B.TECH CSE
APPLICATION LAB REPORT

MALLA REDDY
UNIVERSITY

Department of Computer Science & Engineering

MALLA REDDY UNIVERSITY, HYDERABAD

2022-2023

LIVER DISEASE ANALYSIS

Designed and Developed by

1. N.Sahithi	Roll Number:2011CS010337
2. M.Anirudh Narsaraj	Roll Number:2011CS010325
3. S.Akash Goud	Roll Number:2011CS010284
4. V.Sushanth	Roll Number:2011CS010315

Guided by

Dr.K.Asish Vardhan
Associate Professor

Department of Computer Science & Engineering

MALLA REDDY UNIVERSITY, HYDERABAD

2022-2023

MALLA REDDY UNIVERSITY

(Telangana State Private Universities Act No.13 of 2020 and G.O.Ms.No.14, Higher Education (UE) Department)

CERTIFICATE

This is to certify that this is the Application development lab record entitled "**LIVER DISEASE ANALYSIS**", submitted by **N.SAHITHI (2011CS010337), M.ANIRUDH NARASARAJ (2011CS010325), S.AKASH GOUD (2011CS010284), V.SUSHANTH (2011CS010315)** B.Tech **III** year II semester, Department of CSE during the year 2022-23. The results embodied in this report have not been submitted to any other university or institute for the award of any degree or diploma.

Internal Guide
Dr.K. Asish Vardhan
Associate Professor

HOD-CSE

Dr.ShaikMeeravali

External Examiner

MALLA REDDY UNIVERSITY
(Telangana State Private Universities Act No.13 of 2020 and G.O.Ms.No.14, Higher Education (UE) Department)

DECLARATION

I declare that this project report titled “**LIVER DISEASE ANALYSIS”** submitted in partial fulfillment of the degree of B. Tech in CSE is a record of original work carried out by me under the supervision of **Dr.K.Asish Vardhan , Associate Professor**, and has not formed the basis for the award of any other degree or diploma, in this or any other Institution or University. In keeping with the ethical practice in reporting scientific information, due acknowledgements have been made wherever the findings of others have been cited.

N.Sahithi	(2011CS010337)
M.Anirudh Narsaraj	(2011CS010325)
S.Akash Goud	(2011CS010284)
V.Sushanth	(2011CS010315)

ACKNOWLEDGEMENT

I would like to express my deepest gratitude to my guide, Dr.K.Asish Vardhan Sir for his valuable guidance, consistent encouragement, personal caring, timely help and providing us with an excellent atmosphere for doing research. All through the work, in spite of his busy schedule, he has extended cheerful and cordial support to us for completing this research work. He has been helpful at every step from beginning till the end of this project. We are thankful to him for the encouragement he had given us in completing the project. We express our heart full thanks to the Head of the Department, Department of Computer Science and Engineering, for all the help and infrastructure provided to us to complete the project successfully and her valuable guidance. We are also thankful to all other faculty and staff members of our department for their kind co-operation and help.

TABLE OF CONTENT

ABSTRACT OF THE APPLICATION

In this paper we are going discuss how to predict risk of liver disease for a person, based on the blood test report results of the user. In this paper, the risk of liver disease was predicted using various machine learning algorithms.The final output was predicted based on the most accurate machine learning algorithm.Based on the accurate model we designed a system which asks a person to enter the details of his/her blood test report. Then the system uses the most accurate model which is trained to predict, whether a person has risk of liver disease or not.

Around a million deaths occur due to liver diseases globally. There are several traditional methods to diagnose liver diseases, but they are expensive. Early prediction of liver disease would benefit all individuals prone to liver diseases by providing early treatment. This present study introduces the liver disease prediction (LDP) method in predicting liver disease that can be utilised by health professionals, stakeholders, students and researchers..

The liver is one of the most critical organs of the human body. It plays an essential role in the body's function. Primary purposes include removing toxins from the body, fighting against infections, and balancing the hormones and secretion of bile juice. If these functions are not performed by the liver correctly, it will result in several complications and liver diseases. Therefore if a virus infects the liver or chemicals that injure the liver are consumed, or the immune system's dysfunction occurs, severe damage to the liver or malfunctioning may happen, which ultimately might cause death.

Liver disease is one of the most chronic and threatening diseases globally that can cause various side effects if not treated early. According to World Health Organization (WHO) report in 2018, the number of deaths due to liver diseases is around one million and ranked 11th in the world with a critical number of fatalities (World Total Deaths, n.d.). .Thus, a significant constraint found by health care workers is to predict liver diseases at an early stage, at minimal cost and at the same time provide a better health care system to treat liver diseases.

CHAPTER - 1

INTRODUCTION

1.1 Introduction

With a growing trend of sedentary and lack of physical activities, diseases related to liver have become a common encounter nowadays. In rural areas the intensity is still manageable, but in urban areas, and especially metropolitan areas the liver disease is a very common sighting nowadays. Liver diseases cause millions of deaths every year.

Viral hepatitis alone causes 1.34 million deaths every year. Problems with liver patients are not easily discovered in an early stage as it will be functioning normally even when it is partially damaged. An early diagnosis of liver problems will increase patients survival rate. Liver failures are at high rate of risk among Indians. It is expected that by 2025 India may become the World Capital for Liver Diseases. The widespread occurrence of liver infection in India is contributed due to deskbound lifestyle, increased alcohol consumption and smoking. There are about 100 types of liver infections.

In the human body, the biggest internal glandular organ is the liver, having important functions in metabolism. Its weight is around 3lb which equals 1.36 kg. For human survival, the liver is considered a major organ as it helps most organs. In terms of liver disease, no to minor earlier symptoms occur like lack of energy and weakness. All symptoms rely on both the form and the extent of the infection. A liver function test is sufficient to diagnose liver diseases.

Within the medical field, classification is a commonly used procedure. When it comes to liver infections, it is quite confusing as a liver may be partially damaged however, it still functions at normally. Hence, early detection of liver infection elevates the survival levels of the patient. To detect a liver infection, assessing enzymes levels in the blood is the correct technique.

Severe liver diseases include problems with indigestion, dry mouth, pain in the abdomen, skin colour turning yellow, numbness, memory loss and fainting problems. Unnoticed at the initial stages, these symptoms are only visible when the disease turns chronic. However, even though the liver is partially infected, it can still function .Diagnosis of liver diseases can be divided into three stages i.e., the first stage is liver inflammation, the second is liver scarring (cirrhosis), and the final stage is liver cancer or failure. Since these scenarios are present in liver disease, early prediction is significant to provide better health for New Zealanders. If liver disease is diagnosed early, there will be a chance of early treatment and control of deaths due to liver diseases .

But when the liver fails to function, few treatments are available except liver transplantation , which is very expensive, particularly in New Zealand. Apparently, in New Zealand, 35 - 40% of the population are not diagnosed with Hepatitis C at the early stages because of the asymptomatic behaviour of liver disease. Unfortunately, most of these individuals do not know the risks linked to liver disease. Due to the asymptomatic behaviour and higher costs of liver disease treatment, it is essential to prevent or diagnose early for better treatment.

1.2 Problem Statement

The diagnosis of a disease is the most critical and vital task in medicine, and this mostly depends on a doctor’s intuition based on experiences in the past. Unfortunately, the difficulties in recognizing correct symptoms result in a misdiagnosis. To avoid such medical misdiagnoses, this study utilizes large datasets collected by healthcare industries to automate the diagnosis of diseases. The need to develop a tool that could aid doctors and prevent them from unwarranted errors and unwanted biases in diagnosis is established in this research. Therefore, an automated medical diagnosing system to tackle the problem of correct early detection of liver disease is considered as one of the outputs of this study.

The objective of this project is to develop a machine learning model that can accurately predict the presence or absence of liver disease in patients based on various medical features. The model should be able to analyze a set of input variables, such as age, gender, LB Albumin, alkphos alkaline aminotranferase, and various blood test results, and provide a binary classification of liver disease presence or absence.

Overall, the goal is to develop a reliable and efficient application solution that can aid in the early detection and prediction of liver disease, ultimately contributing to better healthcare outcomes and improving patients' quality of life

1.3 Objective

In India, delayed diagnosis of diseases is a fundamental problem due to a shortage of medical professionals. A typical scenario, prevalent mostly in rural and somewhat in urban areas is:

1. A patient going to a doctor with certain symptoms.
2. The doctor recommending certain tests like blood test, urine test etc depending on the symptoms.
3. The patient taking the aforementioned tests in an analysis lab.
4. The patient taking the reports back to the reports back to the hospital, where they are examined and the disease is identified.

The objective of this paper is to be able to predict the occurrence of liver disease in a sample data set, in order to be able to calculate the predictability to the greater magnitude of accuracy using the Android Studio and fast API.

The primary objective of liver disease prediction using machine learning algorithms is to develop a model that can accurately classify individuals as either having liver disease or being disease-free based on their medical data. The specific objectives include:

1.Early Detection: The model aims to identify individuals who are at risk of developing liver disease at an early stage. Early detection allows for timely medical intervention, monitoring, and treatment, which can potentially prevent the progression of the disease and improve patient outcomes.

2.Prediction Accuracy: The model should strive to achieve high accuracy in predicting the presence or absence of liver disease. This involves training the model on a dataset containing labeled examples of liver disease cases and healthy individuals, and optimizing the model to minimize false positives and false negatives.

3.Feature Importance: By analyzing the model's feature weights or importance scores, the objective is to identify the key factors or variables that contribute significantly to the prediction of liver disease. This information can assist healthcare professionals in understanding the underlying risk factors and making informed decisions about patient care.

4.Generalization: The developed model should be able to generalize well to unseen data. It should not only perform well on the training dataset but also maintain its predictive accuracy on new, unseen instances. This ensures that the model can be reliably used in real-world scenarios and provide accurate predictions for individuals outside the training dataset.

User-Friendly Implementation: The objective is to create a user-friendly system or application that healthcare professionals can easily interact with. The system should accept input variables, such as age, gender, medical test results, and provide a clear and interpretable prediction of liver disease presence or absence. This allows medical practitioners to make informed decisions based on the model's predictions.

Overall, the objective of liver disease prediction is to develop a robust and accurate application that can assist healthcare professionals in early detection, risk assessment, and personalized management of liver disease, ultimately leading to improved patient outcomes and better allocation of healthcare resources.

1.4 Goal

The aim of this project is to somewhat reduce the time delay caused due to the unnecessary back and forth shuttling between the hospital and the pathology lab.A machine learning algorithm will be trained to predict a liver disease in patients.The main aim of this study is to implement an effective Fast API method and Android Studio to predict and diagnose the occurrence of liver diseases in humans in order to eliminate the use of manual methods of analysis relating to liver diseases

The goal of the project is to successfully develop and deploy a application-based liver disease prediction system that achieves the following:

1.High Accuracy: The model should demonstrate a high level of accuracy in predicting the presence or absence of liver disease. This involves achieving a low rate of false positives and false negatives, ensuring that the model provides reliable and trustworthy predictions.

2.Early Detection: The system should be capable of detecting liver disease at an early stage. Early detection allows for prompt medical intervention, enabling healthcare professionals to initiate appropriate treatments, lifestyle modifications, or preventive measures to mitigate the progression of liver disease and improve patient outcomes.

3.Risk Stratification: The model should be able to assess the risk level associated with liver disease for individual patients. By considering various factors and features, the system can categorize patients into different risk groups, providing valuable information to healthcare providers for personalized treatment plans and interventions.

4.Robustness and Generalization: The developed model should exhibit robustness by performing consistently well across different datasets and real-world scenarios. It should be able to generalize to unseen data, ensuring that the predictions remain accurate and reliable when applied to new patient cases.

5.Interpretability and Explainability: The project should aim to provide interpretability and explainability for the predictions made by the model. This allows healthcare professionals to understand the rationale behind the model's predictions and gain insights into the important features contributing to the prediction of liver disease. Transparent and explainable models are crucial for establishing trust and facilitating better decision-making in the clinical setting.

6.User-Friendly Implementation: The ultimate goal is to develop a user-friendly liver disease prediction system that can be easily integrated into existing healthcare systems or used by medical practitioners. The system should provide a simple and intuitive interface for inputting patient data and deliver clear and actionable predictions that can support clinical decision-making.

By achieving these goals, the project aims to provide a valuable tool for healthcare professionals in the early detection, risk assessment, and management of liver disease, ultimately leading to improved patient care, better health outcomes, and effective allocation of medical resources.

CHAPTER - 2
PROBLEM IDENTIFICATION

2.1 Existing System

The existing system had an algorithm called SVM algorithm to calculate that disease details. SVM algorithm is slow algorithm for classifying and it also gives less accuracy. In that main disadvantage is time efficiency. Details of patient liver diseases 8 start from large scale, diverse, fully independent and distributed and seek to explore complex and evolving patterns between data. Existing system shows an algorithm called SVM[Support Vector Machine] theorem that characterizes the options of the massive information revolt, and implement an enormous processing model.this research is pertinent in determining the algorithms that have better accuracy in predicting this dreadful disease.

The stages in the proposed LDP method provide a better alignment of each phase. Once the dataset is
selected, the preprocessing step is conducted by replacing the missing values and balancing the dataset. After that, using R, five different supervised learning methods are applied (i.e., SVM, Naïve Bayes, K-NN, LDA, and CART), and the accuracy with confusion matrix metrics are recorded. The result shows that K-NN has a better accuracy of 72.7% for liver disease prediction. Most of the algorithms are less than the acceptable level of accuracy, which is 75%.

The results from this study would be able to assist health care professionals and relevant stakeholders in the early detection of liver disease.The Existing system shares the same objective but encompass different methodologies to arrive at a relatively less accurate conclusion. The qualitative superiority that these methods have over one another is dependent on the accuracy of the results produced. There are different aspects of the data that are used in order to para metrically come to a definite conclusion over the prediction of liver disease.Typically, the existing mechanisms assumed that the accuracy of prediction was achieved. But this wasn’t the case then, hence, it must be improved further to increase the classification accuracy. Also, other research works addressed these issues by introducing efficient combination. Existing Models based on feature selection and classification raised some issues regarding with training dataset and Test dataset.

