Prediction And Analysis Of Liver Patient Data Using Machine Learning

AN INDUSTRY ORIENTED MINI REPORT
Submitted to

JAWAHARLAL NEHRU TECNOLOGICAL UNIVERSITY, HYDERABAD

In partial fulfillment of the requirements for the award of the degree of

BACHELOR OF TECHNOLOGY

In

COMPUTER SCIENCE AND ENGINEERING(AI&ML)

Submitted By

MOTHE AAKANKSHA **21UK1A66C6**
GUMUDAVELLI SANJAY **21UK1A66A0**
JUTTUKONDA VINAYKUMAR **21UK1A66C5**
DEVARA RAVITEJA **21UK1A6698** Assistant Professor

Under the guidance of

Ms. K.SOUMYA

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

VAAGDEVI ENGINEERING COLLEGE
Affiliated to JNTUH, HYDERABAD
BOLLIKUNTA, WARANGAL (T.S) – 506005

DEPARTMENT OF

COMPUTER SCIENCE AND ENGINEERING(AI&ML)

VAAGDEVI ENGINEERING COLLEGE(WARANGAL)

CERTIFICATE OF COMPLETION

INDUSTRY ORIENTED MINI PROJECT

This is to certify that the UG Project Phase-1 entitled "Prediction And Analysis Of Liver Patient Data Using Machine Learning" MOTHE AAKANKSHA (21UK1A66C6), GUMUDAVELLI SANJAY (21UK1A66A0), JUTTUKONDA VINAY KUMAR (21UK1A66C5), DEVARA RAVITEJA (21UK1A6698) in partial fulfillment of the requirements for the award of the degree of Bachelor of Technology in Computer Science & Engineering to Jawaharlal Nehru Technological University Hyderabad during the academic year 2024- 2025.

Project Guide	**HOD**
Ms. . K.SOUMYA	
	Dr. K. Sharmila
(Assistant Professor)	(Professor)

External

ACKNOWLEDGEMENT

We wish to take this opportunity to express our sincere gratitude and deep sense of respect to our beloved **Dr. P. PRASAD RAO,** Principal, Vaagdevi Engineering College for making us available all the required assistance and for his support and inspiration to carry out this UG Project Phase-1 in the institute.

We extend our heartfelt thanks to **Dr. K. SHARMILA**, Head of the Department of CSE, Vaagdevi Engineering College for providing us necessary infrastructure and thereby giving us freedom to carry out the UG Project Phase-1.

We express heartfelt thanks to Smart Bridge Educational Services Private Limited, for their constant supervision as well as for providing necessary information regarding the UG Project Phase-1 and for their support in completing the Project Phase-1.

We express heartfelt thanks to the guide, **R,** Assistant professor Department CSE for his constant support and giving necessary guidance for completion of this UG Project Phase-1.

Finally, we express our sincere thanks and gratitude to my family members, friends for their encouragement and outpouring their knowledge and experience throughout thesis.

MOTHE AAKANKSHA	**(21UK1A66C6)**
GUMUDAVELLI SANJAY	**(21UK1A66A0)**
JUTTUKONDA VINAYKUMAR	**(21UK1A66C5)**
DEVARA RAVITEJA	**(21UK1A6698)**

ABSTRACT

Abstract:

Liver disease is a significant health issue globally, affecting millions of people each year. Early detection and accurate prediction of liver disorders are crucial for effective treatment and management. Machine learning techniques offer powerful tools for analysing medical data and predicting patient outcomes. This study focuses on the application of machine learning algorithms to predict the likelihood of liver disease based on patient data. The dataset used includes various clinical and demographic features such as age, gender, symptoms, and biochemical test results. Initially, exploratory data analysis (EDA) is conducted to understand the distribution and relationships within the dataset. This step helps in identifying trends, outliers, and potential correlations between variables.

TABLE OF CONTENTS:-

1.INTRODUCTION

1.1.OVERVIEW

Overview:

Liver is the largest organ in the abdomen. This is the primary organ for maintaining the chemicals like glucose, balancing so many nutrients, fat, vitamins, cholesterol and hormones. Liver disease prevents normal liver function. Mainly due to the large amount of alcohol consumption liver disease develops. Early detection of liver disease using classification algorithms is an effective activity that can help doctors diagnose the disease in a short period of time. Early detection of liver disease is a daunting task for doctors. The main purpose of our project is to analyse the parameters of the various classification algorithms and compare their predictive accuracy to determine the best stage for determining liver disease.

1.2.PURPOSE

The purpose of predicting and analysing liver patient data using machine learning can be summarized into several key objectives and benefits:

1. **Early Detection and Diagnosis:** Machine learning models can analyse complex patterns in patient data to identify early signs and symptoms of liver disease. Early detection allows for timely intervention and treatment, which can significantly improve patient outcomes and reduce disease progression.
2. **Improved Accuracy and Reliability:** Traditional diagnostic methods for liver diseases often rely on subjective interpretations or invasive procedures. Machine learning algorithms can enhance diagnostic accuracy by objectively analysing a wide range of patient data, including clinical symptoms and biochemical markers.
3. **Personalized Medicine:** By leveraging patient-specific data, machine learning models can facilitate personalized treatment plans tailored to individual characteristics and disease risks. This approach helps healthcare providers optimize treatment strategies and improve patient care outcomes.

4. **Predictive Analytics:** Machine learning enables predictive modelling to forecast disease progression and patient outcomes based on historical data. This capability supports clinicians in making informed decisions about treatment options and resource allocation.
5. **Clinical Decision Support:** Machine learning algorithms can serve as decision support tools for healthcare professionals, providing insights into patient risk stratification, prognosis, and treatment response. This aids in clinical decision-making and enhances patient care quality.
6. **Research Advancements:** Analysing large-scale datasets using machine learning can uncover novel insights into the ethology, progression, and management of liver diseases. This research contributes to advancing scientific knowledge and developing new therapeutic strategies.
7. **Cost-Efficiency:** By enabling earlier detection and proactive management of liver diseases, machine learning can potentially reduce healthcare costs associated with late-stage interventions and hospitalizations.

2.LITERATURE SURVEY

2.1 EXISTING PROBLEM

The existing problems in predicting and analysing liver patient data using machine learning primarily revolve around several key challenges:

1. **Data Quality and Availability:** One of the primary challenges is the quality and availability of data. Medical datasets used for training machine learning models may be incomplete, inconsistent, or contain missing values. This can affect the reliability and accuracy of predictions and analyses.
2. **Imbalanced Data:** In healthcare datasets, especially for rare diseases or specific patient subgroups, there may be imbalances between classes (e.g., diseased vs. healthy). Imbalanced data can lead to biased models that Favor the majority class and perform poorly in predicting minority classes.
3. **Interpretability of Models:** Machine learning models, particularly complex ones like deep learning algorithms, often lack interpretability. Understanding how and why a model makes specific predictions can be challenging, especially in medical contexts where interpretability is crucial for clinical decision-making.
4. **Generalization to Diverse Populations:** Models trained on one population or dataset may not generalize well to diverse patient populations with different demographics, genetic backgrounds, or environmental factors. Ensuring the robustness and generalizability of models across different populations is essential for their practical application in healthcare settings.
5. **Ethical and Legal Considerations:** Using patient data for machine learning raises ethical concerns regarding patient privacy, consent, and data security. Adhering to regulatory frameworks such as GDPR (General Data Protection Regulation) and ensuring ethical guidelines are followed is critical to maintaining patient trust and compliance with legal requirements.
6. **Integration into Clinical Workflow:** Successfully integrating machine learning predictions into clinical practice requires overcoming barriers such as resistance to adopting new technologies, integrating with existing electronic health record systems, and ensuring healthcare providers understand and trust the outputs of these models.
7. **Validation and Validation Bias:** Proper validation of machine learning models is essential to ensure their reliability and performance. Validation bias, where models perform well on training data but poorly on unseen data, must be addressed through rigorous validation techniques such as cross-validation and external validation on independent datasets.

Addressing these existing problems requires interdisciplinary collaboration between healthcare professionals, data scientists, and policymakers. Overcoming these challenges will lead to more accurate, reliable, and interpretable machine learning models for predicting and analysing liver patient data, ultimately improving patient care and clinical outcomes in liver disease management.

2.2 PROPOSED SOLLUTION

Proposed solutions for improving the prediction and analysis of liver patient data using machine learning involve addressing the existing challenges effectively. Here are some key strategies:

1. **Data Quality Improvement:**
    - **Data Preprocessing:** Implement robust data preprocessing techniques to handle missing values, outliers, and inconsistencies in the medical datasets.
    - **Feature Engineering:** Explore advanced feature engineering methods to extract meaningful features from raw data, enhancing the predictive power of machine learning models.
2. **Handling Imbalanced Data:**
    - **Resampling Techniques:** Use techniques such as oversampling (e.g., SMOTE) or under sampling to balance class distributions in the dataset, ensuring that models are trained effectively on both diseased and healthy patient samples.
    - **Cost-Sensitive Learning:** Incorporate cost-sensitive learning approaches to assign higher penalties to misclassifications in the minority class, improving model performance on imbalanced data.
3. **Model Interpretability:**
    - **Feature Importance Analysis:** Conduct thorough feature importance analysis to understand which variables contribute most significantly to predictions of liver disease. This helps in clinical interpretation and decision-making.
    - **Model Explainability:** Utilize explainable AI techniques (e.g., SHAP values, LIME) to provide transparent explanations for individual predictions made by machine learning models, enhancing trust and usability in clinical settings.
4. **Generalization to Diverse Populations:**
    - **Transfer Learning:** Explore transfer learning approaches where models pre-trained on data from one population are fine-tuned or adapted to perform well on diverse patient

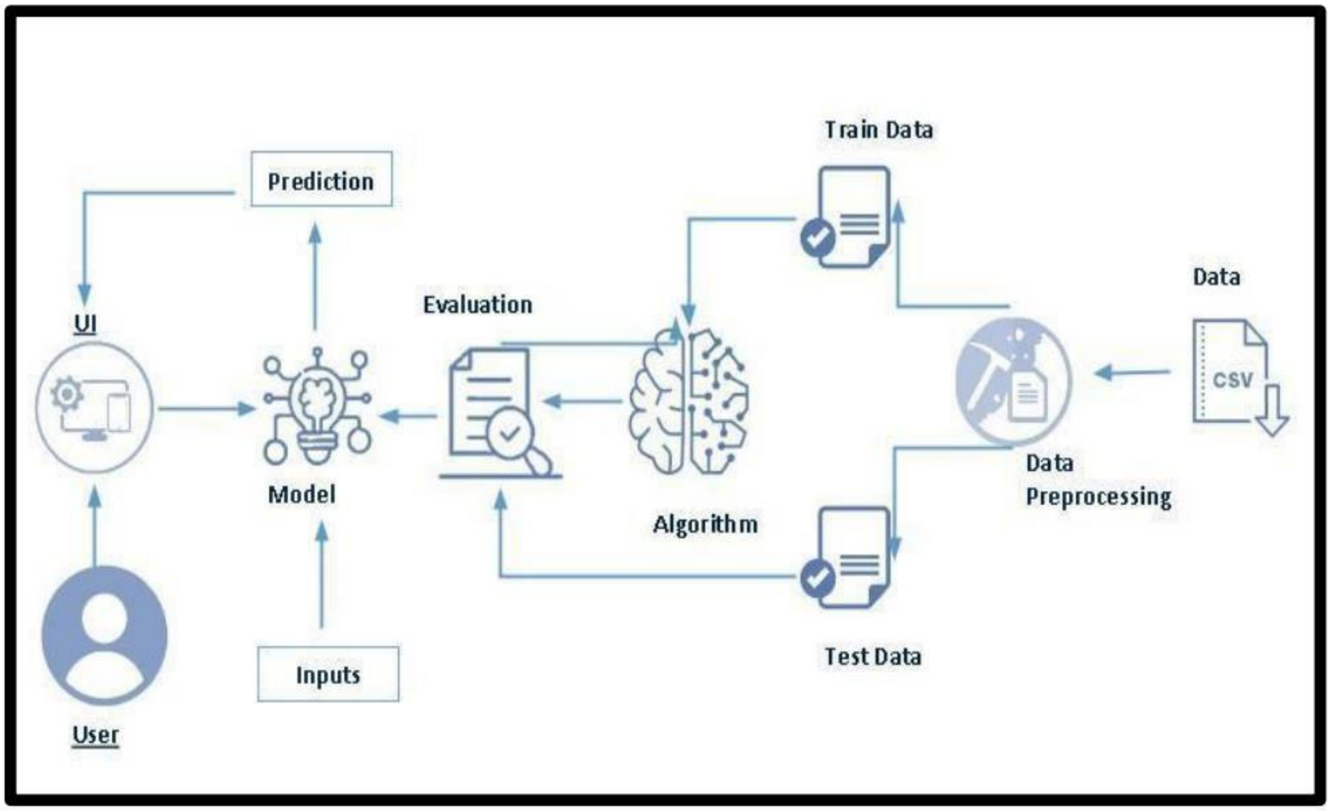
populations, accounting for variations in demographics, genetics, and environmental factors.

