endpoint protection and vulnerability assessment, Sign-in Encryption	IBM Cloud App ID Services
3.	Availability	Available for all data size	IBM Cloud Services
4.	Performance	Can extend the storage according to our needs	Python, IBM Cloud

c. User Stories

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer (Mobile user)	Login	USN-1	As a user, I can login to 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
		USN-3	As a user, I can register for the application through Google account	I can register & access the dashboard with Google Login	Low	Sprint-2
	Dashboard	USN-4	As a user, I must enter my details	I can retrieve information according to entered details	Medium	Sprint-1
	Dashboard	USN-5	As a user, I can request to input test details	I can process the test input after approval	High	Sprint-1
Administrator	Services	USN-6	As an admin, I must validate the result	I can provide validity to result	Medium	Sprint-3
		USN-7	As an admin, I can add suggestions	I can add suggestions	Low	Sprint-3
	Data processing	USN-8	As an admin, I must collect input data for the medical database	Collected data is stored in the database	Medium	Sprint-2
Hospital Administrator	Login	USN-9	As an admin, I need to login with appropriate access levels	I can access admin level details/operations	Medium	Sprint-3
	Dashboard	USN-10	As an admin, I need to proceed with the test details with case head	I can proceed with processing	High	Sprint-1
Doctor / Radiologist	Diagnosis	USN-11	As a radiologist/doctor, I can view the diagnosis/ prediction results	I can view the diagnosis/ prediction	High	Sprint-2

6. PROJECT PLANNING & SCHEDULING

a. 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 to use the application by providing my email ID, password, and confirming password.	5	High	Thanuja Varshini R
Sprint-1		USN-2	As a user,I will receive confirmation registering to use theapplication	5	High	Ezhil Mukhi S
Sprint-1	Login	USN-3	As a user, I can loginto the application by entering registered email ID & password	10	High	Vijayashree S
Sprint-2	Input Necessary Details	USN-4	As a user,I can give input test details topredict the occurrence of Liver Disease.	15	High	Pranusha BavanS
Sprint-2	Data Pre-Processing	USN-5	Transform raw data into appropriate format forprediction.	5	High	Thanuja Varshini R
Sprint-3	Prediction of Liver Disease	USN-6	As a user,I can get the results of predictionof Liver Disease processed using Machine Learning algorithms.	15	High	Ezhil Mukhi S
Sprint-3		USN-7	As a user,I can get accurate results of presence of liver disease.	5	Medi um	Kavipriya C
Sprint-4	Review	USN-8	As an admin,I reinforce the result of prediction.	20	High	Vijayashree S

Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date(Planned)	Story Points Completed (as on Planned End Date)	Sprint Release Date (Actual)
Sprint1	20	6 Days	24 Oct 2022	29 Oct 2022	20	29 Oct 2022
Sprint2	20	6 Days	31 Oct 2022	05 Nov2022	20	05 Nov 2022
Sprint3	20	6 Days	07 Nov 2022	12 Nov2022	20	12 Nov2022
Sprint4	20	6 Days	14 Nov 2022	19 Nov2022	20	19 Nov2022

Velocity:

Sprint 1 Average Velocity:

Average Velocity= 20/6 = 3.3

Sprint 2 Average Velocity:

Average Velocity= 20/6 = 3.3

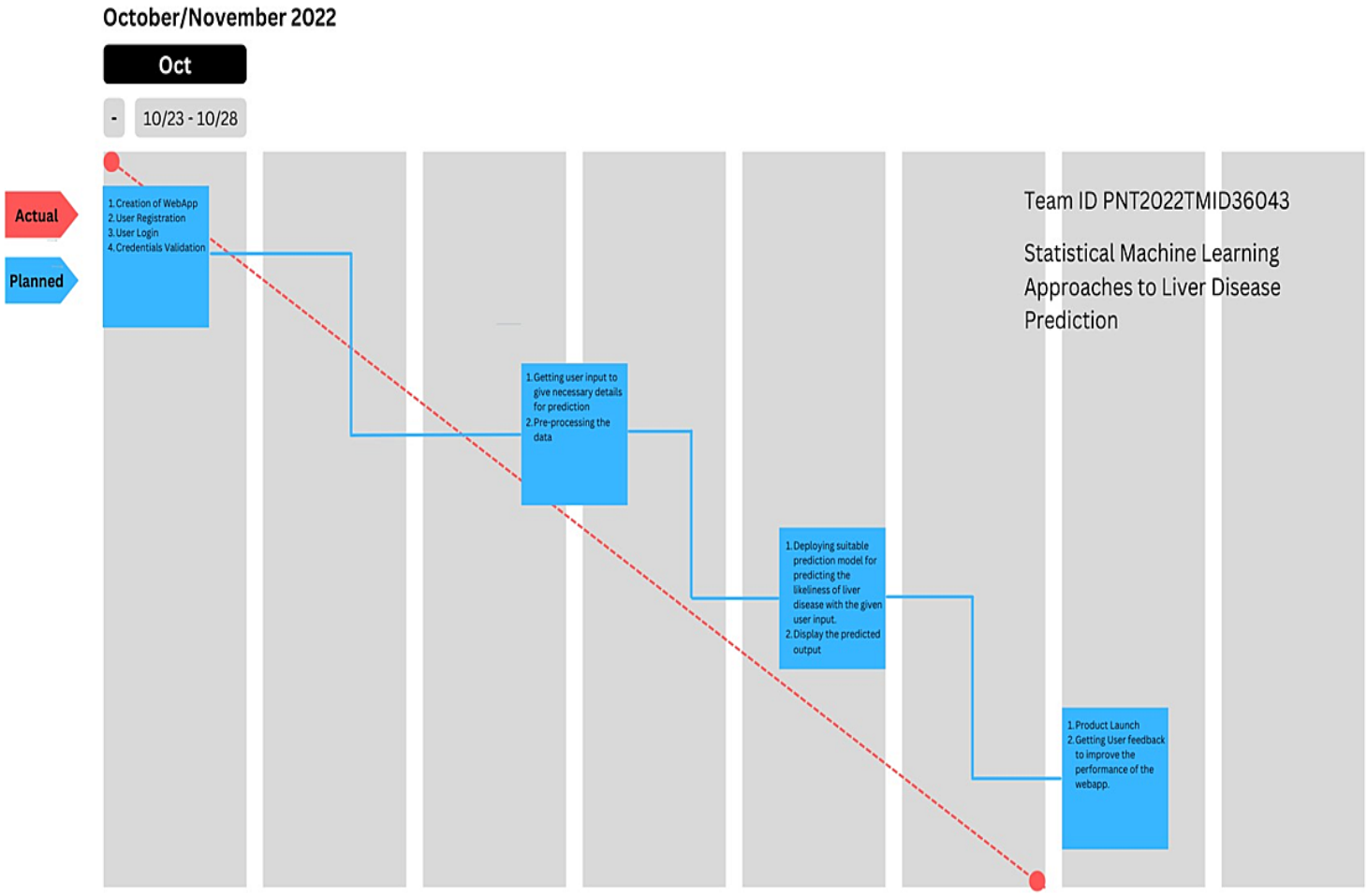
Sprint 3 Average Velocity:

