CONTENTS

1.INTRODUCTION

1.1 Overview

Prediction of the disease in the human being is the very long and difficult process in early days. Now a days, computer aided diagnosis is the important role in the medical industry for predicting, analyzing and storing medical information with the images. In this paper will discuss nand classify the liver patients with the help of the liver patient dataset with the help of the machine learning algorithms. WEKA is the software used here for implement the some of the classification algorithms with the data selected from the liver disease dataset. After the successful implementation of the all the algorithms, the best algorithms selected from the output of the all the algorithms execution.

The liver is an accessory digestive organ that produces bile, an alkaline compound which helps the breakdown of fat. Bile aids in digestion via the emulsification of lipids. The gall bladder, a small pouch that sits just under the liver, stores bile produced by the liver which is afterwards moved to the small intestine to complete digestion [1]. The liver's highly specialized tissue consisting of mostly hepatocytes regulates a wide variety of high-volume biochemical reactions, including the synthesis and breakdown of small and complex molecules, many of which are necessary for normal vital functions [2]. Estimates regarding the organ's total number of functions vary, but textbooks generally cite it being around 500[2]. The liver is a vital organ and supports almost every other organ in the body. Because of its strategic location and multidimensional functions, the liver is also prone to many diseases.

Liver is located in the right upper quadrant of the abdomen, below the diaphragm. Its other roles in metabolism include the regulation of glycogen storage, decomposition of red blood cells and the production of hormones.

Liver diagnosis at an early stage is essential for enhanced handling. In this study, an artificial neural network model was designed and developed using JustNN Tool for predicting weather a person is a liver patient or not based on a dataset for liver patients. The model was trained and validated, most important factors affecting Status of liver patient identified, and the accuracy for the validation was 99.00%.

1.2 Purpose

Liver cirrhosis is the biggest health problem posed by alcohol use, with 1.4 lakh deaths every year. Sadly, no. In fact, it is getting more common in younger people than ever before. Dr. Amrish said that liver disease can set in childhood too as it can pass through genes.Cirrhosis isn't curable, but it's treatable. Alcohol abuse, hepatitis, and fatty liver disease are some of the main causes.

Then you people will get answers like these as I mentioned above, So the purpose and inspiration of this project clearly simplifies the devastating answers from the data available with Google. We do need a system that in some stage reduces the burden on doctors, and today in this article I'll try to frame a practical logic that will help our healthcare system in a long run.

This data set contains 416 liver patient records and 167 non-liver patient records collected from North East of Andhra Pradesh, India. The "Dataset" column is a class label used to divide groups into a liver patient (liver disease) or not (no disease). This data set contains 441 male patient records and 142 female patient records.We have not started any data analysis yet, this is just to show you all the authenticity of the dataset.We can clearly see in the output as well as in the graph that, it is an imbalanced dataset, any patients diagnosed with liver disease are higher compared to the ones who are not diagnosed.

We can clearly see in the output as well as in the graph that, number of patient suffering from liver disease are higher in males than in females.

Here is another interactive plot() that shows, males are at higher risk of chronic liver diseases as compare to females.

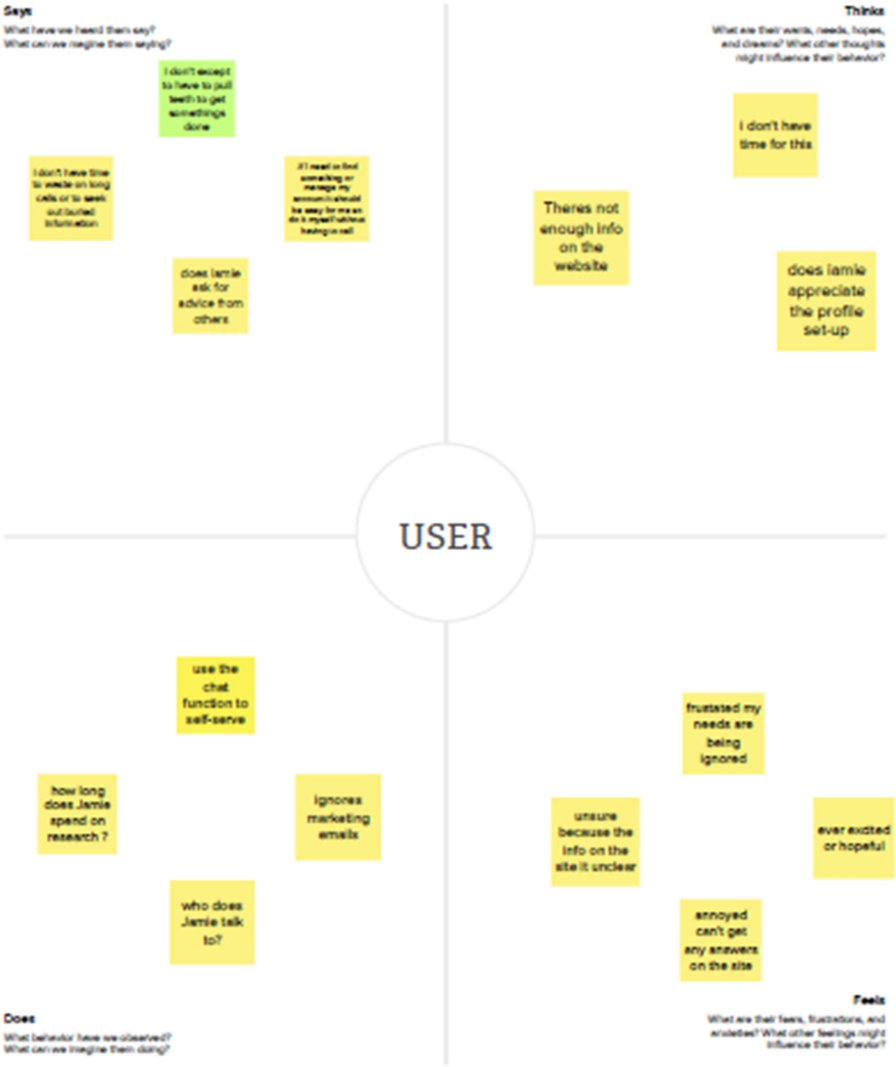
In this FacetGrid plot we are plotting two significant features(Alamine and Aspartate - Aminotransferase) along with Gender as a form of hue and it clearly shows that males are highly effective concerning these two features the most.