- **External Validation:** Validate machine learning models on independent datasets from diverse populations to ensure robustness and generalizability across different patient cohorts.

5. **Ethical and Legal Compliance:**
   - **Data Privacy and Security:** Implement stringent data privacy measures and adhere to regulatory guidelines (e.g., GDPR) to protect patient confidentiality and ensure secure handling of medical data.
   - **Ethical Considerations:** Establish clear guidelines and protocols for obtaining informed consent from patients for using their data in research and model development, respecting ethical principles and patient rights.
6. **Integration into Clinical Workflow:**
   - **User-Centered Design:** Involve healthcare providers in the design and development of machine learning tools, ensuring user-friendly interfaces and seamless integration into existing clinical workflows.
   - **Education and Training:** Provide training programs and resources to healthcare professionals on how to interpret and utilize machine learning predictions effectively in clinical decision-making.
7. **Validation and Transparency:**
   - **Robust Validation:** Conduct rigorous validation studies, including cross-validation and external validation on diverse datasets, to assess model performance and reliability across different scenarios.
   - **Transparent Reporting:** Ensure transparent reporting of methodologies, results, and limitations of machine learning models in scientific publications and clinical reports, fostering trust and reproducibility in research findings.

3.THEORITICAL ANALYSIS

3.1. BLOCK DIAGRAM



3.2. SOFTWARE DESIGNING

The following is the Software required to complete this project:

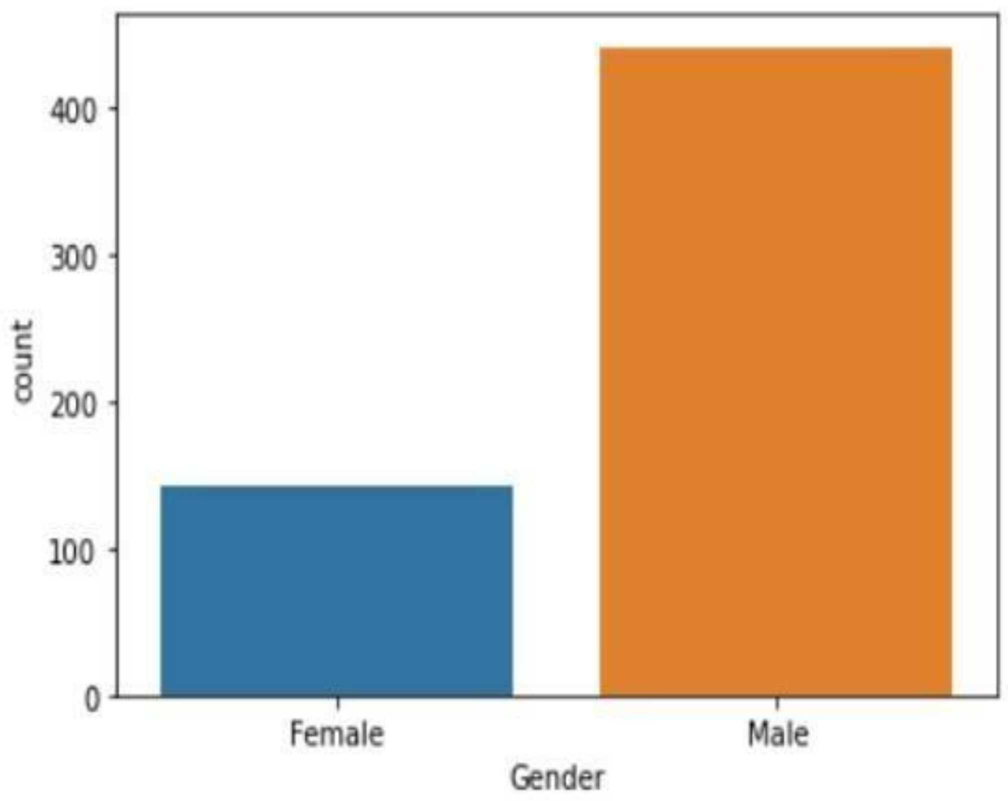
- **Google Colab**: Google Colab as the development and execution environment. It offers a cloudbased Jupyter Notebook environment with Python libraries and hardware acceleration.

- **Dataset (CSV File)**: The dataset in CSV format is containing historical health records, patient demographics, symptoms, medical history, and diagnostic outcomes. This dataset is essential for training and testing the predictive model.
- **Data Preprocessing Tools**: Python libraries like NumPy, Pandas, and Scikit-learn will be Handling missing data, Feature scaling and normalization, Encoding categorical variables, Data cleaning and outlier detection.
- **Feature Selection/Engineering**: Feature selection techniques to identify relevant features that influence disease prediction. Feature engineering might involve creating new features from existing ones to enhance model performance.
- **Model Selection and Training:** classification algorithms such as Logistic Regression, Decision Trees, Random Forests, Support Vector Machines (SVM), or Gradient Boosting Machines (GBM). Train multiple models using cross-validation techniques to optimize hyperparameters and avoid overfitting.
- **Model Accuracy Evaluation**: After model training, Evaluate model performance using metrics like accuracy, precision, recall, F1-score, and ROC-AUC (Receiver Operating Characteristic - Area Under Curve). Utilize techniques such as cross-validation to ensure the model generalizes well to unseen data.
- **UI Based on Flask Environment**: Flask, Develop a web-based user interface using Flask to interact with the trained model.

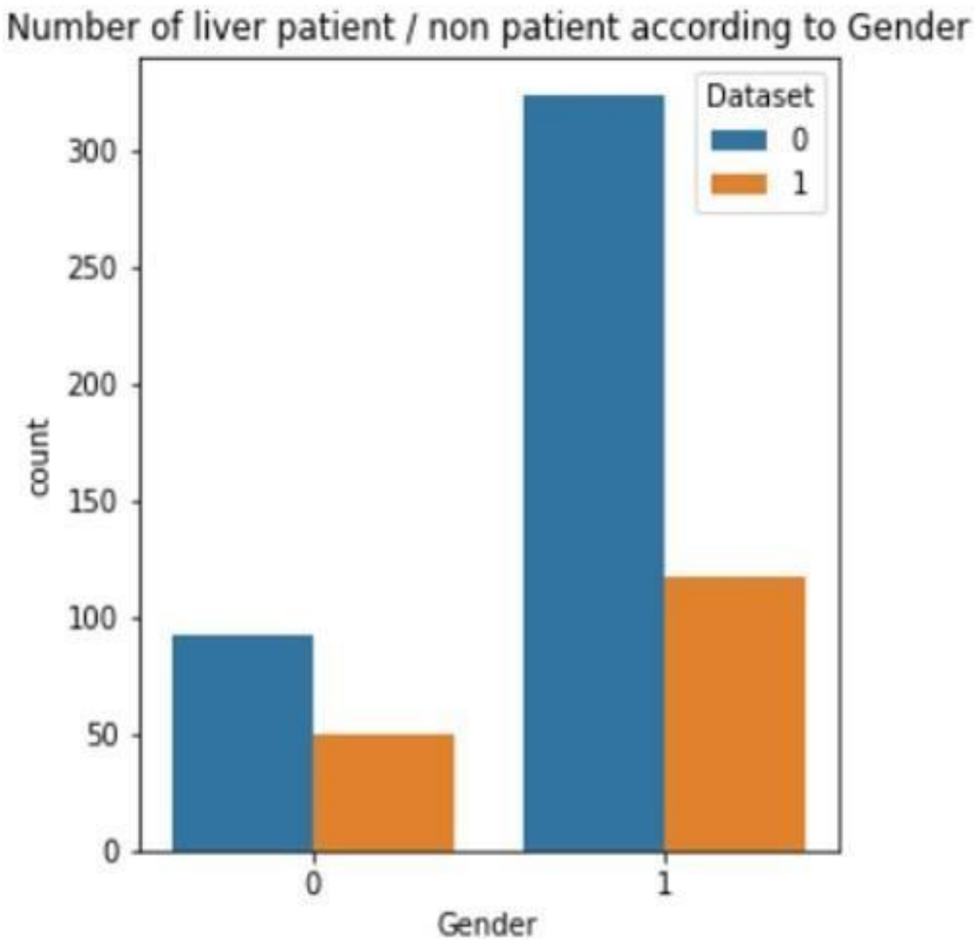
4.EXPERIMENTAL INVESTIGATION

Plotting the number of male and female patients:

