Statistical Machine Learning Approaches to Liver Disease Prediction

Mentor :- Theetchenya S

Industry Mentor :- Nidhi

Team:

S.HARSHITHA

T.G.GOWTHAM

T.DHAYANAND

R.AKRAM

(Final year Computer Science and Engineering)

College:

Sona College of Technology.

INTRODUCTION

1.1**Poject 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.

This Project examines data from liver patients concentrating on relationships between a key list of liver enzymes, proteins, age and gender using them to try and predict the likeliness of liver disease. Here we are building a model by applying various machine learning algorithms find the best accurate model. And integrate to flask based web application. User can predict the disease by entering parameters in the web application.

1.2 **Purpose**

The purpose of this study was to extract significant predictors for liver disease from the medical analysis of 615 humans using ML algorithms. Data visualizations were implemented to reveal significant findings such as missing values

2 LITERATURE SURVEY

2.1 **Existing Problem**

The existing model uses Logistic Regression or Random Forest Classified Machine learning algorithm and has accuracy less than 75%

2.2 **References**

- Wang, Y.; Li, Y.; Wang, X.; Gacesa, R.; Zhang, J.; Zhou, L.; Wang, B. Predicting Liver Disease Risk Using a Combination of Common Clinical Markers: A Screening Model from Routine Health Check-Up. Dis. Markers 2020, 2020, 8460883

- Torkadi, P.P.; Apte, I.C.; Bhute, A.K. Biochemical evaluation of patients of alcoholic liver disease and non-alcoholic liver disease. Indian J. Clin. Biochem. 2014, 29, 79–83

- Ceriotti, F.; Henny, J.; Queraltó, J.; Ziyu, S.; Özarda, Y.; Chen, B.; Boyd, J.C.; Panteghini, M. Common reference intervals for aspartate aminotransferase (AST), alanine aminotransferase (ALT) and γ-glutamyl transferase (GGT) in serum: Results from an IFCC multicenter study. Clin. Chem. Lab. Med. 2010, 48, 1593–1601

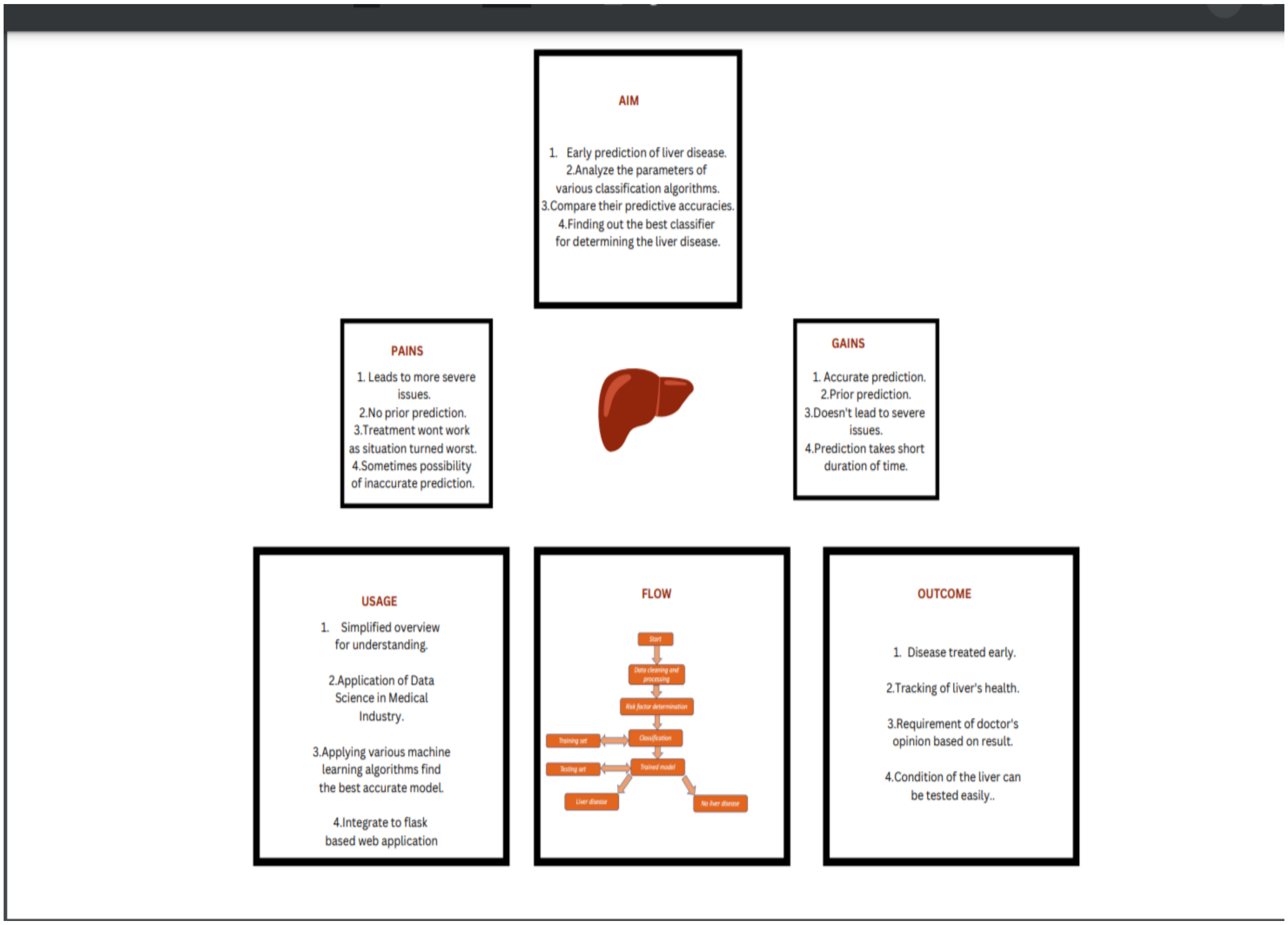
2.3 **Problem Statement Definition**

Liver diseases avert the normal function of the liver. Mainly due to the large amount of alcohol consumption liver disease arises.

