Statistical Machine Learning Approaches to Liver Disease Prediction

Team ID: PNT2022TMID12919

Bachelor of Engineering

Electronics and Communication Engineering

PSG College Of Technology

Coimbatore – 641004

Faculty Evaluator: Dr. P. Kalpana

Faculty Mentor : Dr. C. Ramya

Team Members :-

19L203 - Adhithya S

19L210 - Divya P

19L256 – Sudharshan N

19L261 – Umabharathi P

1 : INTRODUCTION

1.1: Project Overview

Liver diseases avert the normal function of the liver. Mainly due to the large amount of alcohol consumption liver disease arises. 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. Discovering the existence of liver disease at an early stage is a complex task for the doctors. 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.

1.2 : Purpose

Detecting and identifying liver diseases calls for expensive and invasive tests. Therefore, by employing machine learning techniques to detect liver diseases, we can minimise the need for such tests and thus enable more widespread access to early detection of liver diseases. This can allow the liver condition from worsening before it progresses to a point where it becomes very difficult to treat.

2 : LITERATURE SURVEY

2.1 : Existing Problem

The Existing solution for has been studied. It has been that inferred that, the possibility of a liver disease is predicted approximately manually or using some basic software. The result obtained is often unreliable and the margin of error is very high owing to the multiple variables affecting the output. There are a lot of different ways to identify the presence of a liver disease. The output performance may vary significantly even between two people with similar symptoms. A generalized, while also customizable solution to accurately predict liver diseases while taking into consideration all the various parameters is required.

2.2 : References

S.NO	Paper Title	Author	Journal Name	Publication year	Description
1	A Comparative Study on Liver Disease Prediction Using Supervised Machine Learning Algorithms	A.K.M Sazzadur Rahman, F. M. Javed Mehedi Shamrat, Zarrin Tasnim, Joy Roy, Syed Akhter Hossain	ResearchGate	2019	Six machine learning techniques have been applied including Logistic Regression, K Nearest Neighbors, Decision Tree, Support Vector Machine, Naïve Bayes, and Random Forest. The performance was evaluated on different measurement techniques such as accuracy, precision, recall, f-1 score, and specificity and the result was that LR achieved the highest accuracy.
2	Machine learning-based liver disease diagnosis: A systematic review	Rayyan AzamKhan, Yigang Luo, Fang Xiang Wu	ScienceDirect	2022	This paper mainly focuses on the computer-aided diagnosis of hepatic lesions in view of diffuse- and focal liver disorders. This is based on three image acquisition modalities: ultrasonography, computed tomography, and magnetic resonance imaging.

3	Diagnosing of Liver Disease Prediction in Patients using combined Machine Learning Models	Chokka Anuradha, D Swapna, Balamuralikrishnan Thati	IEEE		This paper aims to represent a Diagnosing for Liver disease prediction in Patients using Combined Machine Learning Models. Optimized three machine learning algorithms are used for the accurate diagnosis of liver disease and they are Artificial Neural Networks (ANN), Decision Trees, and K-Nearest Neighbors (KNN).
4	Statistical Machine Learning Approaches to Liver Disease Prediction	Fahad Mostafa, Easin Hasan, Morgan Williamson, Hafiz Khan	MDPI	2021	ML methods were able to identify the liver disease with high accuracy. The PCA results showed five important factors for liver disease diagnosis: AST, ALT, GGT, BIL, and ALP.
5	.Liver Disease Prediction System using Machine Learning Techniques	Rakshith D B, Mrigank Srivastava, Ashwani Kumar, Gururaj S P	IJERT	2021	In this paper risk of liver disease for a person is predicted based on the blood test report results of the user. With the dataset used for this project, 100 % accuracy is obtained for SVM model.

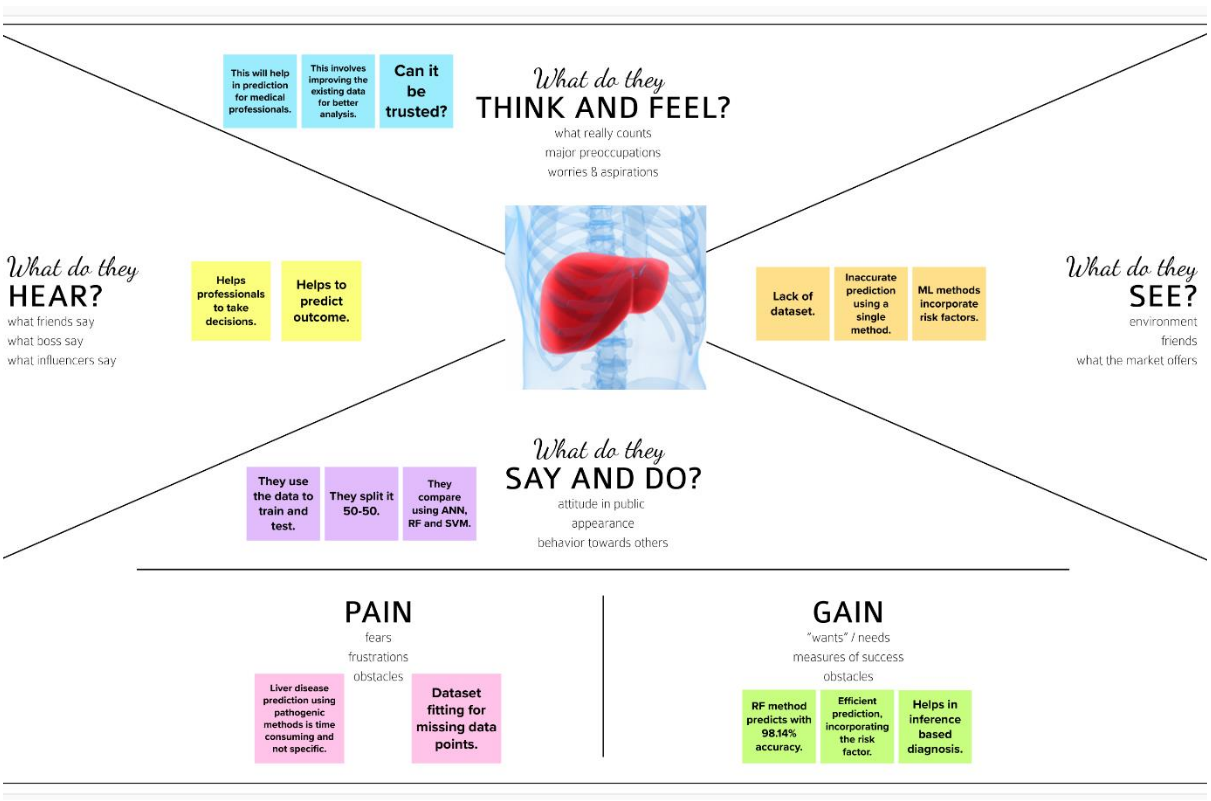
6	Machine Learning Approaches for Liver Disease Diagnosing	Bilal Khan, Rashid Naseem, Mumtaz Ali, Muhammad Arshad, Nazir Jan	International Journal of Data Science and Advanced Analytics	2019	This study proposes a new model based on CHIRP methods for the early finding of liver disease. This examination centre around MAE, RAE, and Accuracy assessment measurements for the benchmarking of the proposed model with other existing models. The exploratory outcomes show a better consequence of applying CHIRP assessing on MAE and RAE while utilizing the Accuracy of the exhibition of RF and MLP is seldom productive than CHIRP.
7	Statistical Machine Learning Approaches to Liver Disease Prediction	Robin Biju	International Journal of Scientific Research and Engineering Development	2022	This study attempts to find an appropriate machine learning algorithm that can determine whether a person has liver disease or not given a dataset containing biological and diagnostic data of 583 Indian patients.

2.3 : Problem Statement Definition

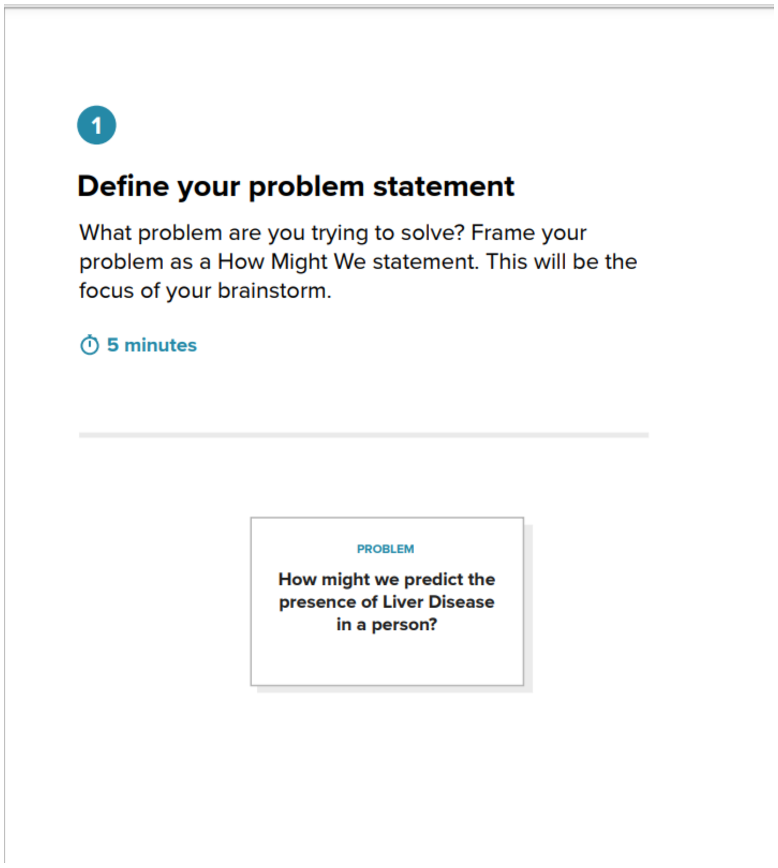
To avoid the expensive and invasive tests, the application of statistical machine learning techniques to CMP results for the extraction of information for a clinician might be helpful for diagnosis. Exploratory data analysis methods are extremely important in healthcare; they can predict patterns across data sets to facilitate the determination of risk or diagnostic factors for disease with more speed and accuracy. The use of these methods can allow for earlier detection and potentially prevent many cases of liver disease from progressing to the point of needing biopsy or complex treatment.