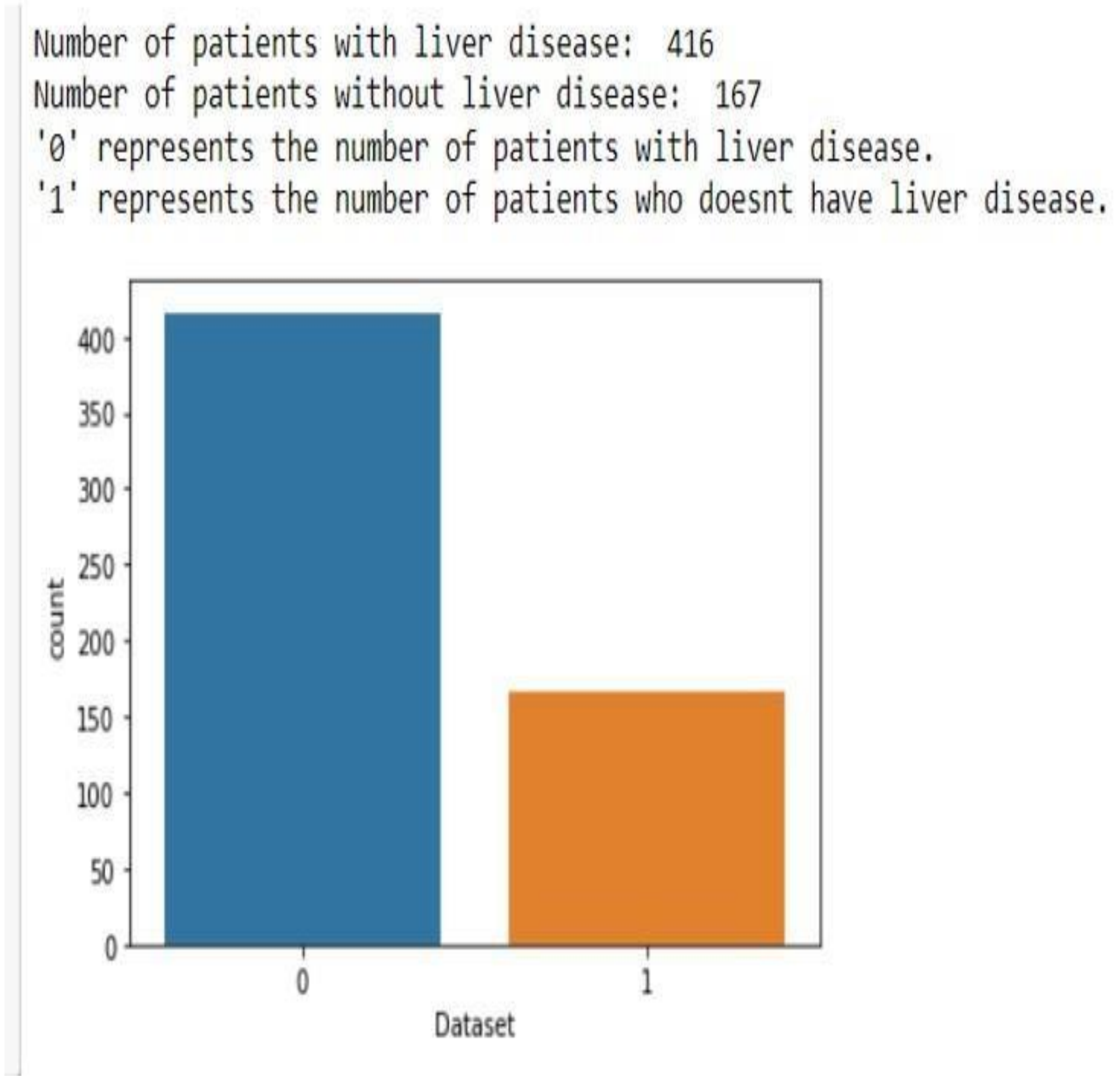
```
Number of female patients:  142
Number of male patients:  441
'0' represents the number of female patients.
'1' represents the number of male patients.
```



Plotting the number of patient / nonpatient according to gender:

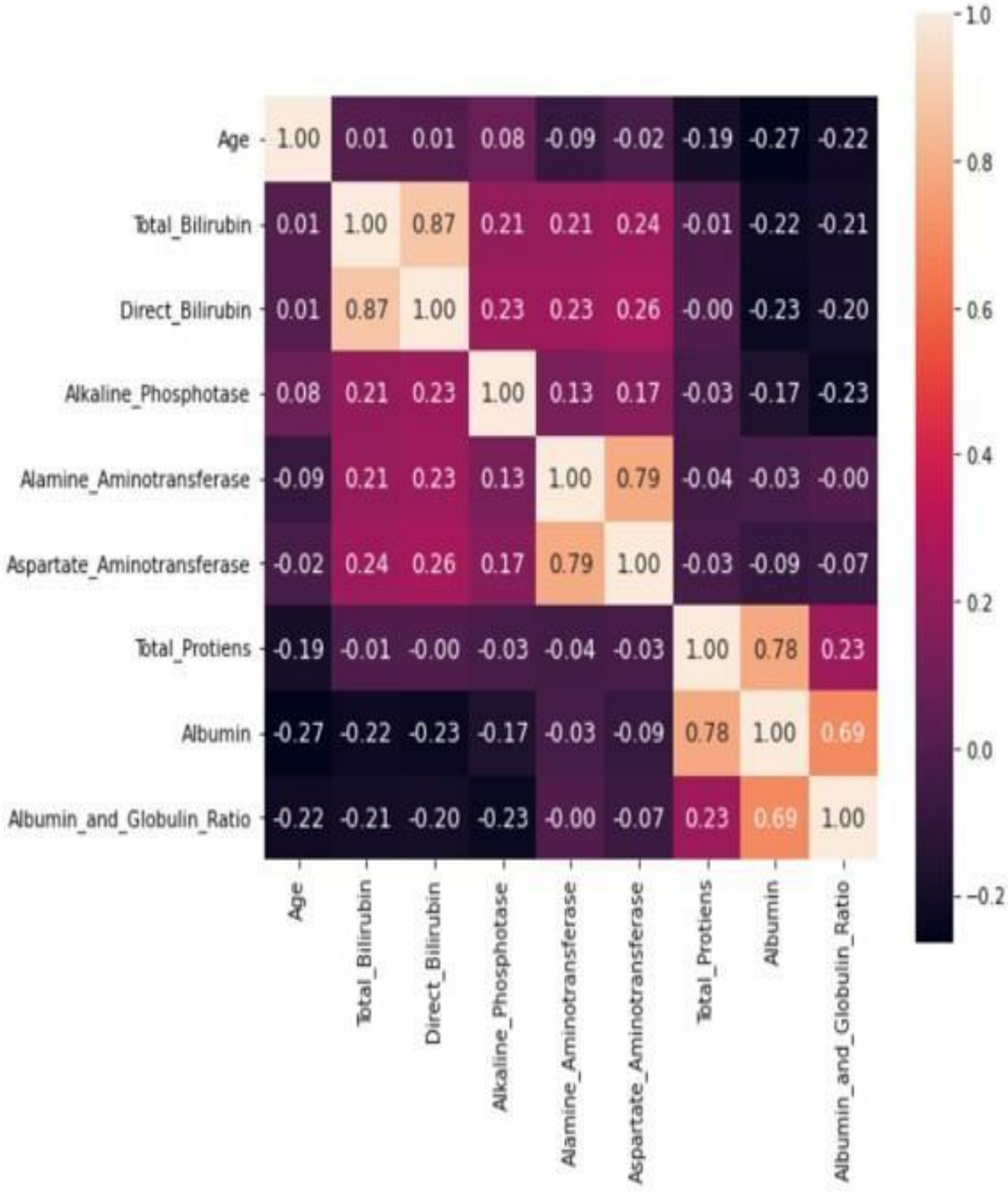


Plotting the number of people with liver disease vs the number of people who doesn't have liver disease:

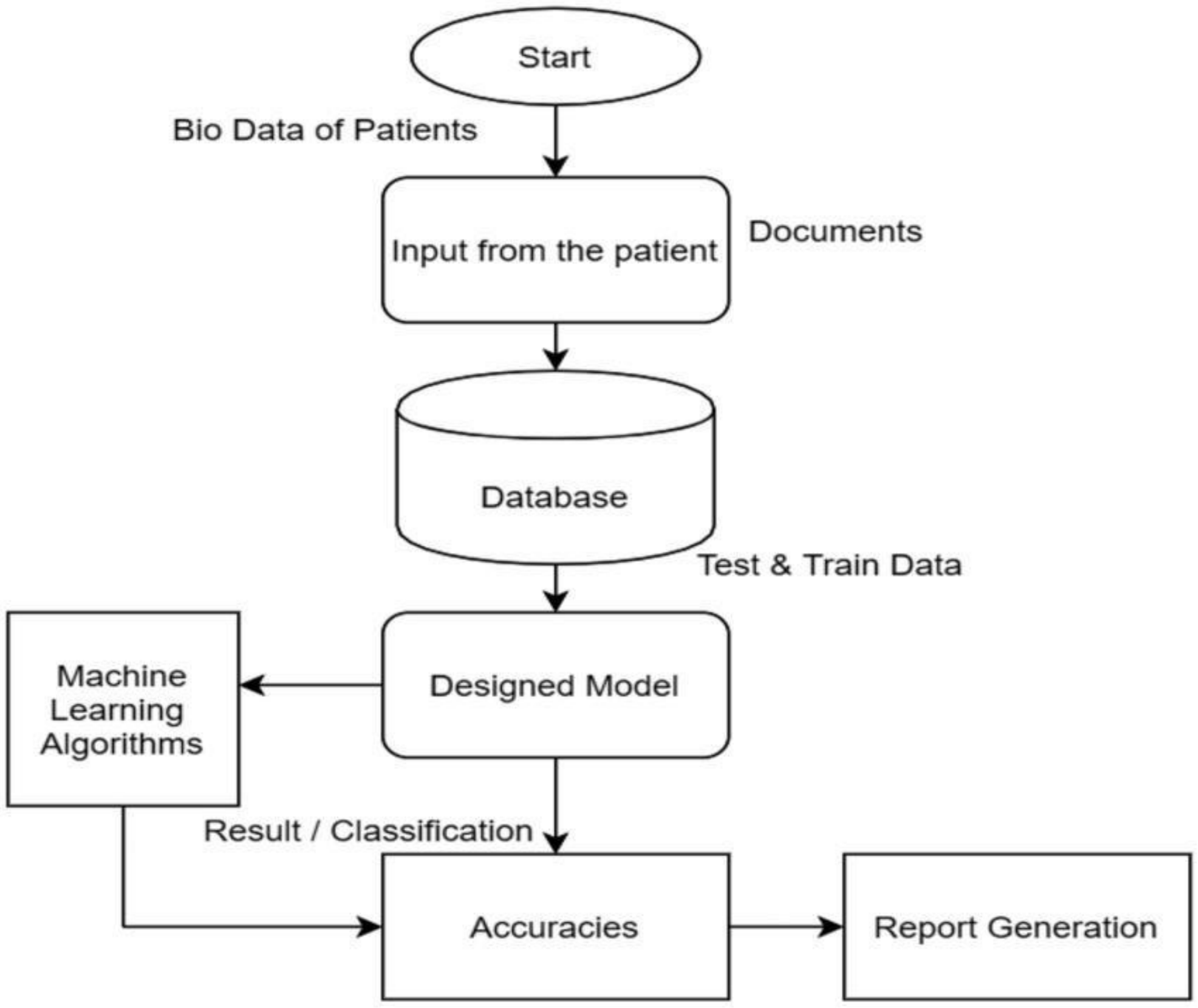


```
Number of patients with liver disease:  416
Number of patients without liver disease:  167
'0' represents the number of patients with liver disease.
'1' represents the number of patients who doesnt have liver disease.
```