Discovering the existence of liver disease at an early stage is a complex task for the doctors

3.IDEATION & PROPOSED SOLUTION

3.1 <u>**Empathy Map Canvas**</u>



AIM

1. Early prediction of liver disease.
2.Analyze the parameters of various classification algorithms.
3.Compare their predictive accuracies.
4.Finding out the best classifier for determining the liver disease.

PAINS

1. Leads to more severe issues.
2.No prior prediction.
3.Treatment wont work as situation turned worst.
4.Sometimes possibility of inaccurate prediction.

GAINS

1. Accurate prediction.
2.Prior prediction.
3.Doesn't lead to severe issues.
4.Prediction takes short duration of time.

USAGE

1. Simplified overview for understanding.

2.Application of Data Science in Medical Industry.

3.Applying various machine learning algorithms find the best accurate model.

4.Integrate to flask based web application

FLOW

OUTCOME

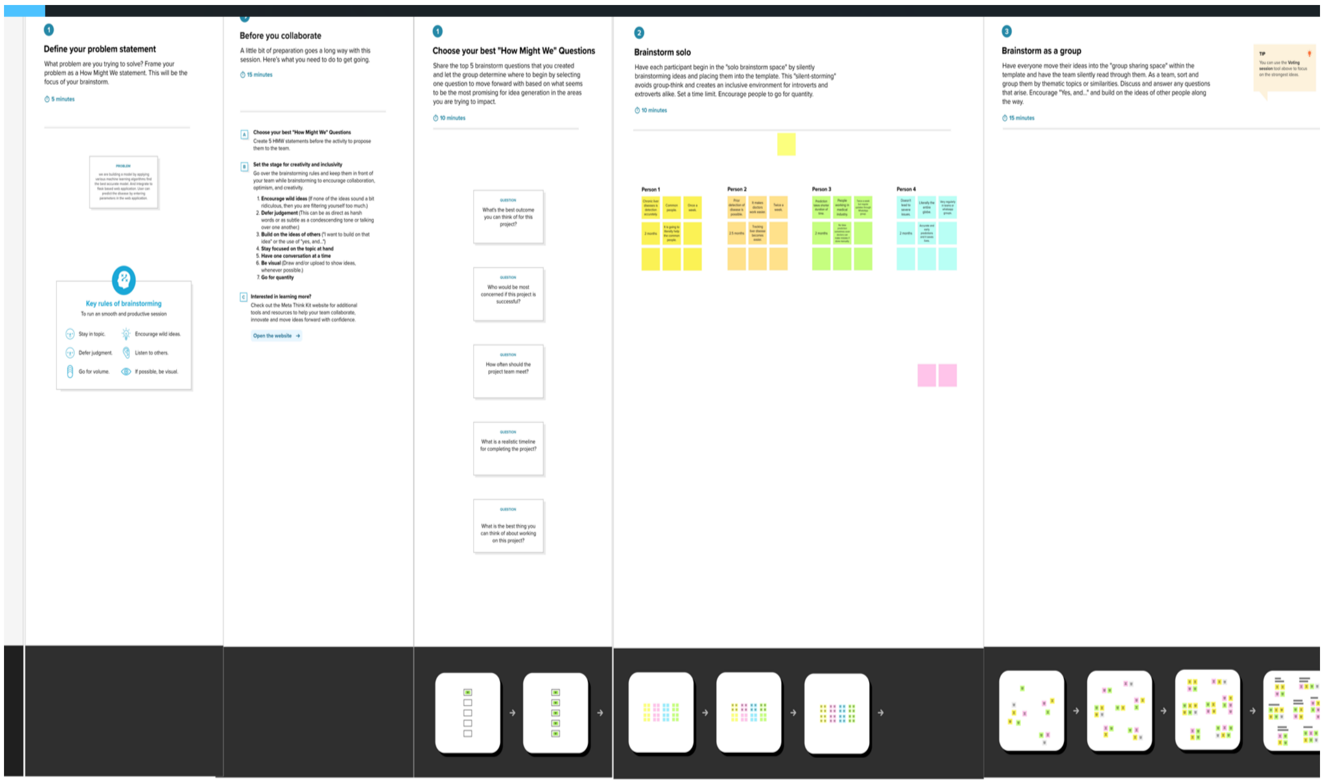
1. Disease treated early.

2.Tracking of liver's health.

3.Requirement of doctor's opinion based on result.

4.Condition of the liver can be tested easily..

3.2 **Ideation & 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

we are building a model by applying various machine learning algorithms find the best accurate model. And integrate to flask based web application. User can predict the disease by entering parameters in the web application.

Key rules of brainstorming

To run an smooth and productive session

Stay in topic.

Encourage wild ideas.

Defer judgment.

Listen to others.

Go for volume.

If possible, be visual.

Before you collaborate

A little bit of preparation goes a long way with this session. Here's what you need to do to get going.

15 minutes

A **Choose your best "How Might We" Questions**
Create 5 HMW statements before the activity to propose them to the team.

B **Set the stage for creativity and inclusivity**
Go over the brainstorming rules and keep them in front of your team while brainstorming to encourage collaboration, optimism, and creativity.

1. **Encourage wild ideas** (If none of the ideas sound a bit ridiculous, then you are filtering yourself too much.)
2. **Defer judgement** (This can be as direct as harsh words or as subtle as a condescending tone or talking over one another.)
3. **Build on the ideas of others** ("I want to build on that idea" or the use of "yes, and...")
4. **Stay focused on the topic at hand**
5. **Have one conversation at a time**
6. **Be visual** (Draw and/or upload to show ideas, whenever possible.)
7. **Go for quantity**

C **Interested in learning more?**
Check out the Meta Think Kit website for additional tools and resources to help your team collaborate, innovate and move ideas forward with confidence.