3 : IDEATION AND PROPOSED SOLUTION

3.1 : Empathy Map Canvas


This will help in prediction for medical professionals.
This involves improving the existing data for better analysis.
Can it be trusted?
What do they THINK AND FEEL?
what really counts
major preoccupations
worries & aspirations
What do they HEAR?
what friends say
what boss say
what influencers say
Helps professionals to take decisions.
Helps to predict outcome.
Lack of dataset.
Inaccurate prediction using a single method.
ML methods incorporate risk factors.
What do they SEE?
environment
friends
what the market offers
They use the data to train and test.
They split it 50-50.
They compare using ANN, RF and SVM.
What do they SAY AND DO?
attitude in public
appearance
behavior towards others
PAIN
fears
frustrations
obstacles
Liver disease prediction using pathogenic methods is time consuming and not specific.
Dataset fitting for missing data points.
GAIN
"wants" / needs
measures of success
obstacles
RF method predicts with 98.14% accuracy.
Efficient prediction, incorporating the risk factor.
Helps in inference based diagnosis.

3.2 : Ideation And Brainstorming



1

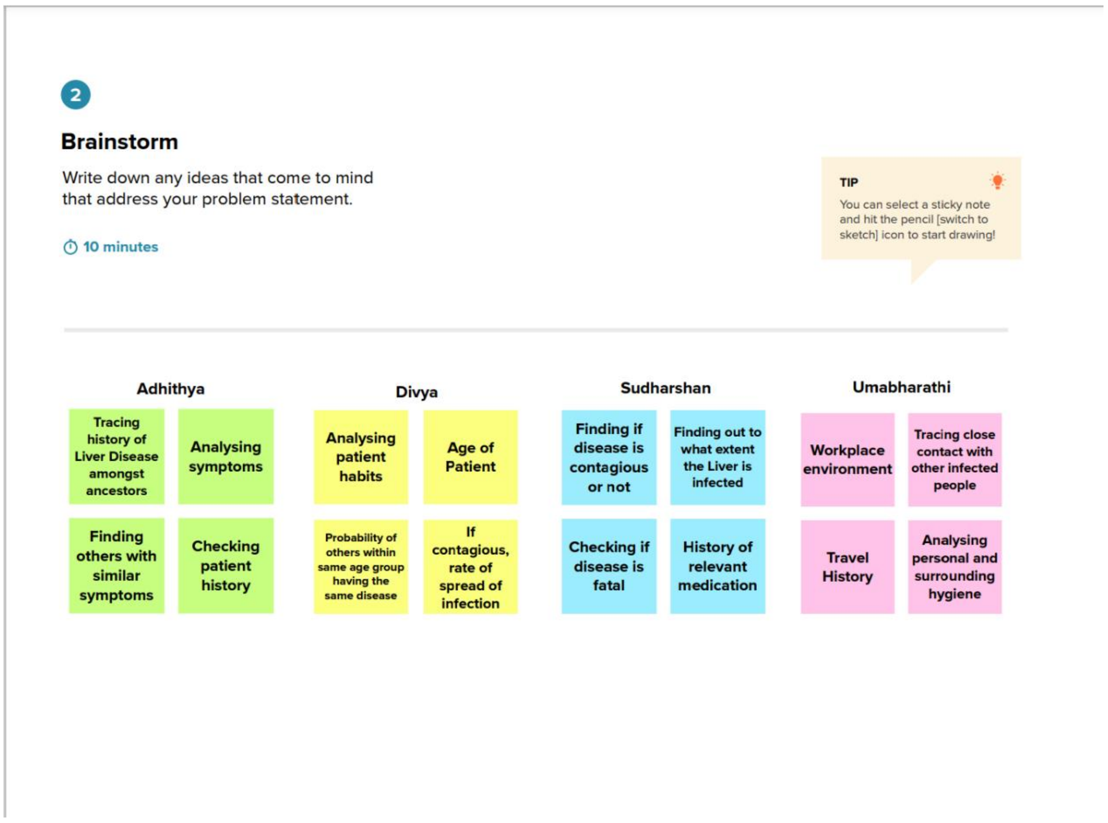
Define your problem statement

What problem are you trying to solve? Frame your problem as a How Might We statement. This will be the focus of your brainstorm.

5 minutes

PROBLEM

How might we predict the presence of Liver Disease in a person?



2

Brainstorm

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

10 minutes

TIP

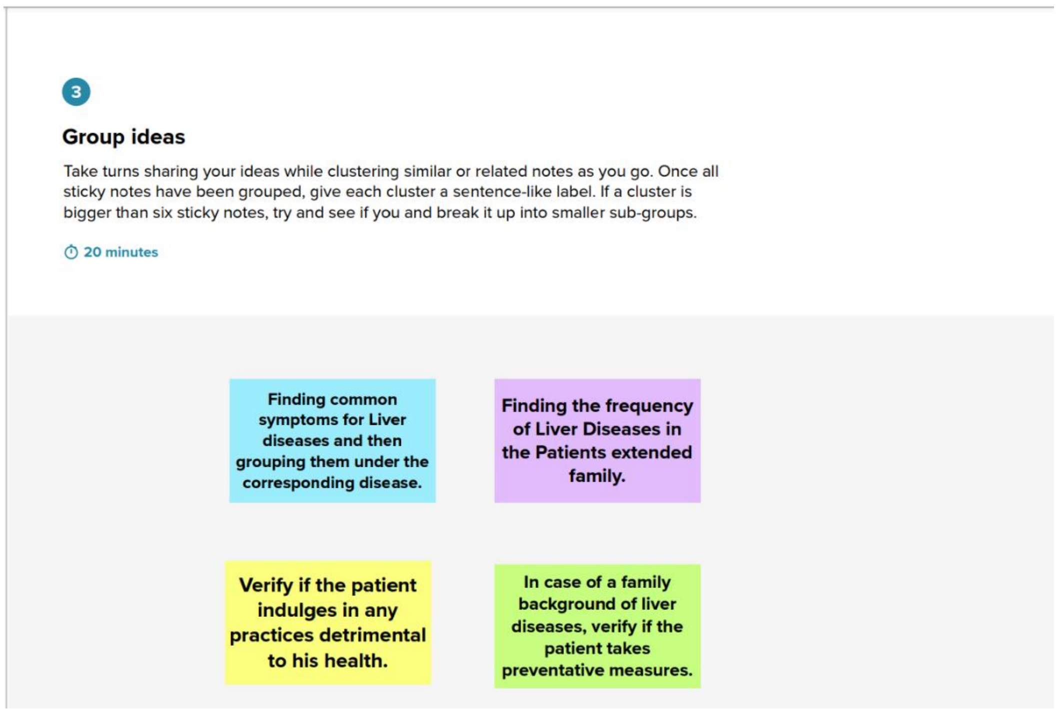
You can select a sticky note and hit the pencil [switch to sketch] icon to start drawing!

Adhithya
- Tracing history of Liver Disease amongst ancestors
- Analysing symptoms
- Finding others with similar symptoms
- Checking patient history

Divya
- Analysing patient habits
- Age of Patient
- Probability of others within same age group having the same disease
- If contagious, rate of spread of infection

Sudharshan
- Finding if disease is contagious or not
- Finding out to what extent the Liver is infected
- Checking if disease is fatal
- History of relevant medication

Umabharathi
- Workplace environment
- Tracing close contact with other infected people
- Travel History
- Analysing personal and surrounding hygiene



3

Group ideas

Take turns sharing your ideas while clustering similar or related notes as you go. Once all sticky notes have been grouped, 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.

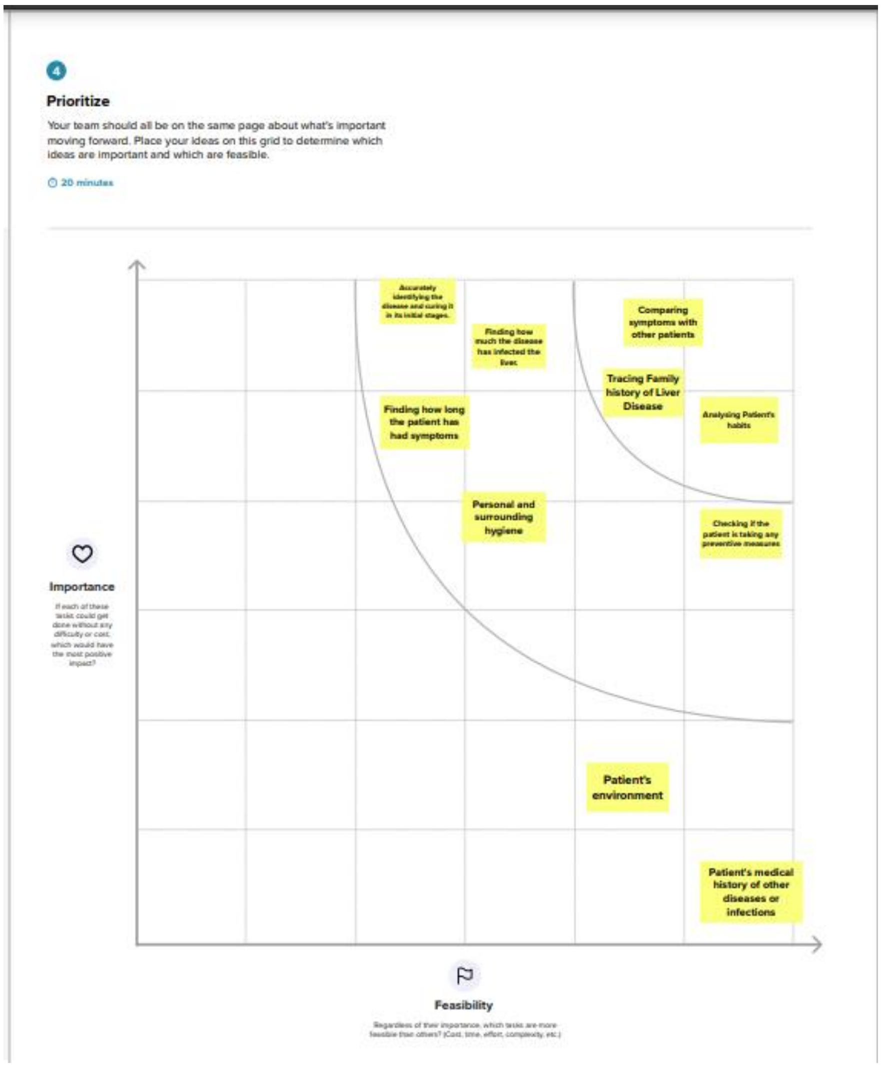
20 minutes

Finding common symptoms for Liver diseases and then grouping them under the corresponding disease.

