PROJECT REPORT

OVERVIEW :

The term “liver disease” refers to any of several conditions that can affect and damage your liver. Over time, liver disease can cause cirrhosis (scarring). As more scar tissue replaces healthy liver tissue, the liver can no longer function properly.

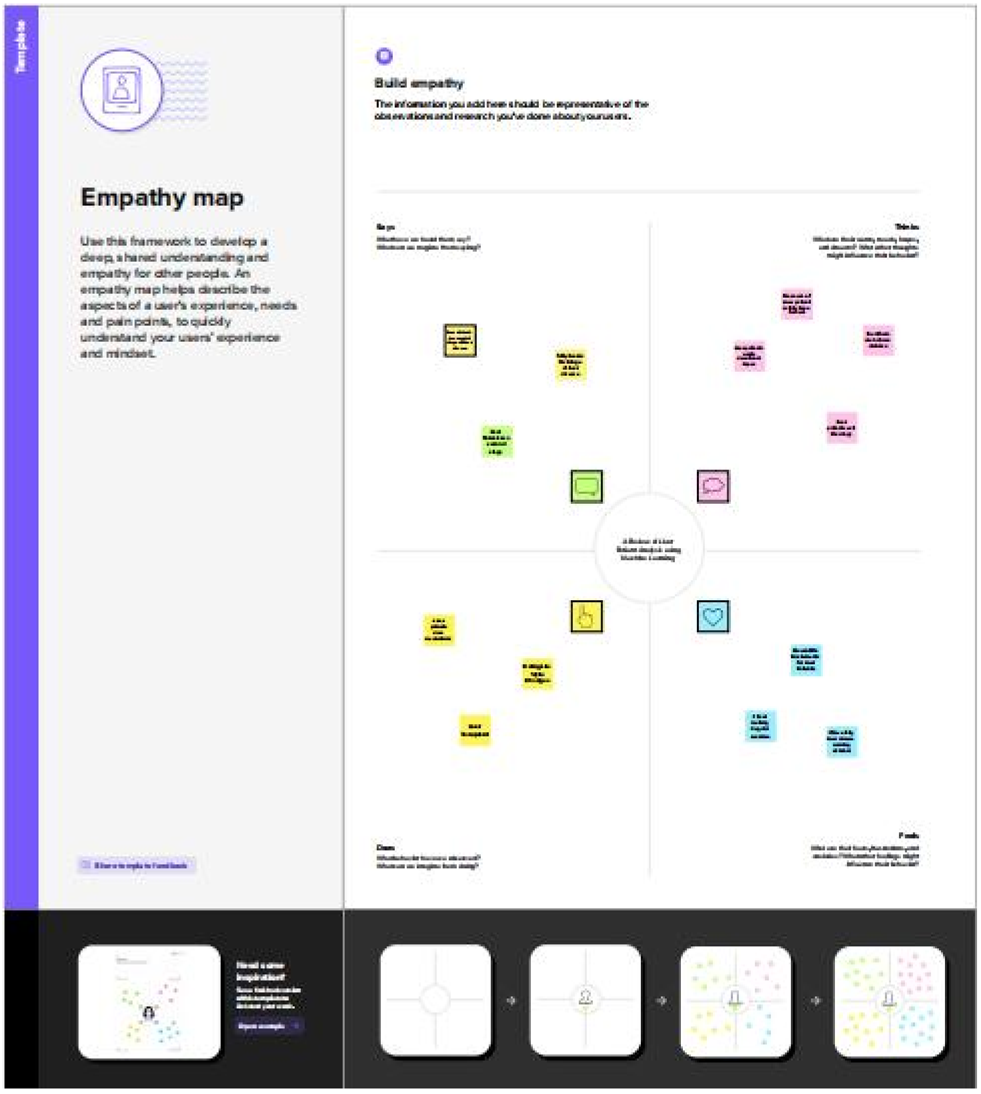
PURPOSE :

The main objective of this research work is to use classification algorithms to predict liver diseases. Naïve Bayes and support vector machine (SVM) are the algorithms used in this work. The algorithms are classify the factors on performance.