Open the website →

1

Choose your best "How Might We" Questions

Share the top 5 brainstorm questions that you created and let the group determine where to begin by selecting one question to move forward with based on what seems to be the most promising for idea generation in the areas you are trying to impact.

10 minutes

QUESTION
What's the best outcome you can think of for this project?

QUESTION
Who would be most concerned if this project is successful?

QUESTION
How often should the project team meet?

QUESTION
What is a realistic timeline for completing the project?

QUESTION
What is the best thing you can think of about working on this project?

2

Brainstorm solo

Have each participant begin in the "solo brainstorm space" by silently brainstorming ideas and placing them into the template. This "silent-storming" avoids group-think and creates an inclusive environment for introverts and extroverts alike. Set a time limit. Encourage people to go for quantity.

10 minutes

Person 1
Chronic liver diseases is detection accurately.
Common people.
Once a week.
2 months
It is going to literally help the common people.

Person 2
Prior detection of disease is possible.
It makes doctors work easier.
Twice a week.
2.5 months
Tracking liver disease becomes easier.

Person 3
Prediction takes shorter duration of time.
People working in medical industry.
[illegible]
2 months
[illegible]

Person 4
Doesn't lead to severe issues.
Literally the entire globe.
Very regularly in teams or whatsapp groups.
2 months
Accurate and early predictions and it saves lives.

3

Brainstorm as a group

Have everyone move their ideas into the "group sharing space" within the template and have the team silently read through them. As a team, sort and group them by thematic topics or similarities. Discuss and answer any questions that arise. Encourage "Yes, and..." and build on the ideas of other people along the way.

15 minutes

TIP
You can use the **Voting session** tool above to focus on the strongest ideas.

3.3 **Proposed Solution**

Researchers are attempting to discover models for early diagnosis of illness utilizing biomedical information. Since the most recent couple of decades, they have utilized a parcel of models for early finding, each with their very own advantages and disadvantages. In this research, a CHIRP based model is proposed for the early forecast of liver disease.

4 REQUIREMENT ANALYSIS

4.1 **Functional Requirements**

- User Registration
- User Confirmation
- patient details like age,proteins, enzyme etc
- prediction

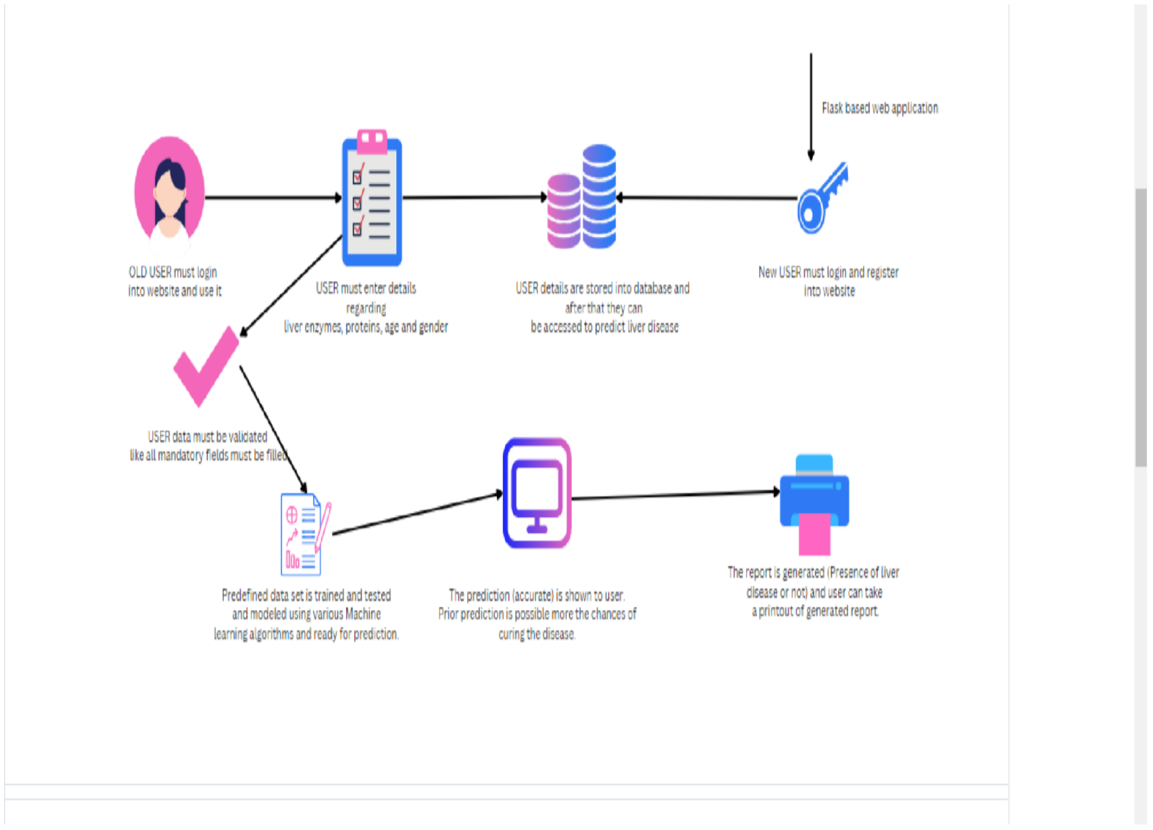
- Retrieval(Database)
- Internet connection