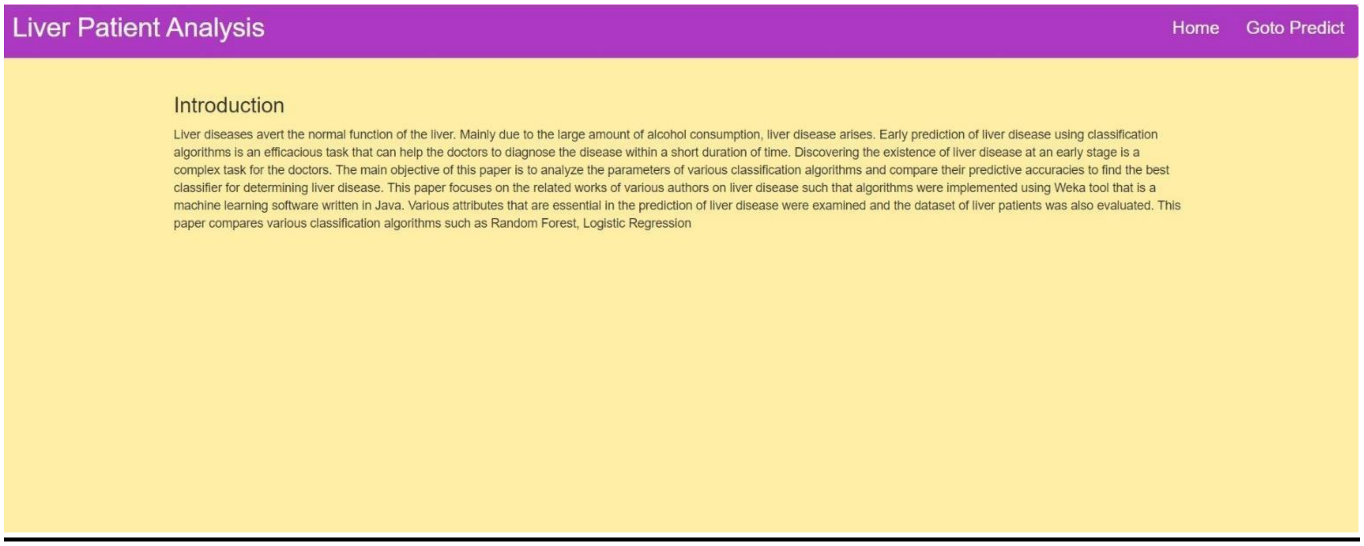
Heat map of the correlation factor b/w each column:


Age 1.00 0.01 0.01 0.08 -0.09 -0.02 -0.19 -0.27 -0.22
Total_Bilirubin 0.01 1.00 0.87 0.21 0.21 0.24 -0.01 -0.22 -0.21
Direct_Bilirubin 0.01 0.87 1.00 0.23 0.23 0.26 -0.00 -0.23 -0.20
Alkaline_Phosphotase 0.08 0.21 0.23 1.00 0.13 0.17 -0.03 -0.17 -0.23
Alamine_Aminotransferase -0.09 0.21 0.23 0.13 1.00 0.79 -0.04 -0.03 -0.00
Aspartate_Aminotransferase -0.02 0.24 0.26 0.17 0.79 1.00 -0.03 -0.09 -0.07
Total_Protiens -0.19 -0.01 -0.00 -0.03 -0.04 -0.03 1.00 0.78 0.23
Albumin -0.27 -0.22 -0.23 -0.17 -0.03 -0.09 0.78 1.00 0.69
Albumin_and_Globulin_Ratio -0.22 -0.21 -0.20 -0.23 -0.00 -0.07 0.23 0.69 1.00
Age
Total_Bilirubin
Direct_Bilirubin
Alkaline_Phosphotase
Alamine_Aminotransferase
Aspartate_Aminotransferase
Total_Protiens
Albumin
Albumin_and_Globulin_Ratio
1.0
0.8
0.6
0.4
0.2
0.0
-0.2

5.FLOWCHART



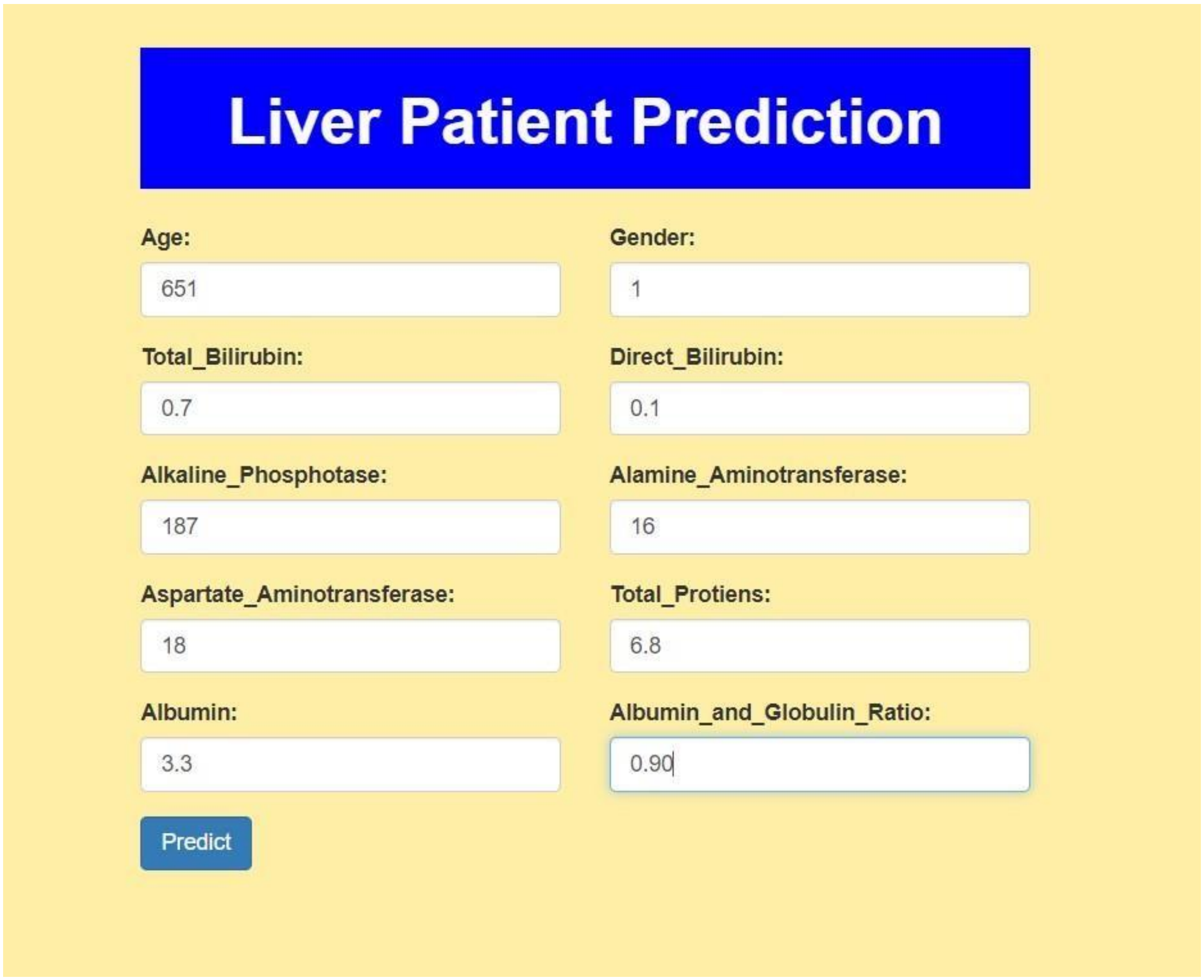
6.RESULT



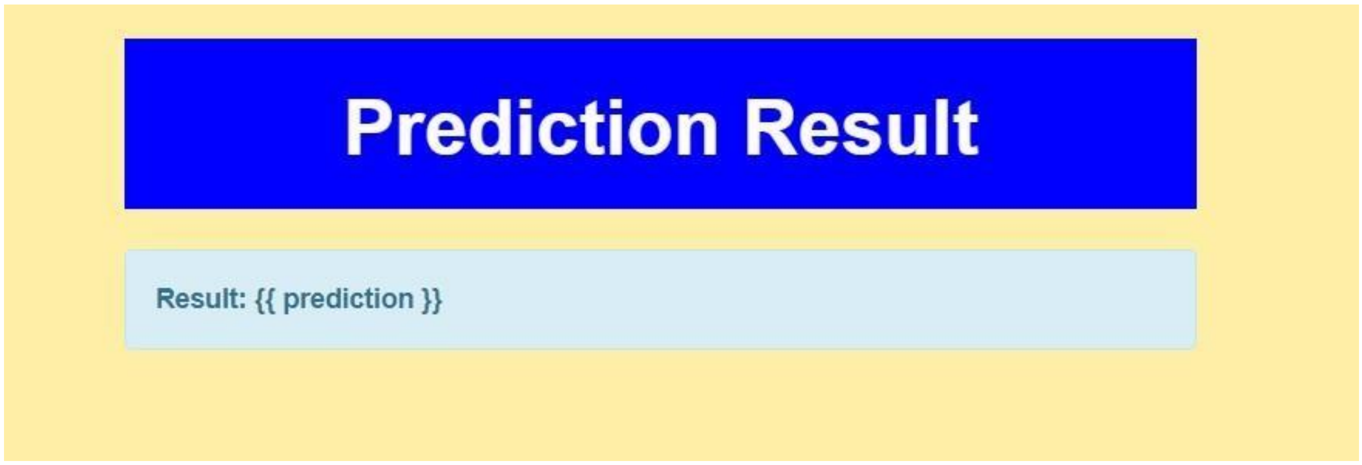
Introduction

Liver diseases avert the normal function of the liver. Mainly due to the large amount of alcohol consumption, liver disease arises. Early prediction of liver disease using classification algorithms is an efficacious task that can help the doctors to diagnose the disease within a short duration of time. Discovering the existence of liver disease at an early stage is a complex task for the doctors. The main objective of this paper is to analyze the parameters of various classification algorithms and compare their predictive accuracies to find the best classifier for determining liver disease. This paper focuses on the related works of various authors on liver disease such that algorithms were implemented using Weka tool that is a machine learning software written in Java. Various attributes that are essential in the prediction of liver disease were examined and the dataset of liver patients was also evaluated. This paper compares various classification algorithms such as Random Forest, Logistic Regression

PREDICTIONS


Liver Patient Prediction

Age:	Gender:
651	1
Total_Bilirubin:	Direct_Bilirubin:
0.7	0.1
Alkaline_Phosphotase:	Alamine_Aminotransferase:
187	16
Aspartate_Aminotransferase:	Total_Protiens:
18	6.8
Albumin:	Albumin_and_Globulin_Ratio:
3.3	0.90



Result: {{ prediction }}

7.ADVANTAGES AND DISADVANTAGES **ADVANTAGES:**

A. It can be widely used in Medical Field to predict liver disease.

B. It would reduce the burden on doctors and medical staff leading to a more accurate treatment of the patients.

DISADVANTAGES:

A. The cost of misclassification of a single patient can be very high.
B. Even the best performing Machine Learning Algorithm

may not be 100% correct hence there will always be a risk of error.

8.APPLICATIONS

1.Early Detection and Diagnosis:

- Machine learning models can analyse diverse datasets including clinical symptoms and biochemical markers to identify early signs of liver disease. Early detection enables timely intervention and improves patient outcomes.

2.Personalized Treatment Planning:

- By analysing individual patient data, machine learning can assist in tailoring treatment plans based on specific patient characteristics, optimizing therapeutic strategies for better outcomes.

3.Predictive Analytics for Disease Progression:

- Predictive models can forecast the progression of liver diseases based on historical data, assisting healthcare providers in anticipating disease trajectories and adjusting treatment accordingly.

4.Risk Stratification and Prevention:

- Machine learning algorithms can stratify patients into different risk categories for liver diseases, allowing for targeted interventions and preventive measures to reduce disease incidence and progression.

5.Optimizing Resource Allocation:

- Predictive analytics helps in allocating healthcare resources more efficiently by identifying high-risk patients who may require intensive monitoring or intervention, thus improving resource utilization.

9.CONCLUSION

The main objective of this study was to provide a summary of the classification algorithms in the field of datadriven predictive data for liver disease. In this study, various classification algorithms were analysed to help doctors predict liver disease early. The purpose of this study was achieved by conducting comparative research in various papers. Based on this study, Random Forest has a much higher accuracy than other algorithms and can be used continuously in predicting user-recommended liver disease.