Average Velocity= 20/6 = 3.3

Sprint 4 Average Velocity:

Average Velocity= 20/6 = 3.3

Burndown Chart :



b. Sprint Delivery Schedule

TITLE	DESCRIPTION	DATE
Literature Survey & Information Gathering	Conducting a literature review on the chosen project and acquiring data by reviewing technical papers, research publications, etc.	18 SEPTEMBER 2022
Prepare Empathy Map	An Empathy Map Canvas is prepared to record the user's gains and pains. A list of the problems is made.	17 SEPTEMBER 2022
Ideation	The ideas generated during the brainstorming session were discussed and the top three were ranked according to relevance and viability.	17 SEPTEMBER 2022
Proposed Solution	The innovation, viability of the concept, business model, social impact, scalability of the solution, etc. are all included in the proposed solution document.	17 SEPTEMBER 2022
Problem Solution Fit	The problem - solution fit document was prepared	17 SEPTEMBER 2022
Solution Architecture	The solution architecture document was prepared.	19 SEPTEMBER 2022

Customer Journey	Customer journey maps are created to comprehend how users engage with and use the application (entry to exit).	11 OCTOBER 2022
Functional Requirement	The functional requirement document is prepared.	21 OCTOBER 2022
Data FlowDiagrams	The data flow was drawn diagrams and submitted for review.	21 OCTOBER 2022
Technology Architecture	The technology architecture diagram was prepared.	21 OCTOBER 2022
Prepare Milestone & Activity List	The milestones & activity list of the project was prepared.	21 OCTOBER 2022
Project Development - Delivery of Sprint-1, 2, 3 &4	The code is to be developed & submitted.	Final sprint submitted on 19 NOVEMBER 2022

c. Reports from JIRA

JIRA has categorized reports in four levels, which are −

i. Agile

ii. Issue Analysis

iii. Forecast & Management

iv. Others

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

a. Machine Learning

	Age	Gender	Total_Bilirubin	Direct_Bilirubin	Alkaline_Phosphotase	Alamine_Aminotransferase	Aspartate_Aminotransferase	Total_Protiens	Albumin
0	65	Female	0.7	0.1	187	16	18	6.8	3.3
1	62	Male	10.9	5.5	699	64	100	7.5	3.2
2	62	Male	7.3	4.1	490	60	68	7.0	3.3
3	58	Male	1.0	0.4	182	14	20	6.8	3.4
4	72	Male	3.9	2.0	195	27	59	7.3	2.4

	Age	Total_Bilirubin	Direct_Bilirubin	Alkaline_Phosphotase	Alamine_Aminotransferase	Aspartate_Aminotransferase	Total_Protiens	Albumin	Al
count	583.000000	583.000000	583.000000	583.000000	583.000000	583.000000	583.000000	583.000000	
mean	44.746141	3.298799	1.486106	290.576329	80.713551	109.910806	6.483190	3.141852	
std	16.189833	6.209522	2.808498	242.937989	182.620356	288.918529	1.085451	0.795519	
min	4.000000	0.400000	0.100000	63.000000	10.000000	10.000000	2.700000	0.900000	
25%	33.000000	0.800000	0.200000	175.500000	23.000000	25.000000	5.800000	2.600000	
50%	45.000000	1.000000	0.300000	208.000000	35.000000	42.000000	6.600000	3.100000	
75%	58.000000	2.600000	1.300000	298.000000	60.500000	87.000000	7.200000	3.800000	
max	90.000000	75.000000	19.700000	2110.000000	2000.000000	4929.000000	9.600000	5.500000	

```
[ ] data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 583 entries, 0 to 582
Data columns (total 11 columns):
 #   Column                      Non-Null Count  Dtype
---  ------                      --------------  -----
 0   Age                         583 non-null    int64
 1   Gender                      583 non-null    object
 2   Total_Bilirubin             583 non-null    float64
 3   Direct_Bilirubin            583 non-null    float64
 4   Alkaline_Phosphotase        583 non-null    int64
 5   Alamine_Aminotransferase    583 non-null    int64
 6   Aspartate_Aminotransferase  583 non-null    int64
 7   Total_Protiens              583 non-null    float64
 8   Albumin                     583 non-null    float64
 9   Albumin_and_Globulin_Ratio  579 non-null    float64
 10  Dataset                     583 non-null    int64
dtypes: float64(5), int64(5), object(1)
memory usage: 50.2+ KB
```

```
[ ] data.shape
```

```
(583, 11)
```

Null values

```
#Checking for null values
data.isna().sum()
```

```
Age                         0
Gender                      0
Total_Bilirubin             0
Direct_Bilirubin            0
Alkaline_Phosphotase        0
Alamine_Aminotransferase    0
Aspartate_Aminotransferase  0
Total_Protiens              0
Albumin                     0
Albumin_and_Globulin_Ratio  4
Dataset                     0
dtype: int64
```

```
[ ] #Filling missing data with mean
data['Albumin_and_Globulin_Ratio']=data['Albumin_and_Globulin_Ratio'].fillna(data['Albumin_and_Globulin_Ratio'].mean())
```

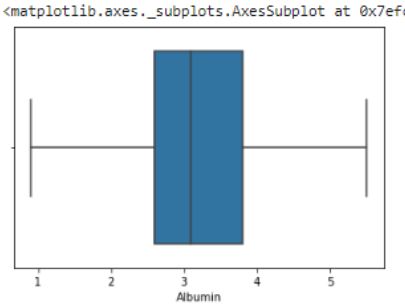
```
[ ] #Checking for null values
data.isna().sum()
```

```
Age                         0
Gender                      0
Total_Bilirubin             0
Direct_Bilirubin            0
Alkaline_Phosphotase        0
Alamine_Aminotransferase    0
Aspartate_Aminotransferase  0
Total_Protiens              0
Albumin                     0
Albumin_and_Globulin_Ratio  0
Dataset                     0
dtype: int64
```