4.2 **Non -Functional Requirements**

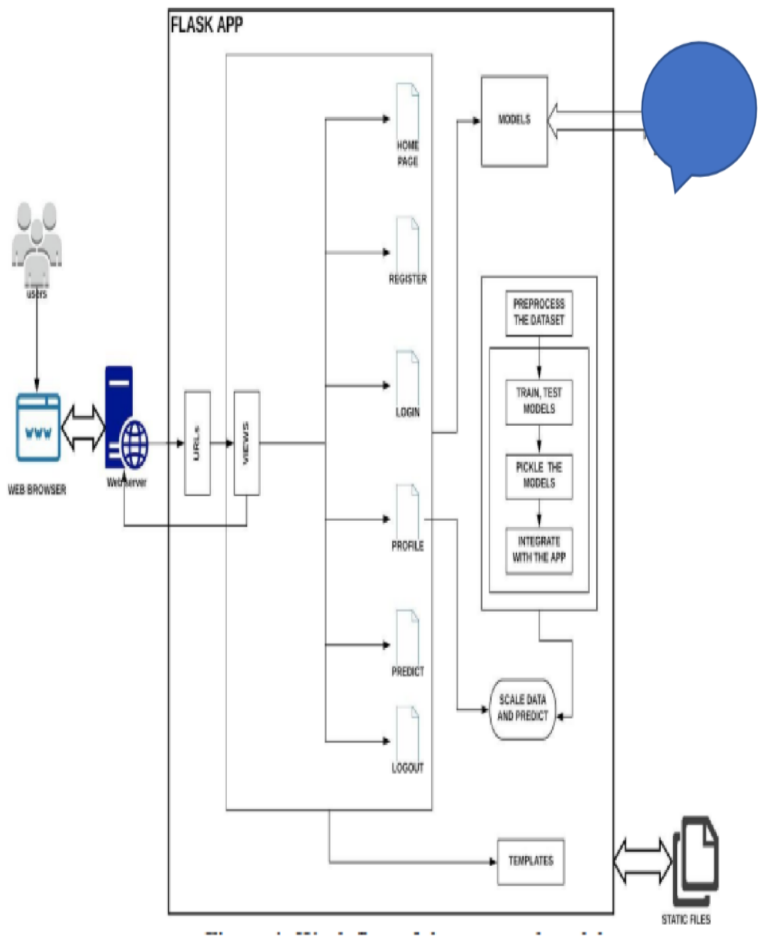
- Usability
- Security
- Reliability
- Performance
- Availability
- Scalability

5 PROJECT DESIGN

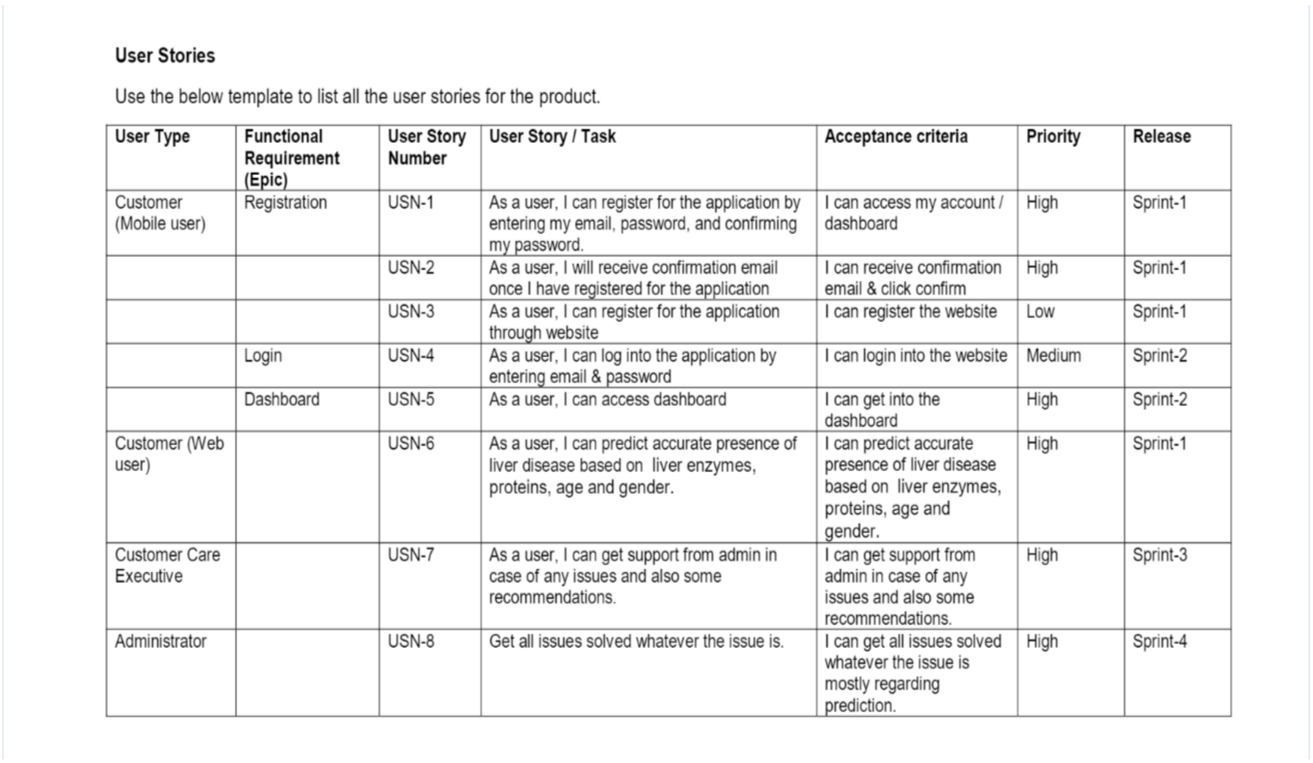
5.1 **Data Flow Diagrams**


Flask based web application
OLD USER must login into website and use it
USER must enter details regarding liver enzymes, proteins, age and gender
USER details are stored into database and after that they can be accessed to predict liver disease
New USER must login and register into website
USER data must be validated like all mandatory fields must be fille
Predefined data set is trained and tested and modeled using various Machine learning algorithms and ready for prediction.
The prediction (accurate) is shown to user. Prior prediction is possible more the chances of curing the disease.
The report is generated (Presence of liver disease or not) and user can take a printout of generated report.