10. FUTURE SCOPE

- In the upcoming future, we will convert our web application to an android application with more enhanced and innovative features.

- We will also integrate digital marketing so that different hospitals and nursing homes can use our application.

11.BIBILOGRAPHY

Certainly! Here's a bibliography for disease prediction using machine learning techniques:

These references cover a range of topics from the application of machine learning in disease prediction using electronic health records to deep learning techniques for medical imaging and clinical decision support systems

[1] Ghassemi M, Naumann T, Schulam P, Beam AL. A review of challenges and opportunities in machine learning for health. AMIA Jt Summits Transl Sci Proc. 2020 Mar 4;2020:191-200. PMID: 32477652.

[2] Rajkomar A, Dean J, Kohane I. Machine learning in medicine. N Engl J Med. 2019 Apr 4;380(14):1347-58. doi: 10.1056/NEJMra1814259. PMID: 30943335.

[3] Obermeyer Z, Emanuel EJ. Predicting the future - big data, machine learning, and clinical medicine. N Engl J Med. 2016 Sep 29;375(13):1216-9. doi: 10.1056/NEJMp1606181. PMID: 27682033.

[4] Becker AS, Mueller M, Stoffel E, Marcon M, Ghafoor S, Boss A. Classification of breast cancer in ultrasound imaging using a generic deep learning analysis software: a pilot study. Br J Radiol. 2018 May;91(1085):20170576. doi: 10.1259/bjr.20170576. Epub 2018 Mar 23. PMID: 29565617; PMCID: PMC6104757.

[5]. Miotto R, Wang F, Wang S, Jiang X, Dudley JT. Deep learning for healthcare: review, opportunities and challenges. Brief Bioinform. 2018 Nov 27;19(6):1236-46. doi: 10.1093/bib/bbx044. PMID: 28582469; PMCID: PMC6522261.

[6] Rajkomar A, Oren E, Chen K, Dai AM, Hajaj N, Hardt M, Liu PJ, Liu X, Marcus J, Sun M, Sundberg P, Yee H, Zhang K, Zhang Y, Flores G, Duggan GE, Irvine J, Le Q, Litsch K, Mossin A, Tansuwan J, Wang

D, Wexler J, Wilson J, Ludwig D, Volchenboum SL, Chou K, Pearson M, Madabushi S, Shah NH, Butte AJ, Howell MD, Cui C, Corrado GS, Dean J. Scalable and accurate deep learning with electronic health records. NPJ Digit Med. 2018 May 8;1:18. doi: 10.1038/s41746-018-0029-1. PMID: 31304301; PMCID: PMC6501990.

[7] Ravi D, Wong C, Deligianni F, Berthelot M, Andreu-Perez J, Lo B, Yang GZ. Deep learning for health informatics. IEEE J Biomed Health Inform. 2017 Jan;21(1):4-21. doi: 10.1109/JBHI.2016.2636665.
PMID: 28113205.

[8]. Shickel B, Tighe PJ, Bihorac A, Rashidi P. Deep EHR: a survey of recent advances in deep learning techniques for electronic health record (EHR) analysis. IEEE J Biomed Health Inform. 2018 Jan;22(5):1589-604. doi: 10.1109/JBHI.2017.2767063. PMID: 28920993; PMCID: PMC5819517.

[9] Lipton ZC, Kale DC, Elkan C, Wetzel R. Learning to diagnose with LSTM recurrent neural networks. arXiv preprint arXiv:1511.03677. 2015 Nov 11.

[10] Chen JH, Asch SM. Machine learning and prediction in medicine - beyond the peak of inflated expectations. N Engl J Med. 2017 Jun 29;376(26):2507-9. doi: 10.1056/NEJMp1702071. PMID: 28657873.

12.APPENDIX

Model building :

1)Dataset
2)Google colab and VS code Application Building
   1. HTML file (Index file, Predict file)
   1. CSS file
   2. Models in pickle format

SOURCE CODE:

INDEX.HTML

```
<!DOCTYPE html>
<html lang="en">
<head>
  <title>Liver Patient Analysis</title>
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap.min.css">
  <style type="text/css">
    body {
      background-color: #ffcf0059;
    }
    nav {
      background-color: #ad38c2;
      height: 60px;
    }
    .navbar-brand {
      color: white;
      font-size: 30px;
    }
    nav ul li a {
      color: white;
      font-size: 20px;
    }
  </style>
</head>
<body>

<nav class="navbar">
  <div class="container-fluid">
    <div class="navbar-header">
      <a class="navbar-brand">Liver Patient Analysis</a>
    </div>
    <ul class="nav navbar-nav navbar-right">
      <li><a href="#">Home</a></li>
      <li><a href="/predict">Goto Predict</a></li>
    </ul>
  </div>
</nav>

<div class="container">
  <h3>Introduction</h3>
  <p>Liver diseases avert the normal function of the liver. Mainly due to the large amount of alcohol consumption,
liver disease arises. Early prediction of liver disease using classification algorithms is an efficacious task that can help
the doctors to diagnose the disease within a short duration of time. Discovering the existence of liver disease at an
early stage is a complex task for the doctors. The main objective of this paper is to analyze the parameters of various
classification algorithms and compare their predictive accuracies to find the best classifier for determining liver
disease. This paper focuses on the related works of various authors on liver disease such that algorithms were
implemented using Weka tool that is a machine learning software written in Java. Various attributes that are
essential in the prediction of liver disease were examined and the dataset of liver patients was also evaluated. This
paper compares various classification algorithms such as Random Forest, Logistic Regression</p>
</div>

</body>
</html>
```

PREDICT.HTML

```
 <!DOCTYPE html>

<html>

<head>

  <title>Liver Patient Prediction</title>

  <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap.min.css">

  <style type="text/css">
```

```
    body {
      background-color: #ffcf0059;
    }
    .page-header {
      background-color: blue;
      width: 100%;
      height: auto;
      text-align: center;
      padding-top: 5px;
      color: #fff;
    }
    h1 {
      font-size: 40px;
      font-weight: bold;
    }
  </style>
</head>
<body>

  <div class="container">
    <div class="row">
      <div class="col-md-3"></div>
      <div class="col-md-6">
        <div class="page-header">
          <h1>Liver Patient Prediction</h1>
        </div>
      </div>
    </div>
  </div>
```

```
 <div class="container">
  <div class="row">
   <div class="col-md-3"></div>
   <div class="col-md-6">
    <form action="/data_predict" method="POST">
     <div class="row">
      <div class="col-md-6">
       <div class="form-group">
        <label for="age">Age:</label>
        <input type="text" class="form-control" id="age" name="age">
       </div>
      </div>

      <div class="col-md-6">
       <div class="form-group">
        <label for="gender">Gender:</label>
        <input type="text" class="form-control" id="gender" name="gender" placeholder="Enter 0 as
male, 1 as female">
       </div>
      </div>
     </div>

     <div class="row">
      <div class="col-md-6">
       <div class="form-group">
        <label for="tb">Total_Bilirubin:</label>
        <input type="text" class="form-control" id="tb" name="tb">
       </div>
  </div>
  <div class="col-md-6">
    <div class="form-group">
      <label for="db">Direct_Bilirubin:</label>
      <input type="text" class="form-control" id="db" name="db">
    </div>
  </div>
</div>

<div class="row">
  <div class="col-md-6">
    <div class="form-group">
      <label for="ap">Alkaline_Phosphotase:</label>
      <input type="text" class="form-control" id="ap" name="ap">
    </div>
  </div>

  <div class="col-md-6">
    <div class="form-group">
      <label for="aa1">Alamine_Aminotransferase:</label>
      <input type="text" class="form-control" id="aa1" name="aa1">
    </div>
  </div>
</div>

<div class="row">
  <div class="col-md-6">
    <div class="form-group">
      <label for="aa2">Aspartate_Aminotransferase:</label>
```

```
        <input type="text" class="form-control" id="aa2" name="aa2">
      </div>
    </div>
    <div class="col-md-6">
      <div class="form-group">
        <label for="tp">Total_Protiens:</label>
        <input type="text" class="form-control" id="tp" name="tp">
      </div>
    </div>
  </div>

  <div class="row">
    <div class="col-md-6">
      <div class="form-group">
        <label for="a">Albumin:</label>
        <input type="text" class="form-control" id="a" name="a">
      </div>
    </div>

    <div class="col-md-6">
      <div class="form-group">
        <label for="agr">Albumin_and_Globulin_Ratio:</label>
        <input type="text" class="form-control" id="agr" name="agr">
      </div>
    </div>
  </div>

  <button type="submit" class="btn btn-primary">Predict</button>
</form>
```

```
      </div>
    </div>
  </div>

</body>
</html>
```

APP.PY

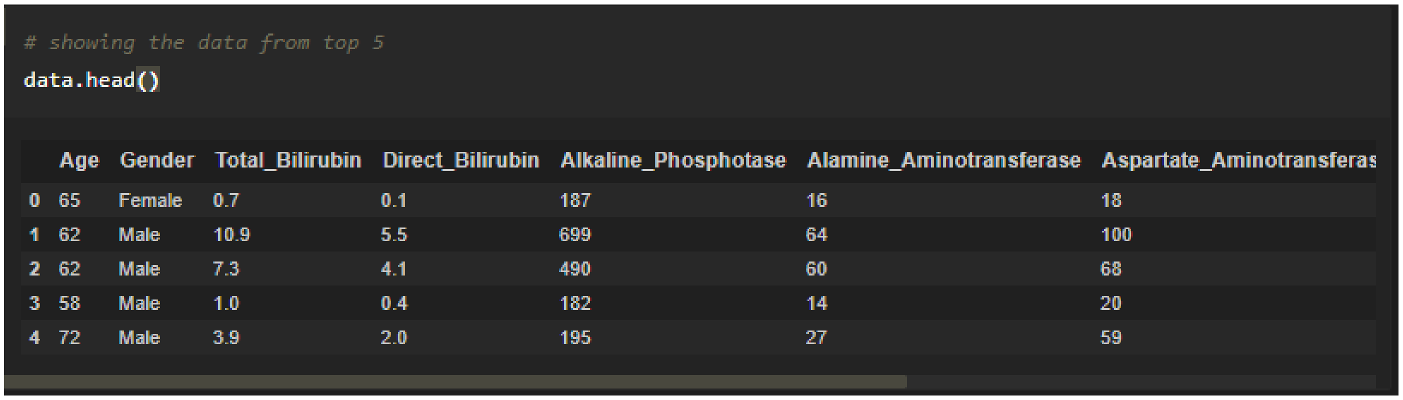
```
# import numpy as np
# from flask import Flask, request, jsonify, render_template
# import pickle

# app = Flask(__name__)
# model = pickle.load(open('indian_liver_patient.pkl', 'rb'))

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

# @app.route('/predict', methods=['POST'])
# def predict():
#     '''
#     For rendering results on HTML GUI
#     '''
#     try:
#         int_features = [int(x) for x in request.form.values()]
#         final_features = [np.array(int_features)]
#         prediction = model.predict(final_features)
#         output = round(prediction[0], 2)
#         return render_template('index.html',
prediction_text='Predicted Value: $ {}'.format(output))
#     except Exception as e:
```

```
#        return str(e)

# @app.route('/predict_api', methods=['POST'])
# def predict_api():
#    '''
#    For direct API calls through request
#    '''
#    try:
#        data = request.get_json(force=True)
#        prediction = model.predict([np.array(list(data.values()))])
#        output = prediction[0]
#        return jsonify(output)
#    except Exception as e:
#        return jsonify({'error': str(e)})

# if __name__ == "__main__":
#    app.run(debug=True)



"""from flask import Flask, request, render_template
import pandas as pd
import pickle

app = Flask(__name__)

# Load the trained model
try:
    with open('indian_liver_patient.pkl', 'rb') as model_file:
        model = pickle.load(model_file)
except (EOFError, FileNotFoundError) as e:
    model = None
    print(f"Error loading model: {e}")
```