PROBLEM DEFINITION AND DESIGN THINKING :

A design thinking problem statement is a concise and actionable sentence or question that defines your UX purpose and direction. Product teams using design thinking develop problem statements to simplify complex problems and identify the gap between what your product has and what your users need.

EMPATHY MAP:



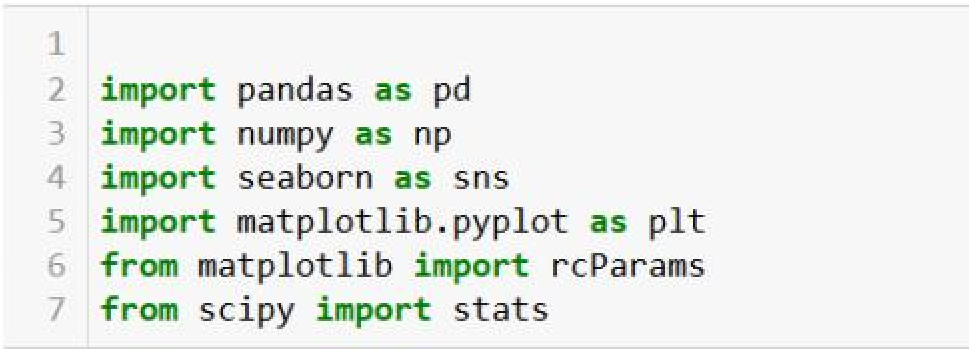
Use this framework to develop a deep, shared understanding and empathy for other people. An empathy map helps describe the aspects of a user's experience, needs and pain points, to quickly understand your users' experience and mindset.

Build empathy

The information you add here should be representative of the observations and research you've done about your users.

BRAINSTROM :

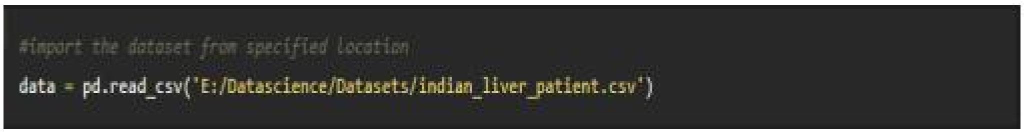
IMPORT LIBRARIES :



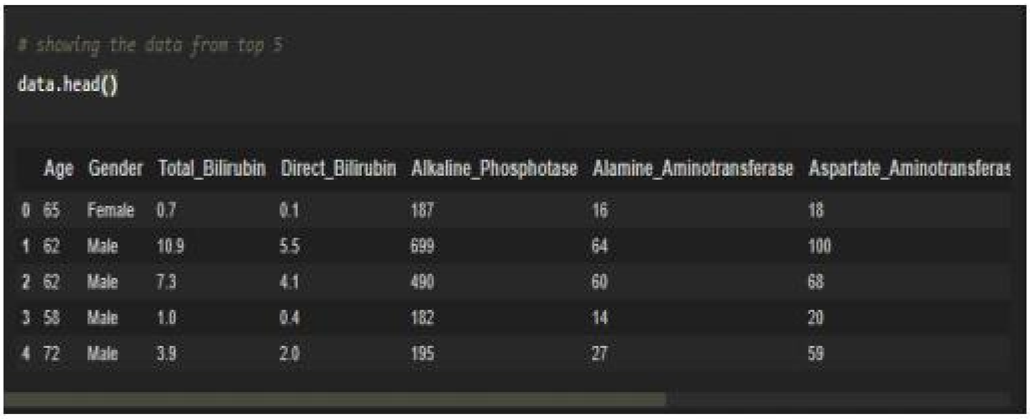
```python

import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from matplotlib import rcParams
from scipy import stats
```

READ DATASET :



```python
#import the dataset from specified location
data = pd.read_csv('E:/Datascience/Datasets/indian_liver_patient.csv')
```



```python
# showing the data from top 5
data.head()
```

```
   Age  Gender  Total_Bilirubin  Direct_Bilirubin  Alkaline_Phosphotase  Alamine_Aminotransferase  Aspartate_Aminotransferas
0  65   Female  0.7              0.1               187                   16                        18
1  62   Male    10.9             5.5               699                   64                        100
2  62   Male    7.3              4.1               490                   60                        68
3  58   Male    1.0              0.4               182                   14                        20
4  72   Male    3.9              2.0               195                   27                        59
```

Descriptive statistical :

Descriptive analysis is to study the basic features of data with the statistical process. Here pandas has a worthy function called describe. With this describe function we can understand the unique, top and frequent values of categorical features. And we can find mean, std, min, max and percentile values of continuous features.