Outliers

```
[ ] #Handling outliers
sns.boxplot(data['Albumin'])
```

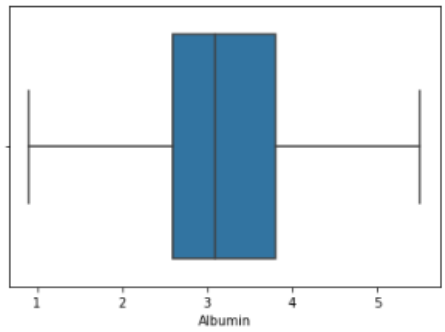
```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: Pass the following variable as a keyword arg: x. From versio
  FutureWarning
<matplotlib.axes._subplots.AxesSubplot at 0x7efc88a9bf50>
```




```
[ ] Q1 = data['Albumin'].quantile(0.25)
Q3 = data['Albumin'].quantile(0.75)
IQR = Q3 - Q1
whisker_width = 1.5
lower_whisker = Q1 -(whisker_width*IQR)
upper_whisker = Q3 +(whisker_width*IQR)
data['Albumin']=np.where(data['Albumin']>upper_whisker,upper_whisker,np.where(data['Albumin']<lower_whisker,lower_whisker,data['Albumin']))
```

```
sns.boxplot(data['Albumin'])
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarni
  FutureWarning
<matplotlib.axes._subplots.AxesSubplot at 0x7efc88547850>
```




```
[ ] #Handling categorical data
    numeric_data = data.select_dtypes(include=[np.number])
    categorical_data = data.select_dtypes(exclude=[np.number])
    print("Number of numerical variables: ", numeric_data.shape[1])
    print("Number of categorical variables: ", categorical_data.shape[1])
```

```
Number of numerical variables:  10
Number of categorical variables:  1
```

```
[ ] print("Number of categorical variables: ", categorical_data.shape[1])
    Categorical_variables = list(categorical_data.columns)
    Categorical_variables
```

```
Number of categorical variables:  1
['Gender']
```

```
[ ] data['Gender'].value_counts()
```

```
Male      441
Female    142
Name: Gender, dtype: int64
```

```
[ ] #Encoding 'Gender' Column
    from sklearn.preprocessing import LabelEncoder
    le = LabelEncoder()
    label = le.fit_transform(data['Gender'])
    data["Gender"] = label
```

```
[ ] data['Gender'].value_counts()
```

```
1    441
0    142
Name: Gender, dtype: int64
```

```
numeric_data = data.select_dtypes(include=[np.number])
categorical_data = data.select_dtypes(exclude=[np.number])
print("Number of numerical variables: ", numeric_data.shape[1])
print("Number of categorical variables: ", categorical_data.shape[1])
```

```
Number of numerical variables:  11
Number of categorical variables:  0
```

```
[ ] # Replacing infinite with nan
    data.replace([np.inf, -np.inf], np.nan, inplace=True)
    # Dropping all the rows with nan values
    data.dropna(inplace=True)
```

```
[ ] #Defining X and Y for independent and dependent variables
    Feature = data[['Age','Total_Bilirubin',  'Direct_Bilirubin', 'Alkaline_Phosphotase', 'Alamine_Aminotransferase', 'Aspartate_Aminotransferase', 'Total_Protiens', 'Albumin',  'Albumin_and_Globulin_Ratio']]
    X = Feature
    Y = data['Dataset']
```

```
[ ] X[:5]
```

	Age	Total_Bilirubin	Direct_Bilirubin	Alkaline_Phosphotase	Alamine_Aminotransferase	Aspartate_Aminotransferase	Total_Protiens	Albumin	Albumin_and_Globulin_Ratio
0	65	0.7	0.1	187	16	18	6.8	3.3	0.90
1	62	10.9	5.5	699	64	100	7.5	3.2	0.74
2	62	7.3	4.1	490	60	68	7.0	3.3	0.89
3	58	1.0	0.4	182	14	20	6.8	3.4	1.00
4	72	3.9	2.0	195	27	59	7.3	2.4	0.40

```
[ ] Y[:5]
```

```
0    1
1    1
2    1
3    1
4    1
Name: Dataset, dtype: int64
```

```
#Feature Scaling
from sklearn.preprocessing import StandardScaler
object= StandardScaler()
scale = object.fit_transform(X)
print(scale)
```

```
[[ 1.25209764 -0.41887783 -0.49396398 ...  0.29211961  0.19896867
  -0.14789798]
 [ 1.06663704  1.22517135  1.43042334 ...  0.93756634  0.07315659
  -0.65069686]
 [ 1.06663704  0.6449187   0.93150811 ...  0.47653296  0.19896867
  -0.17932291]
 ...
 [ 0.44843504 -0.4027597  -0.45832717 ... -0.0767071   0.07315659
   0.16635131]
 [-0.84978917 -0.32216906 -0.35141677 ...  0.29211961  0.32478075
   0.16635131]
 [-0.41704777 -0.37052344 -0.42269037 ...  0.75315299  1.58290153
   1.73759779]]
```

```
[ ] X_scaled  = pd.DataFrame(scale, columns = X.columns)
    X_scaled
```

	Age	Total_Bilirubin	Direct_Bilirubin	Alkaline_Phosphotase	Alamine_Aminotransferase	Aspartate_Aminotransferase	Total_Protiens	Albumin	Albumin_and_Globulin_Ratio
0	1.252098	-0.418878	-0.493964	-0.426715	-0.354665	-0.318393	0.292120	0.198969	-0.147898
1	1.066637	1.225171	1.430423	1.682629	-0.091599	-0.034333	0.937566	0.073157	-0.650697
2	1.066637	0.644919	0.931508	0.821588	-0.113522	-0.145186	0.476533	0.198969	-0.179323
3	0.819356	-0.370523	-0.387054	-0.447314	-0.365626	-0.311465	0.292120	0.324781	0.166351
4	1.684839	0.096902	0.183135	-0.393756	-0.294379	-0.176363	0.753153	-0.933340	-1.719144
...	...	...	...	...	...	...	...	...	...
578	0.942997	-0.451114	-0.493964	0.862786	-0.332743	-0.262967	-0.537740	-1.939837	-1.813419
579	-0.293407	-0.434996	-0.493964	-0.793378	-0.250535	-0.273359	-0.445534	0.073157	0.480601
580	0.448435	-0.402760	-0.458327	-0.187766	-0.179288	-0.211005	-0.076707	0.073157	0.166351
581	-0.849789	-0.322169	-0.351417	-0.439074	-0.283418	-0.269895	0.292120	0.324781	0.166351
582	-0.417048	-0.370523	-0.422690	-0.307240	-0.327263	-0.297608	0.753153	1.582902	1.737598

583 rows × 9 columns

```
[ ] #Splitting the dataset
    from sklearn.model_selection import train_test_split
    X_train, X_test, Y_train, Y_test = train_test_split(X_scaled, Y, test_size=0.20, random_state=0)
```

```
[ ] X_train.shape
```

```
(466, 9)
```

```
[ ] X_test.shape
```

```
(117, 9)
```

```
[ ] Y_train.shape
```

```
(466,)
```

```
[ ] Y_test.shape
```

```
(117,)
```