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


@app.route('/predict', methods=['GET', 'POST'])
def predict():
    if request.method == 'POST':
        try:
            if model is None:
                raise ValueError("Model not loaded properly")

            # Get input data from form
            data = request.form.to_dict()
            data = {k: float(v) for k, v in data.items()}

            # Convert to DataFrame
            df = pd.DataFrame([data])

            # Make prediction
            prediction = model.predict(df)

            # Return result
            result = 'Liver disease' if prediction[0] == 1 else 'No liver
disease'
            return render_template('result.html', result=result)

        except Exception as e:
            return str(e)
    return render_template('predict.html')

if __name__ == "__main__":
    app.run(debug=True)"""



from flask import Flask, request, render_template
import pandas as pd
import pickle

app = Flask(__name__)

# Load the model once during the app initialization
model = pickle.load(open('liver_analysis.pkl', 'rb'))

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

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

@app.route('/data_predict', methods=['POST'])
def predict():
    if request.method == 'POST':
        age = request.form['age']
        gender = request.form['gender']
        tb = request.form['tb']
        db = request.form['db']
        ap = request.form['ap']
        aa1 = request.form['aa1']
        aa2 = request.form['aa2']
        tp = request.form['tp']
        a = request.form['a']
        agr = request.form['agr']
        data = [[float(age), float(gender), float(tb), float(db),
float(ap), float(aa1), float(aa2), float(tp), float(a), float(agr)]]

        prediction = model.predict(data)[0]
        if prediction == 1:
          result = "You have a liver disease"
        else:
          result = "You don't have a liver disease"

        return render_template('result.html', prediction=result)

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

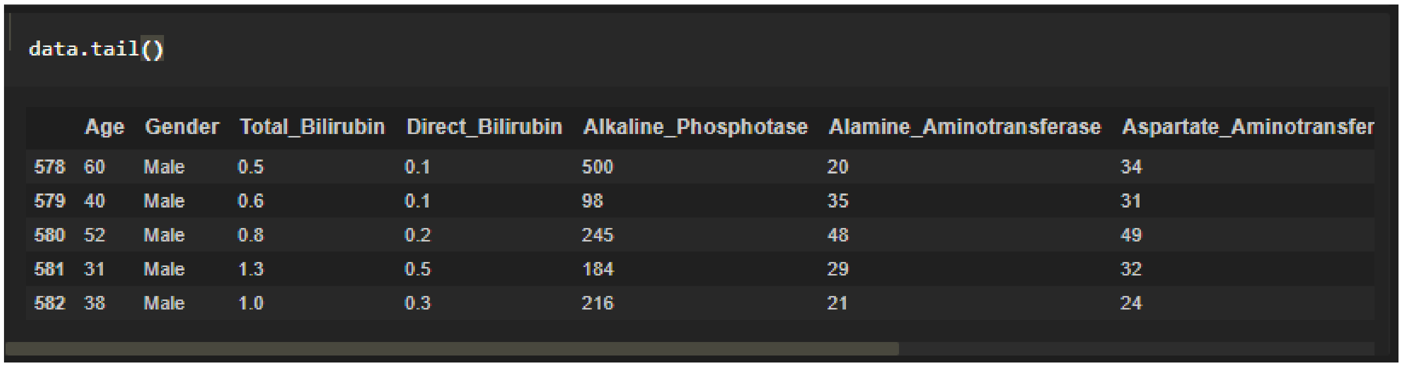
CODE SNIPPETS

MODEL BUILDING



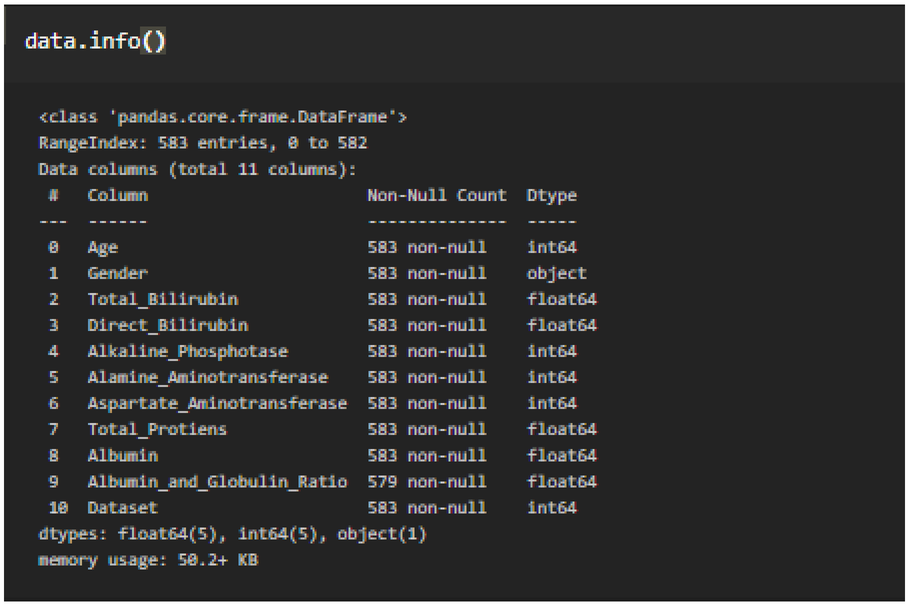
```
# showing the data from top 5
data.head()
```

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



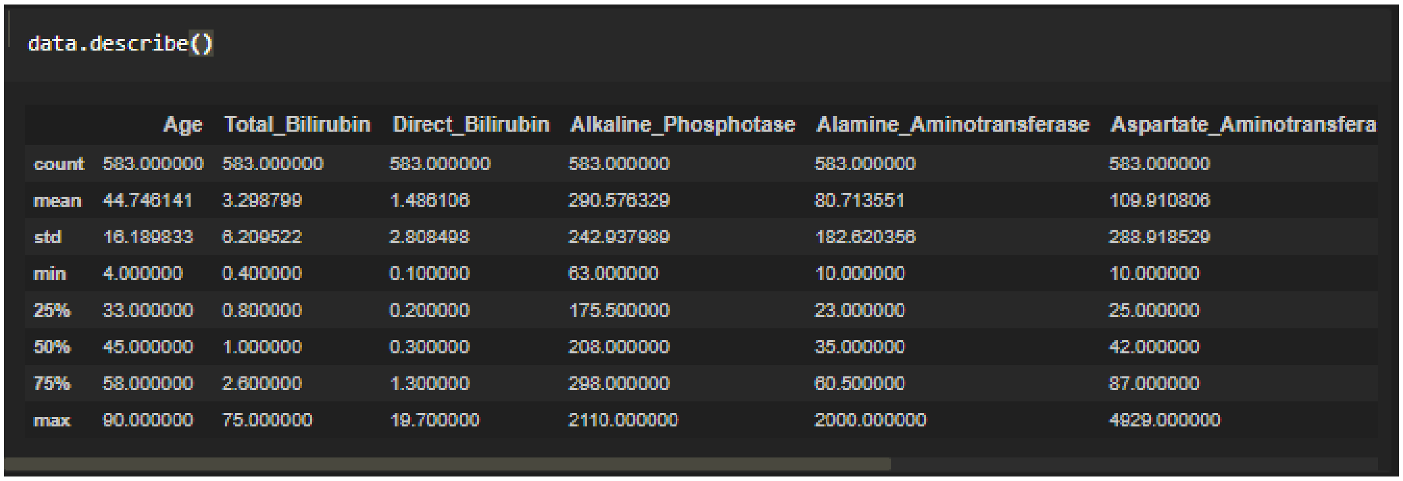
```
data.tail()
```

	Age	Gender	Total_Bilirubin	Direct_Bilirubin	Alkaline_Phosphotase	Alamine_Aminotransferase	Aspartate_Aminotransfer
578	60	Male	0.5	0.1	500	20	34
579	40	Male	0.6	0.1	98	35	31
580	52	Male	0.8	0.2	245	48	49
581	31	Male	1.3	0.5	184	29	32
582	38	Male	1.0	0.3	216	21	24



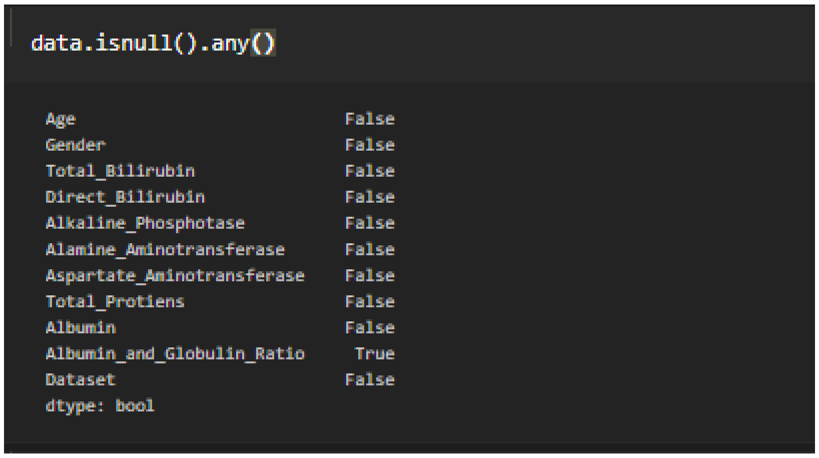
```
data.info()
```

```
<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
```