2. Problem Definition & Design Thinking

2.1 Empathy Map

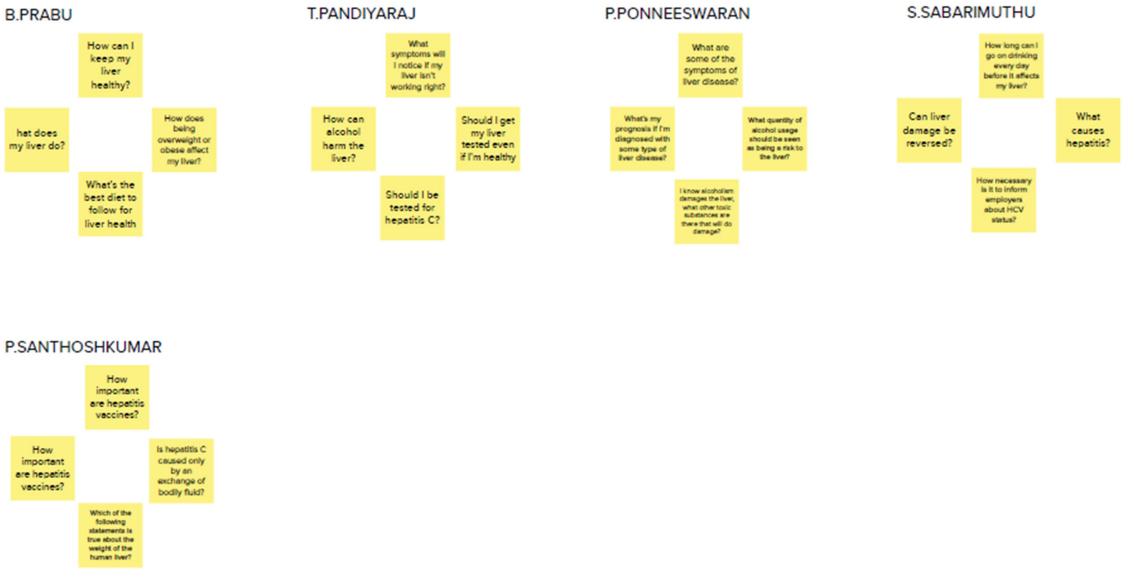
Build empathy

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

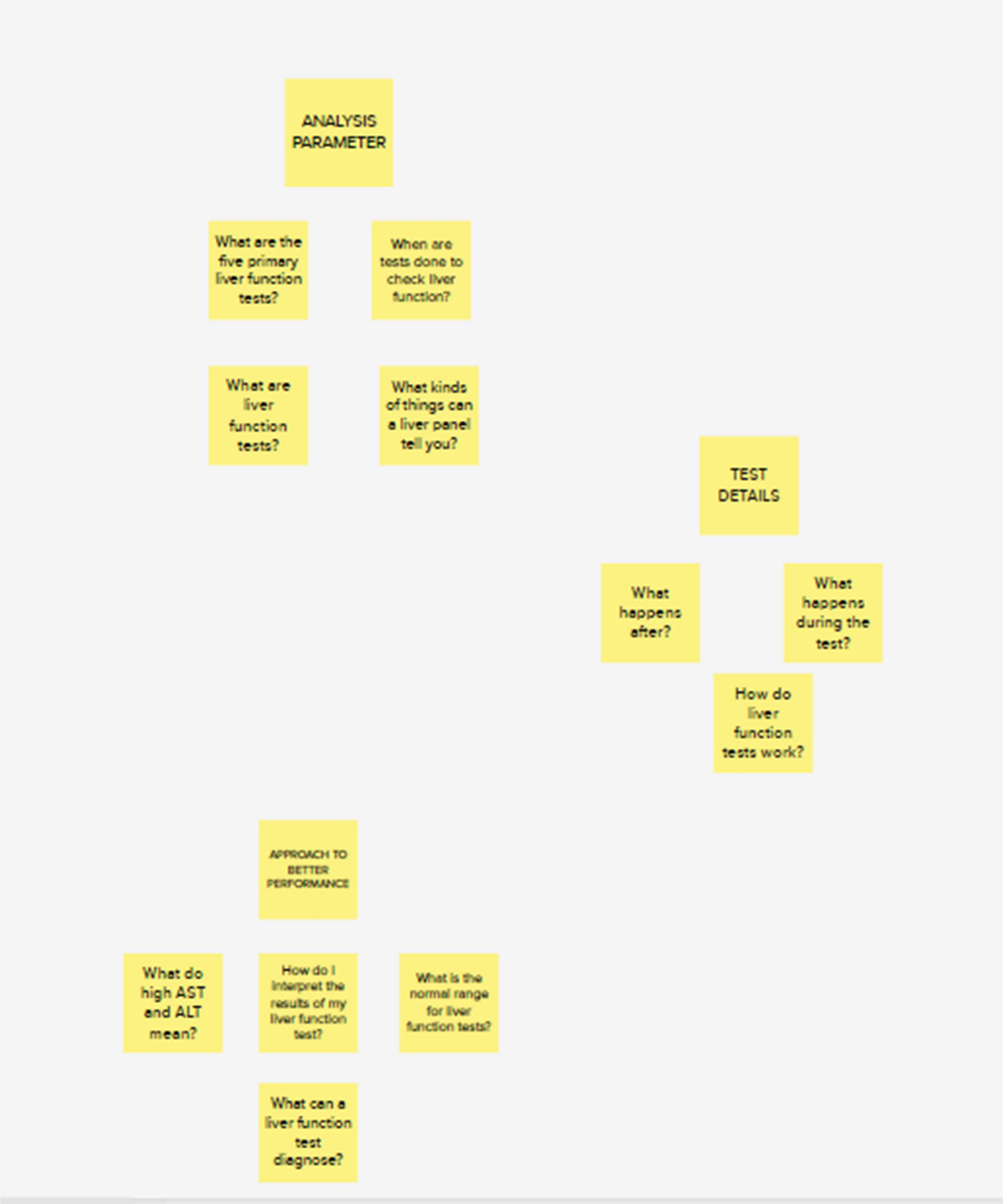


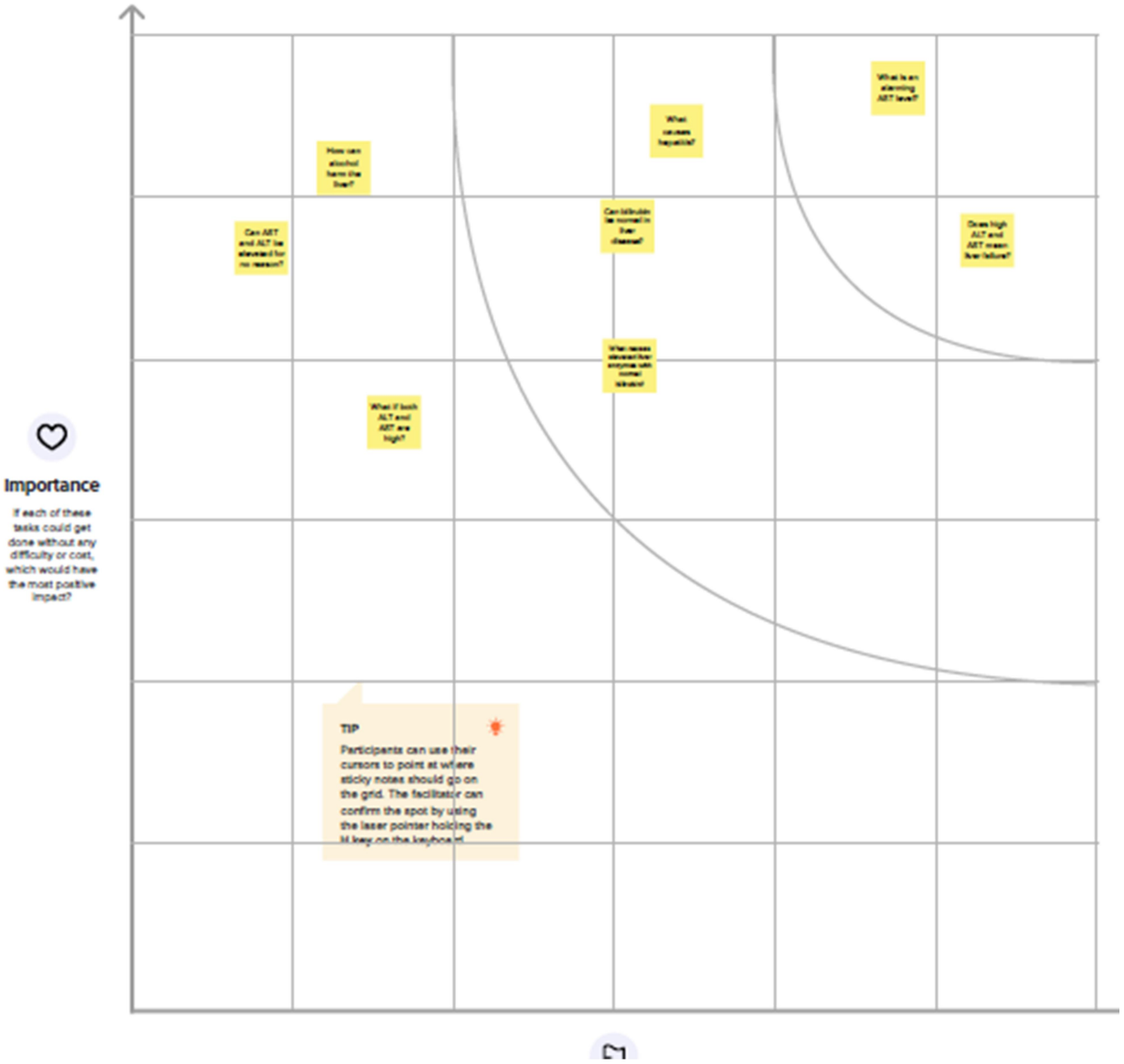
2.2 Ideation & Brainstorming Map

Brainstorm & idea prioritization Use this template in your own brainstorming sessions so your team can unleash their imagination and start shaping concepts even if you're not sitting in the same room. 2-8 people recommended 10 minutes to prepare 1 hour to collaborate Template Share template feedback Team gathering Define who should participate in the session and send an invite. Share relevant information or pre-work ahead. Set the goal Think about the problem you'll be focusing on solving in the brainstorming session. A B Learn how to use the facilitation tools Use the Facilitation Superpowers to run a happy and productive session. C Open article 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. 10 minutes Key rules of brainstorming.