5.2 **Solution & Technical Architecture**



5.3 **User Stories**



User Stories

Use the below template to list all the user stories for the product.

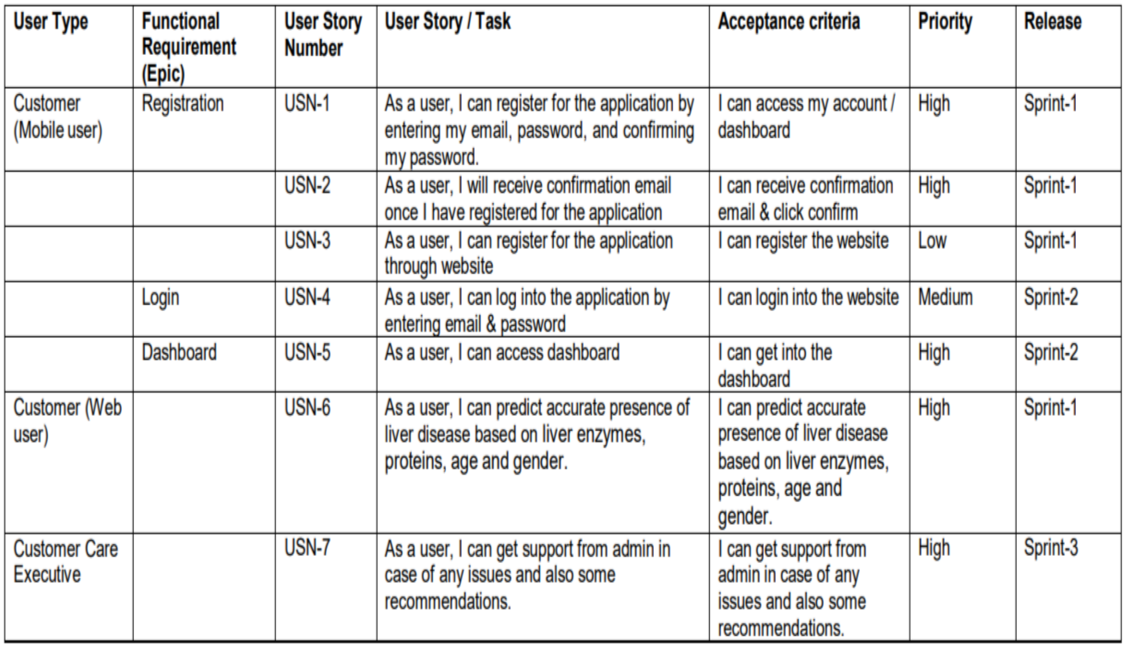
User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer (Mobile user)	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming my password.	I can access my account / dashboard	High	Sprint-1
		USN-2	As a user, I will receive confirmation email once I have registered for the application	I can receive confirmation email & click confirm	High	Sprint-1
		USN-3	As a user, I can register for the application through website	I can register the website	Low	Sprint-1
	Login	USN-4	As a user, I can log into the application by entering email & password	I can login into the website	Medium	Sprint-2
	Dashboard	USN-5	As a user, I can access dashboard	I can get into the dashboard	High	Sprint-2
Customer (Web user)		USN-6	As a user, I can predict accurate presence of liver disease based on liver enzymes, proteins, age and gender.	I can predict accurate presence of liver disease based on liver enzymes, proteins, age and gender.	High	Sprint-1
Customer Care Executive		USN-7	As a user, I can get support from admin in case of any issues and also some recommendations.	I can get support from admin in case of any issues and also some recommendations.	High	Sprint-3
Administrator		USN-8	Get all issues solved whatever the issue is.	I can get all issues solved whatever the issue is mostly regarding prediction.	High	Sprint-4

6 PROJECT PLANNIG & SCHEDULING

6.1 **Sprint Planning & Estimation**

Sprint Planning is an Agile ritual where the team plans the goals and contents of the next sprint. During the Sprint Planning, the team will review already- prioritized items in the backlog, estimate the capacity of the team for the upcoming sprint, and decide which items from the backlog will be worked on. At the end of this session, the team will have a clear picture of what they will work on and what they will deliver next.

6.2 **Sprint Delivery Schedule**



User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer (Mobile user)	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming my password.	I can access my account / dashboard	High	Sprint-1
		USN-2	As a user, I will receive confirmation email once I have registered for the application	I can receive confirmation email & click confirm	High	Sprint-1
		USN-3	As a user, I can register for the application through website	I can register the website	Low	Sprint-1
	Login	USN-4	As a user, I can log into the application by entering email & password	I can login into the website	Medium	Sprint-2
	Dashboard	USN-5	As a user, I can access dashboard	I can get into the dashboard	High	Sprint-2
Customer (Web user)		USN-6	As a user, I can predict accurate presence of liver disease based on liver enzymes, proteins, age and gender.	I can predict accurate presence of liver disease based on liver enzymes, proteins, age and gender.	High	Sprint-1
Customer Care Executive		USN-7	As a user, I can get support from admin in case of any issues and also some recommendations.	I can get support from admin in case of any issues and also some recommendations.	High	Sprint-3