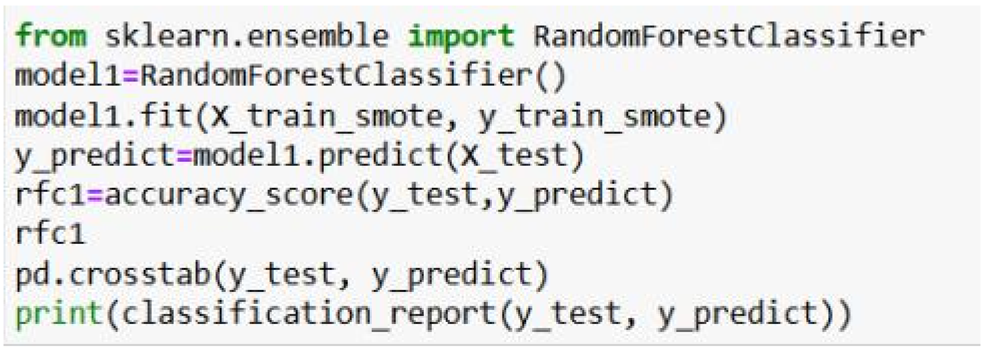
Visual analysis :

Visual analysis is the process of using visual representations, such as charts, plots, and graphs, to explore and understand data. It is a way to quickly identify patterns, trends, and outliers in the data, which can help to gain insights and make informed decisions.

Random Forest model :

A function named RandomForestClassifier is imported and train and test data are passed as the parameters. Inside the function, RandomForestClassifier algorithm is initialised and training data is passed to the model with .fit() function.

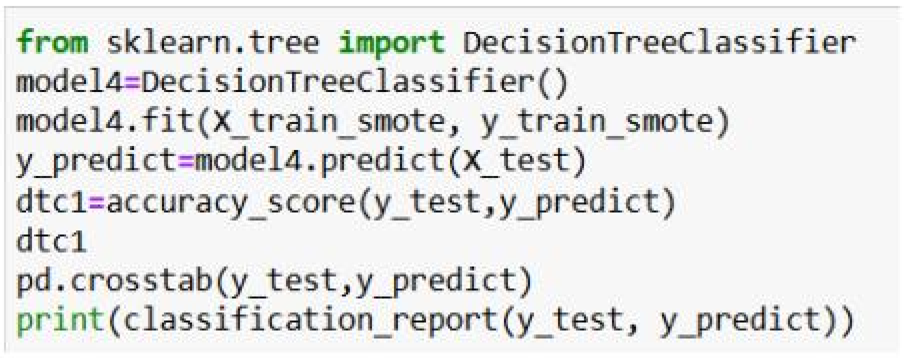
Test data is predicted with .predict() function and saved in a new variable. For evaluating the model, a confusion matrix and classification report is done**.**



```
from sklearn.ensemble import RandomForestClassifier
model1=RandomForestClassifier()
model1.fit(X_train_smote, y_train_smote)
y_predict=model1.predict(X_test)
rfc1=accuracy_score(y_test,y_predict)
rfc1
pd.crosstab(y_test, y_predict)
print(classification_report(y_test, y_predict))
```

Decision tree model :