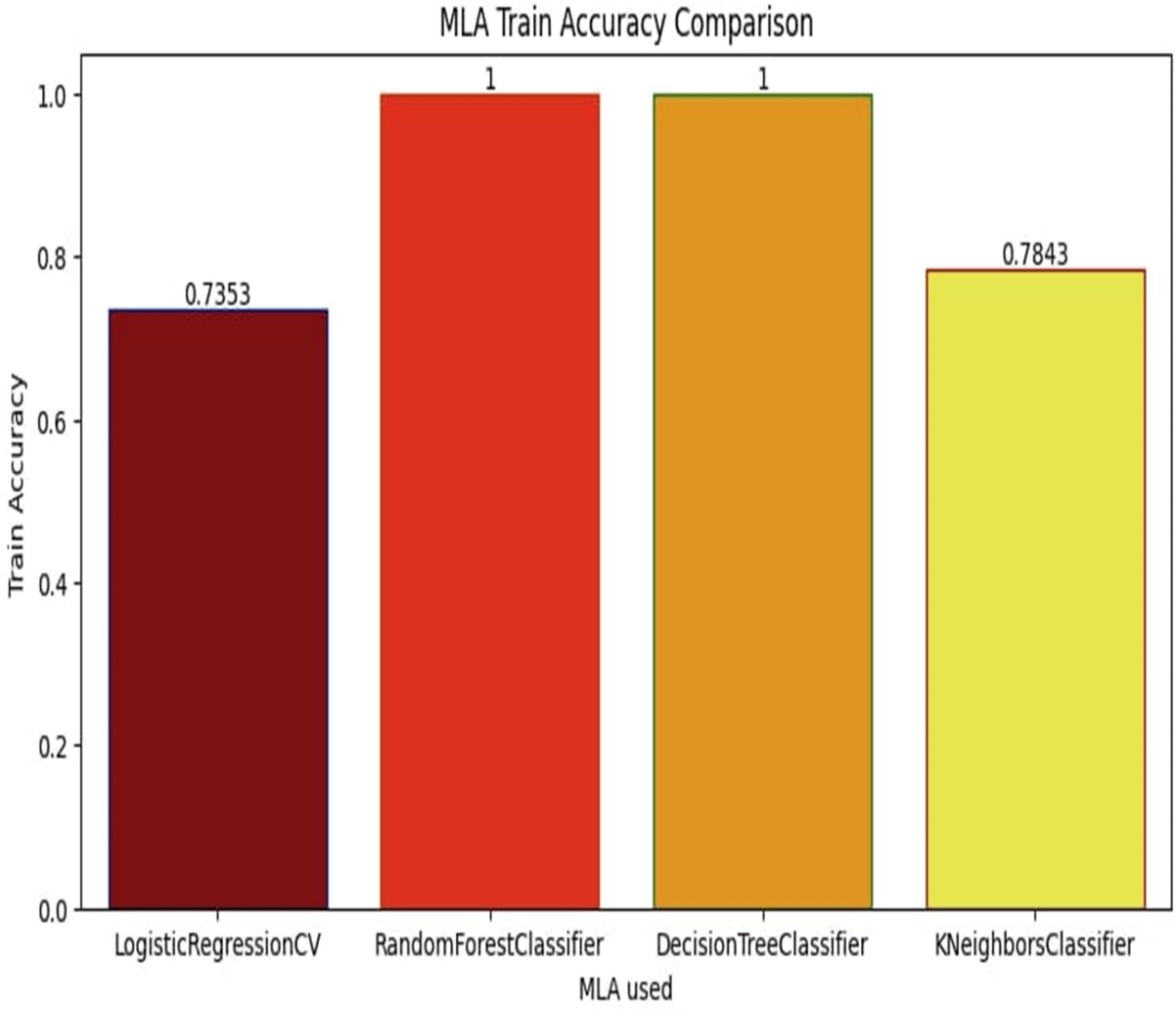


Take turns sharing your ideas while clustering similar or related notes as you go. Once all sticky notes have been grouped, give each cluster a sentence-like label. If a cluster is bigger than six sticky notes, try and see if you and break it up into smaller sub-groups.




How can alcohol harm the liver?
What causes hepatitis?
What is an alarming AST level?
Can AST and ALT be elevated for no reason?
Can bilirubin be normal in liver disease?
Does high ALT and AST mean liver failure?
What causes elevated liver enzymes with normal bilirubin?
What if both ALT and AST are high?
TIP
Participants can use their cursors to point at where sticky notes should go on the grid. The facilitator can confirm the spot by using the laser pointer holding the M key on the keyboard.
Importance
If each of these tasks could get done without any difficulty or cost, which would have the most positive impact?