Finding the frequency of Liver Diseases in the Patients extended family.

Verify if the patient indulges in any practices detrimental to his health.

In case of a family background of liver diseases, verify if the patient takes preventative measures.



4

Prioritize

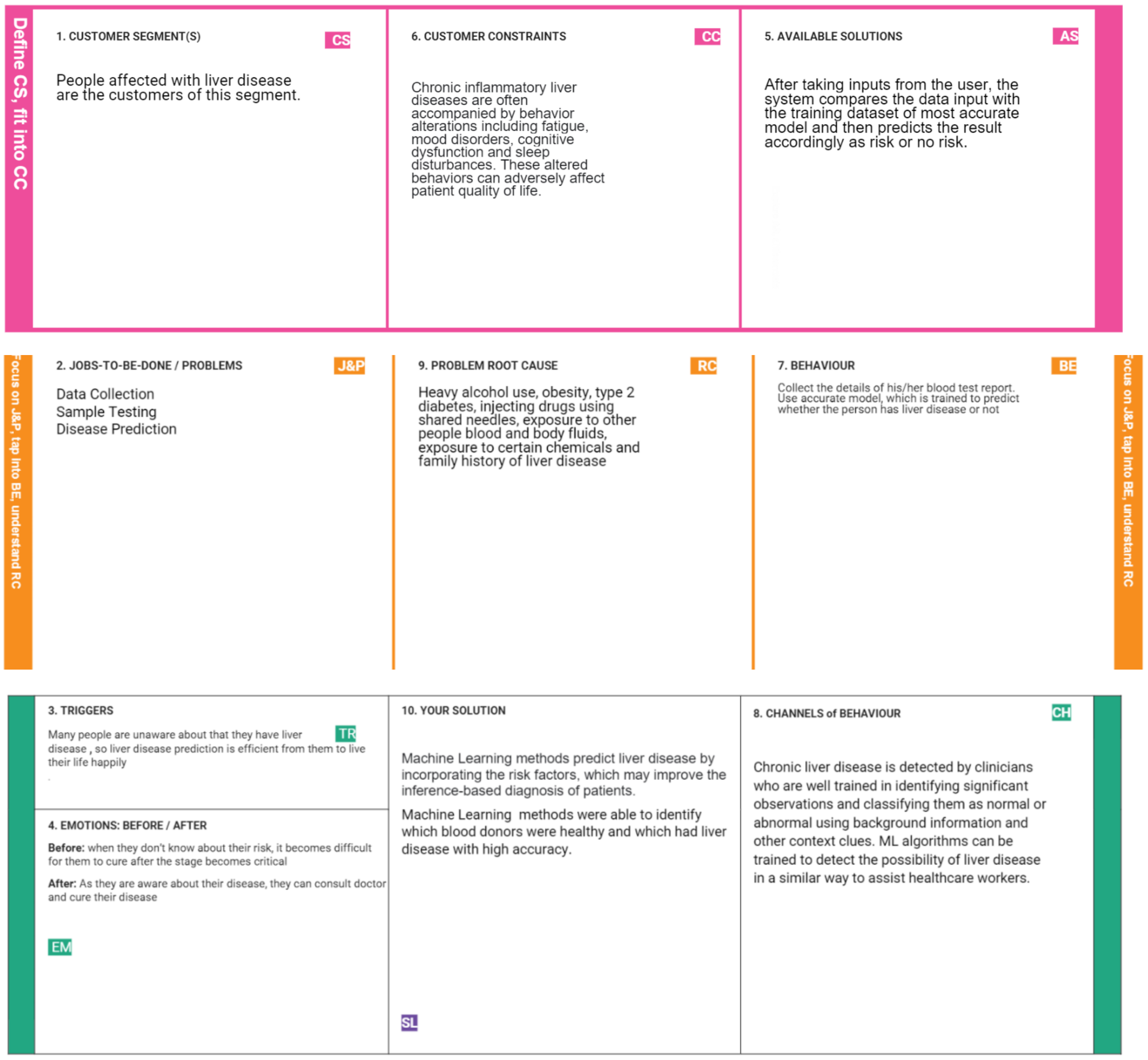
Your team should all be on the same page about what's important moving forward. Place your ideas on this grid to determine which ideas are important and which are feasible.

20 minutes

3.3 : Proposed Solution

S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	Liver diseases avert the normal function of the liver. 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 analyse the parameters of various classification algorithms and compare their predictive accuracies so as to find out the best classifier for determining the liver disease.
2.	Idea / Solution description	Data from liver patients such as liver enzymes, proteins, age and gender are examined to predict the likeliness of liver disease. Then building a model by applying various machine learning algorithms, and integrate it to flask -based web application.
3.	Novelty / Uniqueness	This model takes in the patient's symptoms as inputs and compares it with the symptoms exhibited by others with a liver disease. It also dynamically alters the algorithm based on the predicted value and actual output value.
4.	Social Impact / Customer Satisfaction	This model helps in early prediction of liver disease by doctors. User can predict the disease by entering parameters in the web application. Early prediction of the disease helps save the life.
5.	Business Model (Revenue Model)	Medical experts can use the model to predict the likelihood of the disease without the use of other complicated medical tests, thus reducing the time for diagnosis.
6.	Scalability of the Solution	Picking the right framework increases the scalability of the model. Data must be trained properly so that it produces error free results. Reducing the precision will right away lead to reduced memory requirement.

3.4 : Problem Solution Fit



Define CS, fit into CC

1. CUSTOMER SEGMENT(S) CS

People affected with liver disease are the customers of this segment.

6. CUSTOMER CONSTRAINTS CC

Chronic inflammatory liver diseases are often accompanied by behavior alterations including fatigue, mood disorders, cognitive dysfunction and sleep disturbances. These altered behaviors can adversely affect patient quality of life.

5. AVAILABLE SOLUTIONS AS

After taking inputs from the user, the system compares the data input with the training dataset of most accurate model and then predicts the result accordingly as risk or no risk.

Focus on J&P, tap into BE, understand RC

2. JOBS-TO-BE-DONE / PROBLEMS J&P

Data Collection
Sample Testing
Disease Prediction

9. PROBLEM ROOT CAUSE RC

Heavy alcohol use, obesity, type 2 diabetes, injecting drugs using shared needles, exposure to other people blood and body fluids, exposure to certain chemicals and family history of liver disease

7. BEHAVIOUR BE

Collect the details of his/her blood test report. Use accurate model, which is trained to predict whether the person has liver disease or not

Focus on J&P, tap into BE, understand RC

3. TRIGGERS TR

Many people are unaware about that they have liver disease , so liver disease prediction is efficient from them to live their life happily

.

4. EMOTIONS: BEFORE / AFTER EM

Before: when they don't know about their risk, it becomes difficult for them to cure after the stage becomes critical

After: As they are aware about their disease, they can consult doctor and cure their disease

10. YOUR SOLUTION SL

Machine Learning methods predict liver disease by incorporating the risk factors, which may improve the inference-based diagnosis of patients.

Machine Learning methods were able to identify which blood donors were healthy and which had liver disease with high accuracy.

8. CHANNELS of BEHAVIOUR CH

Chronic liver disease is detected by clinicians who are well trained in identifying significant observations and classifying them as normal or abnormal using background information and other context clues. ML algorithms can be trained to detect the possibility of liver disease in a similar way to assist healthcare workers.

4 : REQUIREMENT ANALYSIS

4.1 : Functional Requirements

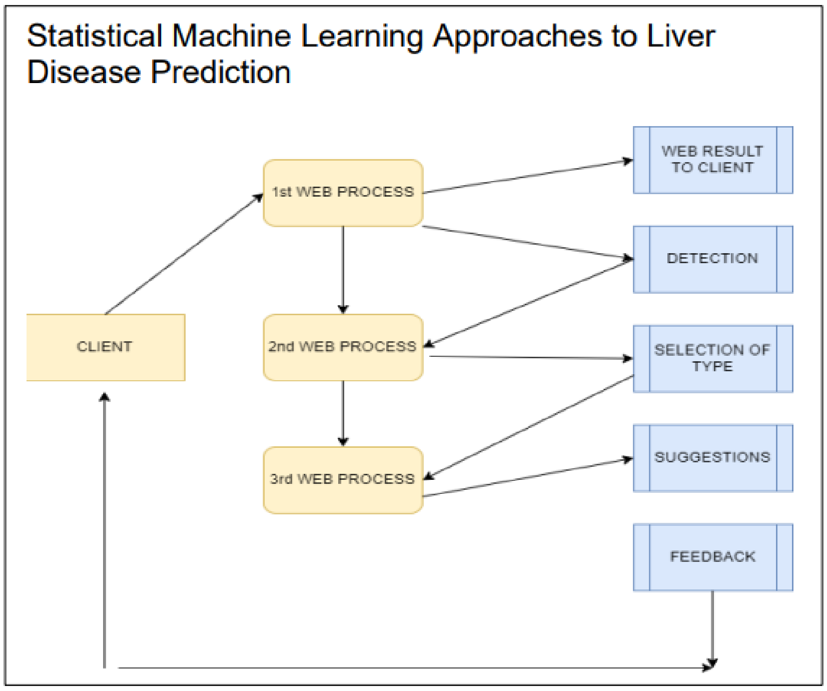
FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	User Registration	Registration through form present in liver disease prediction website
FR-2	User Confirmation	Confirmation done via registered Email
FR-3	Prediction	Based on the data entered (like age, gender, and symptoms) the type of liver disease is predicted.
FR-4	Hardware Requirements	Intel i3 core processor Internet Connectivity
FR-5	Software Requirements	Windows 7 or higher Python 3.6.0 or higher Visual Studio Code Dataset Jupiter notebook
FR-6	Database Retrieval	Data is retrieved from the database

4.2 : Non – Functional Requirements

FR No.	Non-Functional Requirement	Description
NRF-1	Usability	Death rate is decreased as the disease is predicted early
NRF-2	Security	Ensures all data in the system is protected
NRF-3	Reliability	Provides secured storage of data and access
NRF-4	Performance	Performance is high as various Machine learning classification algorithms are used to find the best and accurate model.
NRF-5	Availability	Accessible to all the users.
NRF-6	Scalability	It is acceptable to fit over any place and any resources.