K-Nearest neighbours

```
[ ] from sklearn.neighbors import KNeighborsClassifier as KNN
    knn= KNN()
    knn.fit(X_train, Y_train)
```

```
KNeighborsClassifier()
```

```
[ ] knn.get_params()
```

```
{'algorithm': 'auto',
 'leaf_size': 30,
 'metric': 'minkowski',
 'metric_params': None,
 'n_jobs': None,
 'n_neighbors': 5,
 'p': 2,
 'weights': 'uniform'}
```

```
from sklearn.model_selection import GridSearchCV
n_neighbors = [x for x in range(5, 86, 2)]
algorithm = ['auto', 'ball_tree', 'kd_tree', 'brute']
weights = ['uniform', 'distance']

grid = {'n_neighbors': n_neighbors,
        'algorithm': algorithm,
        'weights': weights}
```

```
[ ] new_model = KNN()
    knn_grid = GridSearchCV(estimator = new_model, param_grid = grid, cv = 8, verbose=0)
    knn_grid.fit(X_train, Y_train)
```

```
GridSearchCV(cv=8, estimator=KNeighborsClassifier(),
             param_grid={'algorithm': ['auto', 'ball_tree', 'kd_tree', 'brute'],
                         'n_neighbors': [5, 7, 9, 11, 13, 15, 17, 19, 21, 23,
                                         25, 27, 29, 31, 33, 35, 37, 39, 41, 43,
                                         45, 47, 49, 51, 53, 55, 57, 59, 61, 63, ...],
                         'weights': ['uniform', 'distance']})
```

```
[ ] knn_grid.best_params_
```

```
{'algorithm': 'auto', 'n_neighbors': 63, 'weights': 'distance'}
```

```
Y_pred = knn_grid.best_estimator_.predict(X_test)

pred_df = pd.DataFrame({'Actual': Y_test, 'Predicted': Y_pred})
pred_df.head()
```

	Actual	Predicted
246	1	1
92	1	1
386	2	1
186	1	1
389	1	1

```
[ ] from sklearn import metrics
    # Measure the Accuracy Score
    print("Accuracy score of the predictions: {value:.2f} %".format(value=metrics.accuracy_score(Y_pred, Y_test)*100))
```

```
Accuracy score of the predictions: 66.67 %
```

Polynomial regression

```
[ ] from sklearn.linear_model import LinearRegression
    lin = LinearRegression()
    lin.fit(X, Y)
```

```
LinearRegression()
```

```
[ ] from sklearn.preprocessing import PolynomialFeatures
    poly = PolynomialFeatures(degree = 4)
    X_poly = poly.fit_transform(X)
    poly.fit(X_poly, Y)
    lin2 = LinearRegression()
    lin2.fit(X_poly, Y)
```

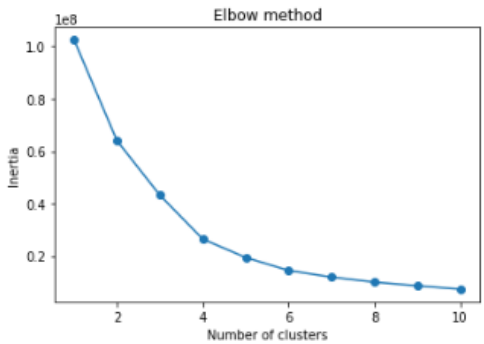
```
LinearRegression()
```

```
y_pred = lin.predict(X_test)
pred_df = pd.DataFrame({'Actual': Y_test, 'Predicted': y_pred})
pred_df.head()
```

	Actual	Predicted
246	1	1.585750
92	1	1.607519
386	2	1.725271
186	1	1.615975
389	1	1.672439

K-Means Clustering

```
from sklearn.cluster import KMeans
inertias = []

for i in range(1,11):
    kmeans = KMeans(n_clusters=i)
    kmeans.fit(X)
    inertias.append(kmeans.inertia_)

plt.plot(range(1,11), inertias, marker='o')
plt.title('Elbow method')
plt.xlabel('Number of clusters')
plt.ylabel('Inertia')
plt.show()
```




```
kmeans = KMeans(n_clusters=2)
kmeans.fit(X)
pred_df = pd.DataFrame({'Actual': Y_test, 'Predicted': y_pred})
pred_df.head()
```

	Actual	Predicted
246	1	1.585750
92	1	1.607519
386	2	1.725271
186	1	1.615975
389	1	1.672439

Logistic Regression

```
[ ] from sklearn.linear_model import LogisticRegression
    logreg = LogisticRegression(random_state = 0)
    logreg.fit(X_train, Y_train)

    LogisticRegression(random_state=0)
```

```
[ ] from sklearn import metrics
    # Measure the Accuracy Score
    print("Accuracy score of the predictions: {value:.2f} %".format(value=metrics.accuracy_score(Y_pred, Y_test)*100))

    Accuracy score of the predictions: 66.67 %
```

Naive Bayes

```
[ ] from sklearn.naive_bayes import GaussianNB
    gnb = GaussianNB()
    gnb.fit(X_train, Y_train)
```

```
GaussianNB()
```

```
[ ] y_pred = gnb.predict(X_test)
    y_pred
```

```
array([1, 1, 2, 1, 2, 2, 1, 2, 1, 2, 2, 2, 2, 2, 2, 2, 1, 2, 2, 2, 1, 2,
       2, 2, 1, 2, 1, 2, 2, 2, 2, 1, 2, 1, 1, 1, 1, 1, 1, 2, 1, 2, 2, 2,
       1, 1, 2, 1, 2, 2, 2, 2, 1, 2, 2, 2, 2, 2, 2, 2, 2, 1, 2, 2, 2, 2,
       2, 2, 2, 2, 2, 2, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 1, 1, 2, 2, 2, 2,
       1, 2, 1, 2, 1, 1, 2, 2, 2, 2, 2, 1, 2, 2, 2, 2, 1, 1, 2, 2, 2, 2,
       2, 2, 2, 2, 2, 2, 2])
```

```
[ ] from sklearn.metrics import accuracy_score
    print('Model accuracy score: {0:0.4f}'. format(accuracy_score(Y_test, y_pred)))
```

```
Model accuracy score: 0.6068
```

Random Forest Classifier

```
from sklearn.ensemble import RandomForestClassifier
rfc = RandomForestClassifier(random_state=0)
rfc.fit(X_train, Y_train)
```

```
RandomForestClassifier(random_state=0)
```

```
[ ] y_pred = rfc.predict(X_test)
```

```
[ ] from sklearn.metrics import accuracy_score
    print('Model accuracy score with 10 decision-trees : {0:0.4f}'. format(accuracy_score(Y_test, y_pred)))
```

```
Model accuracy score with 10 decision-trees : 0.6752
```

```
[ ] import pickle
    filename = '/content/drive/MyDrive/finalized_model_ibm.pkl'
    pickle.dump(rfc, open(filename, 'wb'))
```