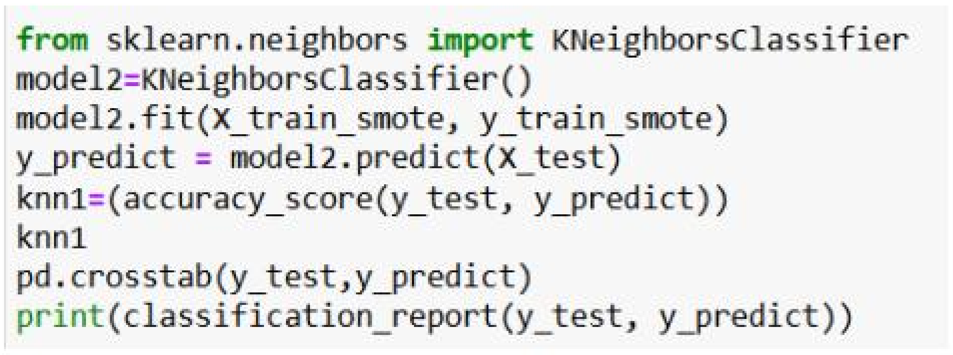
A function named DecisionTreeClassifier is imported and train and test data are passed as the parameters. Inside the function, DecisionTreeClassifier algorithm is initialised and training data is passed to the model with the .fit() function. Test data is predicted with .predict() function and saved in a new variable. For evaluating the model, a confusion matrix and classification report is done



```
from sklearn.tree import DecisionTreeClassifier
model4=DecisionTreeClassifier()
model4.fit(X_train_smote, y_train_smote)
y_predict=model4.predict(X_test)
dtc1=accuracy_score(y_test,y_predict)
dtc1
pd.crosstab(y_test,y_predict)
print(classification_report(y_test, y_predict))
```

KNN model :

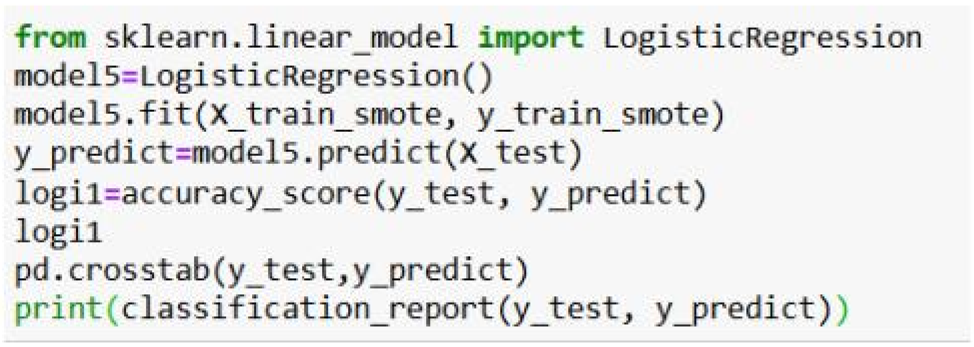
A function named K KNeighborsClassifier is imported and train and test data are passed as the parameters. Inside the function, KNeighborsClassifier algorithm is initialised and training data is passed to the model with .fit() function. Test data is predicted with .predict() function and saved in new variable. For evaluating the model, confusion matrix and classification report is done.



```
from sklearn.neighbors import KNeighborsClassifier
model2=KNeighborsClassifier()
model2.fit(X_train_smote, y_train_smote)
y_predict = model2.predict(X_test)
knn1=(accuracy_score(y_test, y_predict))
knn1
pd.crosstab(y_test,y_predict)
print(classification_report(y_test, y_predict))
```

Logistic Regression model :