5 : PROJECT DESIGN

5.1 : Data Flow Diagrams



1. The images of the potential symptoms are fed as input to the ML model.

2. Images of previously confirmed symptoms are fed as input to the ML model.

3. The images are then tested, and the model determines whether or not there are symptoms.

4. The results of the model are then compared with accurate medical diagnostics.

5. The error in the prediction is calculated and sent as feedback to the model.

6. The accuracy of the model increases over the course of time.

5.2 : Solution and Technical Architecture

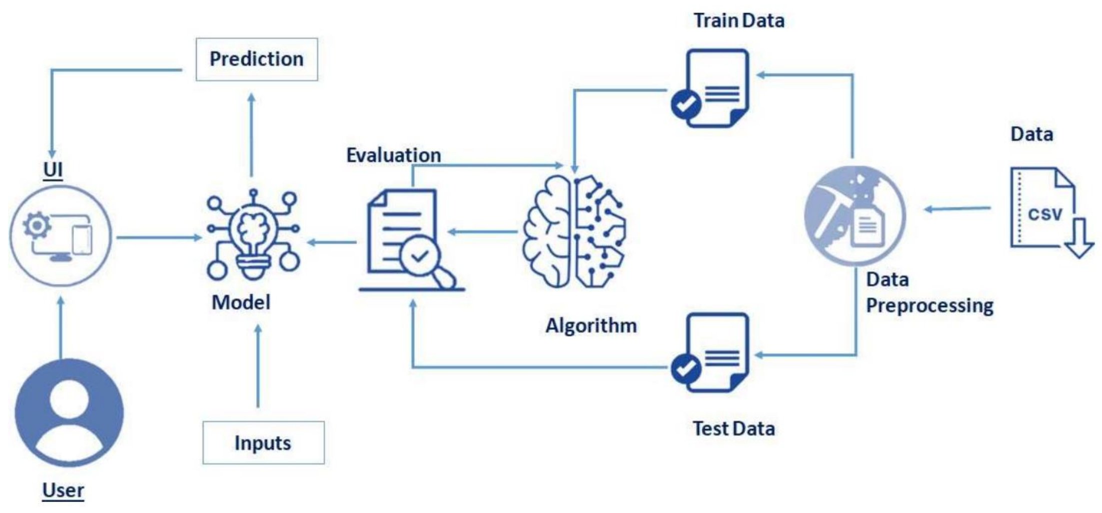


Table-1 : Components & Technologies:

S.No	Component	Description	Technology
1	User Interface	A Web page which gets user input and send it to the backend for predicting the given input data	HTML, CSS.
2	Predicting Model	Model which takes user input and predict whether the person have liver disease or not	Python, Numpy, Pandas, Scikit-learn
3	Web Server	A web server which serves static HTML user interface files and uses Predicting ML model to process output and send back to the client	Python, Flask
4	Machine Learning Model	The model used for classify whether the person have liver disease or not	Support Vector Machine Model
5	Cloud Deployment	The ML model is bind with web server and deployed in to the IBM cloud	IBM cloud/AWS

Table-2: Application Characteristics:

S.No	Characteristics	Description	Technology
1	Open-Source Frameworks	There are several opensource frameworks used for data preprocessing , data analysis , Model building , pickling and web servers	Numpy ,Pandas ,Seaborn ,Scikit-learn ,Pickle ,Flask
2	Security Implementations	Since no user data is stored in the server , There is no security issues in the application side	-
3	Scalable Architecture	It is a monolithic architecture and , if needed the model which is used to predict can be developed separately as a microservice	Microservices using Docker and Kubernetes
4	Availability	If the load increases a load balancer can be used to handle the huge request	Nginx Server, Load Balancer
5	Performance	The performance is still good and has no need the interference of external CDNs, It can able to handle adequate amount of network requests	-

5.3 : User Stories

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer (Web user)	Login	USN-1	As a user, I can register for the application by entering my email, password, and confirming	I can access my account / dashboard	High	Sprint-1

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
			my password.			
		USN-2	As a user, I will receive confirmation email once I have registered	I can receive confirmation email	High	Sprint-1
	Dashboard	USN-3	As a user, I need to enter my details	I can get information pertaining to my details	High	Sprint-2
		USN-4	As a user, I need to provide my Test Details	I can get results based on my test details.	High	Sprint-1
Administator	Services	USN-5	As an admin, I need to provide valid result	I can get a result.	High	Sprint-1
		USN-6	As an admin I need to provide valid/useful Suggestions	I can get suggestions.	Medium	Sprint-1
	Mass Data Process	USN-7	As an admin need to collect all the details and information	I can use it on a later date.	High	Sprint-1
		USN-8	As an admin I need to store all the details and information	I can use it on a later date.	High	Sprint-1

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Hospital Administrator	Login	USN-9	As an admin I need to login and access details of customers	I can use it for further processes.	High	Sprint-1

6 : PROJECT PLANNING AND SCHEDULING

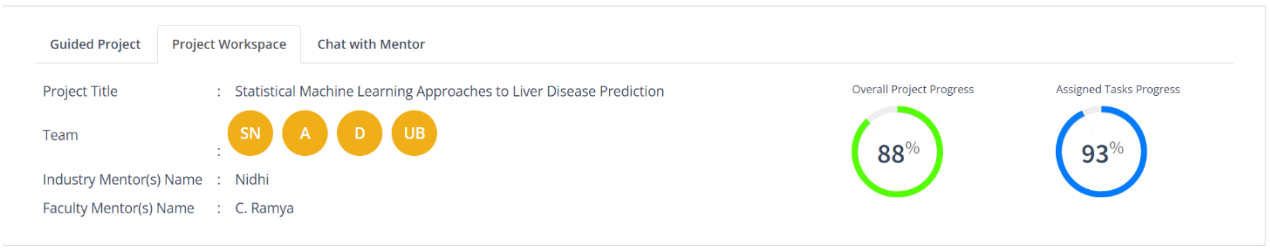
6.1 : Sprint Planning and Estimation

Sprint	Milestone
Sprint 1	1. User Registers into the application through entering Email Id and Password for confirmation. 2. User Receives a confirmation mail for their registered Email. 3. User can also register to the application through Mobile number. 4. User logs in into the website using Email Id and password.
Sprint 2	1. User can access the dashboard. 2. User enters the required details of the patient to get the desired output based on our model's prediction.
Sprint 3	1. Application stores the predictions, that can be used for future analysis. 2. The data stored has to be maintained securely.
Sprint 4	1. Administrator should properly maintain the website and update it whenever required.

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	10	5 Days	24-Oct-22	29-Oct-22	0	3-Nov-22
Sprint-2	15	6 Days	31-Oct-22	5-Nov-22	15	5-Nov-22
Sprint-3	10	6 Days	7-Nov-22	13-Nov-22		
Sprint-4	15	5 Days	14-Nov-22	19-Nov-22		

6.3: Reports from JIRA

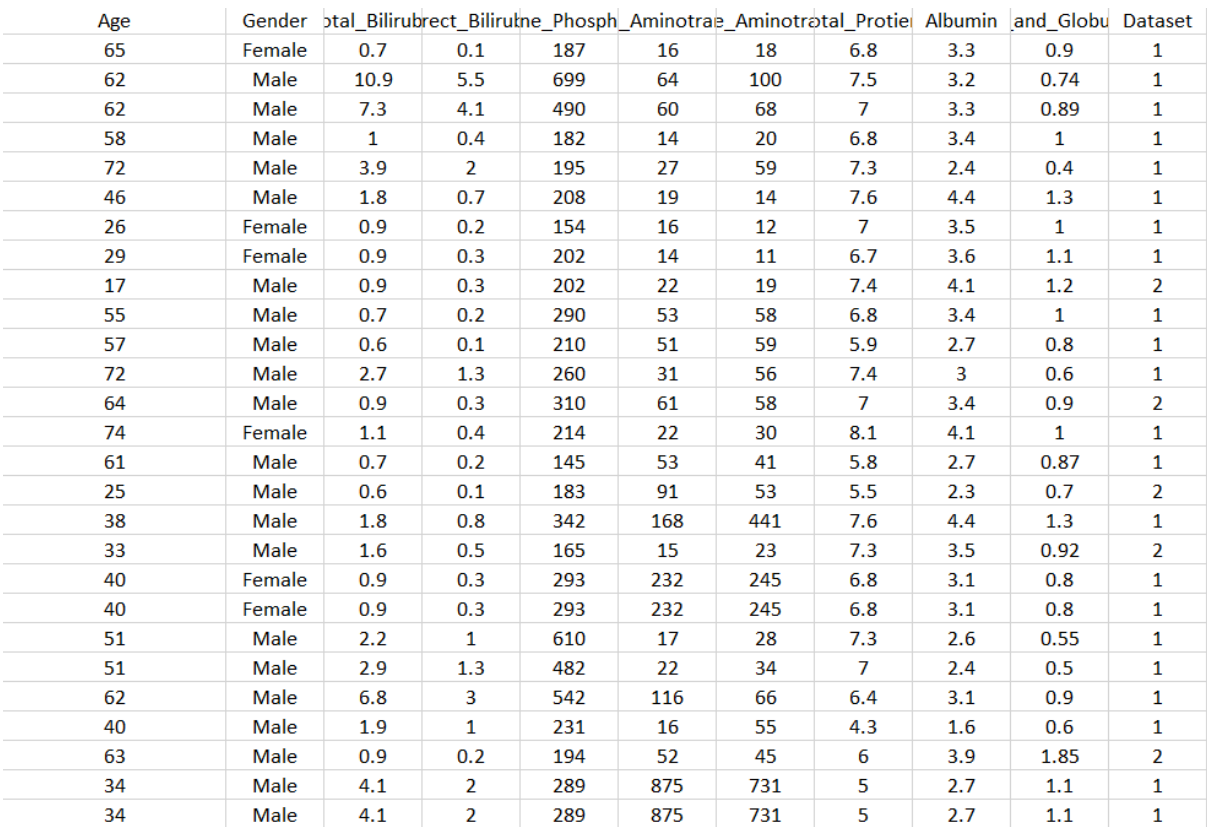


Project Title : Statistical Machine Learning Approaches to Liver Disease Prediction
Team :
Industry Mentor(s) Name : Nidhi
Faculty Mentor(s) Name : C. Ramya