Decision tree Classifier

```
[ ] from sklearn.tree import DecisionTreeClassifier
    clf_gini = DecisionTreeClassifier(criterion='gini', max_depth=3, random_state=0)
    clf_gini.fit(X_train, Y_train)
```

```
DecisionTreeClassifier(max_depth=3, random_state=0)
```

```
[ ] y_pred_gini = clf_gini.predict(X_test)
```

```
[ ] from sklearn.metrics import accuracy_score
    print('Model accuracy score with criterion gini index: {0:0.4f}'. format(accuracy_score(Y_test, y_pred_gini)))
```

```
Model accuracy score with criterion gini index: 0.6667
```

```
[ ] from google.colab import drive
    drive.mount('/content/drive')
```

b. Application

flaskapp.py

```
from flask import Flask, render_template,request
import requests
import joblib
import pandas as pd

# NOTE: you must manually set API_KEY below using information retrieved from
your IBM Cloud account.
API_KEY = "frcKpzAHOm195t5SOcXp_nuH1EUqjG3q4iUpZONkxSP9"
token_response = requests.post('https://iam.cloud.ibm.com/identity/token',
data={"apikey":
 API_KEY, "grant_type": 'urn:ibm:params:oauth:grant-type:apikey'})
print(token_response.json())
mltoken = token_response.json()["access_token"]

header = {'Content-Type': 'application/json', 'Authorization': 'Bearer ' + mltoken}



RanFor = joblib.load(open("finalized_model_ibm.pkl", 'rb'))

app = Flask(_name_)

@app.route('/')
def home():
    return render_template('inputform.html')

@app.route('/', methods=["POST"])
def upload():
    if request.method == "POST":
        name = request.form.get("name")
        emailid = request.form.get("emailid")
        age = int(request.form.get("age"))
        total_bilirubin = float(request.form.get("total_bilirubin"))
        direct_bilirubin = float(request.form.get("direct_bilirubin"))
```

```
    alkaline_phosphate = int(request.form.get("alkaline_phosphate"))
    alamine_aminotransferase = int(request.form.get("alamine_aminotransferase"))
    aspartate_aminotransferase =
int(request.form.get("aspartate_aminotransferase"))
    total_proteins = float(request.form.get("total_proteins"))
    albumin = float(request.form.get("albumin"))
    albumin_and_globulin_ratio =
float(request.form.get("albumin_and_globulin_ratio"))

    X = ['Age', 'Total_Bilirubin', 'Direct_Bilirubin', 'Alkaline_Phosphotase',
'Alamine_Aminotransferase',
         'Aspartate_Aminotransferase', 'Total_Protiens', 'Albumin',
'Albumin_and_Globulin_Ratio']

    index_dict = dict(zip(X, range(len(X))))
    vect={}
    vect1 =
[int(total_bilirubin),int(direct_bilirubin),int(alkaline_phosphate),int(alamine_aminotra
nsferase),int(aspartate_aminotransferase),int(total_proteins),int(albumin),int(albumi
n_and_globulin_ratio)]
    for key, val in index_dict.items():
        vect[key] = 0
    vect['Age'] = age
    vect['Total_Bilirubin'] = total_bilirubin
    vect['Direct_Bilirubin'] = direct_bilirubin
    vect['Alkaline_Phosphotase'] = alkaline_phosphate
    vect['Alamine_Aminotransferase'] = alamine_aminotransferase
    vect['Aspartate_Aminotransferase'] = aspartate_aminotransferase
    vect['Total_Protiens'] = total_proteins
    vect['Albumin'] = albumin
    vect['Albumin_and_Globulin_Ratio'] = albumin_and_globulin_ratio
    df = pd.DataFrame.from_records(vect, index=[0])
    crop_yield = RanFor.predict(df)[0]
    if(str(crop_yield) == '1'):
        msg = "Status: Liver Disease Positive"
    else:
        msg = "Status: Liver Disease Negative"
```

```
    payload_scoring = {"input_data": [{"fields": [X], "values": [vect1]}]}

    response_scoring = requests.post('https://us-
south.ml.cloud.ibm.com/ml/v4/deployments/21e926ef-d143-4738-978f-
25d6f5bc8021/predictions?version=2022-11-18', json=payload_scoring,
     headers={'Authorization': 'Bearer ' + mltoken})
    print("Scoring response")
    print(response_scoring.json())
    return render_template('inputform.html', msg=msg)



if _name_ == '_main_':
  app.run(debug=True)
```

inputform.html

```
<script src="https://cdn.freecodecamp.org/testable-projects-
fcc/v1/bundle.js"></script>
<html>
<body>
  <!-----------------------------------Form-heading--------------------------------->
  <h1 id="title">Survey Form</h1>
  <!----------------------------------Form-section-------------------------------->
  <div id="main-form">
    <p id="description">Details for Liver Disease Prediction</p>
    <form id="survey-form" method="post" action="/" enctype="multipart/form-
data">
      <div class="rowTab">
        <div class="labels">
          <label id="name-label" for="name">* Name: </label>
        </div>
        <div class="rightTab">
          <input autofocus type="text" name="name" id="name" class="input-
field" placeholder="Enter your name" required>
        </div>
      </div>
```