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

7.1 **Features 1**

- RandomForest Classifier Machine Learing algorithm and has accuracy and less than 75%.
- XGBOOST Classification Machine Learning alogithm which has accuracy of 85% and leads to more accurate prediction
- Extratrees Classifier Machine Learing algorithm and has accuracy of 86% and leads to more accurate prediction.

```
# Random Forest
model = RandomForestClassifier(n_jobs=-1,random_state=123)
model.fit(X_train, y_train)
y_train_hat = model.predict(X_train)
y_test_hat = model.predict(X_test)

print(model)
print('Train performance')
print('-------------------------------------------------------')
print(classification_report(y_train, y_train_hat))

print('Test performance')
print('-------------------------------------------------------')
print(classification_report(y_test, y_test_hat))

print('Roc_auc score')
print('-------------------------------------------------------')
print(roc_auc_score(y_test, y_test_hat))
print('')

print('Confusion matrix')
print('-------------------------------------------------------')
print(confusion_matrix(y_test, y_test_hat))



from xgboost import XGBClassifier

model = XGBClassifier(random_state=123)
model.fit(X_train, y_train)
y_train_hat = model.predict(X_train)
y_test_hat = model.predict(X_test)
```

```
print(model)
print('Train performance')
print('-------------------------------------------------------')
print(classification_report(y_train, y_train_hat))

print('Test performance')
print('-------------------------------------------------------')
print(classification_report(y_test, y_test_hat))

print('Roc_auc score')
print('-------------------------------------------------------')
print(roc_auc_score(y_test, y_test_hat))
print('')

print('Confusion matrix')
print('-------------------------------------------------------')
print(confusion_matrix(y_test, y_test_hat))
```

```
# Extra Trees
from sklearn.ensemble import ExtraTreesClassifier

model = ExtraTreesClassifier(random_state=123)
model.fit(X_train, y_train)
y_train_hat = model.predict(X_train)
y_test_hat = model.predict(X_test)

print(model)
print('Train performance')
print('-------------------------------------------------------')
print(classification_report(y_train, y_train_hat))

print('Test performance')
print('-------------------------------------------------------')
print(classification_report(y_test, y_test_hat))

print('Roc_auc score')
print('-------------------------------------------------------')
print(roc_auc_score(y_test, y_test_hat))
print('')

print('Confusion matrix')
print('-------------------------------------------------------')
print(confusion_matrix(y_test, y_test_hat))
```

Voting - we will use these 3 models in final, merge them

```
from sklearn.model_selection import cross_validate
from sklearn.ensemble import VotingClassifier

votes = [
    ('rf', RandomForestClassifier(n_jobs=-1, criterion='gini',
max_leaf_nodes=None, min_samples_leaf=1, min_samples_split=2,
n_estimators=100)),
('xgb', XGBClassifier(n_jobs=-1, base_score=0.2, booster='gbtree', gamma=0,
learning_rate=0.1, n_estimators=500, reg_alpha=0, reg_lambda=1)),
    ('xt', ExtraTreesClassifier(n_jobs=-1, criterion='gini',
max_leaf_nodes=None, min_samples_leaf=1, min_samples_split=2,
n_estimators=500))
]

# votes = [
#     ('rf', gs1.best_estimator_),
#     ('xgb', gs2.best_estimator_),
#     ('xt', gs3.best_estimator_)
# ]

# soft voting based on weights
votesClass = VotingClassifier(estimators=votes, voting='soft', n_jobs=-1)
votesClass_cv = cross_validate(votesClass, X_train, y_train, cv=KFold(3))
votesClass.fit(X_train, y_train)

votesClass_cv


from sklearn.ensemble import ExtraTreesClassifier

model = votesClass
#model.fit(X_train, y_train)
y_train_hat = model.predict(X_train)
y_test_hat = model.predict(X_test)

print(model)
print('Train performance')
print('---------------------------------------------------------')
print(classification_report(y_train, y_train_hat))

print('Test performance')
print('-------------------------------------------------------')
print(classification_report(y_test, y_test_hat))

print('Roc_auc score')
print('-------------------------------------------------------')
print(roc_auc_score(y_test, y_test_hat))
print('')

print('Confusion matrix')
print('-------------------------------------------------------')
print(confusion_matrix(y_test, y_test_hat))
```

7.2 **Features 2**

IBM Cloud

IBM Cloud Paks are software products for hybrid clouds that enable you to develop apps once and deploy them anywhere. Virtual Private Cloud (VPC) is available as a public cloud service that lets you establish your own private cloud-like computing environment on shared public cloud infrastructure.

9 RESULTS

9.1 **Performance Metrics**

In a classification problem, the category or classes of data is identified based on training data. The model learns from the given dataset and then classifies the new data into classes or groups based on the training. It predicts class labels as the output, such as Yes or No, 0 or 1, Spam or Not Spam, etc. To evaluate the performance of a classification model, different metrics are used, and some of them are as follows:

- Accuracy
- Confusion Matrix
- Precision
- Recall
- F-Score
- AUC(Area Under the Curve)-ROC

```
Train performance
-------------------------------------------------------
              precision    recall  f1-score   support

           1       1.00      1.00      1.00       313
           2       1.00      1.00      1.00       311

    accuracy                           1.00       624
   macro avg       1.00      1.00      1.00       624
weighted avg       1.00      1.00      1.00       624

Test performance
-------------------------------------------------------
              precision    recall  f1-score   support

           1       0.92      0.76      0.83       103
           2       0.80      0.93      0.86       105

    accuracy                           0.85       208
   macro avg       0.86      0.85      0.84       208
weighted avg       0.86      0.85      0.84       208

Roc_auc score
```

```
-------------------------------------------------------
0.8453074433656957

Confusion matrix
-------------------------------------------------------
[[78 25]
 [ 7 98]]
```

10 ADVANTAGES & DISADVANTAGES

Advantages

1. Accurate prediction.
2.Prior prediction.
3.Doesn't lead to severe issues.
4.Prediction takes short duration of time.

Disadvantages

1. Data acquation-Massive Dataset needed
2.Time and Resource - Statistical Machine Learning Approaches to Liver Disease Prediction needs enough time to let the algorithms learn and develop enough to fulfill their purpose with a considerable amount of accuracy and relevancy
3.Interpretation of Results -Another major challenge is the ability to accurately interpret results generated by the algorithms. You must also carefully choose the algorithms for your purpose.