7 : CODING AND SOLUTIONING

7.1 : Feature 1

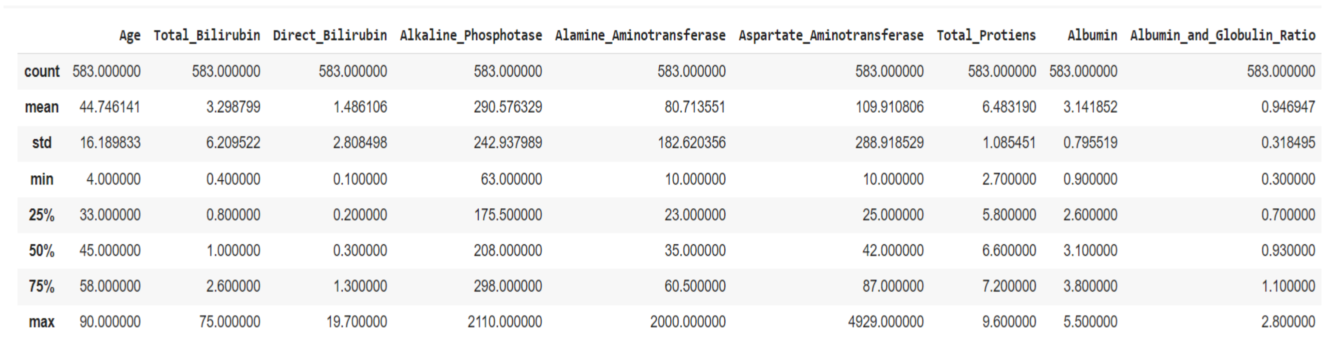
Dataset taken for training :



Age	Gender	otal_Bilirub	rect_Bilirub	ne_Phosph	_Aminotra	e_Aminotra	otal_Protie	Albumin	and_Globu	Dataset
65	Female	0.7	0.1	187	16	18	6.8	3.3	0.9	1
62	Male	10.9	5.5	699	64	100	7.5	3.2	0.74	1
62	Male	7.3	4.1	490	60	68	7	3.3	0.89	1
58	Male	1	0.4	182	14	20	6.8	3.4	1	1
72	Male	3.9	2	195	27	59	7.3	2.4	0.4	1
46	Male	1.8	0.7	208	19	14	7.6	4.4	1.3	1
26	Female	0.9	0.2	154	16	12	7	3.5	1	1
29	Female	0.9	0.3	202	14	11	6.7	3.6	1.1	1
17	Male	0.9	0.3	202	22	19	7.4	4.1	1.2	2
55	Male	0.7	0.2	290	53	58	6.8	3.4	1	1
57	Male	0.6	0.1	210	51	59	5.9	2.7	0.8	1
72	Male	2.7	1.3	260	31	56	7.4	3	0.6	1
64	Male	0.9	0.3	310	61	58	7	3.4	0.9	2
74	Female	1.1	0.4	214	22	30	8.1	4.1	1	1
61	Male	0.7	0.2	145	53	41	5.8	2.7	0.87	1
25	Male	0.6	0.1	183	91	53	5.5	2.3	0.7	2
38	Male	1.8	0.8	342	168	441	7.6	4.4	1.3	1
33	Male	1.6	0.5	165	15	23	7.3	3.5	0.92	2
40	Female	0.9	0.3	293	232	245	6.8	3.1	0.8	1
40	Female	0.9	0.3	293	232	245	6.8	3.1	0.8	1
51	Male	2.2	1	610	17	28	7.3	2.6	0.55	1
51	Male	2.9	1.3	482	22	34	7	2.4	0.5	1
62	Male	6.8	3	542	116	66	6.4	3.1	0.9	1
40	Male	1.9	1	231	16	55	4.3	1.6	0.6	1
63	Male	0.9	0.2	194	52	45	6	3.9	1.85	2
34	Male	4.1	2	289	875	731	5	2.7	1.1	1
34	Male	4.1	2	289	875	731	5	2.7	1.1	1

This is the Excel sheet visualization of the dataset that has been taken for the ML Model. It contains Age, Gender, Total Bilirubin, Aminophosphate and so on. If the Dataset is 1, then the patient has dieasease, if 2, the there is no probability of any dieasease.

Dataset Description:



	Age	Total_Bilirubin	Direct_Bilirubin	Alkaline_Phosphotase	Alamine_Aminotransferase	Aspartate_Aminotransferase	Total_Protiens	Albumin	Albumin_and_Globulin_Ratio
count	583.000000	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	0.946947
std	16.189833	6.209522	2.808498	242.937989	182.620356	288.918529	1.085451	0.795519	0.318495
min	4.000000	0.400000	0.100000	63.000000	10.000000	10.000000	2.700000	0.900000	0.300000
25%	33.000000	0.800000	0.200000	175.500000	23.000000	25.000000	5.800000	2.600000	0.700000
50%	45.000000	1.000000	0.300000	208.000000	35.000000	42.000000	6.600000	3.100000	0.930000
75%	58.000000	2.600000	1.300000	298.000000	60.500000	87.000000	7.200000	3.800000	1.100000
max	90.000000	75.000000	19.700000	2110.000000	2000.000000	4929.000000	9.600000	5.500000	2.800000

```
# Printing How many Unique values present in each feature:
for feature in dataset.columns:
    print(feature,":", len(dataset[feature].unique()))
```

```
Age : 72
Gender : 2
Total_Bilirubin : 113
Direct_Bilirubin : 80
Alkaline_Phosphotase : 263
Alamine_Aminotransferase : 152
Aspartate_Aminotransferase : 177
Total_Protiens : 58
Albumin : 40
Albumin_and_Globulin_Ratio : 69
Dataset : 2
```

The number of unique values is printed.

```
#Computing the Interquantile range of Total_Bilirubin feature to calcul
ate the boundaries:
IQR = dataset.Total_Bilirubin.quantile(0.75)-
dataset.Total_Bilirubin.quantile(0.25)
```

The boundaries of the input values are calculated and are quantised.

```
# Extreme outliers
lower_bridge = dataset['Total_Bilirubin'].quantile(0.25) - (IQR*3)
upper_bridge = dataset['Total_Bilirubin'].quantile(0.75) + (IQR*3)

print(lower_bridge)
print(upper_bridge)

# if value greater than upper bridge, we replace that value with upper_
bridge value:
dataset.loc[dataset['Total_Bilirubin'] >= upper_bridge, 'Total_Bilirubi
n'] = upper_bridge
```

We repeat the same for all the input value columns.

```
# Independent and Dependent Feature:
X = dataset.iloc[:, :-1]
y = dataset.iloc[:, -1]
print(X)
```

```
     Age  Gender  Total_Bilirubin  Alkaline_Phosphotase  \
0     65       0              0.7                 187.0
1     62       1              8.0                 665.5
2     62       1              7.3                 490.0
3     58       1              1.0                 182.0
4     72       1              3.9                 195.0
..   ...     ...              ...                   ...
578   60       1              0.5                 500.0
579   40       1              0.6                  98.0
580   52       1              0.8                 245.0
581   31       1              1.3                 184.0
582   38       1              1.0                 216.0


     Alamine_Aminotransferase  Aspartate_Aminotransferase  Total_Protiens  \
0                          16                          18             6.8
1                          64                         100             7.5
2                          60                          68             7.0
3                          14                          20             6.8
4                          27                          59             7.3
..                        ...                         ...             ...
578                        20                          34             5.9
579                        35                          31             6.0
580                        48                          49             6.4
581                        29                          32             6.8
582                        21                          24             7.3
```

The independent features are displayed. Dataset column is the dependent feature.

```
# Train Test Split:
from sklearn.model_selection import train_test_split
X_train,X_test,y_train,y_test = train_test_split(X_smote,y_smote, test_size=0.3, random_state=33)
```

```
# Feature Importance :
from sklearn.feature_selection import SelectKBest
from sklearn.feature_selection import chi2

### Apply SelectKBest Algorithm
ordered_rank_features=SelectKBest(score_func=chi2,k=9)
ordered_feature=ordered_rank_features.fit(X,y)

dfscores=pd.DataFrame(ordered_feature.scores_,columns=["Score"])
dfcolumns=pd.DataFrame(X.columns)

features_rank=pd.concat([dfcolumns,dfscores],axis=1)

features_rank.columns=['Features','Score']
features_rank.nlargest(9, 'Score')
```

The splitting of data and the best features from which the output can be generated are evaluated.

```python
from sklearn.ensemble import RandomForestClassifier
RandomForest = RandomForestClassifier()
RandomForest = RandomForest.fit(X_train,y_train)

# Predictions:
y_pred = RandomForest.predict(X_test)

# Performance:
print('Accuracy:', accuracy_score(y_test,y_pred))
print(confusion_matrix(y_test,y_pred))
print(classification_report(y_test,y_pred))
```

```
Accuracy: 0.8481012658227848
[[100  18]
 [ 18 101]]
              precision    recall  f1-score   support

           1       0.85      0.85      0.85       118
           2       0.85      0.85      0.85       119

    accuracy                           0.85       237
   macro avg       0.85      0.85      0.85       237
weighted avg       0.85      0.85      0.85       237
```

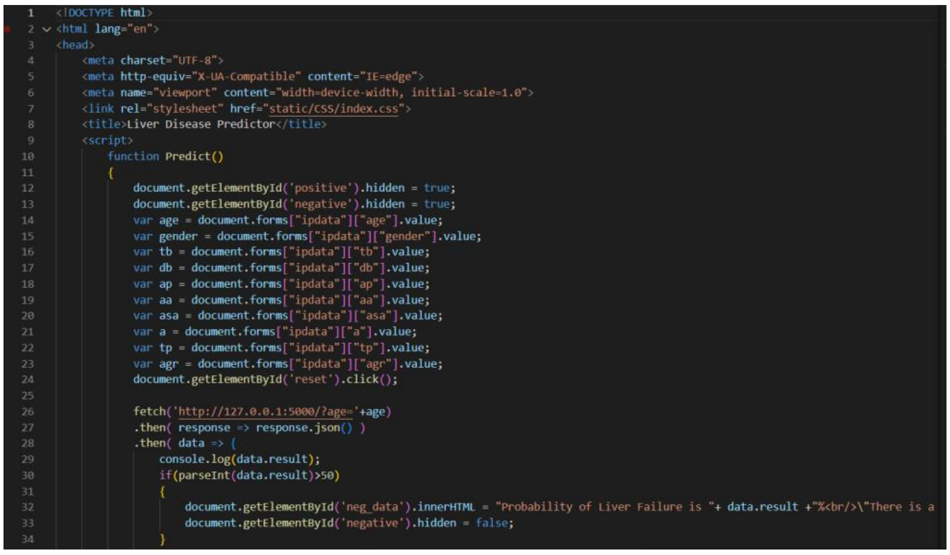
The Random Forest Classifier is used to train the model. It renders an overall Accuracy of 85%.

```python
# Creating a pickle file for the classifier
import pickle
filename = 'Liver.pkl'
pickle.dump(RandomForestClassifier, open(filename, 'wb'))
```

The created ML model is exported to a pickle file, which is nothing but an array of values.