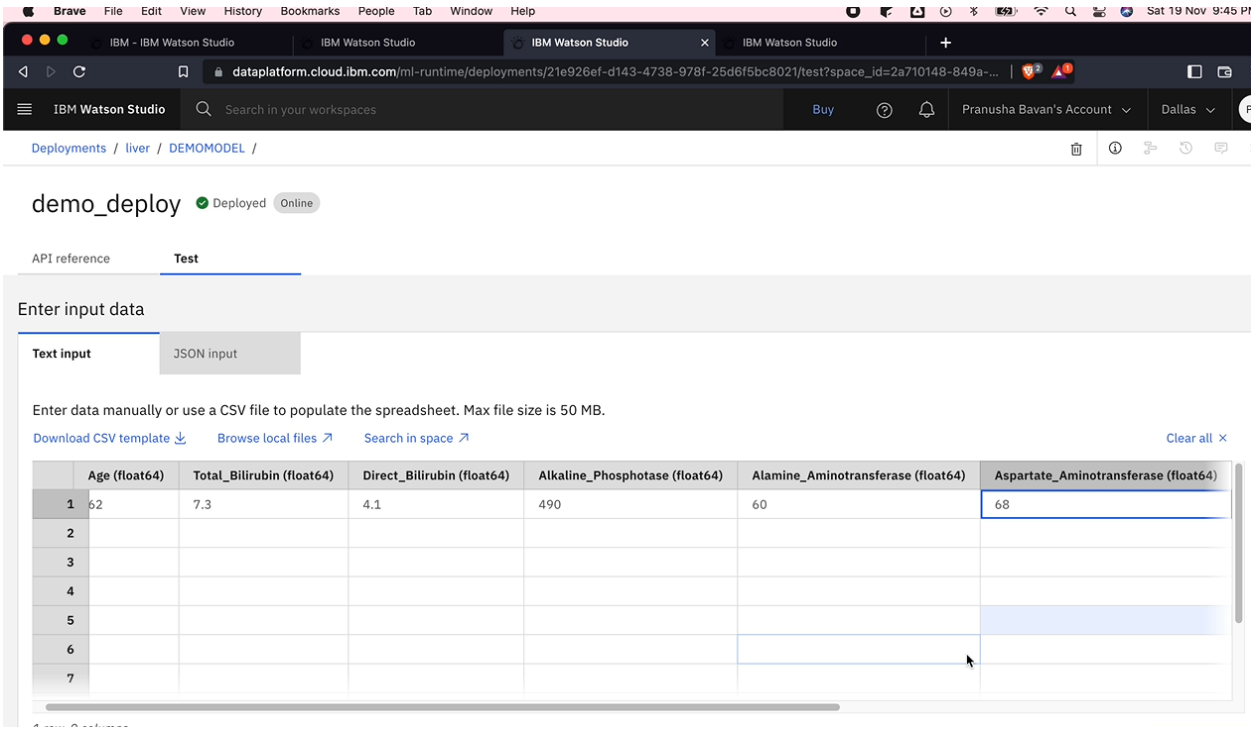
```
        <div class="rowTab">
            <div class="labels">
                <label id="email-label" for="name">* Email: </label>
            </div>
            <div class="rightTab">
                <input type="email" name="emailid" id="emailid" class="input-field"
required placeholder="Enter your Email">
            </div>
        </div>

        <div class="rowTab">
            <div class="labels">
                <label id="number-label1" for="Age">* Age: </label>
            </div>
            <div class="rightTab">
                <input type="number" name="age" id="age" min="1" max="120"
class="input-field" placeholder="Age">
            </div>
        </div>
        <div class="rowTab">
            <div class="labels">
                <label id="number-label2" for="Total Bilirubin">* Total Bilirubin: </label>
            </div>
            <div class="rightTab">
                <input type="number" name="total_bilirubin" id="total_bilirubin"
min="0" max="125" class="input-field" placeholder="Total Bilirubin" step="any">
            </div>
        </div>
        <div class="rowTab">
            <div class="labels">
                <label id="number-label3" for="Direct Bilirubin">* Direct Bilirubin:
</label>
            </div>
            <div class="rightTab">
                <input type="number" name="direct_bilirubin" id="direct_bilirubin"
min="0" max="125" class="input-field" placeholder="Direct Bilirubin" step="any">
```

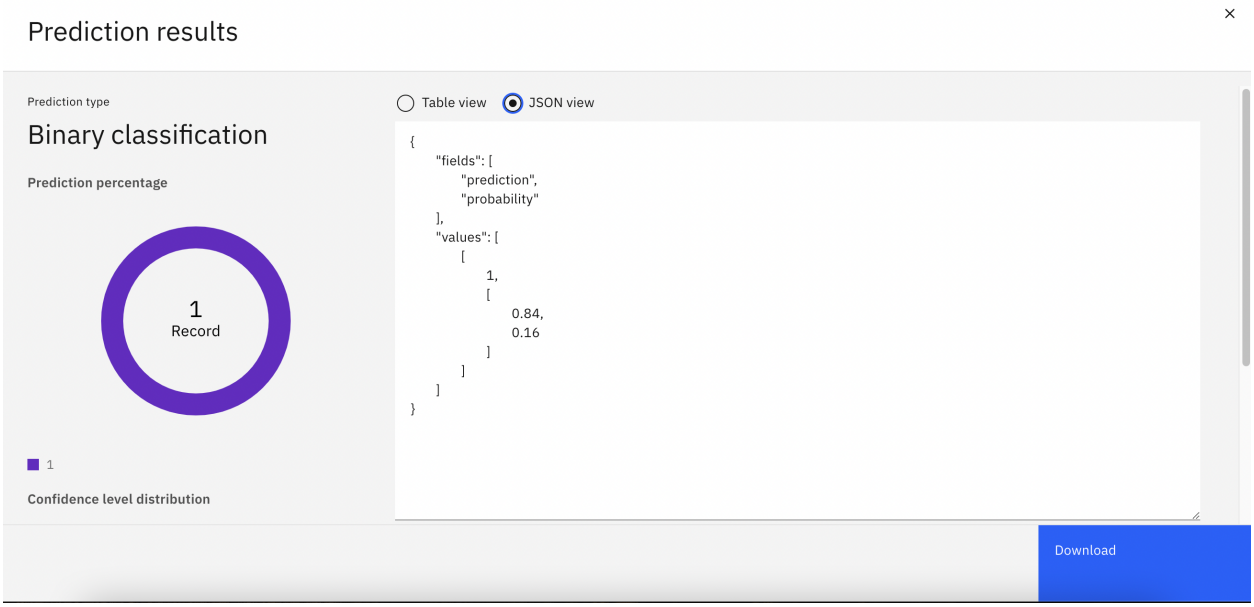
```
            </div>
        </div>
        <div class="rowTab">
            <div class="labels">
                <label id="number-label4" for="Alkaline Phosphate">* Alkaline
Phosphate: </label>
            </div>
            <div class="rightTab">
                <input type="number" name="alkaline_phosphate"
id="alkaline_phosphate" min="0" max="3000" class="input-field"
placeholder="Alkaline Phosphate">
            </div>
        </div>
        <div class="rowTab">
            <div class="labels">
                <label id="number-label5" for="Alamine Aminotransferase">* Alamine
Aminotransferase: </label>
            </div>
            <div class="rightTab">
                <input type="number" name="alamine_aminotransferase"
id="alamine_aminotransferase" min="0" max="3000" class="input-field"
placeholder="Alamine Aminotransferase">
            </div>
        </div>
        <div class="rowTab">
            <div class="labels">
                <label id="number-label6" for="Aspartate Aminotransferase">*
Aspartate Aminotransferase: </label>
            </div>
            <div class="rightTab">
                <input type="number" name="aspartate_aminotransferase"
id="aspartate_aminotransferase" min="0" max="5000" class="input-field"
placeholder="Aspartate Aminotransferase">
            </div>
        </div>
        <div class="rowTab">
            <div class="labels">
```