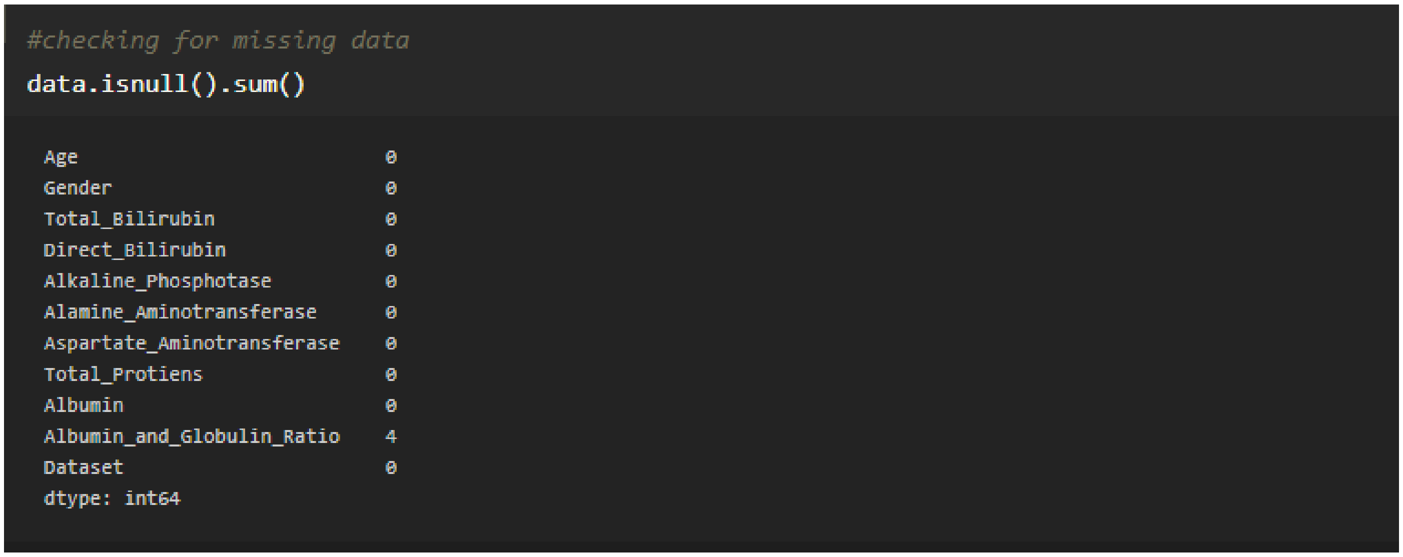
```
data.describe()
```

```
            Age  Total_Bilirubin  Direct_Bilirubin  Alkaline_Phosphotase  Alamine_Aminotransferase  Aspartate_Aminotransfera
count  583.000000  583.000000      583.000000        583.000000            583.000000                583.000000
mean   44.746141   3.298799        1.486106          290.576329            80.713551                 109.910806
std    16.189833   6.209522        2.808498          242.937989            182.620356                288.918529
min    4.000000    0.400000        0.100000          63.000000             10.000000                 10.000000
25%    33.000000   0.800000        0.200000          175.500000            23.000000                 25.000000
50%    45.000000   1.000000        0.300000          208.000000            35.000000                 42.000000
75%    58.000000   2.600000        1.300000          298.000000            60.500000                 87.000000
max    90.000000   75.000000       19.700000         2110.000000           2000.000000               4929.000000
```



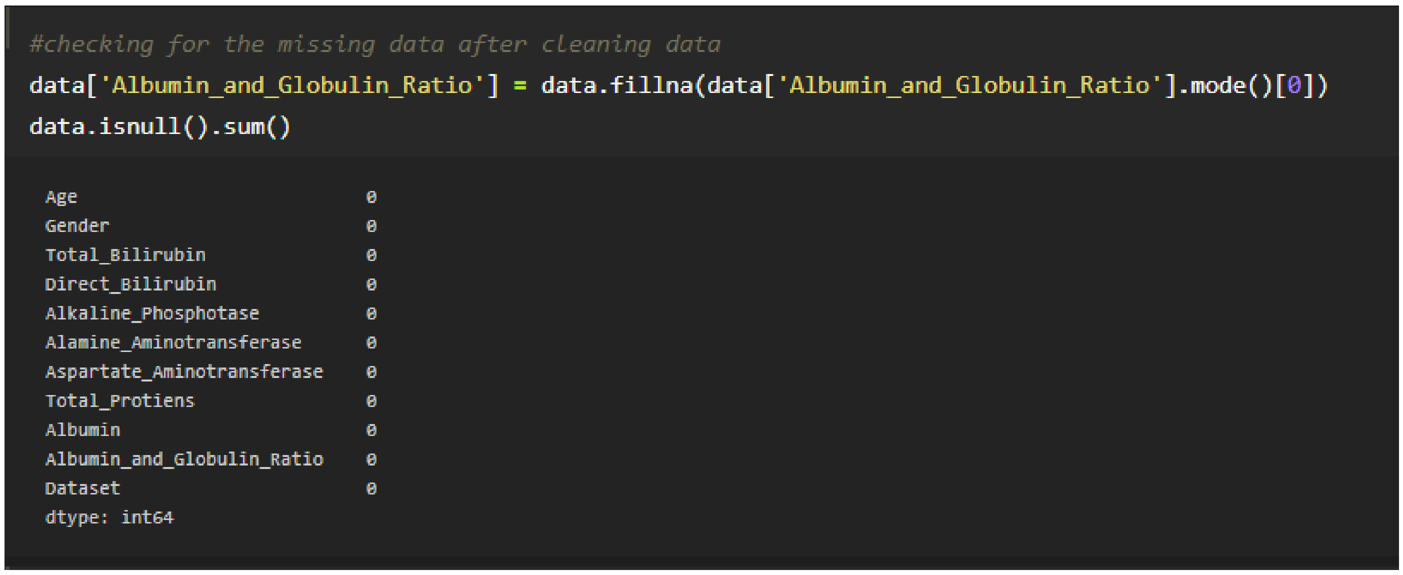
```
data.isnull().any()
```

```
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
```



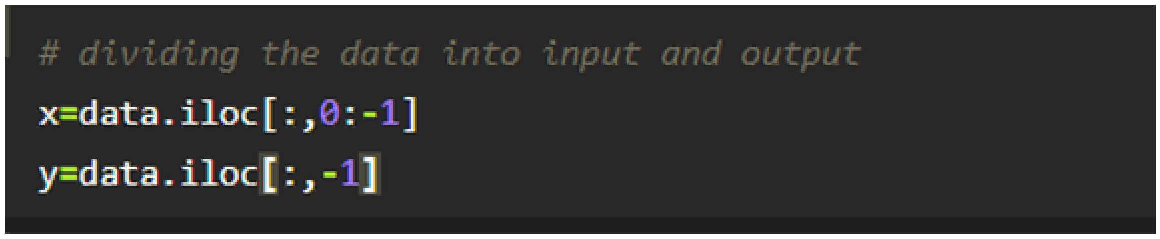
```
#checking for missing data
data.isnull().sum()
```

```
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
```

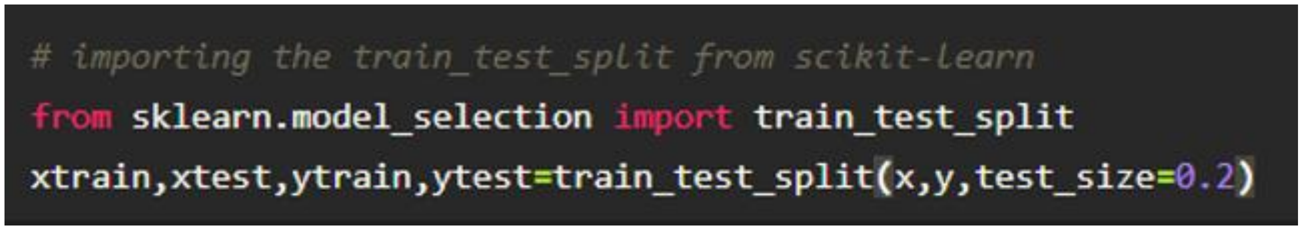


```
#checking for the missing data after cleaning data
data['Albumin_and_Globulin_Ratio'] = data.fillna(data['Albumin_and_Globulin_Ratio'].mode()[0])
data.isnull().sum()
```

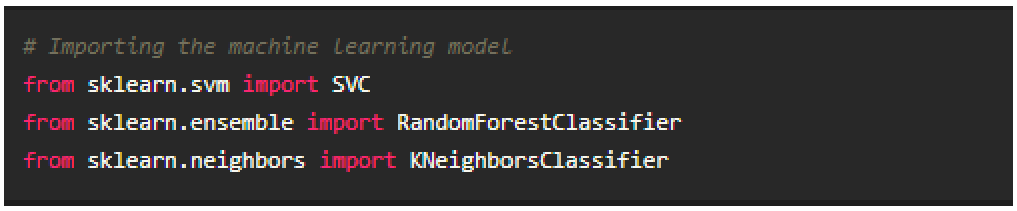
```
Age                           0
Gender                        0
Total_Bilirubin               0
Direct_Bilirubin              0
Alkaline_Phosphotase          0
Alamine_Aminotransferase      0
Aspartate_Aminotransferase    0
Total_Protiens                0
Albumin                       0
Albumin_and_Globulin_Ratio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



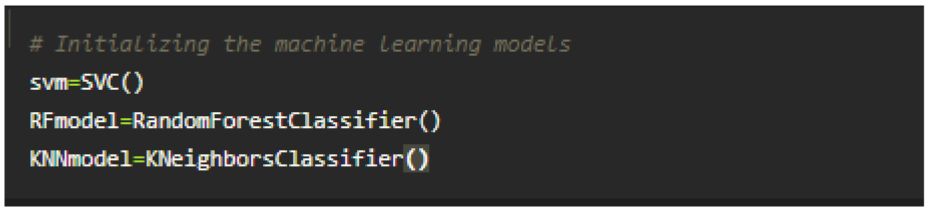
```
# dividing the data into input and output
x=data.iloc[:,0:-1]
y=data.iloc[:,-1]
```



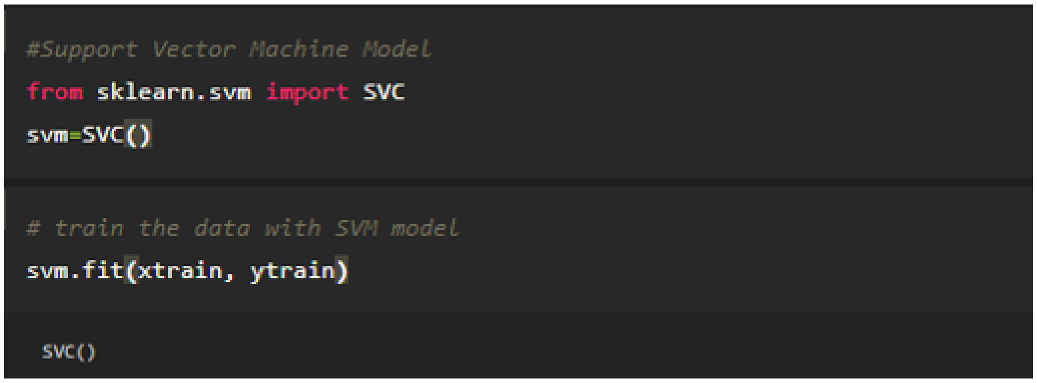
```
# importing the train_test_split from scikit-learn
from sklearn.model_selection import train_test_split
xtrain,xtest,ytrain,ytest=train_test_split(x,y,test_size=0.2)
```



```
# Importing the machine learning model
from sklearn.svm import SVC
from sklearn.ensemble import RandomForestClassifier
from sklearn.neighbors import KNeighborsClassifier
```



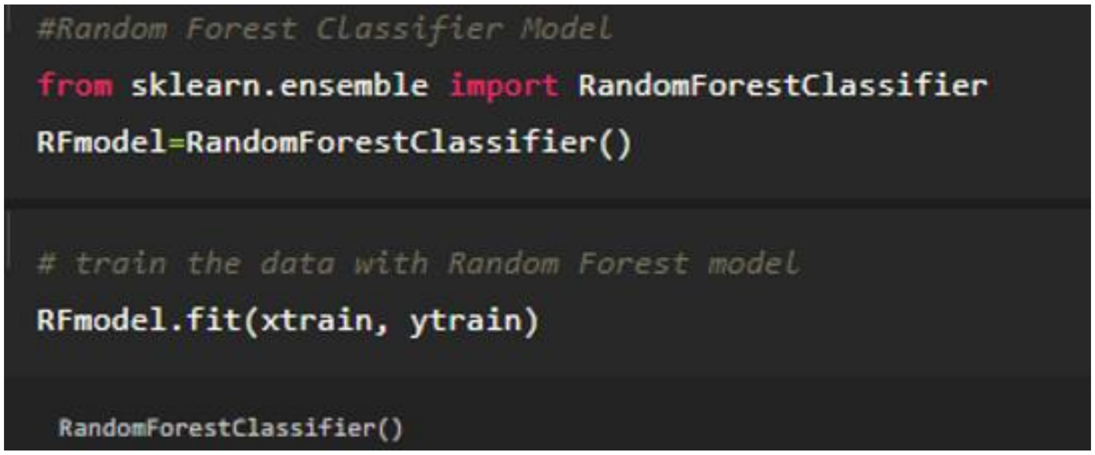
```
# Initializing the machine learning models
svm=SVC()
RFmodel=RandomForestClassifier()
KNNmodel=KNeighborsClassifier()
```