HTML Code:

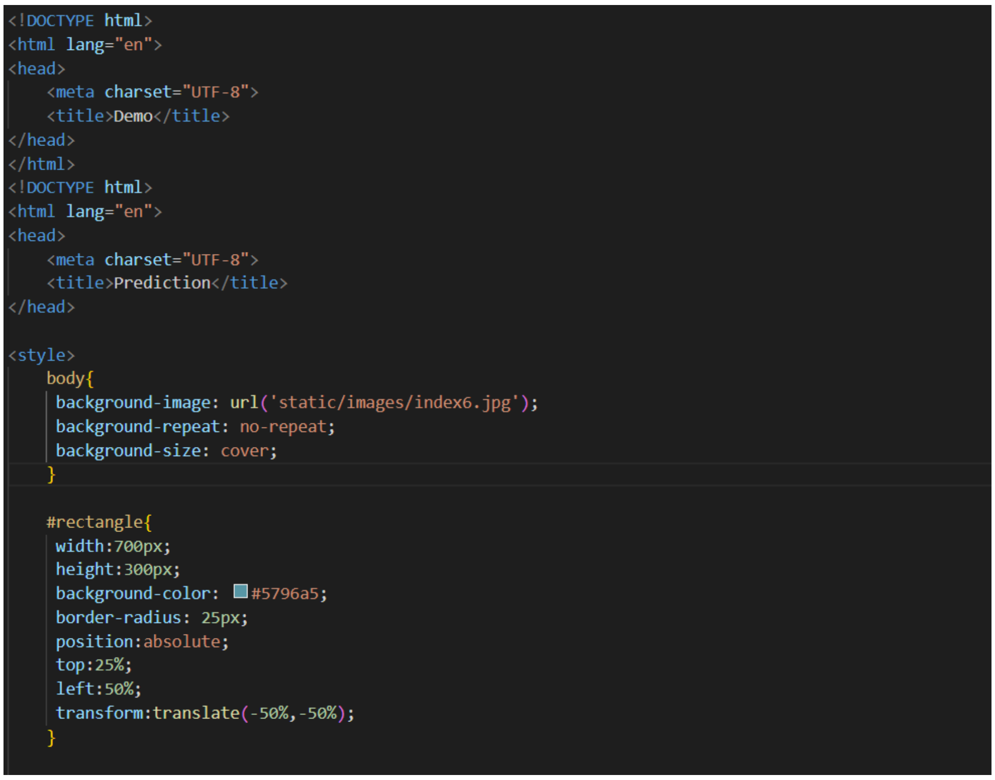
Index Page:



```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link rel="stylesheet" href="static/CSS/index.css">
    <title>Liver Disease Predictor</title>
    <script>
        function Predict()
        {
            document.getElementById('positive').hidden = true;
            document.getElementById('negative').hidden = true;
            var age = document.forms["ipdata"]["age"].value;
            var gender = document.forms["ipdata"]["gender"].value;
            var tb = document.forms["ipdata"]["tb"].value;
            var db = document.forms["ipdata"]["db"].value;
            var ap = document.forms["ipdata"]["ap"].value;
            var aa = document.forms["ipdata"]["aa"].value;
            var asa = document.forms["ipdata"]["asa"].value;
            var a = document.forms["ipdata"]["a"].value;
            var tp = document.forms["ipdata"]["tp"].value;
            var agr = document.forms["ipdata"]["agr"].value;
            document.getElementById('reset').click();

            fetch('http://127.0.0.1:5000/?age='+age)
            .then( response => response.json() )
            .then( data => {
                console.log(data.result);
                if(parseInt(data.result)>50)
                {
                    document.getElementById('neg_data').innerHTML = "Probability of Liver Failure is "+ data.result +"%<br/>\"There is a
                    document.getElementById('negative').hidden = false;
                }
```

Code to design the home page. The page consists of a from wherein the user can enter the vital parameters. When submitted the values are given to the model.

Predict Page:

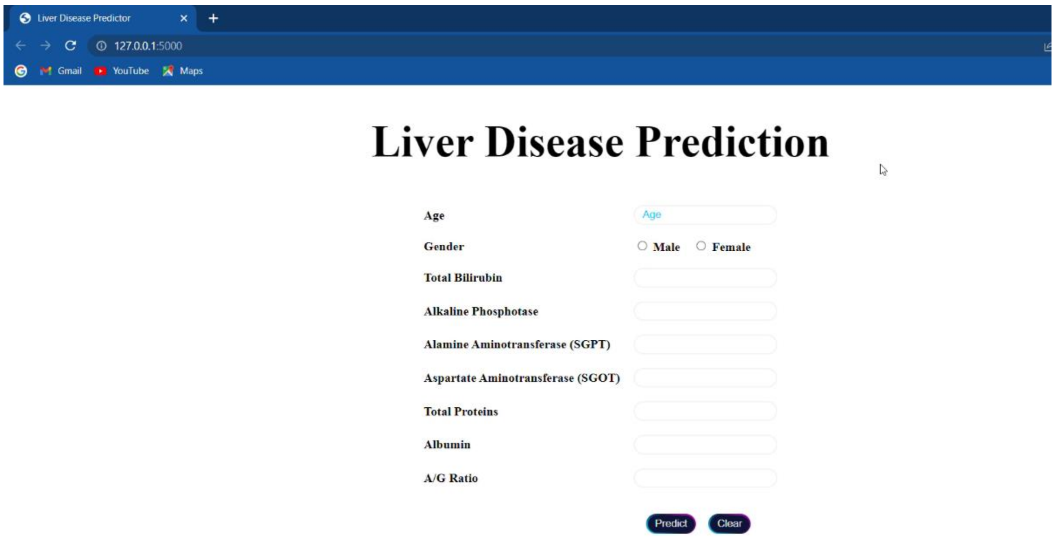


```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Demo</title>
</head>
</html>
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Prediction</title>
</head>

<style>
    body{
     background-image: url('static/images/index6.jpg');
     background-repeat: no-repeat;
     background-size: cover;
    }

    #rectangle{
     width:700px;
     height:300px;
     background-color: #5796a5;
     border-radius: 25px;
     position:absolute;
     top:25%;
     left:50%;
     transform:translate(-50%,-50%);
    }
```

This page displays the output predicted value. This is a post method and hence receives the value from model and displays on the web page.

Web Page Design :

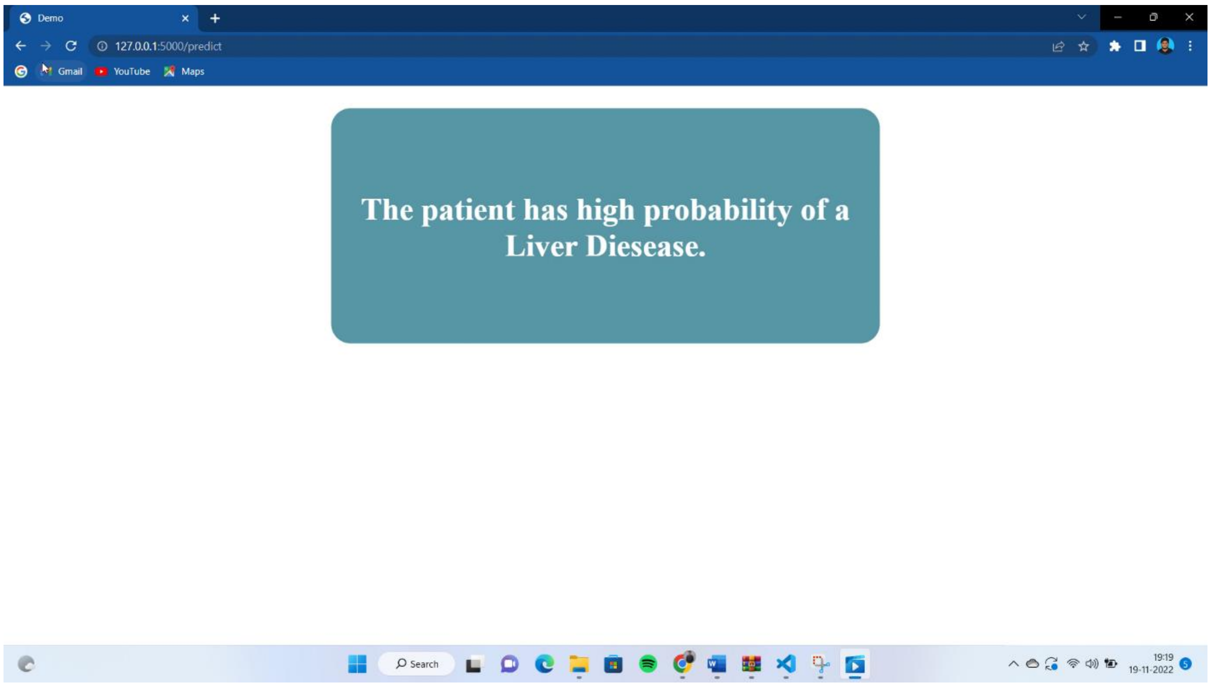
Home Page :



Liver Disease Prediction

Age	Age
Gender	Male Female
Total Bilirubin	
Alkaline Phosphotase	
Alamine Aminotransferase (SGPT)	
Aspartate Aminotransferase (SGOT)	
Total Proteins	
Albumin	
A/G Ratio	

Output Page :