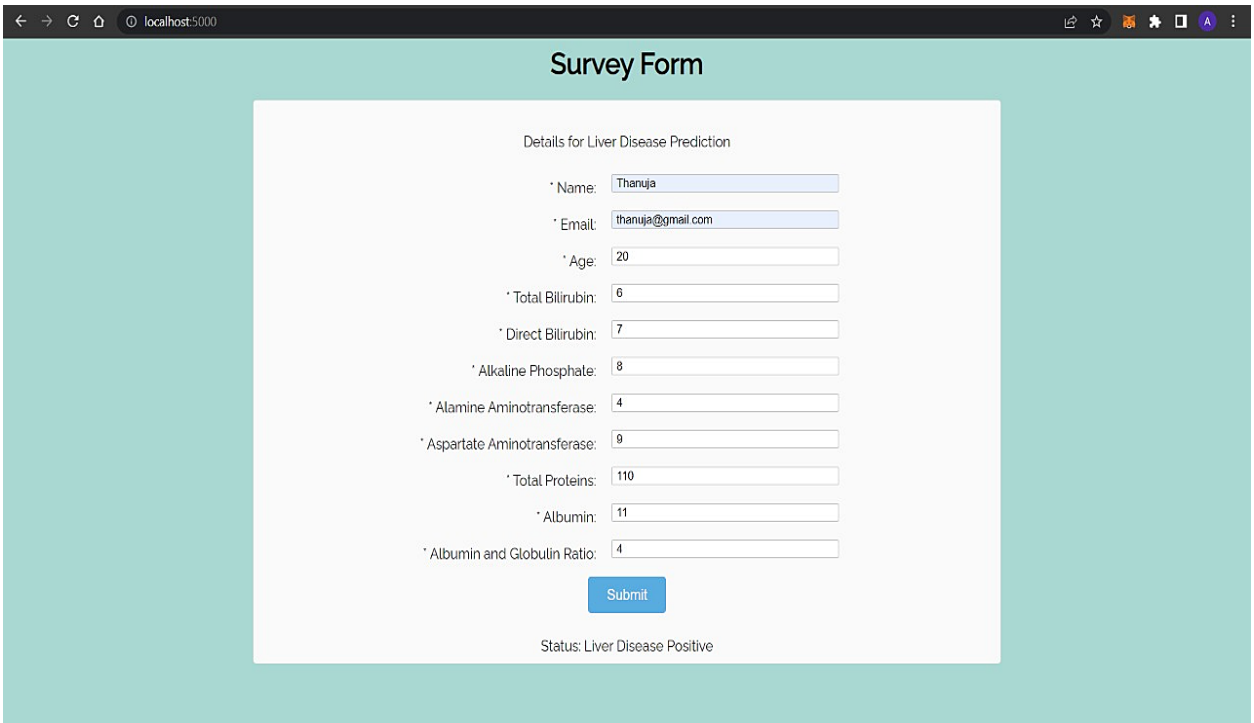
```
                <label id="number-label7" for="Total Proteins">* Total Proteins: </label>
            </div>
            <div class="rightTab">
                <input type="number" name="total_proteins" id="total_proteins"
min="0" max="125" class="input-field" placeholder="Total Proteins" step="any">
            </div>
        </div>
        <div class="rowTab">
            <div class="labels">
                <label id="number-label8" for="Albumin">* Albumin: </label>
            </div>
            <div class="rightTab">
                <input type="number" name="albumin" id="albumin" min="0"
max="125" class="input-field" placeholder="Albumin" step="any">
            </div>
        </div>
        <div class="rowTab">
            <div class="labels">
                <label id="number-label9" for="Albumin and Globulin Ratio">* Albumin
and Globulin Ratio: </label>
            </div>
            <div class="rightTab">
                <input type="number" name="albumin_and_globulin_ratio"
id="albumin_and_globulin_ratio" min="0" max="125" class="input-field"
placeholder="Albumin and Globulin Ratio" step="any">
            </div>
        </div>
        <button id="submit" type="submit">Submit</button>
    </form>
        <div id="prediction" >
                        {{ msg }}
        </div>
    </div>
    <script href="/index.js"></script>
</body>

</html>
```

8. TESTING

a. Test Cases







b. User Acceptance Testing

Defect Analysis

This report shows the number of resolved or closed bugs at each severity level, and how they were resolved

Resolution	Severity 1	Severity 2	Severity 3	Severity 4	Subtotal
By Design	12	2	2	2	18
Duplicate	1	1	3	0	5
External	2	3	0	1	6
Fixed	8	4	1	20	33
Not Reproduced	0	0	1	2	3
Skipped	1	0	1	1	3
Won't Fix	0	7	2	1	10
Totals	24	17	10	27	78

Test Case Analysis

This report shows the number of test cases that have passed, failed,and untested

Section	Total Cases	Not Tested	Fail	Pass
Print Engine	8	0	0	8
Client Application	45	0	0	45
Security	2	0	0	2
Outsource Shipping	4	0	0	4
Exception Reporting	8	0	0	8
Final Report Output	5	0	0	5
Version Control	1	0	0	1

9. RESULTS

a. Performance Metrics

RANDOM FOREST

Random Forest Classifier

```
from sklearn.ensemble import RandomForestClassifier
rfc = RandomForestClassifier(random_state=0)
rfc.fit(X_train, Y_train)
```

```
RandomForestClassifier(random_state=0)
```

```
[ ] y_pred = rfc.predict(X_test)
```

```
[ ] from sklearn.metrics import accuracy_score
    print('Model accuracy score with 10 decision-trees : {0:0.4f}'. format(accuracy_score(Y_test, y_pred)))
```

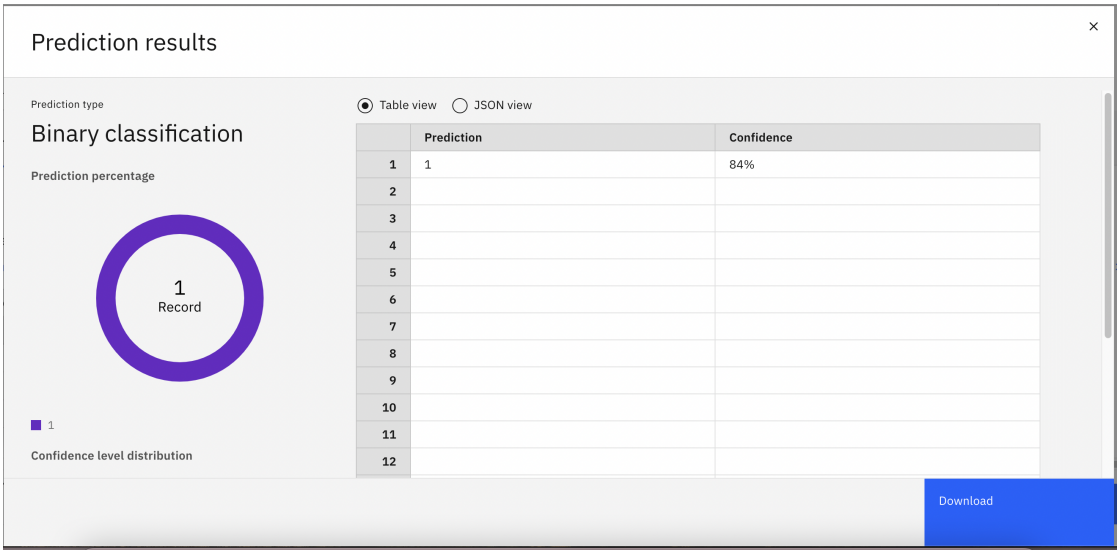
```
Model accuracy score with 10 decision-trees : 0.6752
```