2.2 Proposed System

Biomedical science is most important in the field of machine learning to identify liver disease. The liver is the most interior limb of the human corpse. It was acting a major role for relocates blood throughout our stiff. The most matching cases from the previous cases can be retrieved by using Fast API . The whole ideology of the research paper is to help people detect liver disease at the earliest stage and describe the usage of various classification techniques like Android Studio ,Fast API, AWS(Service).

By this we can get a clear picture of how the methods works on liver dataset and how it gives a high efficiency model for predicting liver disease.The mobile app that presently gives all the information about liver diseases will soon be turned into interactive application.It will help the patient in diagnosis of liver diseases, will provide information on the causes of liver disease and answer doubts in a patient’s mind regarding liver transplantation.

"Patients will have to fill a form giving details of symptoms and treatment. At first, the system asks you to enter your details including age, gender, total Bilirubin, direct Bilirubin, total proteins, albumin, A/G ratio, SGPT, SGOT and Alkphos. Values of last eight parameters mentioned here, can be known by blood test report of the user. After taking these 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.

The system has following advantages:

1. No medical expertise required: You don’t need to have any knowledge of medical science and liver diseases to predict the liver disease using this application. All you need to do is enter the details being asked, which are already present in the blood test report(some like age, gender are already known) and then you will get the results of prediction.

2.High accuracy: The system predicts the results with 100 % accuracy for the data-set that we have used while creating this application. While the accuracy might be different in some cases, it will still be high enough to be trustworthy at a large scale.

3.Immediate results: The results here are predicted within seconds of entering the details. You don’t need to wait for a doctor to come, unlike in traditional method.

CHAPTER - 3
REQUIREMENTS

3.1 Software Requirements

● Application Platform	: Mobile Application
● Application Development Framework	: Android
● Programming Language	: Python
● User Interface Design	: XML
● Back-end Application	: FastAPI
● IDE	: Android Studio

3.2Hardware Requirements

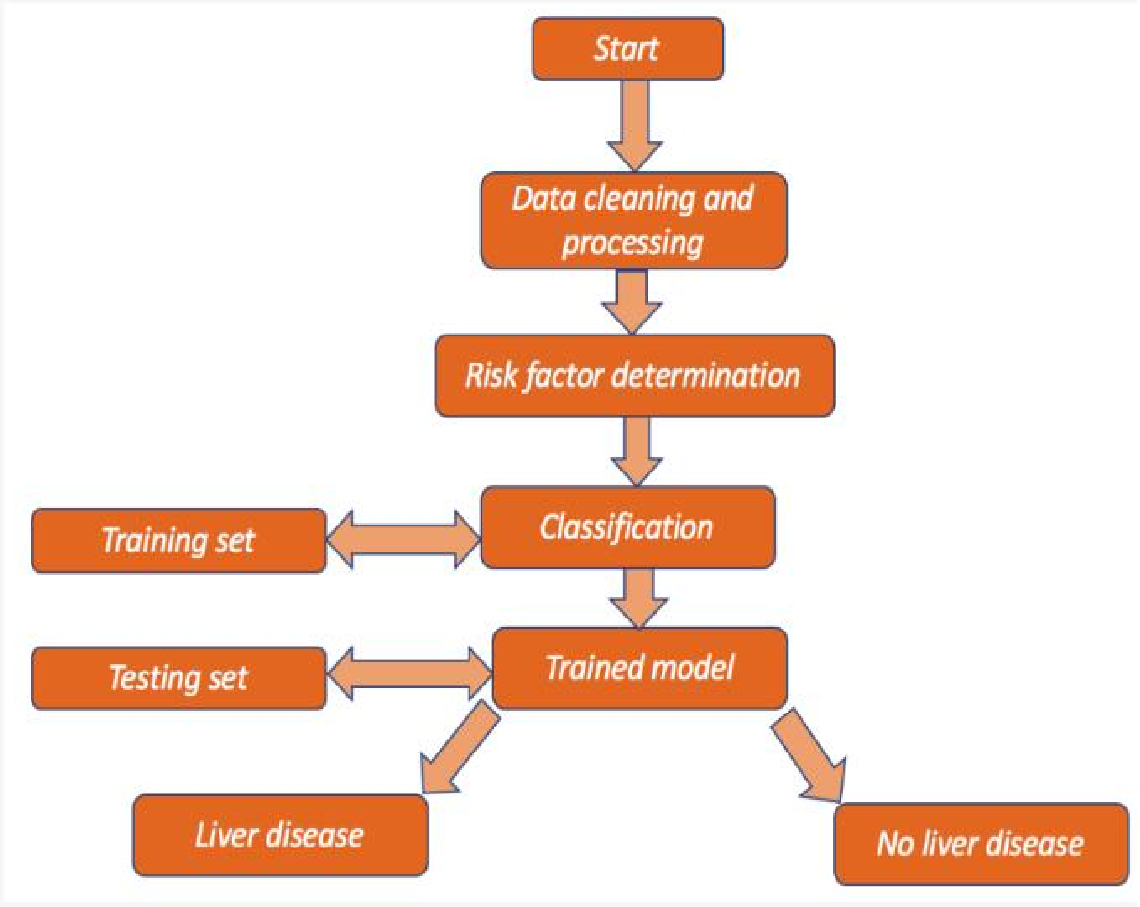
- A 64-bit environment is required for Android 2.3.x (Gingerbread) and higher versions, including the master branch. You can compile older versions on 32-bit systems.
- At least 250GB of free disk space to check out the code and an extra 150 GB to build it. If you conduct multiple builds, you need additional space Note: If you're checking out a mirror, you need more space as full Android Open-Source Project (AOSP) mirrors contain all Git repositories that have ever been used.
- Google recommends at least 64 GB of RAM and doesn't test with less. Lower amounts lead to builds being OOM killed.

CHAPTER – 4

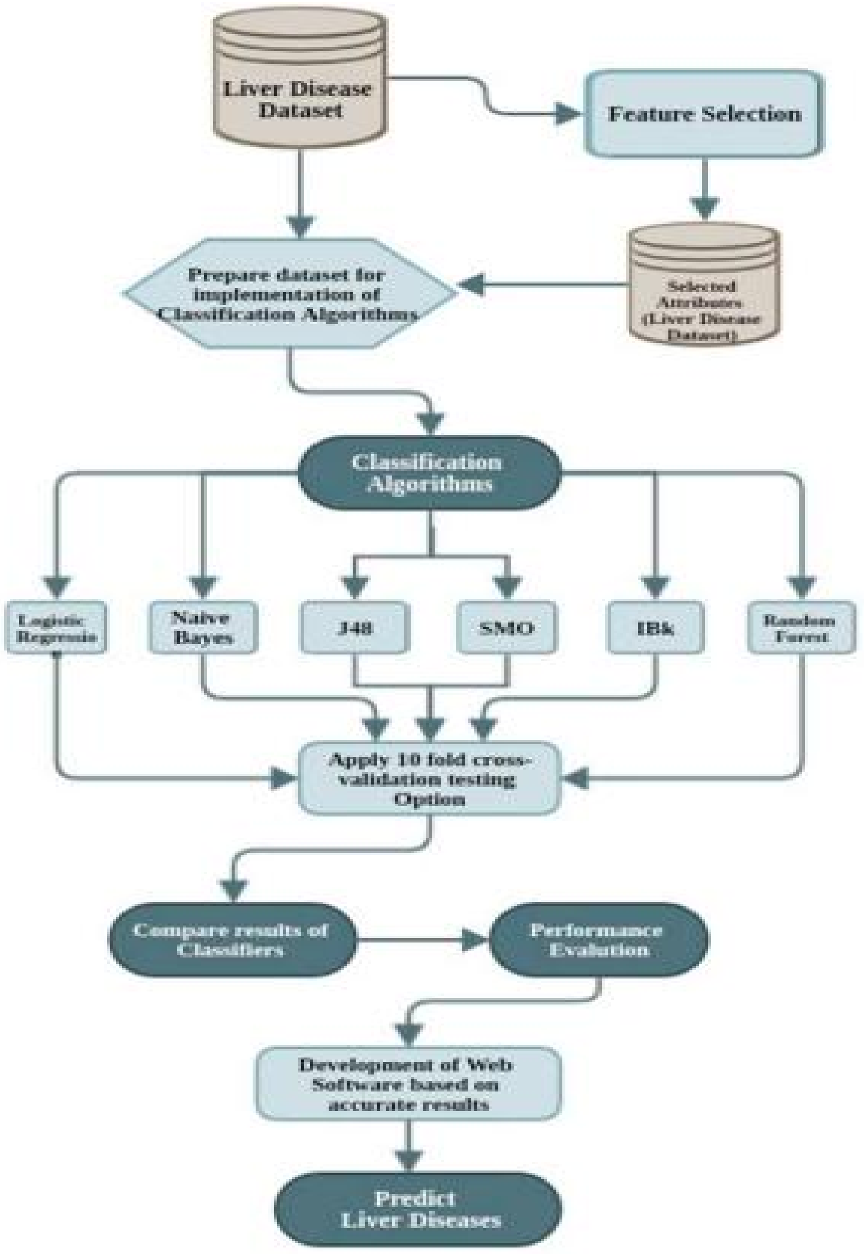
DESIGN AND IMPLEMENTATION

4.1 Design

Figure 1. Flow chart of the study design.



4.2 Implementation


Liver Disease Dataset
Feature Selection
Selected Attributes (Liver Disease Dataset)
Prepare dataset for implementation of Classification Algorithms
Classification Algorithms
Logistic Regressio
Naive Bayes
J48
SMO
IBk
Random Forest
Apply 10 fold cross-validation testing Option
Compare results of Classifiers
Performance Evaluation
Development of Web Software based on accurate results
Predict Liver Diseases

The implementation of a liver disease prediction system involves several steps. Here's an overview of the implementation process:

1.Data Collection: Gather a comprehensive dataset that includes relevant medical features and their corresponding labels (liver disease presence or absence). The dataset should be diverse, representative, and adequately labeled to ensure accurate model training.

2.Data Preprocessing: Perform preprocessing steps to clean and prepare the data for model training. This may involve handling missing values, removing outliers, normalizing or scaling numerical features, and encoding categorical variables.

3.Feature Selection/Engineering: Analyze the dataset to identify the most informative features for liver disease prediction. Conduct feature selection techniques (e.g., correlation analysis, feature importance from models) to select the most relevant features. Additionally, feature engineering may be performed to create new features or transform existing ones to enhance the predictive power of the model.

4.Model Selection: Choose an appropriate machine learning algorithm for liver disease prediction. Consider algorithms such as logistic regression, decision trees, random forests, support vector machines (SVM), gradient boosting, or deep learning models like neural networks. Select the algorithm based on its performance, interpretability, and suitability for the dataset.

5.Model Training: Split the dataset into training and validation sets. Train the selected model using the training data and optimize its parameters through techniques like cross-validation and hyperparameter tuning. Evaluate the model's performance on the validation set, considering metrics such as accuracy, precision, recall, and F1 score.

5.Interpretability and Explainability: Analyze the trained model to interpret and explain its predictions. Use techniques like feature importance analysis, SHAP (SHapley Additive exPlanations), or LIME (Local Interpretable Model-agnostic Explanations) to understand the impact of individual features on the model's predictions.

6.System Development: Develop a user-friendly interface or application that allows healthcare professionals to input patient data and obtain liver disease predictions. Design the interface to be intuitive, visually appealing, and seamlessly integrated into the existing clinical workflow.

7.Validation and Evaluation: Validate the developed liver disease prediction system using independent datasets or clinical studies. Assess its performance, accuracy, and generalizability compared to existing diagnostic methods or expert opinions. Incorporate feedback from healthcare professionals to refine and improve the system.

8.Deployment and Monitoring: Deploy the liver disease prediction system in a production environment, ensuring scalability and efficient real-time predictions. Continuously monitor the system's performance, update the model periodically as new data becomes available, and refine the system based on user feedback and evolving medical knowledge.

It is important to note that implementing a liver disease prediction system requires expertise in data preprocessing, machine learning, software development, and domain knowledge in liver diseases. Collaboration with healthcare professionals and data scientists can greatly contribute to the success of the implementation process.

Importing Modules

For further process we need to import some important modules present in python:

import **pandas** as **pd**

import **numpy** as **np**

import **seaborn** as **sns**

import **matplotlib.pyplot** as **plt**

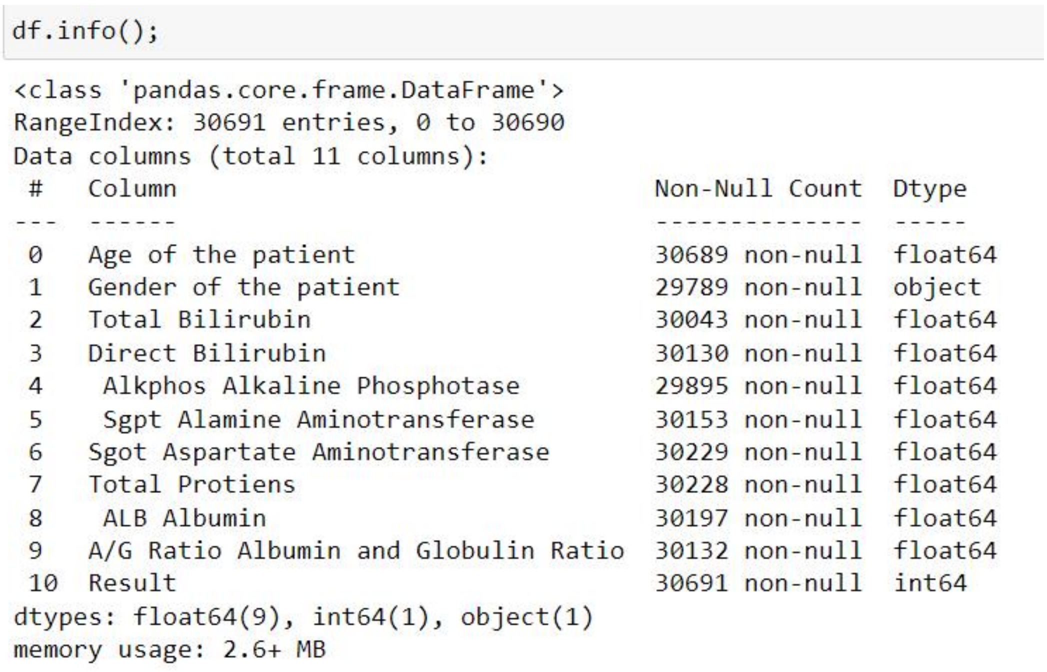
So, we import pandas for data analysis, **NumPy** for calculating N-dimensional array, **seaborn**, and **matplotlib** to visualize the data.