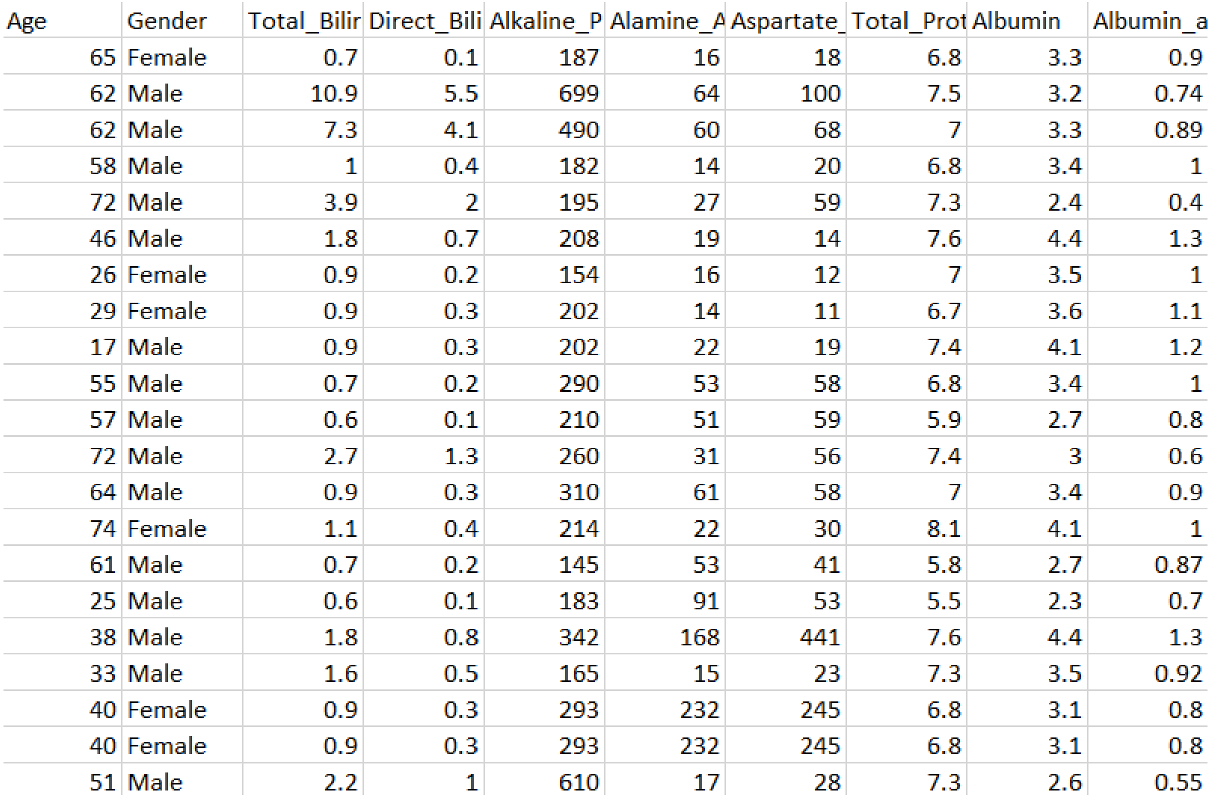


The patient has high probability of a Liver Diesease.

8: TESTING

8.1: Test Cases

A wide range of test cases are generated and the model is checked for prediction. A wide range of the test case values of the vital parameters are generated and the corresponding output from the dataset is also verified.



Age	Gender	Total_Bilir	Direct_Bili	Alkaline_P	Alamine_A	Aspartate_	Total_Prot	Albumin	Albumin_a
65	Female	0.7	0.1	187	16	18	6.8	3.3	0.9
62	Male	10.9	5.5	699	64	100	7.5	3.2	0.74
62	Male	7.3	4.1	490	60	68	7	3.3	0.89
58	Male	1	0.4	182	14	20	6.8	3.4	1
72	Male	3.9	2	195	27	59	7.3	2.4	0.4
46	Male	1.8	0.7	208	19	14	7.6	4.4	1.3
26	Female	0.9	0.2	154	16	12	7	3.5	1
29	Female	0.9	0.3	202	14	11	6.7	3.6	1.1
17	Male	0.9	0.3	202	22	19	7.4	4.1	1.2
55	Male	0.7	0.2	290	53	58	6.8	3.4	1
57	Male	0.6	0.1	210	51	59	5.9	2.7	0.8
72	Male	2.7	1.3	260	31	56	7.4	3	0.6
64	Male	0.9	0.3	310	61	58	7	3.4	0.9
74	Female	1.1	0.4	214	22	30	8.1	4.1	1
61	Male	0.7	0.2	145	53	41	5.8	2.7	0.87
25	Male	0.6	0.1	183	91	53	5.5	2.3	0.7
38	Male	1.8	0.8	342	168	441	7.6	4.4	1.3
33	Male	1.6	0.5	165	15	23	7.3	3.5	0.92
40	Female	0.9	0.3	293	232	245	6.8	3.1	0.8
40	Female	0.9	0.3	293	232	245	6.8	3.1	0.8
51	Male	2.2	1	610	17	28	7.3	2.6	0.55

8.2: User Acceptance Testing

The project has been tested extensively with a number of users. The users found the interface very easy to use. The Web pages were colourful and attractive. There were no unnecessary details in the web page. It was clean and simple that any new user could master it. The data input format was also simple. The user need not enter any unit. He could simply enter the value. The prediction time is fairly low at an average time of 3 seconds. This delay primarily varies depending on the internet connectivity. The model has been hosted in IBM cloud. Thus, with the API available, the model can be accessed remotely from any system provided IBM access key is given. The model predicts the power output close to the actual power generated. The users are satisfied with the predicted output power. Although the prediction is not very accurate it comes closer to the actual power. Various inputs have been given by the users to test the consistency of the model. The model proved itself and all the users accepted the model as a reliable and convenient.

9: RESULTS

9.1: Performance Metrics

The RandomForestClassifier ML model that we have used here has better performance in speed and accuracy compared to other models. We have compared the performance metrics of 3 models and selected this as the best for the application. The model performed well for all the test cases. The API developed also performed good with no glitches or lag found during the testing phase.

10: Advantages and Disadvantages

10.1: Advantages

The advantage is that with a constrained dataset, we were able to achieve high efficiency. The prediction accuracy nearly modelled upto 90%. Moreover, the input data is nothing but a set of blood parameters. This can be easily taken using a blood test/urine test. Thus, this prediction is cost effective and is readily available.

10.2 : Disadvantages

It is not completely reliable as this does not involve thousands of data points. This can be used for initial testing and not for critical cases of affected patients.

11 : CONCLUSION

The Random Forest Classifier model that has been used above performs well for our dataset. The model is fast and consumes less resources. The API developed is also simple and user-friendly. By using this model, we could predict the output power of a wind turbine provided the required input parameter. The model is not 100% accurate but it performs sufficiently. Other factors such as the weight, habits of the person are not considered as parameters here, but can have a significant impact on the final result.

12 : FUTURE SCOPE

The further works that can be done in this project is to include more features in model training to study the effect on the output. A long history of data (dataset of more than 1000 people) can be used for training for increased accuracy. More web pages can be designed so that the user can get more information in the same API. The dashboard can be made for User Interactive by making it to show real time prediction and analysis. Automated report generation can also be done.