3. RESULT


MLA Train Accuracy Comparison
Train Accuracy
1.0
0.8
0.6
0.4
0.2
0.0
0.7353
1
1
0.7843
LogisticRegressionCV
RandomForestClassifier
DecisionTreeClassifier
KNeighborsClassifier
MLA used

4.ADVANTAGES & DISADVANTAGES

Advantages of being a liver patient:

- Increased awareness of the importance of liver health
- Access to specialized medical care and treatments
- Support from patient advocacy organizations and support groups
- Improved lifestyle choices, such as avoiding alcohol and unhealthy foods
- Regular monitoring and testing to detect any changes or complications.

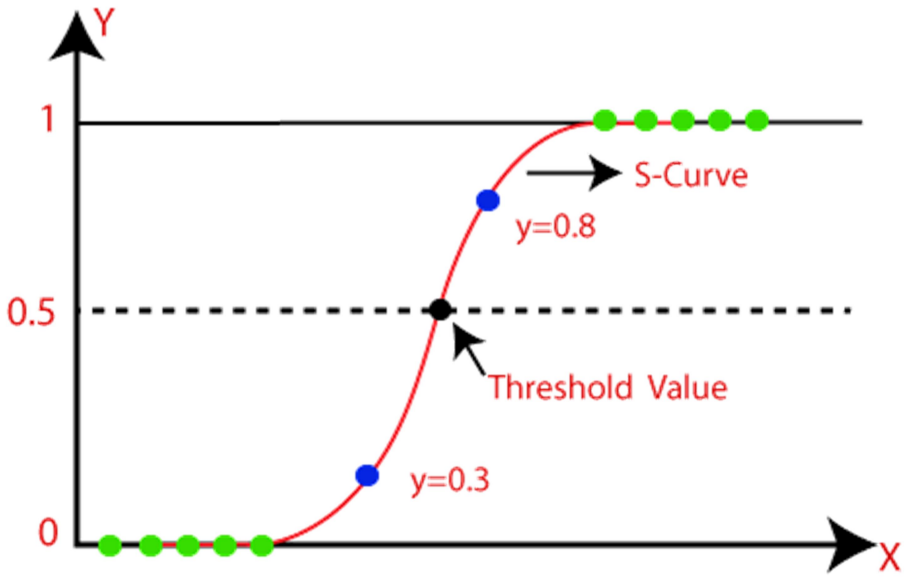
Disadvantages of being a liver patient:

- Reduced quality of life due to symptoms and restrictions on activities
- Increased risk of complications such as liver failure or cancer
- Dependence on medications and medical interventions
- Financial burden due to medical expenses and potentially needing to take time off work
- Emotional stress and anxiety related to living with a chronic condition.

5 .APPLICATIONS

5.1.LOGISTIC REGRESSION

- Logistic regression is one of the most popular Machine Learning algorithms, which comes under the Supervised Learning technique. It is used for predicting the categorical dependent variable using a given set of independent variables.
- Logistic regression predicts the output of a categorical dependent variable. Therefore the outcome must be a categorical or discrete value. It can be either Yes or No, 0 or 1, true or False, etc. but instead of giving the exact value as 0 and 1, it gives the probabilistic values which lie between 0 and 1.
- Logistic Regression is much similar to the Linear Regression except that how they are used. Linear Regression is used for solving Regression problems, whereas Logistic regression is used for solving the classification problems.
- In Logistic regression, instead of fitting a regression line, we fit an "S" shaped logistic function, which predicts two maximum values (0 or 1).
- The curve from the logistic function indicates the likelihood of something such as whether the cells are cancerous or not, a mouse is obese or not based on its weight, etc.
- Logistic Regression is a significant machine learning algorithm because it has the ability to provide probabilities and classify new data using continuous and discrete datasets.
- Logistic Regression can be used to classify the observations using different types of data and can easily determine the most effective variables used for the classification. The below image is showing the logistic function:

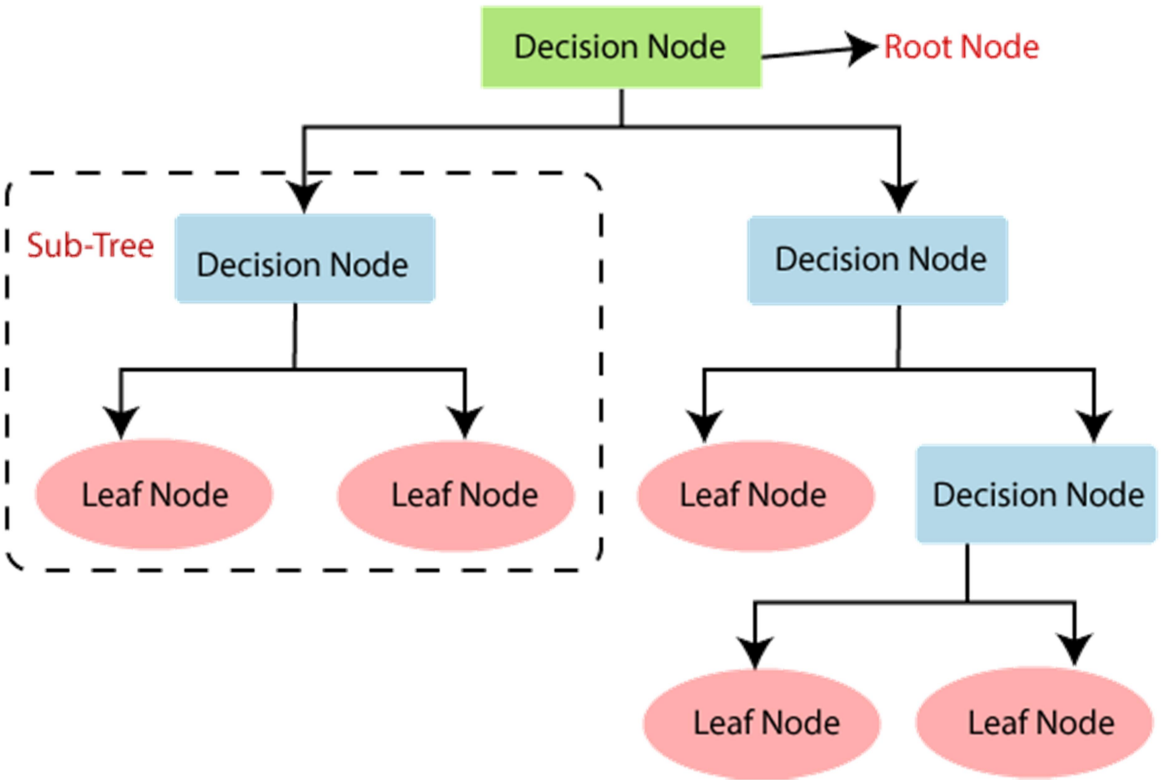

Y
1
0.5
0
S-Curve
y=0.8
Threshold Value
y=0.3
X

5.2. KNN

- K-Nearest Neighbour is one of the simplest Machine Learning algorithms based on Supervised Learning technique.
- K-NN algorithm assumes the similarity between the new case/data and available cases and put the new case into the category that is most similar to the available categories.
- K-NN algorithm stores all the available data and classifies a new data point based on the similarity. This means when new data appears then it can be easily classified into a well suite category by using K- NN algorithm.
- K-NN algorithm can be used for Regression as well as for Classification but mostly it is used for the Classification problems.
- K-NN is a non-parametric algorithm, which means it does not make any assumption on underlying data.
- It is also called a lazy learner algorithm because it does not learn from the training set immediately instead it stores the dataset and at the time of classification, it performs an action on the dataset.
- KNN algorithm at the training phase just stores the dataset and when it gets new data, then it classifies that data into a category that is much similar to the new data.
- Suppose, we have an image of a creature that looks similar to cat and dog, but we want to know either it is a cat or dog. So for this identification, we can use the KNN algorithm, as it works on a similarity measure. Our KNN model will find the similar features of the new data set to the cats and dogs images and based on the most similar features it will put it in either cat or dog category.