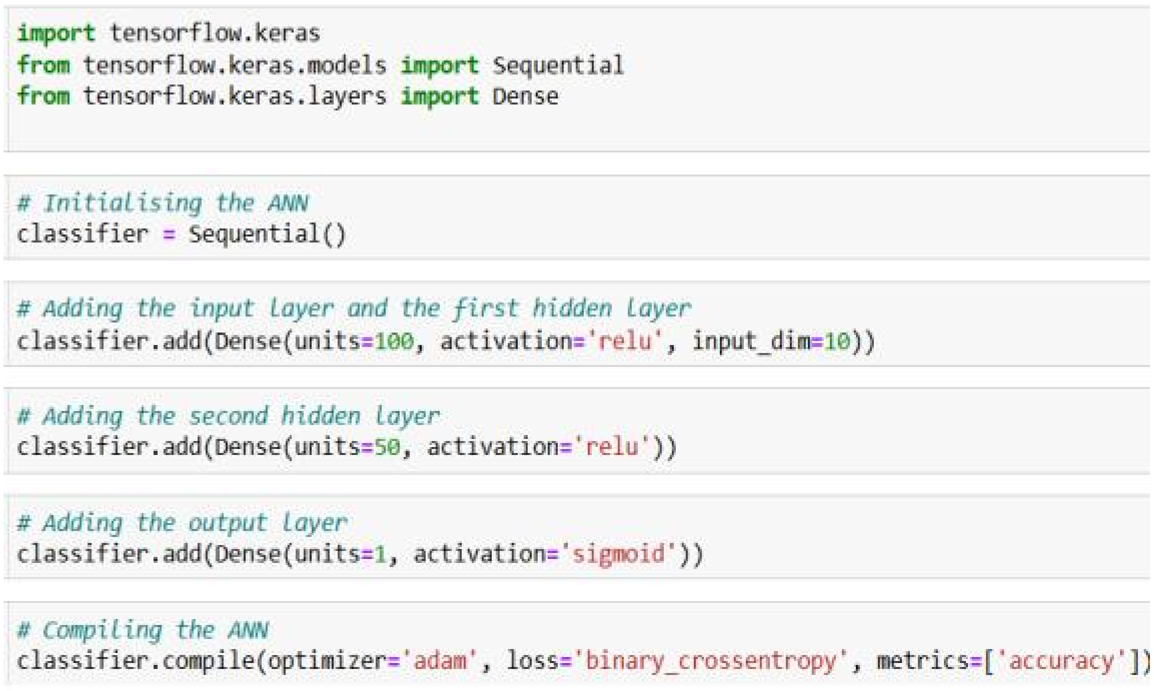
A function named Logistic Regression is imported and train and test data are passed as the parameters. Inside the function, Logistic Regression algorithm is initialised and training data is passed to the model with .fit() function. Test data is predicted with .predict() function and saved in new variable. For evaluating the model, confusion matrix and classification report is done



```
from sklearn.linear_model import LogisticRegression
model5=LogisticRegression()
model5.fit(X_train_smote, y_train_smote)
y_predict=model5.predict(X_test)
logi1=accuracy_score(y_test, y_predict)
logi1
pd.crosstab(y_test,y_predict)
print(classification_report(y_test, y_predict))
```

ANN model :

Building and training an Artificial Neural Network (ANN) using the Keras library with TensorFlow as the backend. The ANN is initialised as an instance of the Sequential class, which is a linear stack of layers. Then, the input layer and two hidden layers are added to the model using the Dense class, where the number of units and activation function are specified. The output layer is also added using the Dense class with a sigmoid activation function. The model is then compiled with the Adam optimizer, binary cross-entropy loss function, and accuracy metric. Finally, the model is fit to the training data with a batch size of 100, 20% validation split, and 100 epochs.



```
import tensorflow.keras
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense
```

```
# Initialising the ANN
classifier = Sequential()
```

```
# Adding the input layer and the first hidden layer
classifier.add(Dense(units=100, activation='relu', input_dim=10))
```

```
# Adding the second hidden layer
classifier.add(Dense(units=50, activation='relu'))
```

```
# Adding the output layer
classifier.add(Dense(units=1, activation='sigmoid'))
```

```
# Compiling the ANN
classifier.compile(optimizer='adam', loss='binary_crossentropy', metrics=['accuracy'])
```