Reading data

Generally, we use a dataset in the form of a CSV file, for reading this CSV file we will use the panda's library, let's see:

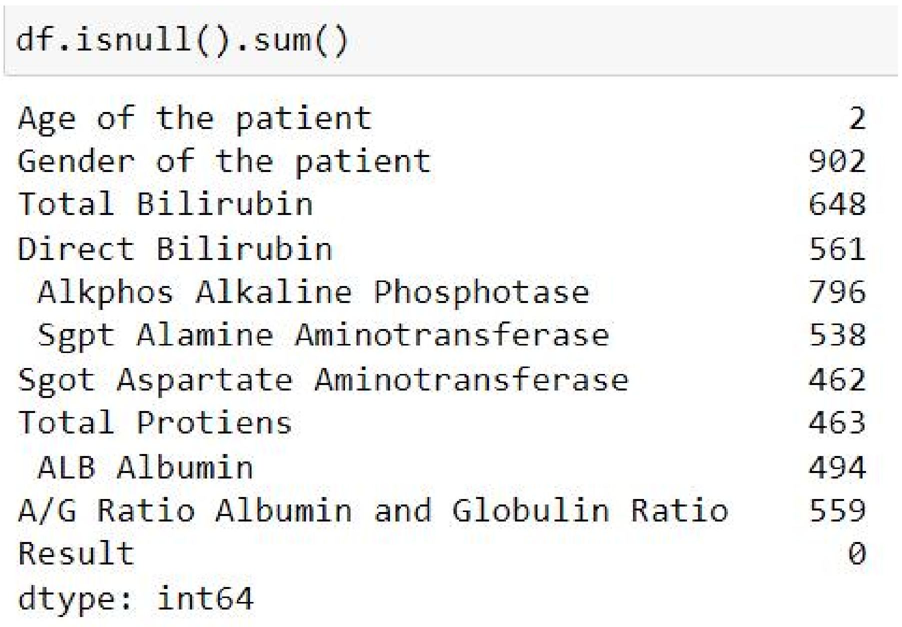
df = pd.read_csv(r'C:\Users\HP\Downloads\archive (5)\Liver Patient Dataset (LPD)_train.csv', encoding= 'unicode_escape')
print(df)

StudyDataset



```
df.info();
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 30691 entries, 0 to 30690
Data columns (total 11 columns):
 #   Column                                Non-Null Count  Dtype  
---  ------                                --------------  -----  
 0   Age of the patient                    30689 non-null  float64
 1   Gender of the patient                 29789 non-null  object 
 2   Total Bilirubin                       30043 non-null  float64
 3   Direct Bilirubin                      30130 non-null  float64
 4    Alkphos Alkaline Phosphotase         29895 non-null  float64
 5    Sgpt Alamine Aminotransferase        30153 non-null  float64
 6   Sgot Aspartate Aminotransferase       30229 non-null  float64
 7   Total Protiens                        30228 non-null  float64
 8    ALB Albumin                          30197 non-null  float64
 9   A/G Ratio Albumin and Globulin Ratio  30132 non-null  float64
 10  Result                                30691 non-null  int64  
dtypes: float64(9), int64(1), object(1)
memory usage: 2.6+ MB
```



```
df.isnull().sum()
```

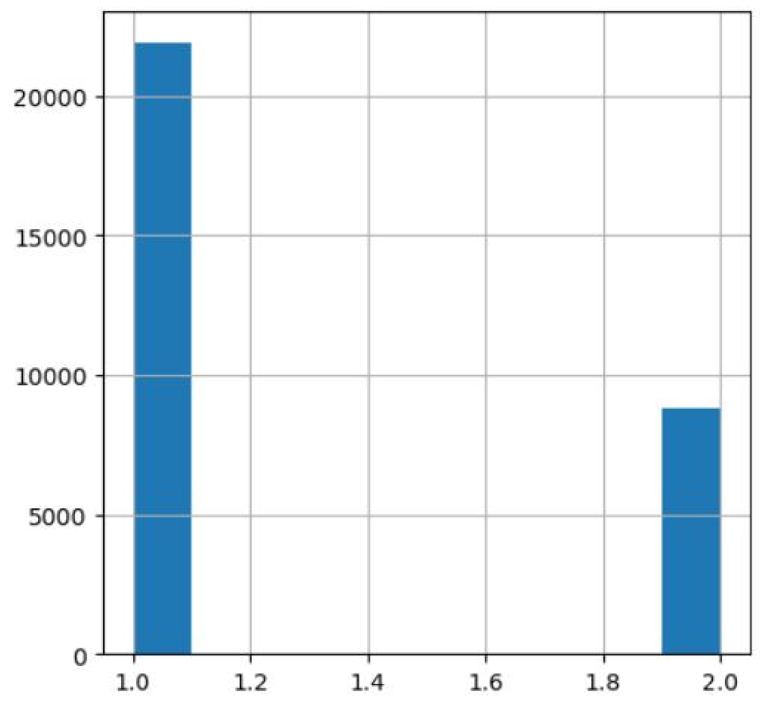
```
Age of the patient                        2
Gender of the patient                   902
Total Bilirubin                         648
Direct Bilirubin                        561
 Alkphos Alkaline Phosphotase           796
 Sgpt Alamine Aminotransferase          538
Sgot Aspartate Aminotransferase         462
Total Protiens                          463
 ALB Albumin                            494
A/G Ratio Albumin and Globulin Ratio    559
Result                                    0
dtype: int64
```

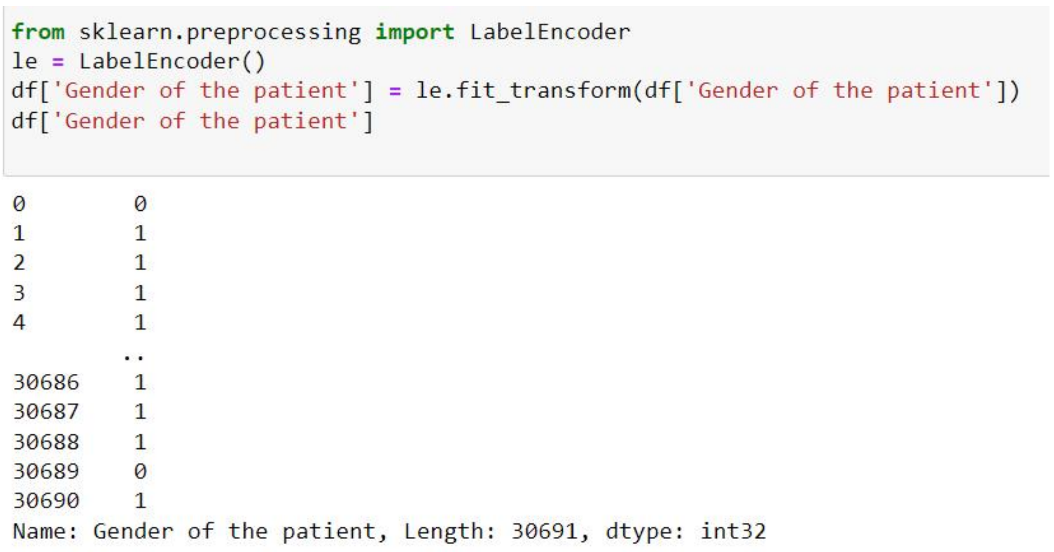
```
df=df.fillna(method='bfill')
df.isnull().sum()
```

```
Age of the patient                      0
Gender of the patient                   0
Total Bilirubin                         0
Direct Bilirubin                        0
 Alkphos Alkaline Phosphotase           0
 Sgpt Alamine Aminotransferase          0
Sgot Aspartate Aminotransferase         0
Total Protiens                          0
 ALB Albumin                            0
A/G Ratio Albumin and Globulin Ratio    0
Result                                  0
dtype: int64
```

```
df['Result'].hist(figsize=(5,5))
```

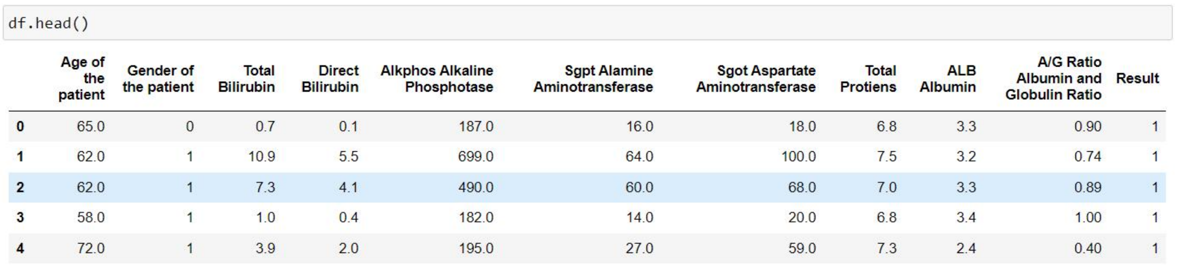
```
<Axes: >
```





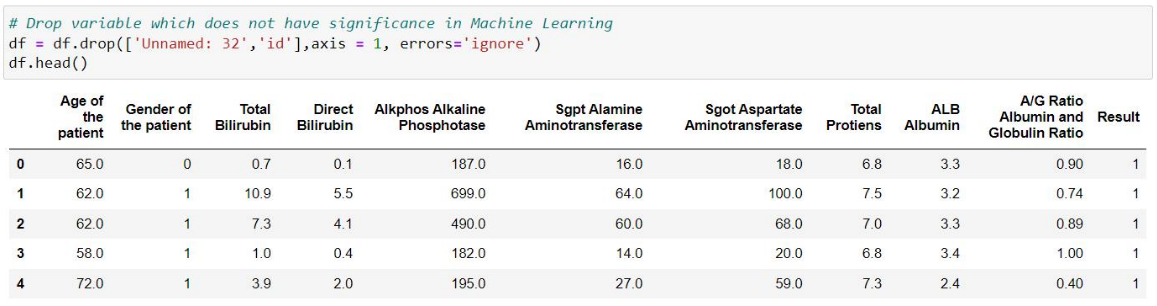
```
from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()
df['Gender of the patient'] = le.fit_transform(df['Gender of the patient'])
df['Gender of the patient']
```

```
0        0
1        1
2        1
3        1
4        1
        ..
30686    1
30687    1
30688    1
30689    0
30690    1
Name: Gender of the patient, Length: 30691, dtype: int32
```



```
df.head()
```

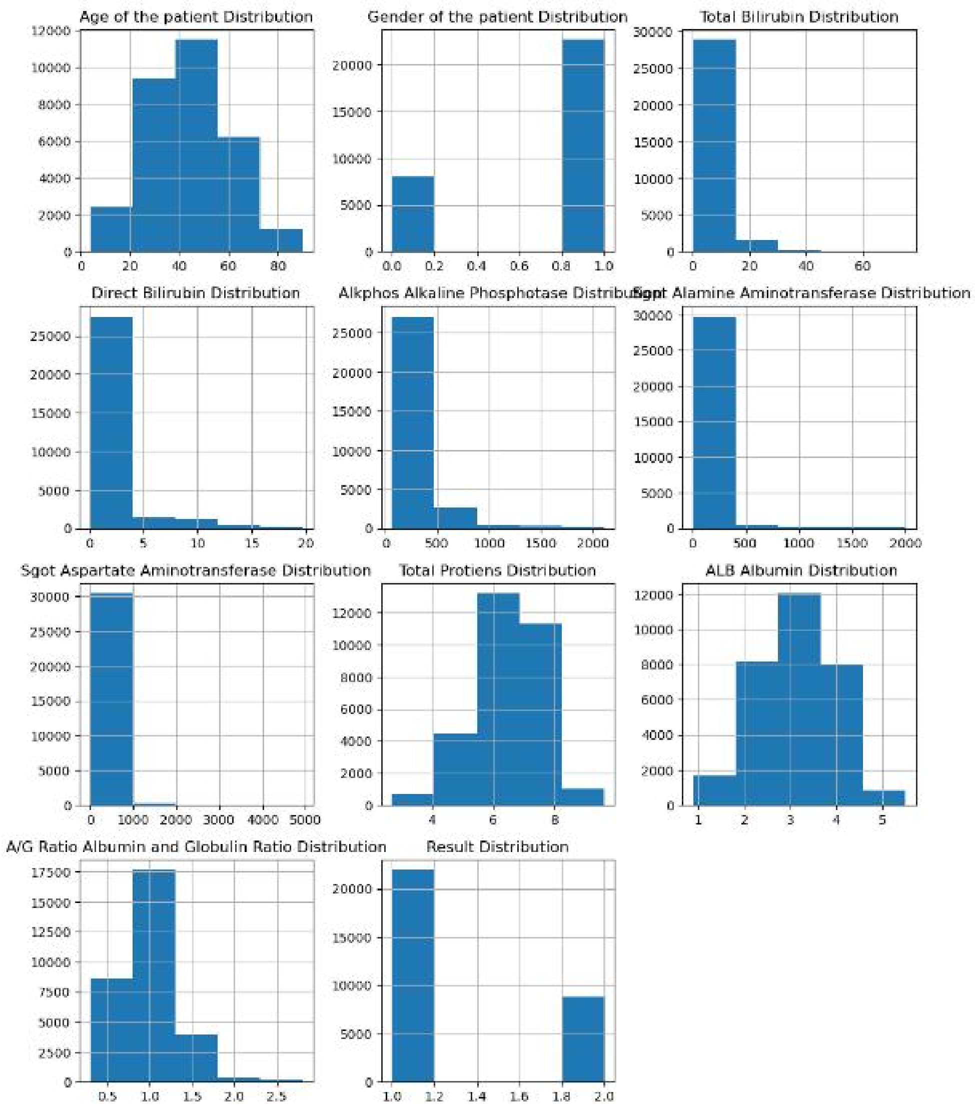
	Age of the patient	Gender of the patient	Total Bilirubin	Direct Bilirubin	Alkphos Alkaline Phosphotase	Sgpt Alamine Aminotransferase	Sgot Aspartate Aminotransferase	Total Protiens	ALB Albumin	A/G Ratio Albumin and Globulin Ratio	Result
0	65.0	0	0.7	0.1	187.0	16.0	18.0	6.8	3.3	0.90	1
1	62.0	1	10.9	5.5	699.0	64.0	100.0	7.5	3.2	0.74	1
2	62.0	1	7.3	4.1	490.0	60.0	68.0	7.0	3.3	0.89	1
3	58.0	1	1.0	0.4	182.0	14.0	20.0	6.8	3.4	1.00	1
4	72.0	1	3.9	2.0	195.0	27.0	59.0	7.3	2.4	0.40	1