5.3.DECISION TREE

- Decision Tree is a Supervised learning technique that can be used for both classification and Regression problems, but mostly it is preferred for solving Classification problems. It is a tree-structured classifier, where internal nodes represent the features of a dataset, branches represent the decision rules and each leaf node represents the outcome.
- In a Decision tree, there are two nodes, which are the Decision Node and Leaf Node. Decision nodes are used to make any decision and have multiple branches, whereas Leaf nodes are the output of those decisions and do not contain any further branches.
- The decisions or the test are performed on the basis of features of the given dataset.
- It is a graphical representation for getting all the possible solutions to a problem/decision based on given conditions.
- It is called a decision tree because, similar to a tree, it starts with the root node, which expands on further branches and constructs a tree-like structure.
- In order to build a tree, we use the CART algorithm, which stands for Classification and Regression Tree algorithm.
- A decision tree simply asks a question, and based on the answer (Yes/No), it further split the tree into subtrees.
- Below diagram explains the general structure of a decision tree:


Decision Node
Root Node
Sub-Tree
Decision Node
Decision Node
Leaf Node
Leaf Node
Leaf Node
Decision Node
Leaf Node
Leaf Node

6 .CONCLUSION

As an AI language model, I cannot draw conclusions or provide analysis on a specific liver patient analysis without access to data and context. However, in general, machine learning can be used to analyze data related to liver disease and predict outcomes, such as the likelihood of developing liver cancer or the effectiveness of a particular treatment. Machine learning algorithms can also be used to identify patterns and correlations in liver disease data that may be difficult for humans to detect, which can lead to improved diagnosis and treatment. It is important to note that the accuracy and reliability of machine learning models are heavily dependent on the quality and quantity of the data used for training. Additionally, the use of machine learning in healthcare requires careful consideration of ethical and legal implications, such as data privacy and bias in algorithms.

7 .FUTURE SCOPE

There is a lot of potential for using machine learning techniques to analyze liver patient data, particularly in the areas of diagnosis, prognosis, and treatment. Here are some potential applications of machine learning in liver patient analysis:

Diagnosis: Machine learning algorithms can be trained on large datasets of liver patient information to identify patterns and indicators of liver diseases. These algorithms can be used to develop predictive models that can help physicians diagnose liver diseases earlier and with greater accuracy.

Prognosis: Once a liver disease has been diagnosed, machine learning can be used to predict the progression of the disease and the likelihood of complications. This can help physicians develop personalized treatment plans for individual patients based on their unique risk factors.

Treatment: Machine learning algorithms can be used to analyze the effectiveness of different treatments for liver diseases. This can help physicians identify the most effective treatments for individual patients and adjust treatment plans as needed.

Biomarker discovery: Machine learning can be used to analyze large datasets of liver patient information to identify new biomarkers for liver diseases. These biomarkers can be used to develop new diagnostic tests and treatments for liver diseases.

Overall, the use of machine learning in liver patient analysis has the potential to improve the accuracy of diagnosis and treatment, as well as improve patient outcomes and quality of life

8 . APPENDIX

SOURCE CODE

```
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
from sklearn.model_selection import train_test_split
```

```
[ ] data.head()
```

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

```
[ ] #import the dataset from specified Location
    data = pd.read_csv('/content/indian_liver_patient.csv')
```

```
[ ] data.head()
```

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

```
data.info()
```

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

```
[ ]  data.isnull().any()
```

```
Age                         False
Gender                      False
Total_Bilirubin             False
Direct_Bilirubin            False
Alkaline_Phosphotase        False
Alamine_Aminotransferase    False
Aspartate_Aminotransferase  False
Total_Protiens              False
Albumin                     False
Albumin_and_Globulin_Ratio   True
Dataset                     False
dtype: bool
```

```
data.isnull().sum()
```

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

```
[ ] data.fillna(data['Albumin_and_Globulin_Ratio'].mode()[0])
    data.isnull().sum()
```

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

```
[ ] data[['Gender', 'Dataset','Age']].groupby(['Dataset','Gender'], as_index=False).count().sort_values(by='Dataset', ascending=False)
```

	Dataset	Gender	Age
2	2	Female	50
3	2	Male	117
0	1	Female	92
1	1	Male	324

```
data[['Gender', 'Dataset','Age']].groupby(['Dataset','Gender'], as_index=False).mean().sort_values(by='Dataset', ascending=False)
```

	Dataset	Gender	Age
2	2	Female	42.740000
3	2	Male	40.598291
0	1	Female	43.347826
1	1	Male	46.950617

```
pd.get_dummies(data['Gender'], prefix = 'Gender').head()
```

	Gender_Female	Gender_Male
0	1	0
1	0	1
2	0	1
3	0	1
4	0	1

```
data = pd.concat([data,pd.get_dummies(data['Gender'], prefix = 'Gender')], axis=1)
```

```
data.head()
```

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

```
data.describe()
```

	Age	Total_Bilirubin	Direct_Bilirubin	Alkaline_Phosphotase	Alamine_Aminotransferase	Aspartate_Aminotransferase	Total_Protiens	Albumin	Albumin_and_Globulin_Ratio	Dataset	Gender_Female
count	583.000000	583.000000	583.000000	583.000000	583.000000	583.000000	583.000000	583.000000	579.000000	583.000000	583.000000
mean	44.746141	3.298799	1.486106	290.576329	80.713551	109.910806	6.483190	3.141852	0.947064	1.286449	0.243568
std	16.189833	6.209522	2.808498	242.937989	182.620356	288.918529	1.085451	0.795519	0.319592	0.452490	0.429603
min	4.000000	0.400000	0.100000	63.000000	10.000000	10.000000	2.700000	0.900000	0.300000	1.000000	0.000000
25%	33.000000	0.800000	0.200000	175.500000	23.000000	25.000000	5.800000	2.600000	0.700000	1.000000	0.000000
50%	45.000000	1.000000	0.300000	208.000000	35.000000	42.000000	6.600000	3.100000	0.930000	1.000000	0.000000
75%	58.000000	2.600000	1.300000	298.000000	60.500000	87.000000	7.200000	3.800000	1.100000	2.000000	0.000000
max	90.000000	75.000000	19.700000	2110.000000	2000.000000	4929.000000	9.600000	5.500000	2.800000	2.000000	1.000000

```
data[data['Albumin_and_Globulin_Ratio'].isnull()]
```