11 CONCLUSION

This project, "Statistical Machine Learning Approaches To Liver Disease Prediction" helps in early prediction of liver disease using classification algorithms by concentrating on relationships between a key list of liver enzymes, proteins, age and gender using them to try and predict the likeliness of liver disease.

12 FUTURE SCOPE

Diseases related to liver and heart are becoming more and more common with time. With continuous technological advancements, these are only going to increase in the future.

Although people are becoming more conscious of health nowadays and are joining yoga classes, dance classes; still the sedentary lifestyle and luxuries that are continuously being introduced and enhanced; the problem is going to last long.

So, in such a scenario, our project will be extremely helpful to the society. With the dataset that we used for this project, we got 85 % accuracy for RandomForest classifier,Xgboost, Extratrees models, and though it might be difficult to get such accuracies with very large datasets, from this projects results, one can clearly conclude that we can predict the risk of liver diseases with accuracy of 80% or more.

Today almost everybody above the age of 12 years has smartphones with them, and so we can incorporate these solutions into an android app or ios app. Also it can be incorporated into a website and these app and website will be highly beneficial for a large section of society

13 APPENDIX

13.1 **Source Code**

Import Libraries :

```
from sklearn.linear_model import LogisticRegression
from sklearn.neural_network import MLPClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.svm import SVC
from sklearn.ensemble import RandomForestClassifier

from sklearn.metrics import classification_report, confusion_matrix,
roc_auc_score

# Random Forest
model = RandomForestClassifier(n_jobs=-1,random_state=123)
model.fit(X_train, y_train)
y_train_hat = model.predict(X_train)
y_test_hat = model.predict(X_test)

print(model)
print('Train performance')
print('--------------------------------------------------------')
print(classification_report(y_train, y_train_hat))

print('Test performance')
print('--------------------------------------------------------')
print(classification_report(y_test, y_test_hat))

print('Roc_auc score')
print('--------------------------------------------------------')
print(roc_auc_score(y_test, y_test_hat))
print('')

print('Confusion matrix')
print('--------------------------------------------------------')
print(confusion_matrix(y_test, y_test_hat))



from xgboost import XGBClassifier

model = XGBClassifier(random_state=123)
model.fit(X_train, y_train)
y_train_hat = model.predict(X_train)
y_test_hat = model.predict(X_test)

print(model)
print('Train performance')
print('--------------------------------------------------------')
print(classification_report(y_train, y_train_hat))

print('Test performance')
print('--------------------------------------------------------')
print(classification_report(y_test, y_test_hat))

print('Roc_auc score')
print('--------------------------------------------------------')
print(roc_auc_score(y_test, y_test_hat))
print('')

print('Confusion matrix')
print('--------------------------------------------------------')
print(confusion_matrix(y_test, y_test_hat))

# Extra Trees
```

```
from sklearn.ensemble import ExtraTreesClassifier

model = ExtraTreesClassifier(random_state=123)
model.fit(X_train, y_train)
y_train_hat = model.predict(X_train)
y_test_hat = model.predict(X_test)

print(model)
print('Train performance')
print('-------------------------------------------------------')
print(classification_report(y_train, y_train_hat))

print('Test performance')
print('-------------------------------------------------------')
print(classification_report(y_test, y_test_hat))

print('Roc_auc score')
print('-------------------------------------------------------')
print(roc_auc_score(y_test, y_test_hat))
print('')

print('Confusion matrix')
print('-------------------------------------------------------')
print(confusion_matrix(y_test, y_test_hat))
```

Voting - we will use these 3 models in final, merge them

```
from sklearn.model_selection import cross_validate
from sklearn.ensemble import VotingClassifier

votes = [
    ('rf', RandomForestClassifier(n_jobs=-1, criterion='gini',
max_leaf_nodes=None, min_samples_leaf=1, min_samples_split=2,
n_estimators=100)),
    ('xgb', XGBClassifier(n_jobs=-1, base_score=0.2, booster='gbtree', gamma=0,
learning_rate=0.1, n_estimators=500, reg_alpha=0, reg_lambda=1)),
    ('xt', ExtraTreesClassifier(n_jobs=-1, criterion='gini',
max_leaf_nodes=None, min_samples_leaf=1, min_samples_split=2,
n_estimators=500))
]

# votes = [
#     ('rf', gs1.best_estimator_),
#     ('xgb', gs2.best_estimator_),
#     ('xt', gs3.best_estimator_)
# ]

# soft voting based on weights
votesClass = VotingClassifier(estimators=votes, voting='soft', n_jobs=-1)
votesClass_cv = cross_validate(votesClass, X_train, y_train, cv=KFold(3))
votesClass.fit(X_train, y_train)

votesClass_cv



from sklearn.ensemble import ExtraTreesClassifier

model = votesClass
#model.fit(X_train, y_train)
y_train_hat = model.predict(X_train)
y_test_hat = model.predict(X_test)

print(model)
print('Train performance')
print('-------------------------------------------------------')
print(classification_report(y_train, y_train_hat))

print('Test performance')
print('-------------------------------------------------------')
print(classification_report(y_test, y_test_hat))

print('Roc_auc score')
print('-------------------------------------------------------')
print(roc_auc_score(y_test, y_test_hat))
print('')

print('Confusion matrix')
print('-------------------------------------------------------')
print(confusion_matrix(y_test, y_test_hat))
```