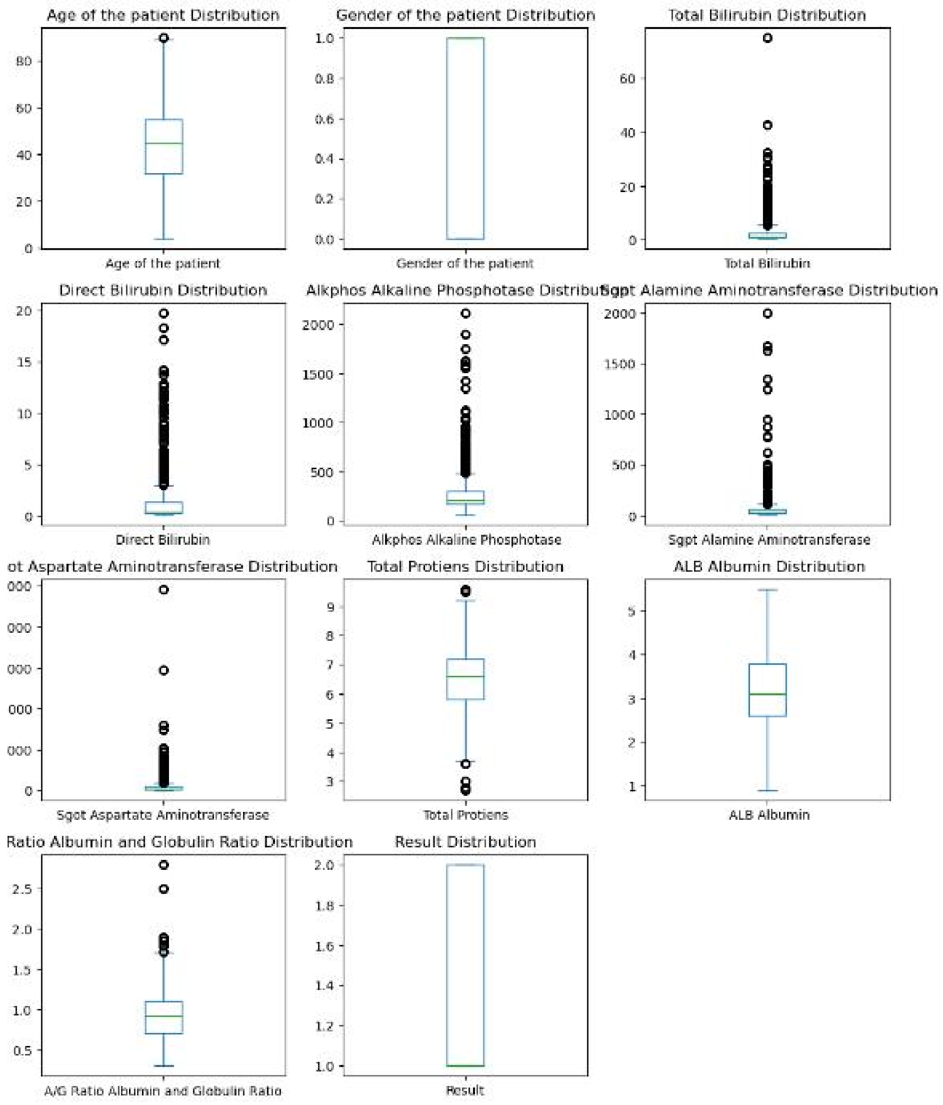
```
# Drop variable which does not have significance in Machine Learning
df = df.drop(['Unnamed: 32','id'],axis = 1, errors='ignore')
df.head()
```

	Age of the patient	Gender of the patient	Total Bilirubin	Direct Bilirubin	Alkphos Alkaline Phosphotase	Sgpt Alamine Aminotransferase	Sgot Aspartate Aminotransferase	Total Protiens	ALB Albumin	A/G Ratio Albumin and Globulin Ratio	Result
0	65.0	0	0.7	0.1	187.0	16.0	18.0	6.8	3.3	0.90	1
1	62.0	1	10.9	5.5	699.0	64.0	100.0	7.5	3.2	0.74	1
2	62.0	1	7.3	4.1	490.0	60.0	68.0	7.0	3.3	0.89	1
3	58.0	1	1.0	0.4	182.0	14.0	20.0	6.8	3.4	1.00	1
4	72.0	1	3.9	2.0	195.0	27.0	59.0	7.3	2.4	0.40	1

```
def histograms_plot(dataframe, features, rows, cols):
    fig=plt.figure(figsize=(10,75))
    for i, feature in enumerate(features):
        ax=fig.add_subplot(rows,cols,i+1)
        df[feature].hist(bins=5,ax=ax)
        ax.set_title(feature+" Distribution",color='black')

    fig.tight_layout()
    plt.show()

histograms_plot(df,df.columns,25,3)
```

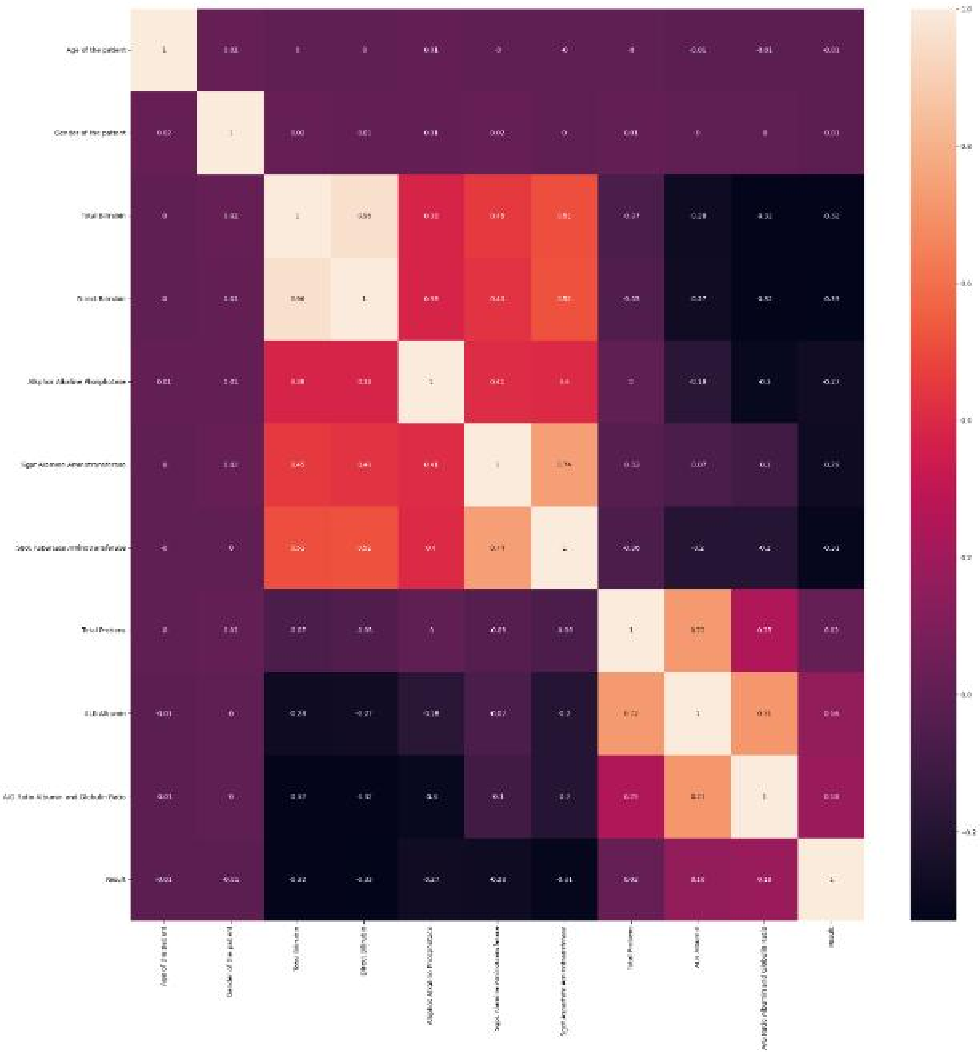


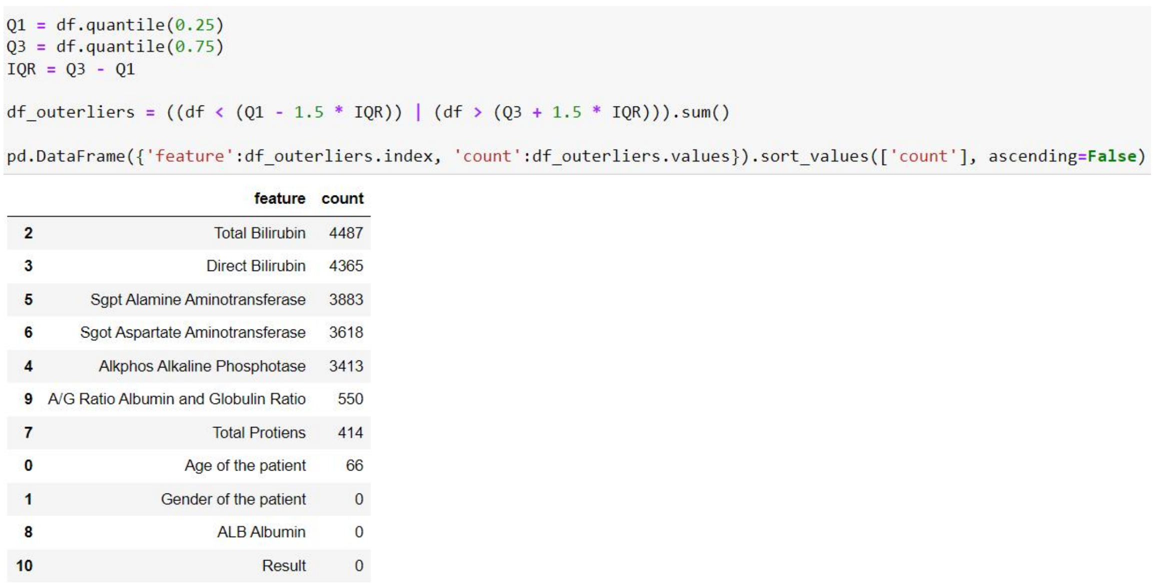
```
def box_plot(dataframe, features, rows, cols):
    fig=plt.figure(figsize=(10,75))
    for i, feature in enumerate(features):
        ax=fig.add_subplot(rows,cols,i+1)
        df[feature].plot(kind='box', title='compactness_mean',ax=ax)
        ax.set_title(feature+" Distribution",color='black')

    fig.tight_layout()
    plt.show()

box_plot(df,df.columns,25,3)
```



```
import seaborn as sns

plt.figure(figsize =(25,25))
sns.heatmap(df.corr(method='pearson',min_periods=1).round(2), annot = True)
```

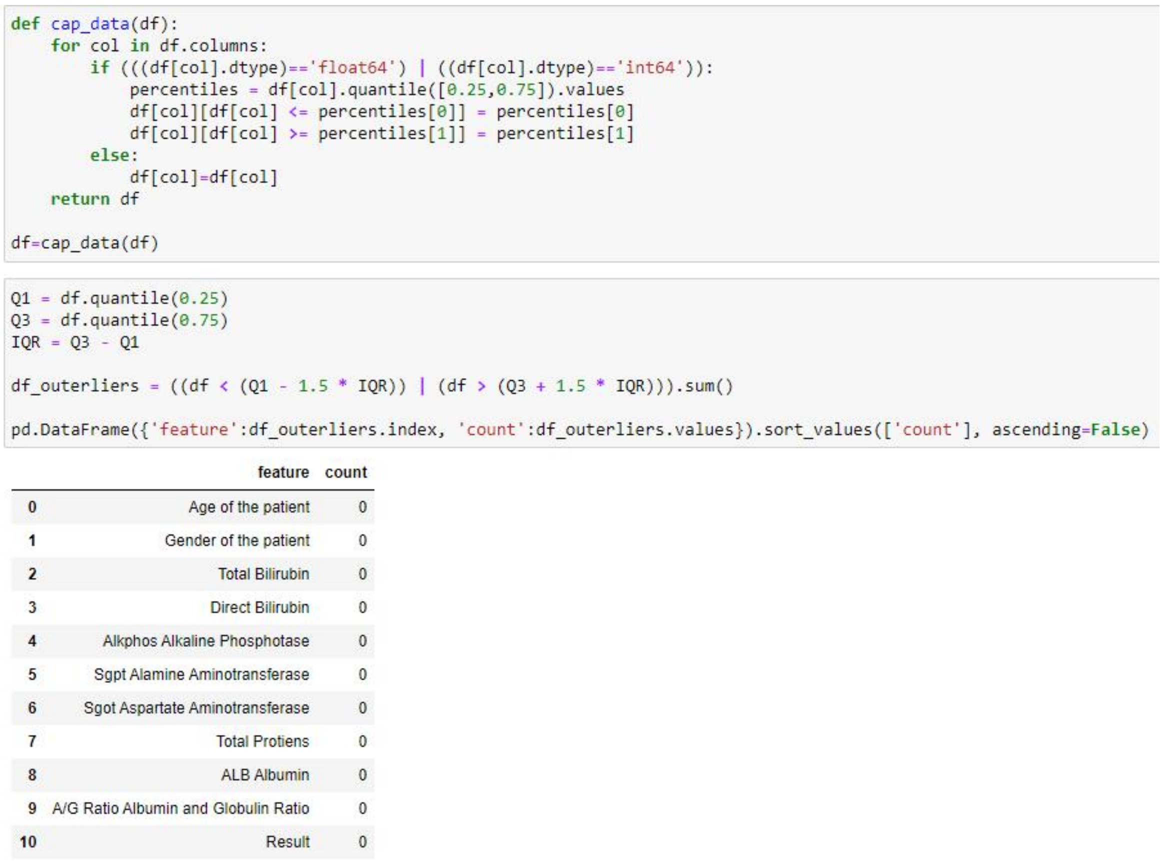
```
<Axes: >
```





```
Q1 = df.quantile(0.25)
Q3 = df.quantile(0.75)
IQR = Q3 - Q1

df_outerliers = ((df < (Q1 - 1.5 * IQR)) | (df > (Q3 + 1.5 * IQR))).sum()

pd.DataFrame({'feature':df_outerliers.index, 'count':df_outerliers.values}).sort_values(['count'], ascending=False)
```