Age	Gender	Total_Bilirubin	Direct_Bilirubin	Alkaline_Phosphotase	Alamine_Aminotransferase	Aspartate_Aminotransferase	Total_Protiens	Albumin	Albumin_and_Globulin_Ratio	Dataset	Gender_Female	Gender_Male
45	Female	0.9	0.3	189	23	33	6.6	3.9	NaN	1	1	0
51	Male	0.8	0.2	230	24	46	6.5	3.1	NaN	1	0	1
35	Female	0.6	0.2	180	12	15	5.2	2.7	NaN	2	1	0
27	Male	1.3	0.6	106	25	54	8.5	4.8	NaN	2	0	1

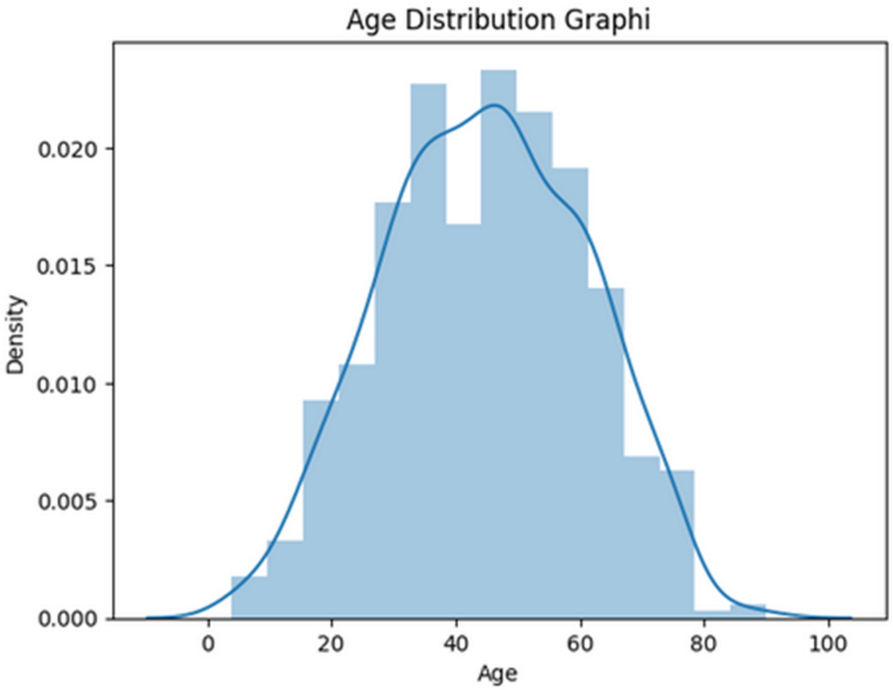
```
data["Albumin_and_Globulin_Ratio"] = data.Albumin_and_Globulin_Ratio.fillna(data['Albumin_and_Globulin_Ratio'].mean())
```

```
from sklearn.preprocessing import LabelEncoder
lc = LabelEncoder()
data['Gender'] = lc.fit_transform(data['Gender'])
```

```
data.describe()
```

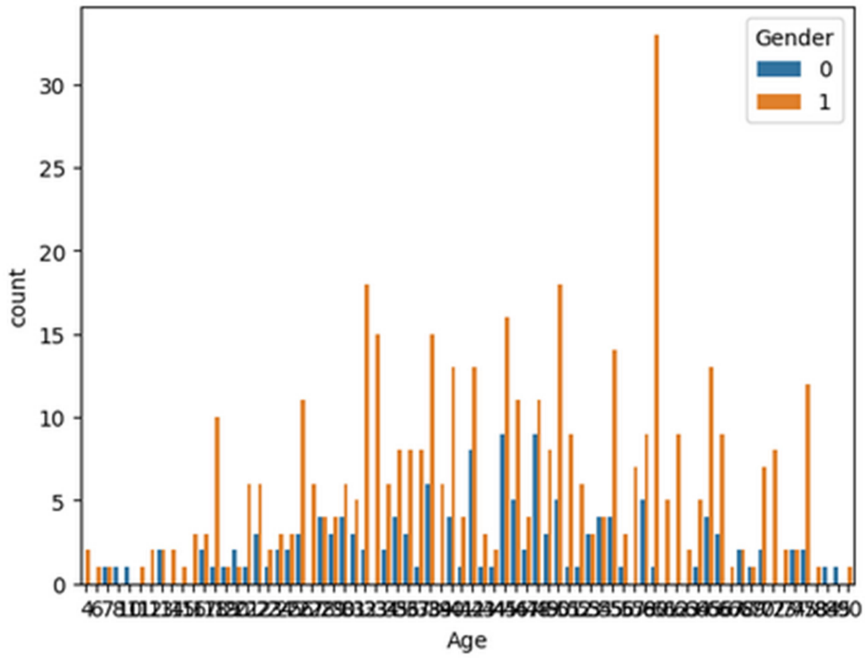
	Age	Gender	Total_Bilirubin	Direct_Bilirubin	Alkaline_Phosphotase	Alamine_Aminotransferase	Aspartate_Aminotransferase	Total_Protiens	Albumin	Albumin_and_Globulin_Ratio	Dataset
count	583.000000	583.000000	583.000000	583.000000	583.000000	583.000000	583.000000	583.000000	583.000000	583.000000	583.000000
mean	44.746141	0.756432	3.298799	1.486106	290.576329	80.713551	109.910806	6.483190	3.141852	0.947064	1.286449
std	16.189833	0.429603	6.209522	2.808498	242.937989	182.620356	288.918529	1.085451	0.795519	0.318492	0.452490
min	4.000000	0.000000	0.400000	0.100000	63.000000	10.000000	10.000000	2.700000	0.900000	0.300000	1.000000
25%	33.000000	1.000000	0.800000	0.200000	175.500000	23.000000	25.000000	5.800000	2.600000	0.700000	1.000000
50%	45.000000	1.000000	1.000000	0.300000	208.000000	35.000000	42.000000	6.600000	3.100000	0.947064	1.000000
75%	58.000000	1.000000	2.600000	1.300000	298.000000	60.500000	87.000000	7.200000	3.800000	1.100000	2.000000
max	90.000000	1.000000	75.000000	19.700000	2110.000000	2000.000000	4929.000000	9.600000	5.500000	2.800000	2.000000

```
[ ]  sns.distplot(data['Age'])
     plt.title('Age Distribution Graphi')
     plt.show()
```




```
[ ]  sns.countplot(x=data['Age'], hue=data['Gender'])
```

```
<Axes: xlabel='Age', ylabel='count'>
```



```
X = data.drop(['Gender','Dataset'], axis=1)
X.head()
```

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

```
[ ] Y = data['Dataset']
```

```
liver_corr = X.corr()

liver_corr
```