13 : APPENDIX

13.1 : Source Code

Model Training :

```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt

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

dataset=pd.read_csv('/content/drive/MyDrive/indian_liver_patient.csv')

pd.pandas.set_option('display.max_columns', None)

# Last 5 records:
dataset.tail()

# Shape of dataset:
dataset.shape

# Cheaking Missing (NaN) Values:
dataset.isnull().sum()
```

```python
# Mean & Median of "Albumin_and_Globulin_Ratio" feature:
print(dataset['Albumin_and_Globulin_Ratio'].median())
print(dataset['Albumin_and_Globulin_Ratio'].mean())

dataset['Albumin_and_Globulin_Ratio'] =
dataset['Albumin_and_Globulin_Ratio'].fillna(dataset['Albumin_and_Globulin_Rat
io'].median())

# Datatypes:
dataset.dtypes

dataset.describe()

# Target feature:
print("Liver Disease Patients      :", dataset['Dataset'].value_counts()[1])
print("Non Liver Disease Patients  :", dataset['Dataset'].value_counts()[2])

# Visualization:
sns.countplot(dataset['Dataset'])
plt.show()

# Histrogram of Age:
plt.figure(figsize=(8,5))
sns.histplot(dataset['Age'], kde=True)
plt.title('Age', fontsize=20)
plt.show()

# Gender feature:
print("Total Male   :", dataset['Gender'].value_counts()[0])
print("Total Female :", dataset['Gender'].value_counts()[1])

# Visualization:
sns.countplot(dataset['Gender'])
plt.show()

# Printing How many Unique values present in each feature:
for feature in dataset.columns:
    print(feature,":", len(dataset[feature].unique()))

# Label Encoding
dataset['Gender'] = np.where(dataset['Gender']=='Male', 1,0)

dataset.head()

# Correlation using Heatmap:
plt.figure(figsize=(12,8))
sns.heatmap(dataset.corr(), annot=True, cmap='YlGnBu')
plt.show()
```

```
# Droping 'Direct_Bilirubin' feature:
dataset = dataset.drop('Direct_Bilirubin', axis=1)

sns.distplot(dataset['Albumin'])

# Calculate the boundaries of Total_Protiens feature which differentiates the
outliers:
uppper_boundary=dataset['Total_Protiens'].mean() + 3*
dataset['Total_Protiens'].std()
lower_boundary=dataset['Total_Protiens'].mean() - 3*
dataset['Total_Protiens'].std()

print(dataset['Total_Protiens'].mean())
print(lower_boundary)
print(uppper_boundary)

##### Calculate the boundaries of Albumin feature which differentiates the
outliers:
uppper_boundary=dataset['Albumin'].mean() + 3* dataset['Albumin'].std()
lower_boundary=dataset['Albumin'].mean() - 3* dataset['Albumin'].std()

print(dataset['Albumin'].mean())
print(lower_boundary)
print(uppper_boundary)

# Lets compute the Interquantile range of Total_Bilirubin feature to calculate
the boundaries:
IQR = dataset.Total_Bilirubin.quantile(0.75)-
dataset.Total_Bilirubin.quantile(0.25)

# Extreme outliers
lower_bridge = dataset['Total_Bilirubin'].quantile(0.25) - (IQR*3)
upper_bridge = dataset['Total_Bilirubin'].quantile(0.75) + (IQR*3)

print(lower_bridge)
print(upper_bridge)

# if value greater than upper bridge, we replace that value with upper_bridge
value:
dataset.loc[dataset['Total_Bilirubin'] >= upper_bridge, 'Total_Bilirubin'] =
upper_bridge

# Lets compute the Interquantile range of Alkaline_Phosphotase feature to
calculate the boundaries:
IQR = dataset.Alkaline_Phosphotase.quantile(0.75) -
dataset.Alkaline_Phosphotase.quantile(0.25)
```

```
# Extreme outliers
lower_bridge = dataset['Alkaline_Phosphotase'].quantile(0.25) - (IQR*3)
upper_bridge = dataset['Alkaline_Phosphotase'].quantile(0.75) + (IQR*3)

print(lower_bridge)
print(upper_bridge)

# if value greater than upper bridge, we replace that value with upper_bridge
value:
dataset.loc[dataset['Alkaline_Phosphotase'] >= upper_bridge,
'Alkaline_Phosphotase'] = upper_bridge

# Lets compute the Interquantile range of Alamine_Aminotransferase feature to
calculate the boundaries:
IQR = dataset.Alamine_Aminotransferase.quantile(0.75) -
dataset.Alamine_Aminotransferase.quantile(0.25)

# Extreme outliers
lower_bridge = dataset['Alamine_Aminotransferase'].quantile(0.25) - (IQR*3)
upper_bridge = dataset['Alamine_Aminotransferase'].quantile(0.75) + (IQR*3)

print(lower_bridge)
print(upper_bridge)

# if value greater than upper bridge, we replace that value with upper_bridge
value:
dataset.loc[dataset['Alamine_Aminotransferase'] >= upper_bridge,
'Alamine_Aminotransferase'] = upper_bridge

IQR = dataset.Aspartate_Aminotransferase.quantile(0.75) -
dataset.Aspartate_Aminotransferase.quantile(0.25)

# Extreme outliers
lower_bridge = dataset['Aspartate_Aminotransferase'].quantile(0.25) - (IQR*3)
upper_bridge = dataset['Aspartate_Aminotransferase'].quantile(0.75) + (IQR*3)

print(lower_bridge)
print(upper_bridge)

# if value greater than upper bridge, we replace that value with upper_bridge
value:
dataset.loc[dataset['Aspartate_Aminotransferase'] >= upper_bridge,
'Aspartate_Aminotransferase'] = upper_bridge

# Lets compute the Interquantile range of Albumin_and_Globulin_Ratio feature
to calculate the boundaries
IQR = dataset.Albumin_and_Globulin_Ratio.quantile(0.75) -
dataset.Albumin_and_Globulin_Ratio.quantile(0.25)
```

```
# Extreme outliers
lower_bridge = dataset['Albumin_and_Globulin_Ratio'].quantile(0.25) - (IQR*3)
upper_bridge = dataset['Albumin_and_Globulin_Ratio'].quantile(0.75) + (IQR*3)

print(lower_bridge)
print(upper_bridge)

# if value greater than upper bridge, we replace that value with upper_bridge
value:
dataset.loc[dataset['Albumin_and_Globulin_Ratio'] >= upper_bridge,
'Albumin_and_Globulin_Ratio'] = upper_bridge

# Top 5 records:
dataset.head()

# Independent and Dependent Feature:
X = dataset.iloc[:, :-1]
y = dataset.iloc[:, -1]
print(X)

# SMOTE Technique:
from imblearn.combine import SMOTETomek
smote = SMOTETomek()
X_smote, y_smote = smote.fit_resample(X,y)

# Counting before and after SMOTE:
from collections import Counter
print('Before SMOTE : ', Counter(y))
print('After SMOTE  : ', Counter(y_smote))

# Train Test Split:
from sklearn.model_selection import train_test_split
X_train,X_test,y_train,y_test = train_test_split(X_smote,y_smote,
test_size=0.3, random_state=33)

# Feature Importance :
from sklearn.feature_selection import SelectKBest
from sklearn.feature_selection import chi2

### Apply SelectKBest Algorithm
ordered_rank_features=SelectKBest(score_func=chi2,k=9)
ordered_feature=ordered_rank_features.fit(X,y)

dfscores=pd.DataFrame(ordered_feature.scores_,columns=["Score"])
dfcolumns=pd.DataFrame(X.columns)

features_rank=pd.concat([dfcolumns,dfscores],axis=1)
```

```
features_rank.columns=['Features','Score']
features_rank.nlargest(9, 'Score')

# Importing Performance Metrics:
from sklearn.metrics import accuracy_score, confusion_matrix,
classification_report

from sklearn.ensemble import RandomForestClassifier
RandomForest = RandomForestClassifier()
RandomForest = RandomForest.fit(X_train,y_train)

# Predictions:
y_pred = RandomForest.predict(X_test)

# Performance:
print('Accuracy:', accuracy_score(y_test,y_pred))
print(confusion_matrix(y_test,y_pred))
print(classification_report(y_test,y_pred))

# Creating a pickle file for the classifier
import pickle
filename = 'Liver.pkl'
pickle.dump(RandomForestClassifier, open(filename, 'wb'))
```

FLASK Application

```
import numpy as np
import os
from PIL import Image
from flask import Flask, request, render_template, url_for
from werkzeug.utils import secure_filename, redirect
from gevent.pywsgi import WSGIServer
from flask import send_from_directory
from joblib import Parallel,delayed
import joblib
import pandas as pd
from scipy.sparse import issparse
UPLOAD_FOLDER = 'D:/sdhi/PROJECT DEVELOPMENT PHASE/SPRINT 3/UPLOADS'
app = Flask(__name__)
app.config['UPLOAD_FOLDER'] = UPLOAD_FOLDER

@app.route('/')
def index():
   return render_template('index.html')
```

```
@app.route('/predict', methods=['GET', 'POST'])
def upload():
   if request.method == "POST":
       data =
[[request.form.get('age'),request.form.get('gender'),request.form.get('tb'),request.fo
rm.get('ap'),request.form.get('aa'),request.form.get('asa')
              ,request.form.get('tp'),request.form.get('a'),request.form.get('agr')]]

       df = pd.DataFrame(data,
columns=['Age','Gender','Total_Bilirubin','Alkaline_Phosphotase',

'Alamine_Aminotransferase','Aspartate_Aminotransferase','Total_Protiens',
                                      'Albumin','Albumin_and_Globulin_Ratio'])

       gh=joblib.load('Liver.pkl')
       num=gh.predict(df)
       if num[0]==1:
          k="The patient has high probability of a Liver Diesease."
       else:
          k="The patient has low probability of a Liver Diesease."
       return render_template('predict.html', num=k)


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

Home Web Page :

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link rel="stylesheet" href="static/CSS/index.css">
    <title>Liver Disease Predictor</title>
    <script>
        function Predict()
        {
            document.getElementById('positive').hidden = true;
            document.getElementById('negative').hidden = true;
            var age = document.forms["ipdata"]["age"].value;
            var gender = document.forms["ipdata"]["gender"].value;
            var tb = document.forms["ipdata"]["tb"].value;
            var db = document.forms["ipdata"]["db"].value;
            var ap = document.forms["ipdata"]["ap"].value;
            var aa = document.forms["ipdata"]["aa"].value;
            var asa = document.forms["ipdata"]["asa"].value;
            var a = document.forms["ipdata"]["a"].value;
            var tp = document.forms["ipdata"]["tp"].value;
            var agr = document.forms["ipdata"]["agr"].value;
            document.getElementById('reset').click();

            fetch('http://127.0.0.1:5000/?age='+age)
            .then( response => response.json() )
            .then( data => {
                console.log(data.result);
                if(parseInt(data.result)>50)
                {
                    document.getElementById('neg_data').innerHTML =
"Probability of Liver Failure is "+ data.result +"%<br/>\"There is a
Possiblity that you are having Liver Disease.\"";
                    document.getElementById('negative').hidden = false;
                }
                else
                {
                    document.getElementById('pos_data').innerHTML =
"Probability of Liver Failure is "+ data.result +"%<br/>\"There is a Very Less
Possiblity that you are have Liver Disease. Stay Healthy\"";
                    document.getElementById('positive').hidden = false;
                }

            })
            .catch( error => console.log(error) )
        }
```

```html
    </script>
</head>
<body>
    <h2 id="h">Liver Disease Prediction</h2>
    <div id="positive" hidden><h4 id="pos_data">Positive</h4></div>
    <div id="negative" hidden><h4 id="neg_data">Negative</h4></div>
    <table>
        <form action="/predict" method="POST" enctype="multipart/form-data">
        <tr>
            <td>Age</td>
            <td><input name="age" type="number" min="0" placeholder="Age"
required></td>
        </tr>
        <tr>
            <td>Gender</td>
            <td><input name="gender" type="radio" value=1
required>&nbsp;Male&ensp;&ensp;<input id="gender" name="gender" type="radio"
value=0>&nbsp;Female</td>
        </tr>
        <tr>
            <td>Total Bilirubin</td>
            <td><input name="tb" type="number" min="0" step="0.01"
placeholder="" required></td>
        </tr>
        <tr>
            <td>Alkaline Phosphotase</td>
            <td><input name="ap" type="number" min="0" step="0.01"
placeholder="" required></td>
        </tr>
        <tr>
            <td>Alamine Aminotransferase (SGPT)</td>
            <td><input name="aa" type="number" min="0" step="0.01"
placeholder="" required></td>
        </tr>
        <tr>
            <td>Aspartate Aminotransferase (SGOT)</td>
            <td><input name="asa" type="number" min="0" step="0.01"
placeholder="" required></td>
        </tr>
        <tr>
            <td>Total Proteins</td>
            <td><input name="tp" type="number" min="0" step="0.01"
placeholder="" required></td>
        </tr>
        <tr>
            <td>Albumin</td>
            <td><input name="a" type="number" min="0" step="0.01"
placeholder="" required></td>
```

```
        </tr>
        <tr>
            <td>A/G Ratio</td>
            <td><input name="agr" type="number" min="0" step="0.01"
placeholder="" required></td>
        </tr>
        <tr>
            <td></td>
            <td id="btn"><div id="btndiv"><input id="submit" type="submit"
value="Predict"></div>&emsp;<div id="btndiv"><input id="reset" type="reset"
value="Clear"></div></td>
        </tr>

        </form>
    </table>
</body>
</html>
```

Output Web Page :

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Demo</title>
</head>
</html>
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Prediction</title>
</head>

<style>
    body{
     background-image: url('static/images/index6.jpg');
     background-repeat: no-repeat;
     background-size: cover;
    }

    #rectangle{
     width:700px;
     height:300px;
     background-color: #5796a5;
     border-radius: 25px;
     position:absolute;
     top:25%;
     left:50%;
     transform:translate(-50%,-50%);
    }

    #ans{
  text-align: center;
  font-size: 40px;
  margin: 0 auto;
  padding: 3% 5%;
  padding-top: 15%;
  color: white;
    }

</style>
<body>
    <div id="rectangle">
        <h1 id="ans">{{num}}</h1>
    </div>
</body>
</html>
```

13.2 : GitHub & Project Demo Link

GitHub Repo : https://github.com/IBM-EPBL/IBM-Project-37354-1660305522

Project Demo Link : https://youtu.be/_nDTXZon-eE