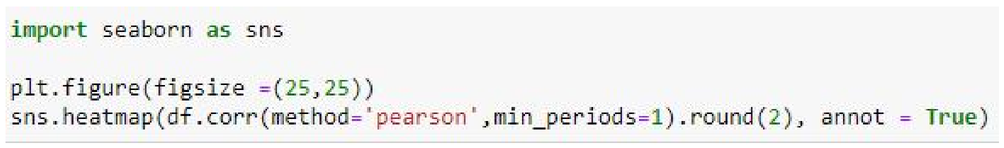
	feature	count
2	Total Bilirubin	4487
3	Direct Bilirubin	4365
5	Sgpt Alamine Aminotransferase	3883
6	Sgot Aspartate Aminotransferase	3618
4	Alkphos Alkaline Phosphotase	3413
9	A/G Ratio Albumin and Globulin Ratio	550
7	Total Protiens	414
0	Age of the patient	66
1	Gender of the patient	0
8	ALB Albumin	0
10	Result	0



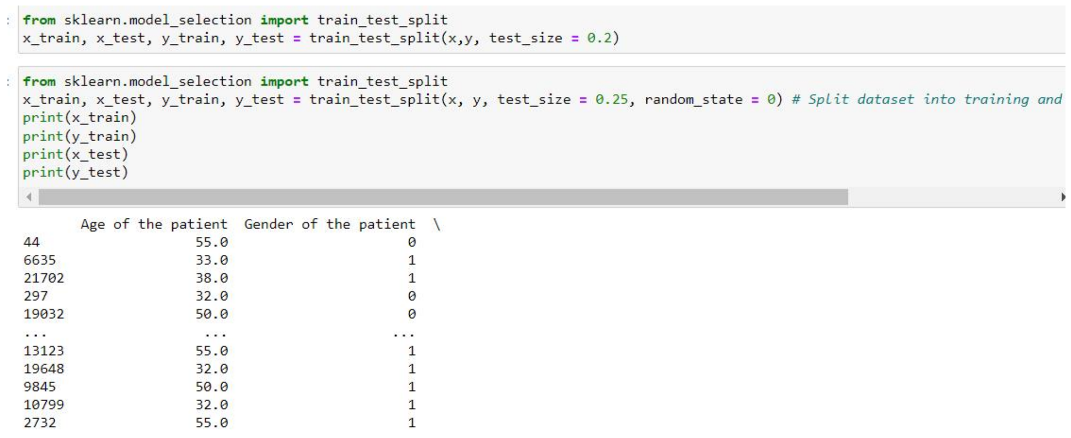
```
def cap_data(df):
    for col in df.columns:
        if (((df[col].dtype)=='float64') | ((df[col].dtype)=='int64')):
            percentiles = df[col].quantile([0.25,0.75]).values
            df[col][df[col] <= percentiles[0]] = percentiles[0]
            df[col][df[col] >= percentiles[1]] = percentiles[1]
        else:
            df[col]=df[col]
    return df

df=cap_data(df)
```

```
Q1 = df.quantile(0.25)
Q3 = df.quantile(0.75)
IQR = Q3 - Q1

df_outerliers = ((df < (Q1 - 1.5 * IQR)) | (df > (Q3 + 1.5 * IQR))).sum()

pd.DataFrame({'feature':df_outerliers.index, 'count':df_outerliers.values}).sort_values(['count'], ascending=False)
```

	feature	count
0	Age of the patient	0
1	Gender of the patient	0
2	Total Bilirubin	0
3	Direct Bilirubin	0
4	Alkphos Alkaline Phosphotase	0
5	Sgpt Alamine Aminotransferase	0
6	Sgot Aspartate Aminotransferase	0
7	Total Protiens	0
8	ALB Albumin	0
9	A/G Ratio Albumin and Globulin Ratio	0
10	Result	0



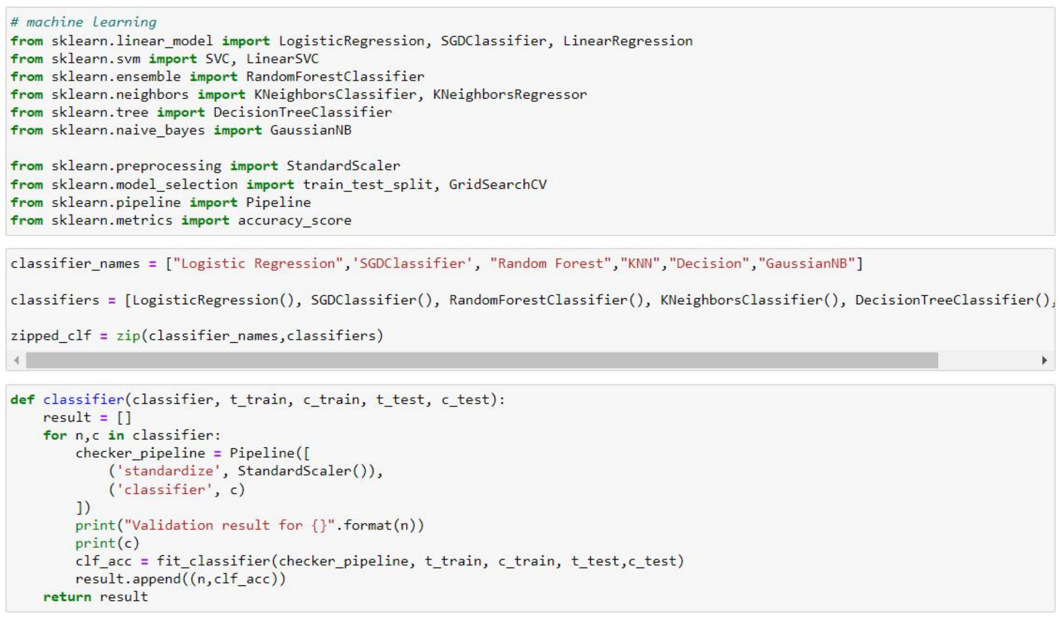
```
import seaborn as sns

plt.figure(figsize =(25,25))
sns.heatmap(df.corr(method='pearson',min_periods=1).round(2), annot = True)
```



```
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(x,y, test_size = 0.2)
```

```
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size = 0.25, random_state = 0) # Split dataset into training and
print(x_train)
print(y_train)
print(x_test)
print(y_test)
```

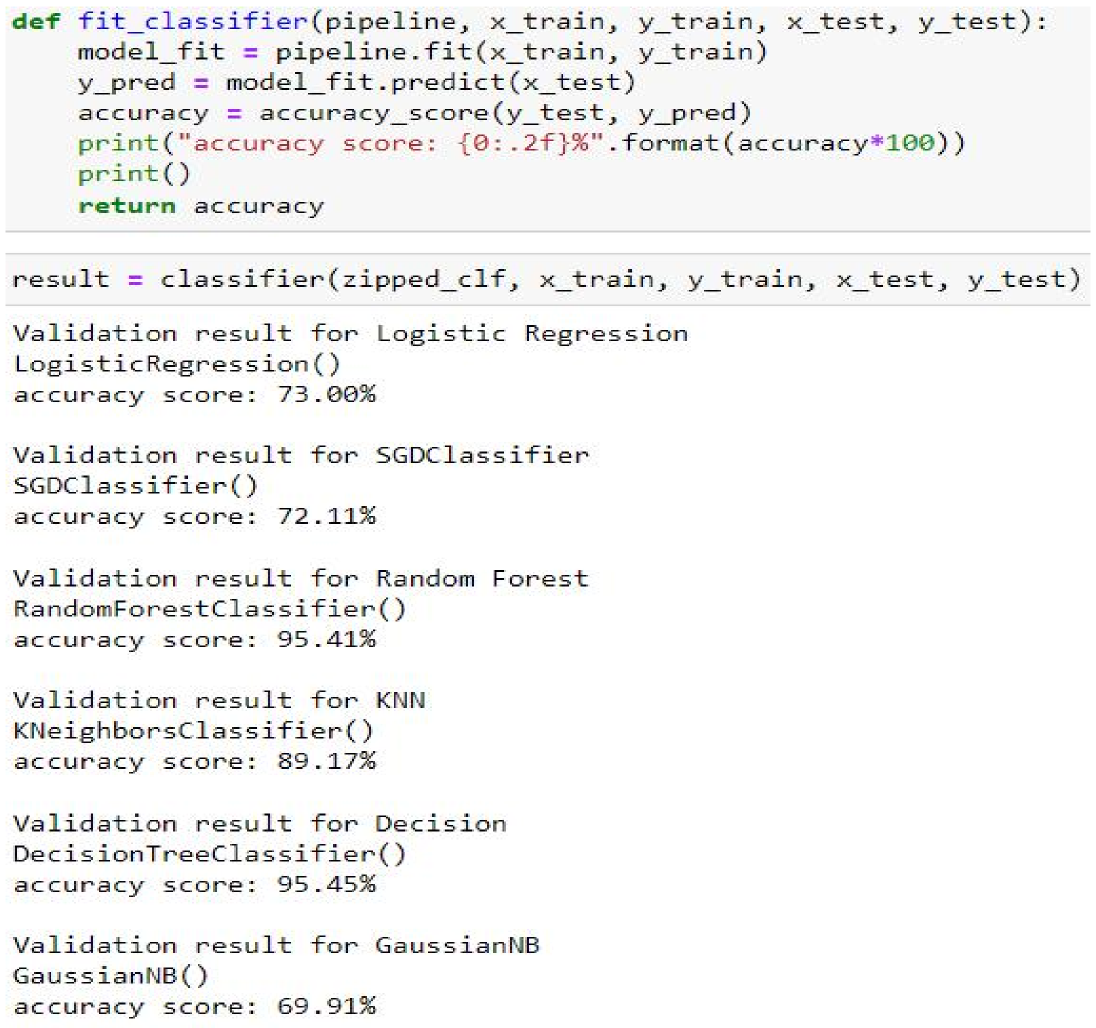
```
       Age of the patient  Gender of the patient  \
44                   55.0                      0
6635                 33.0                      1
21702                38.0                      1
297                  32.0                      0
19032                50.0                      0
...                   ...                    ...
13123                55.0                      1
19648                32.0                      1
9845                 50.0                      1
10799                32.0                      1
2732                 55.0                      1
```



```
# machine learning
from sklearn.linear_model import LogisticRegression, SGDClassifier, LinearRegression
from sklearn.svm import SVC, LinearSVC
from sklearn.ensemble import RandomForestClassifier
from sklearn.neighbors import KNeighborsClassifier, KNeighborsRegressor
from sklearn.tree import DecisionTreeClassifier
from sklearn.naive_bayes import GaussianNB

from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split, GridSearchCV
from sklearn.pipeline import Pipeline
from sklearn.metrics import accuracy_score
```

```
classifier_names = ["Logistic Regression",'SGDClassifier', "Random Forest","KNN","Decision","GaussianNB"]

classifiers = [LogisticRegression(), SGDClassifier(), RandomForestClassifier(), KNeighborsClassifier(), DecisionTreeClassifier(),

zipped_clf = zip(classifier_names,classifiers)
```

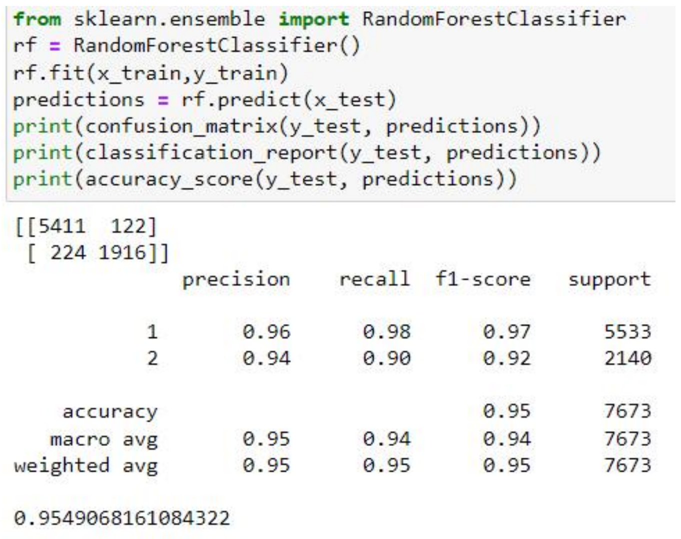
```
def classifier(classifier, t_train, c_train, t_test, c_test):
    result = []
    for n,c in classifier:
        checker_pipeline = Pipeline([
            ('standardize', StandardScaler()),
            ('classifier', c)
        ])
        print("Validation result for {}".format(n))
        print(c)
        clf_acc = fit_classifier(checker_pipeline, t_train, c_train, t_test,c_test)
        result.append((n,clf_acc))
    return result
```