	Age	Total_Bilirubin	Direct_Bilirubin	Alkaline_Phosphotase	Alamine_Aminotransferase	Aspartate_Aminotransferase	Total_Protiens	Albumin	Albumin_and_Globulin_Ratio	Gender_Female	Gender_Male
	1.000000	0.011763	0.007529	0.080425	-0.086883	-0.019910	-0.187461	-0.265924	-0.216089	-0.056560	0.056560
n	0.011763	1.000000	0.874618	0.206669	0.214065	0.237831	-0.008099	-0.222250	-0.206159	-0.089291	0.089291
in	0.007529	0.874618	1.000000	0.234939	0.233894	0.257544	-0.000139	-0.228531	-0.200004	-0.100436	0.100436
otase	0.080425	0.206669	0.234939	1.000000	0.125680	0.167196	-0.028514	-0.165453	-0.233960	0.027496	-0.027496
sferase	-0.086883	0.214065	0.233894	0.125680	1.000000	0.791966	-0.042518	-0.029742	-0.002374	-0.082332	0.082332
nsferase	-0.019910	0.237831	0.257544	0.167196	0.791966	1.000000	-0.025645	-0.085290	-0.070024	-0.080336	0.080336
s	-0.187461	-0.008099	-0.000139	-0.028514	-0.042518	-0.025645	1.000000	0.784053	0.233904	0.089121	-0.089121
	-0.265924	-0.222250	-0.228531	-0.165453	-0.029742	-0.085290	0.784053	1.000000	0.686322	0.093799	-0.093799
lin_Ratio	-0.216089	-0.206159	-0.200004	-0.233960	-0.002374	-0.070024	0.233904	0.686322	1.000000	0.003404	-0.003404
le	-0.056560	-0.089291	-0.100436	0.027496	-0.082332	-0.080336	0.089121	0.093799	0.003404	1.000000	-1.000000
e	0.056560	0.089291	0.100436	-0.027496	0.082332	0.080336	-0.089121	-0.093799	-0.003404	-1.000000	1.000000

```
[ ] import warnings
    warnings.filterwarnings('ignore')
    from sklearn.metrics import accuracy_score
    from sklearn.model_selection import train_test_split
    from sklearn.metrics import classification_report, confusion_matrix
    from sklearn import linear_model
    from sklearn.linear_model import LogisticRegression
    from sklearn.svm import SVC, LinearSVC
    from sklearn.ensemble import RandomForestClassifier
    from sklearn.naive_bayes import GaussianNB
```

```
[ ] X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.30, random_state=101)
```

```
[ ] logreg = LogisticRegression()
```

```
[ ] # Train the model using the training sets and check score
    l1=logreg.fit(X_train, Y_train)
```

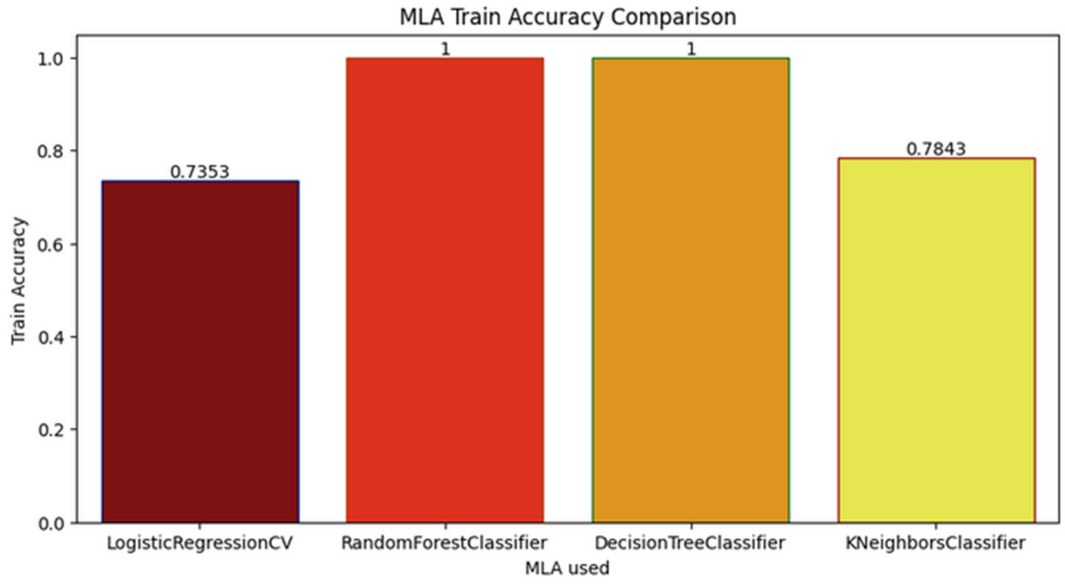
```
[ ] log_predicted= logreg.predict(X_test)
```