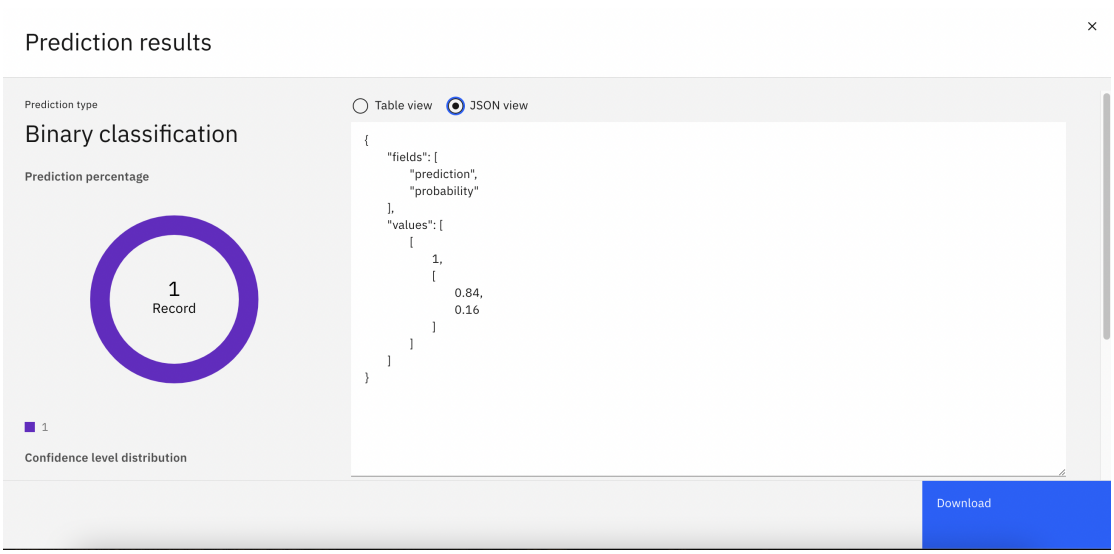
```
[ ] import pickle
    filename = '/content/drive/MyDrive/finalized_model_ibm.pkl'
    pickle.dump(rfc, open(filename, 'wb'))
```

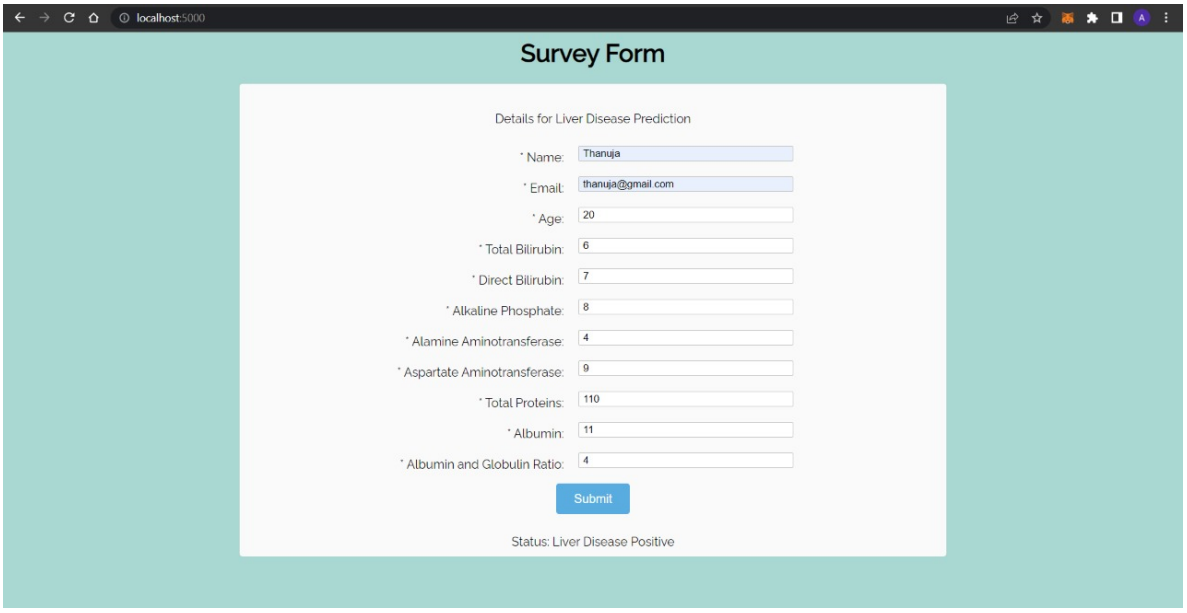
```
In [193]: deployment = wml_client.deployments.create(
              artifact_uid=model_id,
              meta_props=deployment_props
          )
```

```
#######################################################################################

Synchronous deployment creation for uid: '75015591-899f-4af9-9856-819f7e961798' started

#######################################################################################


initializing
Note: online_url is deprecated and will be removed in a future release. Use serving_urls instead.

ready


------------------------------------------------------------------------------------------------
Successfully finished deployment creation, deployment_uid='21e926ef-d143-4738-978f-25d6f5bc8021'
------------------------------------------------------------------------------------------------
```

b. Application Output







10. ADVANTAGES & DISADVANTAGES

Advantages :

- This helps in early diagnosis and prevention of severity of liver disease.
- This helps patients who are not in a condition to directly consult a doctor.

Disadvantages:

- Inaccuracies could have been avoided.
- This isn't a feasible solution for people who have no access to the Internet or gadgets.

11. CONCLUSION

Random Forest Classification model was the best suited model for this application based on the training and testing accuracies identified with this project. The application that enables the user to upload the necessary details for prediction displays if the person is suspected to have liver disease or not whih has been hosted on IBM Cloud. The application will enable users to predict the possibility of liver disease and thereby, prevent or inhibit its severity.

12. FUTURE SCOPE

The model accuracy can be further improved. The user application can be incorporated with a database to upload the testing query details and results to be appended to it. The User Interface can be enhanced further for better user experience. Suggestions for clinics/hospitals or treatment procedures can be displayed accordingly.

13. APPENDIX

GitHub Link - https://github.com/IBM-EPBL/IBM-Project-1940-1658420948

Project Demo Link -
https://drive.google.com/file/d/1pIu6JOv_eeOJhas2D3YU9mS1cZS5Z1sY/view?usp=share_link