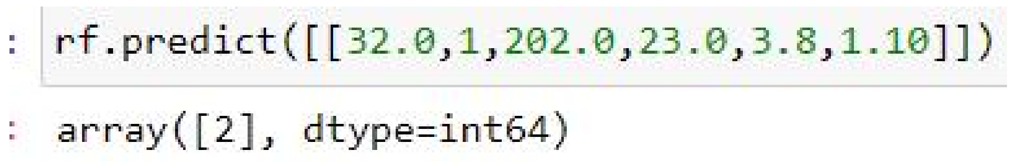
```
def fit_classifier(pipeline, x_train, y_train, x_test, y_test):
    model_fit = pipeline.fit(x_train, y_train)
    y_pred = model_fit.predict(x_test)
    accuracy = accuracy_score(y_test, y_pred)
    print("accuracy score: {0:.2f}%".format(accuracy*100))
    print()
    return accuracy
```

```
result = classifier(zipped_clf, x_train, y_train, x_test, y_test)
```

```
Validation result for Logistic Regression
LogisticRegression()
accuracy score: 73.00%

Validation result for SGDClassifier
SGDClassifier()
accuracy score: 72.11%

Validation result for Random Forest
RandomForestClassifier()
accuracy score: 95.41%

Validation result for KNN
KNeighborsClassifier()
accuracy score: 89.17%

Validation result for Decision
DecisionTreeClassifier()
accuracy score: 95.45%

Validation result for GaussianNB
GaussianNB()
accuracy score: 69.91%
```


```
from sklearn.ensemble import RandomForestClassifier
rf = RandomForestClassifier()
rf.fit(x_train,y_train)
predictions = rf.predict(x_test)
print(confusion_matrix(y_test, predictions))
print(classification_report(y_test, predictions))
print(accuracy_score(y_test, predictions))
```

```
[[5411  122]
 [ 224 1916]]
              precision    recall  f1-score   support

           1       0.96      0.98      0.97      5533
           2       0.94      0.90      0.92      2140

    accuracy                           0.95      7673
   macro avg       0.95      0.94      0.94      7673
weighted avg       0.95      0.95      0.95      7673

0.9549068161084322
```


```
: rf.predict([[32.0,1,202.0,23.0,3.8,1.10]])

: array([2], dtype=int64)
```

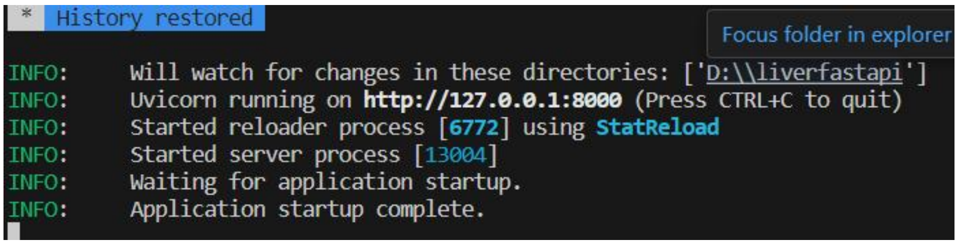
APP.py

```
# 1. Install uvicorn and fastapi
# pip install fastapi uvicorn

# 2. Imports Libraries
import uvicorn
from fastapi import FastAPI
from fastapi.middleware.cors import CORSMiddleware
import pickle

# 3. Create App
app = FastAPI()

# 4. Configure CORS to access API from anywhere
app.add_middleware(
    CORSMiddleware,
    allow_origins=["*"],
    allow_credentials=True,
    allow_methods=["*"],
    allow_headers=["*"],
)

# 5. Index route, opens automatically on http://127.0.0.1:8000
@app.get('/')
def index():
    return {'message': 'Hello World'}

# 6. Run App
if __name__ == '__main__':
    uvicorn.run(app, port=8080, host='0.0.0.0')

# 7a. Using GET Method
@app.get("/predictDisease")
def getPredictDisease(Age_of_the_patient: float,Gender_of_the_patient: float,
                Alkphos_Alkaline_Phosphotase: float,Sgpt_Alamine_Aminotransferase:
float,ALB_Albumin: float,
                A_G_Ratio_Albumin_and_Globulin_Ratio: float):
    rgModel = pickle.load(open("rf.pkl", "rb"))

    prediction =
rgModel.predict([[Age_of_the_patient,Gender_of_the_patient,Alkphos_Alkaline_Phosph
otase,Sgpt_Alamine_Aminotransferase,ALB_Albumin,A_G_Ratio_Albumin_and_Globu
lin_Ratio]])
    return [{
        'Result': str(prediction[0])
    }]

# 7b. Using POST Method

#-------------
# 8. Run the API with uvicorn with Reload Option - Auto Run after edit source code
```

# uvicorn app:app --reload

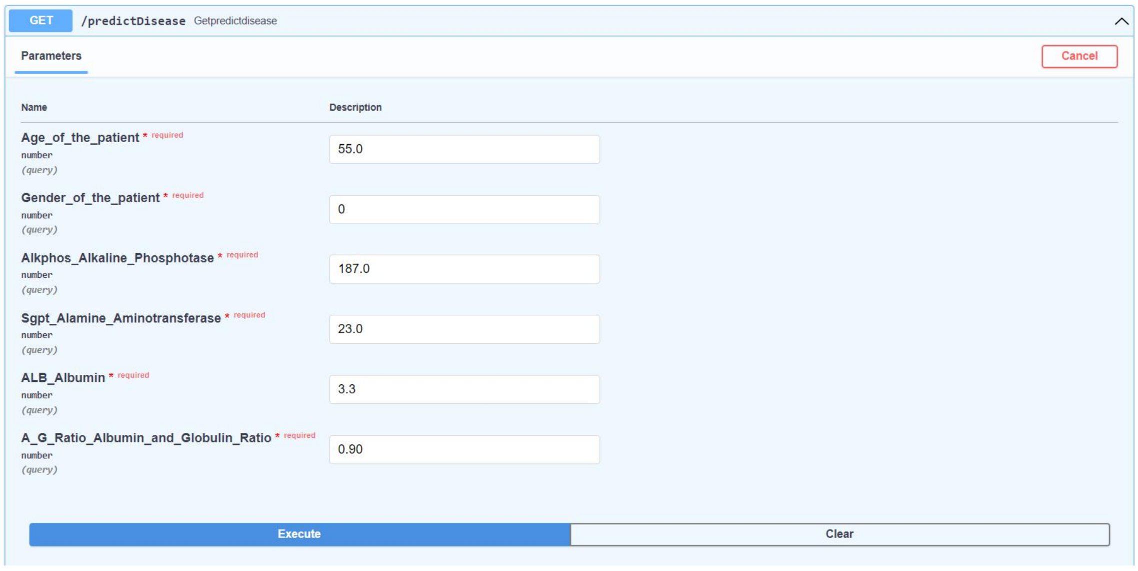
# 9. Test API from Web Browser

# http://127.0.0.1:8000/predictPrice?Area=1400&BedRooms=3&BathRooms=3

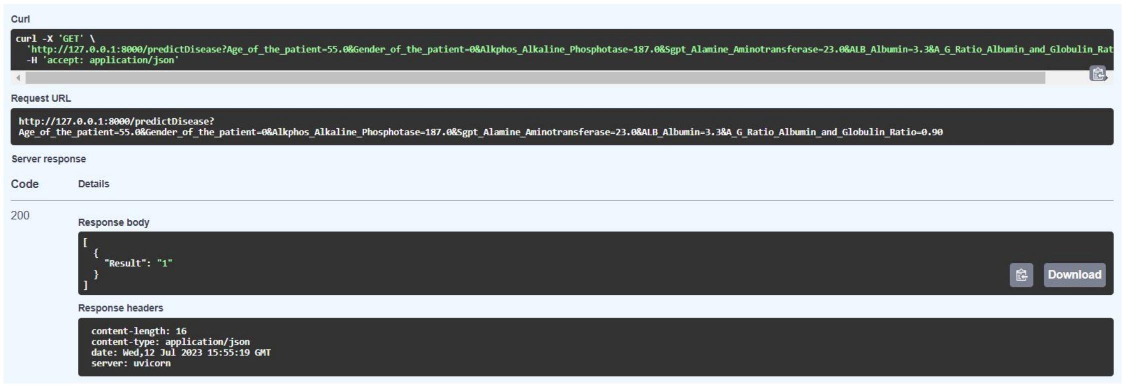


```
INFO:     Will watch for changes in these directories: ['D:\\liverfastapi']
INFO:     Uvicorn running on http://127.0.0.1:8000 (Press CTRL+C to quit)
INFO:     Started reloader process [6772] using StatReload
INFO:     Started server process [13004]
INFO:     Waiting for application startup.
INFO:     Application startup complete.
```

FastAPI



Name	Description
Age_of_the_patient * required number (query)	55.0
Gender_of_the_patient * required number (query)	0
Alkphos_Alkaline_Phosphotase * required number (query)	187.0
Sgpt_Alamine_Aminotransferase * required number (query)	23.0
ALB_Albumin * required number (query)	3.3
A_G_Ratio_Albumin_and_Globulin_Ratio * required number (query)	0.90


Curl

```
curl -X 'GET' \
  'http://127.0.0.1:8000/predictDisease?Age_of_the_patient=55.0&Gender_of_the_patient=0&Alkphos_Alkaline_Phosphotase=187.0&Sgpt_Alamine_Aminotransferase=23.0&ALB_Albumin=3.3&A_G_Ratio_Albumin_and_Globulin_Rat
  -H 'accept: application/json'
```

Request URL

```
http://127.0.0.1:8000/predictDisease?
Age_of_the_patient=55.0&Gender_of_the_patient=0&Alkphos_Alkaline_Phosphotase=187.0&Sgpt_Alamine_Aminotransferase=23.0&ALB_Albumin=3.3&A_G_Ratio_Albumin_and_Globulin_Ratio=0.90
```

Server response

Code Details

200

Response body

```
[
  {
    "Result": "1"
  }
]
```

Response headers

```
content-length: 16
content-type: application/json
date: Wed,12 Jul 2023 15:55:19 GMT
server: uvicorn
```

CHAPTER – 5
CODE

5.1 Source Code

MainActivity.java

```
package com.example.diabetespredictionapp;

import androidx.annotation.NonNull;
import androidx.appcompat.app.AppCompatActivity;
import androidx.fragment.app.Fragment;
import androidx.fragment.app.FragmentTransaction;
import android.os.Bundle;
import android.util.Log;
import android.view.MenuItem;
import com.google.android.material.bottomnavigation.BottomNavigationView;
import com.google.android.material.navigation.NavigationBarView;
public class MainActivity extends AppCompatActivity {
    BottomNavigationView bottomNavigationView;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        bottomNavigationView = findViewById(R.id.bottomNavigationView);
        openFragment(new Home());
        bottomNavigationView.setOnItemSelectedListener(new
NavigationBarView.OnItemSelectedListener()
        {
            @Override
            public boolean onNavigationItemSelected(@NonNull MenuItem item) {
                switch (item.getItemId()) {
                    case R.id.home:
                        openFragment(new Home());
                        return true;
                    case R.id.team:
                        openFragment(new team());
                        return true;
                    case R.id.aboutproject:
                        openFragment(new aboutproject());
                        return true;
                }
                return false;
            }
        });
    }
    private void openFragment(Fragment fragment) {
        FragmentTransaction transaction =
getSupportFragmentManager().beginTransaction();
        transaction.replace(R.id.fragmentView, fragment);
        transaction.addToBackStack(null);
        transaction.commit();
    }
}
```

Aboutproject.java

```
package com.example.diabetespredictionapp;

import android.os.Bundle;
import androidx.fragment.app.Fragment;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.webkit.WebView;
import android.webkit.WebViewClient;
public class aboutproject extends Fragment {
   public aboutproject() {
// Required empty public constructor
   }
   public static aboutproject newInstance(String param1, String param2) {
      aboutproject fragment = new aboutproject();
      return fragment;
   }
   @Override
   public void onCreate(Bundle savedInstanceState) {
      super.onCreate(savedInstanceState);
      getActivity().setTitle("about project");
   }
   @Override
   public View onCreateView(LayoutInflater inflater, ViewGroup container,
                     Bundle savedInstanceState) {
// Inflate the layout for this fragment
      View view = inflater.inflate(R.layout.fragment_team, container, false);
// Find the WebView by its unique ID
      WebView webView = view.findViewById(R.id.webView);
      webView.loadUrl("file:///android_asset/index.html");
      webView.getSettings().setJavaScriptEnabled(true);
      webView.setWebViewClient(new WebViewClient());
      return view;
   }
}
```

Liver.java

```
package com.example.diabetespredictionapp;

public class Diabetes {

    public String Age_of_the_patient;
    public String Gender_of_the_patient;
    public String Alkphos_Alkaline_Phosphotase;
    public String Sgpt_Alamine_Aminotransferase;
    public String ALB_Albumin;
    public String A_G_Ratio_Albumin_and_Globulin_Ratio;

}
```

Home,java

```
package com.example.diabetespredictionapp;

import android.annotation.SuppressLint;
import android.content.Context;
import android.os.Bundle;
import androidx.fragment.app.Fragment;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;
import android.widget.Toast;
import com.android.volley.Request;
import com.android.volley.RequestQueue;
import com.android.volley.Response;
import com.android.volley.VolleyError;
import com.android.volley.toolbox.JsonArrayRequest;
import com.android.volley.toolbox.Volley;
import org.json.JSONArray;
import org.json.JSONException;
import org.json.JSONObject;
public class Home extends Fragment implements View.OnClickListener {
  Context context;
  Diabetes diabetes;
  TextView textLiver;
  EditText textage, textgender, textalkphos,textalamine,textALB,textAlbumin;

  public Home() {
  }

  public static Home newInstance(String param1, String param2) {
    Home fragment = new Home();
    return fragment;
  }

  @Override
  public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    getActivity().setTitle("Liver Disease");
    diabetes = new Diabetes();
  }

  @SuppressLint("MissingInflatedId")
  @Override
  public View onCreateView(LayoutInflater inflater, ViewGroup container,
               Bundle savedInstanceState) {
// Inflate the layout for this fragment
    View view = inflater.inflate(R.layout.fragment_home, container, false);

    this.context = container.getContext();
```

```
        textage = view.findViewById(R.id.textage);
        textgender = view.findViewById(R.id.textgender);
        textalkphos = view.findViewById(R.id.textalkphos);
        textalamine = view.findViewById(R.id.textalamine);
        textALB = view.findViewById(R.id.textALB);
        textAlbumin = view.findViewById(R.id.textAlbumin);
        textLiver = view.findViewById(R.id.textLiver);
        Button btnPredict = view.findViewById(R.id.btnPredict);
        btnPredict.setOnClickListener(this);
        return view;
    }

    @Override
    public void onClick(View view) {
        switch (view.getId()) {
            case R.id.btnPredict:
                diabetes.Age_of_the_patient = textage.getText().toString();
                diabetes.Gender_of_the_patient = textgender.getText().toString();
                diabetes.Alkphos_Alkaline_Phosphotase = textalkphos.getText().toString();
                diabetes.Sgpt_Alamine_Aminotransferase = textalamine.getText().toString();
                diabetes.ALB_Albumin = textALB.getText().toString();
                diabetes.A_G_Ratio_Albumin_and_Globulin_Ratio =
textAlbumin.getText().toString();
                GetpredictOutcome();
                break;
        }
    }

    private void GetpredictOutcome() {
        String url = "http://3.208.241.10/predictDisease?";
        url += "Age_of_the_patient=" + diabetes.Age_of_the_patient + "&";
        url += "Gender_of_the_patient=" + diabetes.Gender_of_the_patient + "&";
        url += "Alkphos_Alkaline_Phosphotase=" +
diabetes.Alkphos_Alkaline_Phosphotase + "&";
        url += "Sgpt_Alamine_Aminotransferase=" +
diabetes.Sgpt_Alamine_Aminotransferase + "&";
        url += "ALB_Albumin=" + diabetes.ALB_Albumin + "&";
        url += "A_G_Ratio_Albumin_and_Globulin_Ratio=" +
diabetes.A_G_Ratio_Albumin_and_Globulin_Ratio;
// creating a new variable for our request queue
        RequestQueue queue = Volley.newRequestQueue(context);
// make json array request and then extracting data from each json object.
        JsonArrayRequest jsonArrayRequest = new
JsonArrayRequest(Request.Method.GET, url, null, new
                Response.Listener<JSONArray>() {
                    @Override
                    public void onResponse(JSONArray response) {
                        try {
                            JSONObject responseObj = response.getJSONObject(0);
                            String returnValue = responseObj.getString("Result");
                            if (returnValue.equals("2")) {
                                textLiver.setText("You have Liver Disease");
                            } else {
                                textLiver.setText("You Dont have Liver Disease");
                            }
                        } catch (JSONException e) {
                            throw new RuntimeException(e);
                        }
                    }
                }, new Response.ErrorListener() {
            @Override
            public void onErrorResponse(VolleyError error) {
                Toast.makeText(context, "Fail to get the data..",
Toast.LENGTH_SHORT).show();
            }
        });
        queue.add(jsonArrayRequest);
    }
}
```