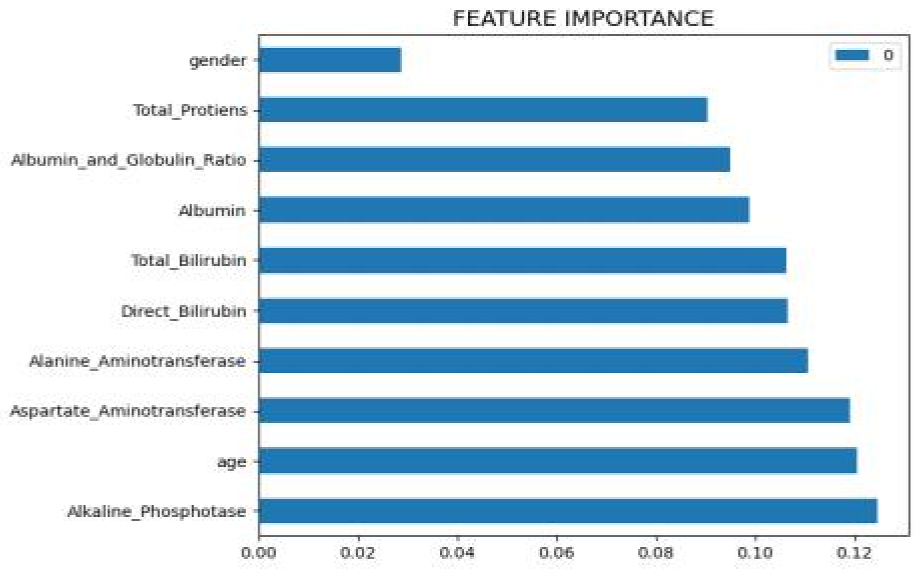
Identifying Important Features :

10 attributes are passed to predict the actucal outcome, Its necessary to identify the l important feature to determin the output. Here we are using function called feature_importance to identify the important features

among the available attributes and understand with a visualization

```
dd.plot(kind='barh', figsize=(7,6))
plt.title("FEATURE IMPORTANCE",fontsize=14)
```

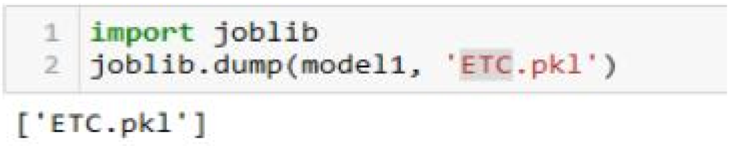
```
Text(0.5, 1.0, 'FEATURE IMPORTANCE')
```



Direct_Bilirubin & Total_Bilirubin are the most important features to predict the outcome

Save the best model :

Saving the best model after comparing its performance using different evaluation metrics means selecting the model with the highest performance and saving its weights and configuration. This can be useful in avoiding the need to retrain the model every time it is needed and also to be able to use it in the future.

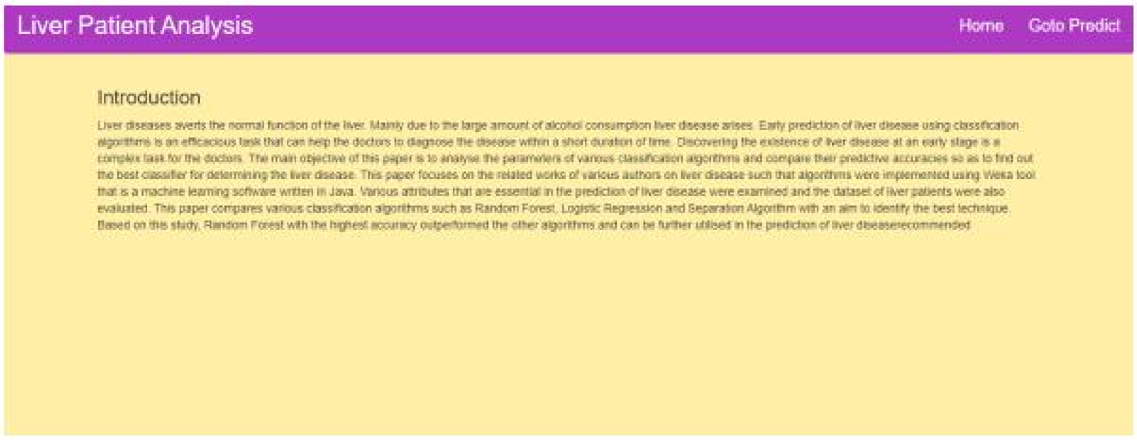


```
import joblib
joblib.dump(model1, 'ETC.pkl')
```

```
['ETC.pkl']
```