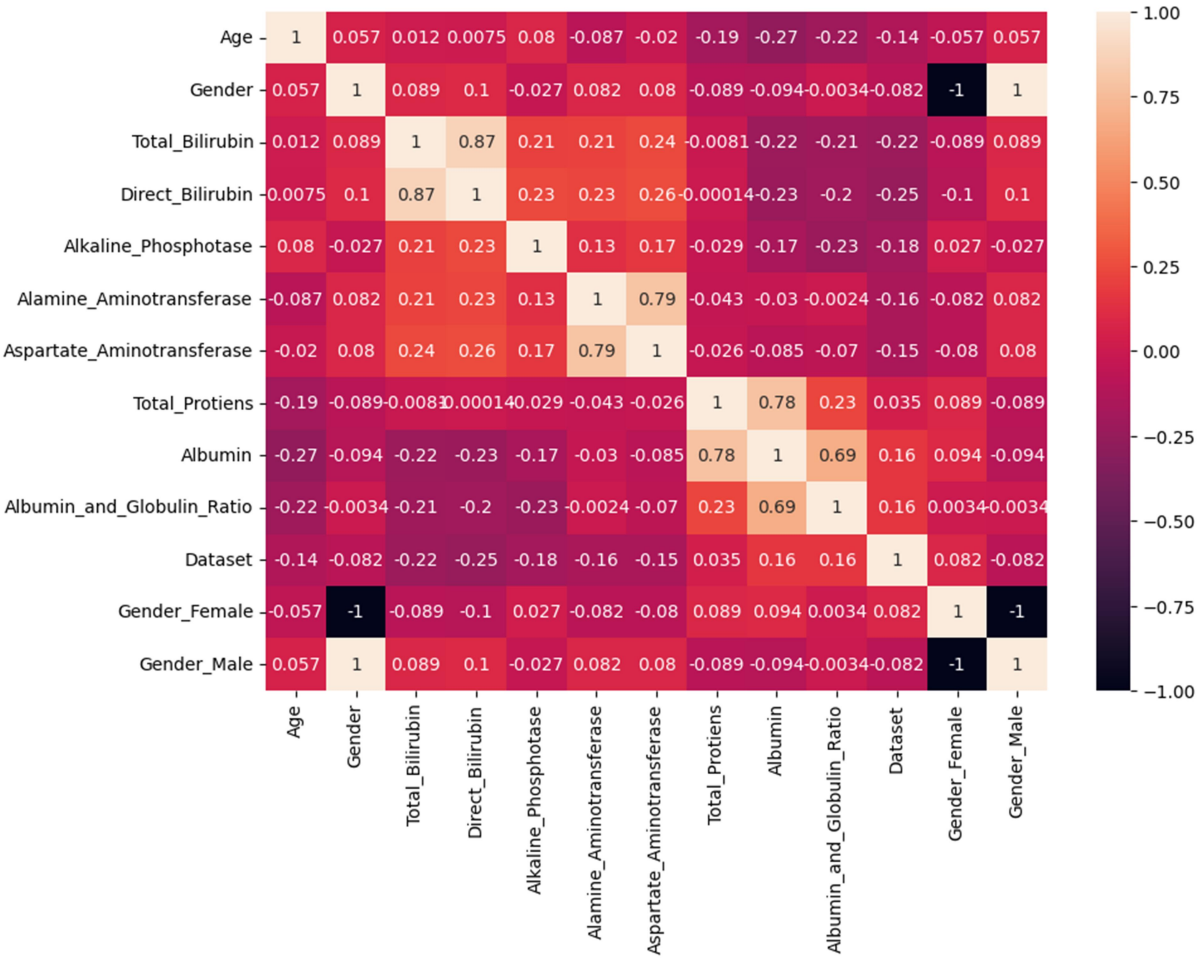
```
[ ]  #Importing sklearn modules
     from sklearn.metrics import mean_squared_error,confusion_matrix, precision_score, recall_score, auc,roc_curve
     from sklearn import ensemble, linear_model, neighbors, svm, tree, neural_network
     from sklearn.linear_model import Ridge
     from sklearn.preprocessing import PolynomialFeatures
     from sklearn.model_selection import train_test_split, cross_val_score
     from sklearn.pipeline import make_pipeline
     from sklearn import svm,model_selection, tree, linear_model, neighbors, naive_bayes, ensemble, discriminant_analysis, gaussian_process
     from sklearn.discriminant_analysis import LinearDiscriminantAnalysis
     from sklearn.neighbors import KNeighborsClassifier
     from sklearn.tree import DecisionTreeClassifier
     from sklearn.naive_bayes import GaussianNB
     from sklearn.svm import SVC
```

```
[ ]  #Application of all Machine Learning methods
     MLA = [
         #GLM
         linear_model.LogisticRegressionCV(),


         ensemble.RandomForestClassifier(),


         #Trees
         tree.DecisionTreeClassifier(),

         #Nearest Neighbor
         neighbors.KNeighborsClassifier(),

         ]
```

```
MLA_columns = []
MLA_compare = pd.DataFrame(columns = MLA_columns)

row_index = 0
for alg in MLA:

    predicted = alg.fit(X_train, Y_train).predict(X_test)
    MLA_name = alg.__class__.__name__
    MLA_compare.loc[row_index,'MLA used'] = MLA_name
    MLA_compare.loc[row_index, 'Train Accuracy'] = round(alg.score(X_train, Y_train), 4)
    MLA_compare.loc[row_index, 'Test Accuracy'] = round(alg.score(X_test, Y_test), 4)
    MLA_compare.loc[row_index, 'Precission'] = precision_score(Y_test, predicted)
    MLA_compare.loc[row_index, 'Recall'] = recall_score(Y_test, predicted)


    row_index+=1
```

```
# Creating plot to show the train accuracy
plt.subplots(figsize=(10,5))
ax=sns.barplot(x="MLA used", y="Train Accuracy",data=MLA_compare,palette='hot',edgecolor=sns.color_palette('dark',7))
plt.xticks()
plt.title('MLA Train Accuracy Comparison')

for i in ax.containers:
  ax.bar_label(i)
```




Age 1 0.057 0.012 0.0075 0.08 -0.087 -0.02 -0.19 -0.27 -0.22 -0.14 -0.057 0.057
Gender 0.057 1 0.089 0.1 -0.027 0.082 0.08 -0.089 -0.094 -0.0034 -0.082 -1 1
Total_Bilirubin 0.012 0.089 1 0.87 0.21 0.21 0.24 -0.0081 -0.22 -0.21 -0.22 -0.089 0.089
Direct_Bilirubin 0.0075 0.1 0.87 1 0.23 0.23 0.26 -0.00014 -0.23 -0.2 -0.25 -0.1 0.1
Alkaline_Phosphotase 0.08 -0.027 0.21 0.23 1 0.13 0.17 -0.029 -0.17 -0.23 -0.18 0.027 -0.027
Alamine_Aminotransferase -0.087 0.082 0.21 0.23 0.13 1 0.79 -0.043 -0.03 -0.0024 -0.16 -0.082 0.082
Aspartate_Aminotransferase -0.02 0.08 0.24 0.26 0.17 0.79 1 -0.026 -0.085 -0.07 -0.15 -0.08 0.08
Total_Protiens -0.19 -0.089 -0.083 0.00014 0.029 -0.043 -0.026 1 0.78 0.23 0.035 0.089 -0.089
Albumin -0.27 -0.094 -0.22 -0.23 -0.17 -0.03 -0.085 0.78 1 0.69 0.16 0.094 -0.094
Albumin_and_Globulin_Ratio -0.22 -0.0034 -0.21 -0.2 -0.23 -0.0024 -0.07 0.23 0.69 1 0.16 0.0034 -0.0034
Dataset -0.14 -0.082 -0.22 -0.25 -0.18 -0.16 -0.15 0.035 0.16 0.16 1 0.082 -0.082
Gender_Female -0.057 -1 -0.089 -0.1 0.027 -0.082 -0.08 0.089 0.094 0.0034 0.082 1 -1
Gender_Male 0.057 1 0.089 0.1 -0.027 0.082 0.08 -0.089 -0.094 -0.0034 -0.082 -1 1
Age
Gender
Total_Bilirubin
Direct_Bilirubin
Alkaline_Phosphotase
Alamine_Aminotransferase
Aspartate_Aminotransferase
Total_Protiens
Albumin
Albumin_and_Globulin_Ratio
Dataset
Gender_Female
Gender_Male
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00

```
[ ]
    import keras
```

```
[ ] from keras.models import Sequential
    from keras.layers import Dense
    from keras.layers import Dropout
```

```
[ ] from sklearn.preprocessing import StandardScaler
    scaler=StandardScaler()
```

```
[ ]
    X_train=scaler.fit_transform(X_train)
```

```
[ ]
    X_test=scaler.transform(X_test)
```

```
[ ] #Initialising the model
    model=Sequential()
```

```
[ ] #adding the first layer
    model.add(Dense(units=6,kernel_initializer='uniform',activation='relu',input_dim=10))
```

```
[ ]  #new_Data.info()
     #adding the second layer
     model.add(Dense(units=6,kernel_initializer='uniform',activation='relu'))
```

```
[ ]  #adding the output layer
     model.add(Dense(units=1,kernel_initializer='uniform',activation='sigmoid'))
```

```
[ ]  #compiling all the layer together
     model.compile(optimizer='adam',loss='binary_crossentropy',metrics=['accuracy'])
```

```
[ ]  # Creating plot to show the test accuracy
     plt.subplots(figsize=(10,5))
     ax=sns.barplot(x="MLA used", y="Test Accuracy",data=MLA_compare,palette='hot',edgecolor=sns.color_palette('dark',7))
     plt.xticks()
     plt.title('Accuracy of different machine learning models')

     for i in ax.containers:
       ax.bar_label(i)
```