Team.java

```
package com.example.diabetespredictionapp;

import android.os.Bundle;

import androidx.fragment.app.Fragment;

import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.webkit.WebView;
import android.webkit.WebViewClient;


public class team extends Fragment {

    public team() {
        // Required empty public constructor
    }

    public static team newInstance(String param1, String param2) {
        team fragment = new team();
        return fragment;
    }

    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);

        getActivity().setTitle("Team");
    }

    @Override
    public View onCreateView(LayoutInflater inflater, ViewGroup container,
                             Bundle savedInstanceState) {
```

```
        // Inflate the layout for this fragment
        View view = inflater.inflate(R.layout.fragment_team, container, false);

        // Find the WebView by its unique ID
        WebView webView = view.findViewById(R.id.webView);

        webView.loadUrl("https://madlabs.s3.amazonaws.com/aboutmad/ads3bu.html");
        webView.getSettings().setJavaScriptEnabled(true);
        webView.setWebViewClient(new WebViewClient());

        return view;
    }
}
```

activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">
    <com.google.android.material.bottomnavigation.BottomNavigationView
        android:id="@+id/bottomNavigationView"
        android:layout_width="match_parent"
        android:layout_height="60dp"
        app:menu="@menu/bottom_nav_menu"
        app:layout_constraintBottom_toBottomOf="parent"
        app:itemIconSize="40dp"
        tools:layout_editor_absoluteX="1dp" />
    <FrameLayout
        android:id="@+id/fragmentView"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        app:layout_constraintBottom_toTopOf="@+id/bottomNavigationView"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent">
    </FrameLayout>
</androidx.constraintlayout.widget.ConstraintLayout>
```

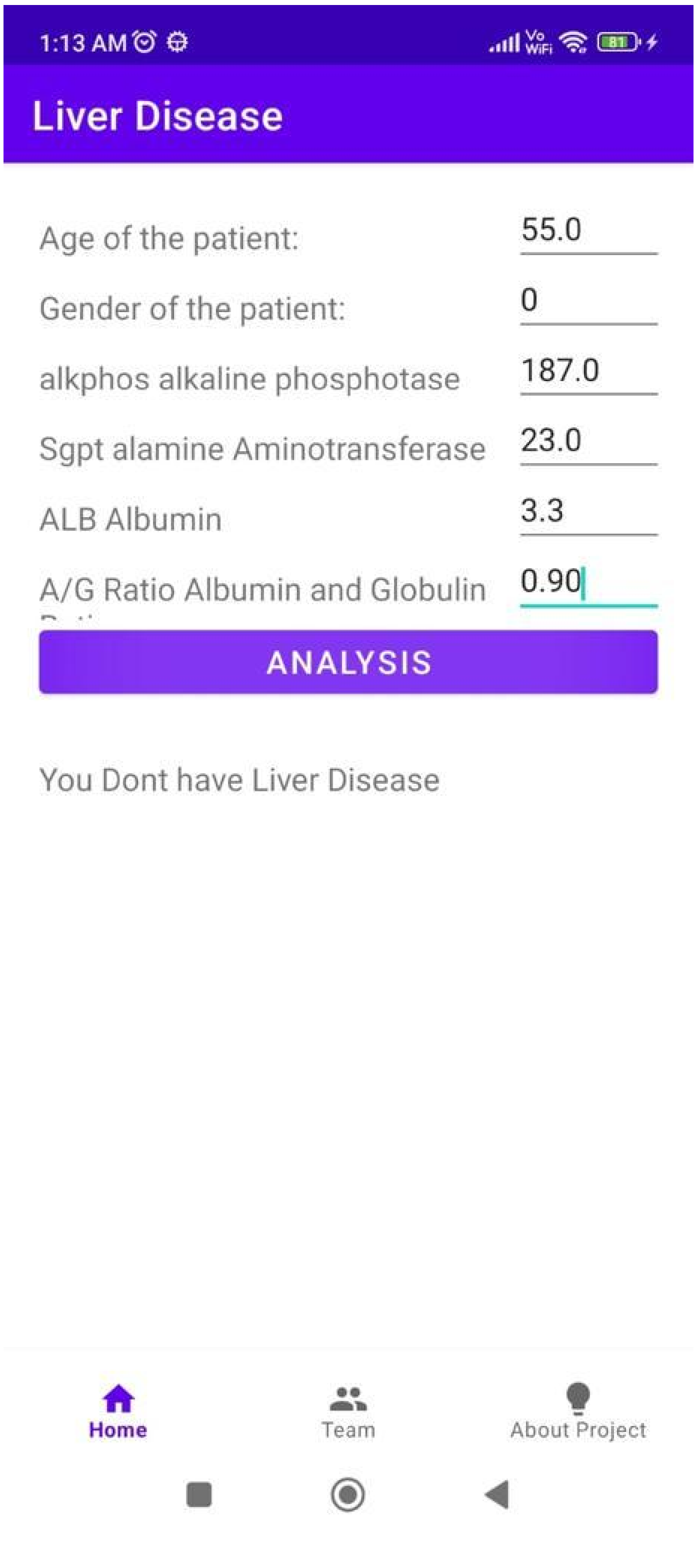
fragment_home.xml

```
<?xml version="1.0" encoding="utf-8"?>

    <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
        xmlns:app="http://schemas.android.com/apk/res-auto"
        xmlns:tools="http://schemas.android.com/tools"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:orientation="vertical"
        android:layout_marginLeft="20dp"
        android:layout_marginRight="20dp"
        android:layout_marginTop="20dp"
        android:layout_marginBottom="20dp"
        tools:context=".MainActivity">
        <LinearLayout
            android:layout_width="match_parent"
            android:layout_height="40dp">
            <TextView
                android:layout_width="260dp"
                android:layout_height="40dp"
                android:text="Age of the patient:"
                android:textSize="18dp"
                ></TextView>
            <EditText
                android:id="@+id/textage"
                android:text=""
                android:inputType="text"
                android:textSize="18dp"
                android:layout_marginLeft="10dp"
                android:layout_width="100dp"
                android:layout_height="40dp">
            </EditText>
        </LinearLayout>
        <LinearLayout
            android:layout_width="match_parent"
            android:layout_height="40dp">

            <TextView
                android:layout_width="260dp"
                android:layout_height="40dp"
                android:text="Gender of the patient:"
                android:textSize="18dp"></TextView>

            <EditText
                android:id="@+id/textgender"
                android:text=""
                android:inputType="text"
                android:textSize="18dp"
                android:layout_marginLeft="10dp"
                android:layout_width="100dp"
                android:layout_height="40dp">
            </EditText>
        </LinearLayout>
        <LinearLayout
            android:layout_width="match_parent"
            android:layout_height="40dp">

            <TextView
                android:layout_width="260dp"
                android:layout_height="40dp"
                android:text="alkphos alkaline phosphotase"
                android:textSize="18dp"></TextView>
                <EditText
                    android:id="@+id/textalkphos"
                    android:text=""
                    android:inputType="text"
                    android:textSize="18dp"
                    android:layout_width="100dp"
                    android:layout_marginLeft="10dp"
                    android:layout_height="40dp">
                </EditText>
            </LinearLayout>
        <LinearLayout
            android:layout_width="match_parent"
            android:layout_height="40dp">

            <TextView
                android:layout_width="260dp"
                android:layout_height="40dp"
                android:text="Sgpt alamine Aminotransferase"
                android:textSize="18dp"></TextView>
            <EditText
                android:id="@+id/textalamine"
                android:text=""
                android:inputType="text"
                android:textSize="18dp"
                android:layout_width="100dp"
                android:layout_marginLeft="10dp"
                android:layout_height="40dp">
            </EditText>
        </LinearLayout>
        <LinearLayout
            android:layout_width="match_parent"
            android:layout_height="40dp">

            <TextView
                android:layout_width="260dp"
                android:layout_height="40dp"
                android:text="ALB Albumin"
                android:textSize="18dp"></TextView>
            <EditText
                android:id="@+id/textALB"
                android:text=""
                android:inputType="text"
                android:textSize="18dp"
                android:layout_width="100dp"
                android:layout_marginLeft="10dp"
                android:layout_height="40dp">
            </EditText>
        </LinearLayout>
        <LinearLayout
            android:layout_width="match_parent"
            android:layout_height="40dp">

            <TextView
                android:layout_width="260dp"
                android:layout_height="40dp"
                android:text="A/G Ratio Albumin and Globulin Ratio"
                android:textSize="18dp"></TextView>
            <EditText
                android:id="@+id/textAlbumin"
                android:text=""
                android:inputType="text"
                android:textSize="18dp"
                android:layout_width="100dp"
                android:layout_marginLeft="10dp"
                android:layout_height="40dp">
            </EditText>
        </LinearLayout>

            <Button
                android:id="@+id/btnPredict"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:onClick="predict"
                android:text="ANALYSIS"
                android:textSize="18dp"></Button>

            <TextView
                android:id="@+id/textLiver"
                android:layout_width="match_parent"
                android:layout_height="40dp"
                android:layout_marginTop="30dp"
                android:text=""
                android:textSize="18dp" />
        </LinearLayout>
```

index.html