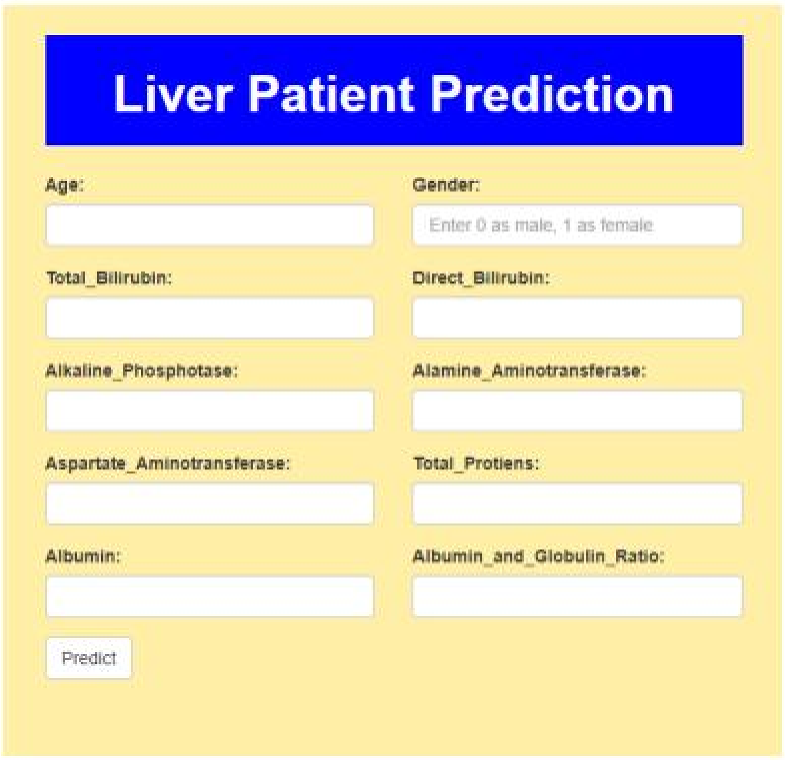
Building Html Pages:

- home.html
- predict.html



Introduction

Liver diseases averts 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 paper 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. This paper focuses on the related works of various authors on liver disease such that algorithms were implemented using Weka tool that is a machine learning software written in Java. Various attributes that are essential in the prediction of liver disease were examined and the dataset of liver patients were also evaluated. This paper compares various classification algorithms such as Random Forest, Logistic Regression and Separation Algorithm with an aim to identify the best technique. Based on this study, Random Forest with the highest accuracy outperformed the other algorithms and can be further utilised in the prediction of liver diseaserecommended


Liver Patient Prediction

Age:

Gender: Enter 0 as male, 1 as female

Total_Bilirubin:

Direct_Bilirubin:

Alkaline_Phosphotase:

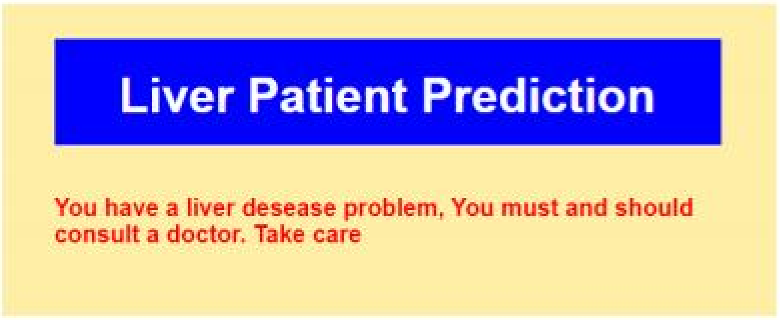
Alamine_Aminotransferase:

Aspartate_Aminotransferase:

Total_Protiens:

Albumin:

Albumin_and_Globulin_Ratio:


Liver Patient Prediction

You have a liver desease problem, You must and should consult a doctor. Take care

END