app.py

```
import os

import numpy as np #used for numerical analysis
```

```
from flask import Flask,request,render_template # Flask-It is our framework
which we are going to use to run/serve our application.

#request-for accessing file which was uploaded by the user on our application.

#render_template- used for rendering the html pages


from werkzeug.datastructures import ImmutableMultiDict

import pickle

from sklearn.preprocessing import RobustScaler


app=Flask(__name__)#our flask app

model=pickle.load(open("./model/model.pkl","rb"))#loading the model

@app.route("/") #default route

def about():

    return render_template("about.html")#rendering html page


@app.route("/about") #route about page

def home():

    return render_template("about.html")#rendering html page


@app.route("/info") # route for info page

def information():

    return render_template("info.html")#rendering html page


@app.route("/classify") # route for uploads

def test():

    return render_template("index6.html")#rendering html page
```

```
@app.route("/predict",methods=["GET","POST"]) #route for our prediction

def upload():

    if request.method=='POST':

        print("hi")

        print(request.form)

        # imd = ImmutableMultiDict(request.form)

        res = request.form.to_dict(flat=False)

        print(res)

        Age = int(res['Age'][0])

        Gender = res['gender'][0]

        if(Gender == 'male'):

            Gender = 0

        else:

            Gender = 1

        Total_Bilirubin = int(res['Total_Bilirubin'][0])

        Direct_Bilirubin = int(res['Direct_Bilirubin'][0])

        Alkaline_Phosphotase = int(res['Alkaline_Phosphotase'][0])

        Alamine_Aminotransferase = int(res['Alamine_Aminotransferase'][0])

        Aspartate_Aminotransferase = int(res['Aspartate_Aminotransferase'][0])

        Total_Protiens = float(res['Total_Protiens'][0])

        Albumin = float(res['Albumin'][0])

        Albumin_and_Globulin_Ratio = float(res['Albumin_and_Globulin_Ratio'][0])

        arr =
[Age,Gender,Total_Bilirubin,Alkaline_Phosphotase,Alamine_Aminotransferase,Albumi
n_and_Globulin_Ratio]
```

```
        print(arr)

        pred = model.predict([arr])

        pred = pred[0]

        print("prediction",pred)#printing the prediction

        if(pred==0):#checking the results

            result="UnInfected"

        else:

            result="Infected"

        return result#resturing the result

    return None


#port = int(os.getenv("PORT"))

if __name__=="__main__":

    app.run(debug=False)#running our app

    #app.run(host='0.0.0.0', port=8000,debug=False)
```

app_ibm.py

```
import os

import numpy as np #used for numerical analysis

from flask import Flask,request,render_template # Flask-It is our framework
which we are going to use to run/serve our application.

#request-for accessing file which was uploaded by the user on our application.

#render_template- used for rendering the html pages


from werkzeug.datastructures import ImmutableMultiDict

import pickle
```

```
from sklearn.preprocessing import RobustScaler


import requests


# NOTE: you must manually set API_KEY below using information retrieved from
your IBM Cloud account.

API_KEY = "j0BgO8c07rllOiKYd-fzN6fUKss9_b9Sd633gGNKCTgx"

token_response = requests.post('https://iam.cloud.ibm.com/identity/token',
data={"apikey":

 API_KEY, "grant_type": 'urn:ibm:params:oauth:grant-type:apikey'})

mltoken = token_response.json()["access_token"]


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


app=Flask(__name__)#our flask app

#model=pickle.load(open("./model/model.pkl","rb"))#loading the model

@app.route("/") #default route

def about():

    return render_template("about.html")#rendering html page


@app.route("/about") #route about page

def home():

    return render_template("about.html")#rendering html page


@app.route("/info") # route for info page
```

```
def information():

    return render_template("info.html")#rendering html page


@app.route("/classify") # route for uploads

def test():

    return render_template("index6.html")#rendering html page


@app.route("/predict",methods=["GET","POST"]) #route for our prediction

def upload():

    if request.method=='POST':

        print(request.form)

        # imd = ImmutableMultiDict(request.form)

        res = request.form.to_dict(flat=False)

        print(res)

        Age = int(res['Age'][0])

        Gender = res['gender'][0]

        if(Gender == 'male'):

            Gender = 0

        else:

            Gender = 1

        Total_Bilirubin = int(res['Total_Bilirubin'][0])

        Direct_Bilirubin = int(res['Direct_Bilirubin'][0])

        Alkaline_Phosphotase = int(res['Alkaline_Phosphotase'][0])

        Alamine_Aminotransferase = int(res['Alamine_Aminotransferase'][0])

        Aspartate_Aminotransferase = int(res['Aspartate_Aminotransferase'][0])
        Total_Protiens = float(res['Total_Protiens'][0])

        Albumin = float(res['Albumin'][0])

        Albumin_and_Globulin_Ratio = float(res['Albumin_and_Globulin_Ratio'][0])

        arr =
[[Age,Gender,Total_Bilirubin,Alkaline_Phosphotase,Alamine_Aminotransferase,Album
in_and_Globulin_Ratio]]


        payload_scoring = {"input_data": [{"fields":
[['Age','Gender','Total_Bilirubin','Alkaline_Phosphotase','Alamine_Aminotransfer
ase','Albumin_and_Globulin_Ratio']], "values": arr}]}


        response_scoring = requests.post('https://us-
south.ml.cloud.ibm.com/ml/v4/deployments/61158191-4047-4d6d-ae8a-
b137b0c8c2d2/predictions?version=2022-11-14', json=payload_scoring,

        headers={'Authorization': 'Bearer ' + mltoken})


        print(response_scoring)

        predictions = response_scoring.json()

        predict = predictions['predictions'][0]['values'][0][0]

        print("Final prediction :",predict)


    # showing the prediction results in a UI# showing the prediction results in
a UI

        return render_template('predict.html', predict=predict)


if __name__ == '__main__' :

    app.run(debug= False)
```

13.2 **Github link & Poject Demo Link**

Github : https://github.com/IBM-EPBL/IBM-Project-26409-1660026280
Project Demo Link:
https://drive.google.com/file/d/1OcqksN7kTPzpViiXTqVns1NdoxiFE0Eh/view?usp=drivesdk