```
<html>
<h1>LiverDisease</h1>
<h4>Your Personal liverDisease Guidence</h4>
<p align="justify">This project aims to predict patient has liver Disease or not using
different supervised machine learning methods including: Random Forest classifier,
Decision Trees and SVM. Based on a number of factors such as the user’s
Age,gender,ALB Albumin etc</p>
<img src="https://ad-69project.s3.amazonaws.com/diabetes.jpg.jpeg"  width="330"
height="350">
</body>
</html>
```

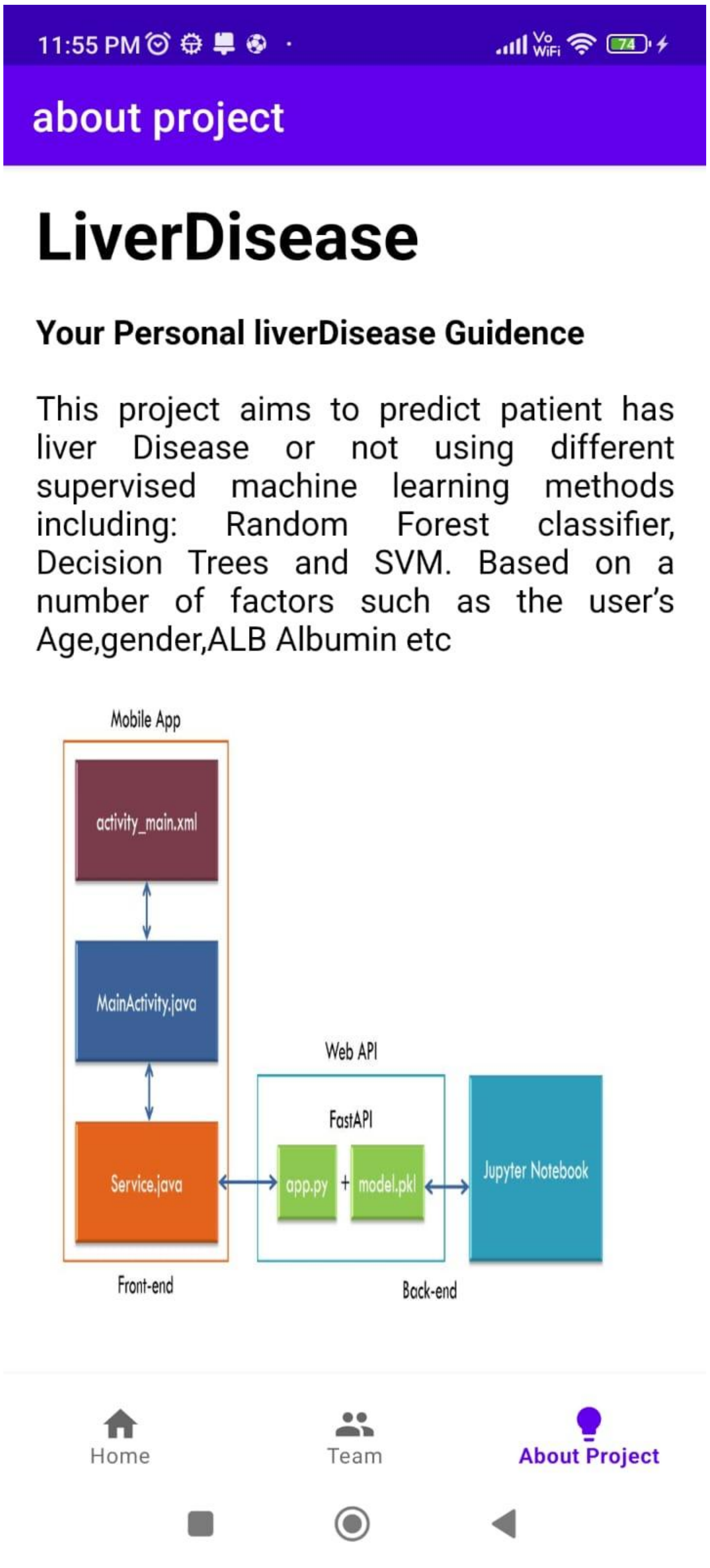
5.2 Screenshot of Application



1:13 AM

Age of the patient:	55.0
Gender of the patient:	0
alkphos alkaline phosphotase	187.0
Sgpt alamine Aminotransferase	23.0
ALB Albumin	3.3
A/G Ratio Albumin and Globulin	0.90

You Dont have Liver Disease

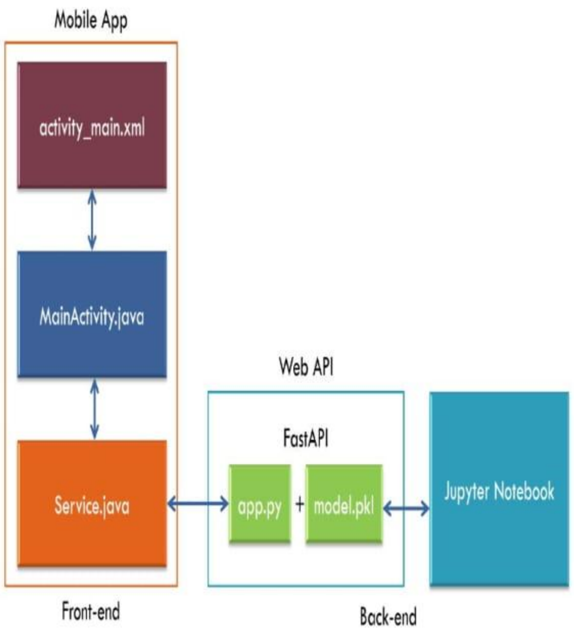

11:55 PM

about project

LiverDisease

Your Personal liverDisease Guidence

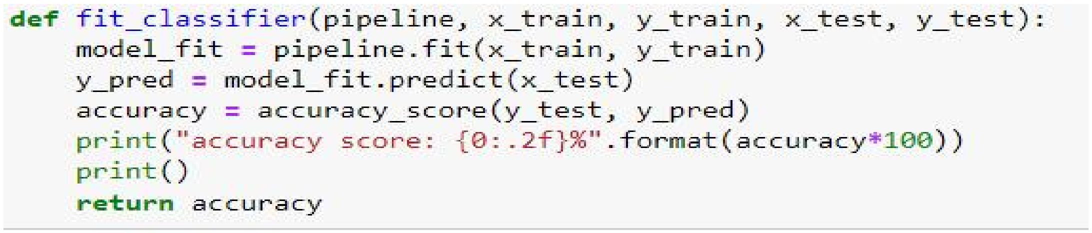
This project aims to predict patient has liver Disease or not using different supervised machine learning methods including: Random Forest classifier, Decision Trees and SVM. Based on a number of factors such as the user's Age,gender,ALB Albumin etc



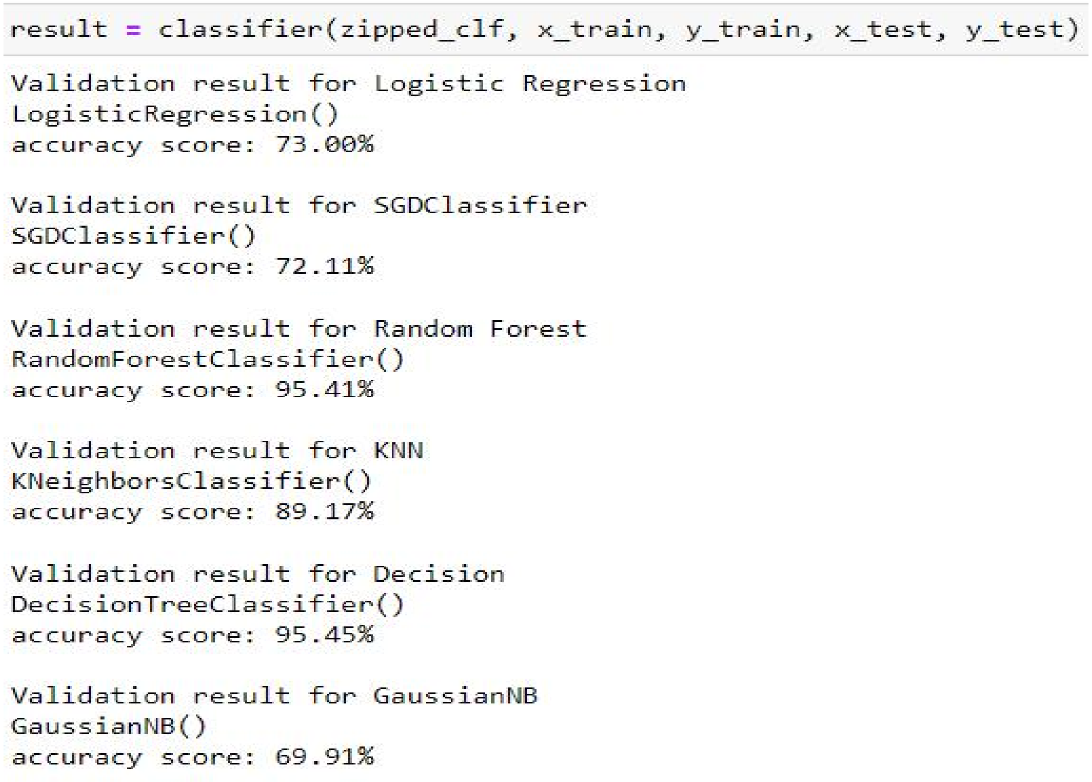
Home Team **About Project**

CHAPTER - 6
RESULTS & CONCLUSION

6.1 Result



```
def fit_classifier(pipeline, x_train, y_train, x_test, y_test):
    model_fit = pipeline.fit(x_train, y_train)
    y_pred = model_fit.predict(x_test)
    accuracy = accuracy_score(y_test, y_pred)
    print("accuracy score: {0:.2f}%".format(accuracy*100))
    print()
    return accuracy
```



```
result = classifier(zipped_clf, x_train, y_train, x_test, y_test)
```

```
Validation result for Logistic Regression
LogisticRegression()
accuracy score: 73.00%

Validation result for SGDClassifier
SGDClassifier()
accuracy score: 72.11%

Validation result for Random Forest
RandomForestClassifier()
accuracy score: 95.41%

Validation result for KNN
KNeighborsClassifier()
accuracy score: 89.17%

Validation result for Decision
DecisionTreeClassifier()
accuracy score: 95.45%

Validation result for GaussianNB
GaussianNB()
accuracy score: 69.91%
```

The results of the proposed work have been obtained by implementing feature selection techniques and different classifiers. The judgment of different results was done based on the following categories. First, the results of different classifiers are compared the on basis of correctly classified instances with feature selection techniques shown in table 3 and without using feature selection techniques shown in table 2. Secondly, the parameters like Kappa statistic value, mean absolute error are compared using on 10-fold cross-validation testing option. Finally, the execution time of different classifiers also compares for both the above techniques shown in table 4. The table 2 show the results of different classifiers without using feature selection technique. Logistic Regression shows the higher accuracy value 73.00 and (Naive bayas) least value 69.91 percentage. The shows the results of different classifiers using feature selection technique. Using this technique Decision Tree shows the higher accuracy value 95.45 and (SGDclassifer) shows 72.11 percentage.

6.2 Conclusion

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 100 % accuracy for SVM model, 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 90 % 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.

REFERENCES

[1] S. Karthim, J. A. Priyadarshin and B. K. Tripathi. (2011), “Classification and rule Extraction using Rough Set for Diagnosis of Liver Disease and its Types”, Advances in Applied Science Research, vol. 2 no.3, pp. 334-345.

[2] W. Nanyue, Y. Youhua, H. Dawei, X. Bin, L. Jia, L. Tongda X. Liyuan, S, Zengyu, C. Yanping and W. Jia. (2015), “Pulse Diagnosis Signals Analysis of Fatty Liver Disease and Cirrhosis Patients by Using Machine Learning,” The Scientific World Journal, vol. 2015, Article ID 859192, pp. 1-9.

[3] B. Y. Ramana, M. S. P. Babu and N. B. Venkateswarlu. (2011), "A Critical Study of Selected Classification Algorithms for Liver Disease Diagnosis", International Journal of Database Management Systems (IJDMS), vol.3, no.2, pp. 101-114.

[4] I. Arshad, C. Dutta, T. Choudhury, and A. Thakral. (2018), "Liver Disease Detection Due to Excessive Alcoholism Using Data Mining Techniques." In IEEE International Conference on Advances in Computing and Communication Engineering (ICACCE), pp. 163-168.

[5] H. Jin,S. Kim and J. Kim. (2014),"Decision Factors on Effective Liver Patient Data Prediction", International Journal of Bio-Science and BioTechnology, vol.6, no.4, pp. 167-178.

[6] J. Singh, A. Kamra, H. Singh. (2014), “Recent Trends in Data Mining: A Review”, In preceding of 3rd International Conference on Biomedical Engineering & Assistive Technologies, Chandigarh pp. 130-144.

[7] E. Frank, M. A. Hall, and I. H. Witten, The WEKA Workbench. Online Appendix for. (2016) "Data Mining: Practical Machine Learning Tools and Techniques", Morgan Kaufmann, Fourth Edition.

[8] A. Arora, S. Bagga and R. S. Cheema. (2012), “Distributed cluster processing to evaluate interlaced run-length compression schemes”. International journal of computer applications, vol. 46, no. 6.

[9] A. M. Hall and A. L. Smith. (1999), “Feature Selection for Machine Learning: Comparing a Correlation-Based Filter Approach to the Wrapper”, In Proceedings of the Twelfth International Florida Artificial Intelligence Research Society Conference, AAAI Press pp. 235- 239.

[10] P. Kuppan, N. Manoharan. (2017) “A Tentative analysis of Liver Disorder using Data Mining Algorithms J48, Decision Table and Naive Bayes”, International Journal of Computing Algorithm, vol. 6, no. 1, pp. 2278-239.

[11] A. Gulia, R. Vohra, P. Rani. (2014), “Liver Patient Classification Using Intelligent Techniques,” International Journal of Computer Science and Information Technologies (IJCSIT), vol. 5, no. 4, pp. 5110-5115.
[12] Y. Kumar and G. Sahoo. (2013) “Prediction of different types of liver diseases using rule-based classification model”, Technology and Health Care, vol. 21, pp. 417-432.
[13] M. Pasha, M. Fatima. (2017), “Comparative Analysis of Meta Learning Algorithms for Liver Disease Detection”, Journal of Software, Vol. 12, No. 12, pp 923-933.
[14] M. Abdar, N.Y. Yen and J. CS. J. Hung. (2017), “Improving the Diagnosis of Liver Disease Using Multilayer Perceptron Neural Network and Boosted Decision Trees” Journal of Medical and Biological Engineering, vol. 4, no. 22, pp. 1-13.
[15] A. El-Shafeiy, L. Ali. Engy, El-Desouky and S. M. Elghamrawy. (2018) "Prediction of Liver Diseases Based on Machine Learning Technique for Big Data." In International Conference on Advanced Machine Learning Technologies and Applications, pp. 362-374. Springer, Cham.
[16] S. Vijayarani, and S. Dhayanand. (2015) "Liver disease prediction using SVM and Naïve Bayes algorithms." International Journal of Science, Engineering and Technology Research (IJSETR) vol. 4, no. 4, pp. 816-820.
[17] B. V. Ramana, M. S. P. Babu and N. B. Venkateswarlu (2012). “Indian Liver Patientdataset.”[Online]Available:https://archive.ics.uci.edu/ml/datasets/ILPD+(Indian+Liver+Patient+Dataset) [accessed on July, 2018].
[18] M. B. Priya, P. L. Juliet, P.R. Tamilselvi. (2018), “Performance Analysis of Liver Disease Prediction Using Machine Learning Algorithms”, International Research Journal of Engineering and Technology, vol. 5 no. 1, pp. 206-211.
[19] M. Dash, H. Liu. (1997), “Feature Selection for Classification,” Intelligent Data Analysis, Elsevier, pp. 131 -156.
[20] B. M. S. Prasad, M. Ramjee, S. Katta, and K. Swapna. (2016), "Implementation of partitional clustering on ILPD dataset to predict liver disorders." In Software Engineering and Service Science (ICSESS), 7th IEEE International Conference, pp. 1094-1097.
[21] S. Bagga, A. Girdhar, M. C. Trivedi. (2017), “SPMD based time sharing intelligent approach for image denoising”, Journal of Intelligent & Fuzzy Systems, vol.32, no. 5, pp. 3561-3573.
[22] H. Kaur, S. Bagga and A. Arora. (2015), “RMI approach to cluster based Winograd's variant of Strassen's method”. In MOOCs, Innovation and Technology in Education (MITE), IEEE 3rd International Conference, pp. 156-162. healthcare: Encouraging clinical experience of simple e-prescription system and m-health system development for

mother and childcare”. In e-Health Networking Applications and Services (Healthcom) 13th IEEE International Conference, pp. 102-105.

[23] S. Skevoulis, J. Campedelli, K. Holdsworth, J. Verel, and S. Tavales. (2009) “Engineering a Software Supported Health Risk Appraisal Method: A Joint Effort between Academia and Health Care Industry”. In Software Engineering Education and Training, IEEE CSEET'09 22nd Conference, pp. 113-116.

[24] J.H. Weber-Jahnke, M. Price and J. Williams. (2013), “Software engineering in health care: Is it really different? And how to gain impact”. In Proceedings of the 5th International Workshop on Software Engineering in Health Care, IEEE Press, pp. 1-4.