```
#Support Vector Machine Model
from sklearn.svm import SVC
svm=SVC()
```

```
# train the data with SVM model
svm.fit(xtrain, ytrain)
```

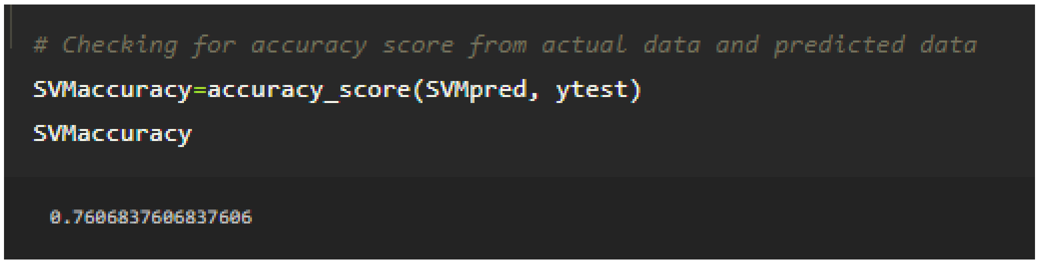
```
SVC()
```



```
#Random Forest Classifier Model
from sklearn.ensemble import RandomForestClassifier
RFmodel=RandomForestClassifier()
```

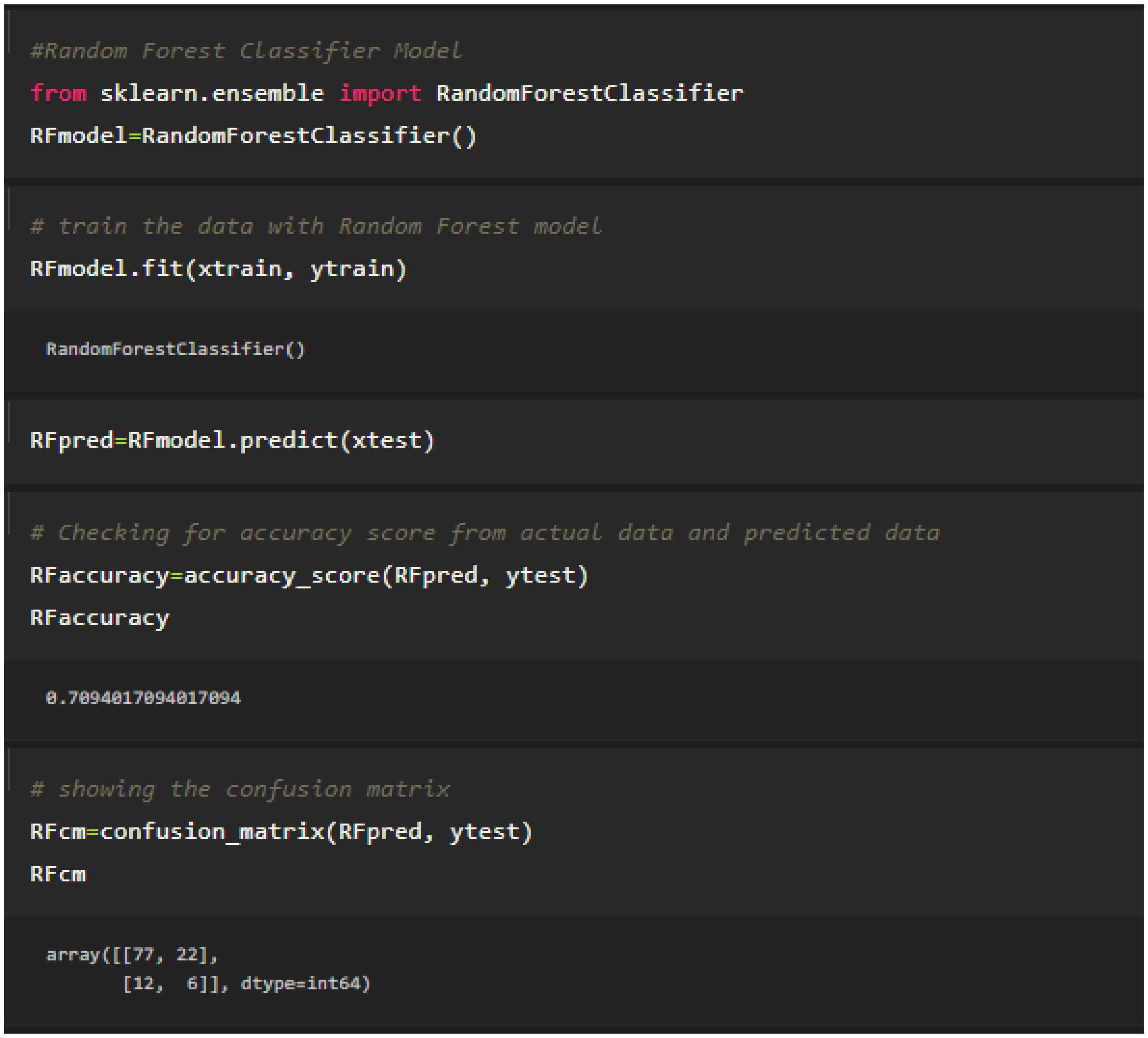
```
# train the data with Random Forest model
RFmodel.fit(xtrain, ytrain)
```

```
RandomForestClassifier()
```



```
# Checking for accuracy score from actual data and predicted data
SVMaccuracy=accuracy_score(SVMpred, ytest)
SVMaccuracy
```

```
0.7606837606837606
```



```
#Random Forest Classifier Model
from sklearn.ensemble import RandomForestClassifier
RFmodel=RandomForestClassifier()
```

```
# train the data with Random Forest model
RFmodel.fit(xtrain, ytrain)
```

```
RandomForestClassifier()
```

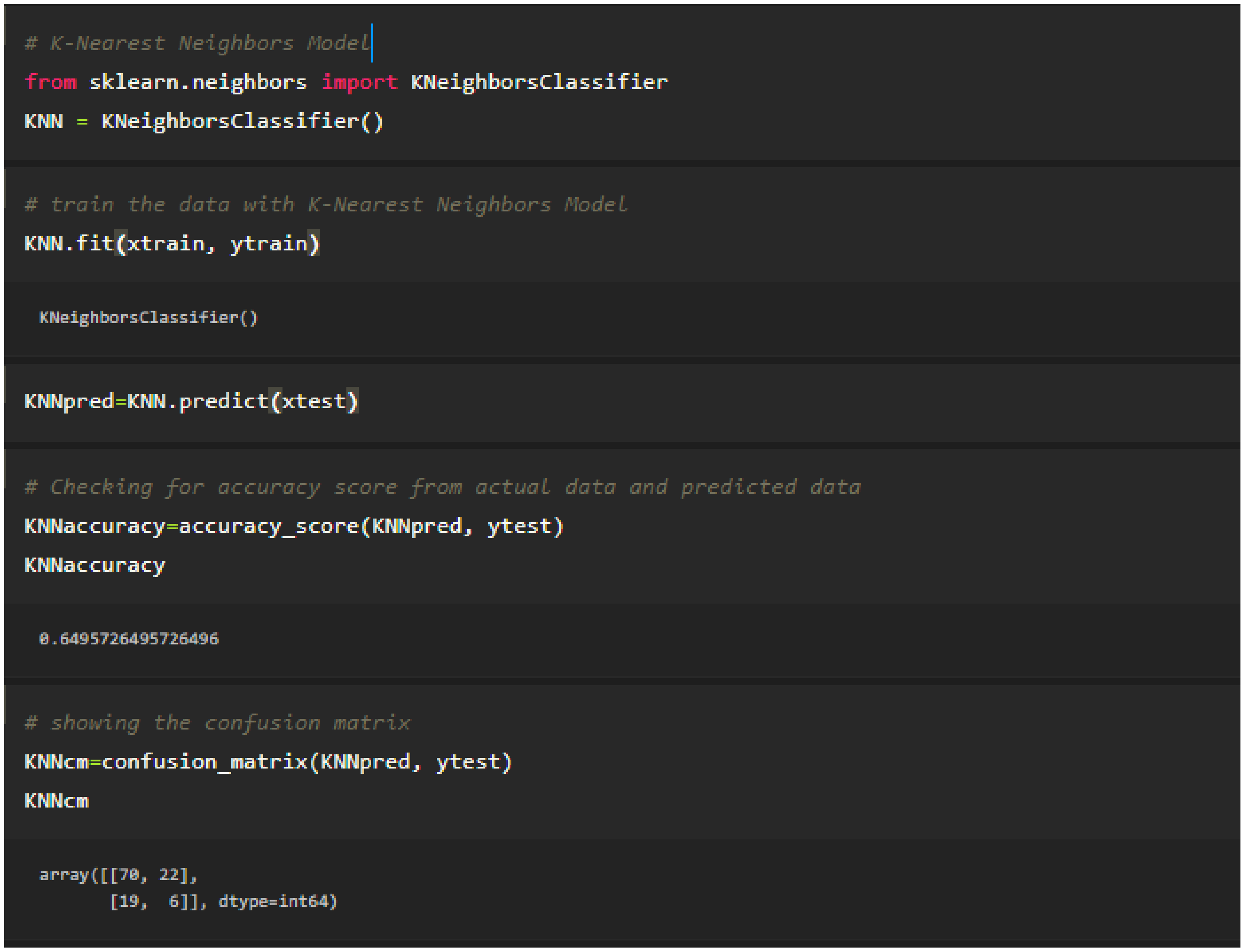
```
RFpred=RFmodel.predict(xtest)
```

```
# Checking for accuracy score from actual data and predicted data
RFaccuracy=accuracy_score(RFpred, ytest)
RFaccuracy
```

```
0.7094017094017094
```

```
# showing the confusion matrix
RFcm=confusion_matrix(RFpred, ytest)
RFcm
```

```
array([[77, 22],
       [12,  6]], dtype=int64)
```



```
# K-Nearest Neighbors Model
from sklearn.neighbors import KNeighborsClassifier
KNN = KNeighborsClassifier()
```

```
# train the data with K-Nearest Neighbors Model
KNN.fit(xtrain, ytrain)
```

```
KNeighborsClassifier()
```

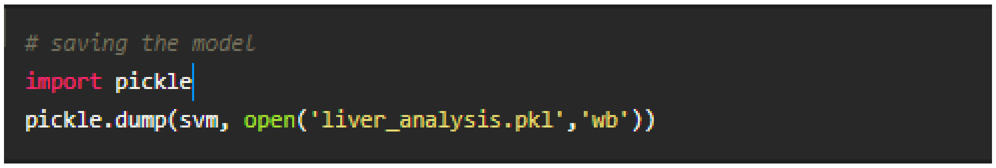
```
KNNpred=KNN.predict(xtest)
```

```
# Checking for accuracy score from actual data and predicted data
KNNaccuracy=accuracy_score(KNNpred, ytest)
KNNaccuracy
```

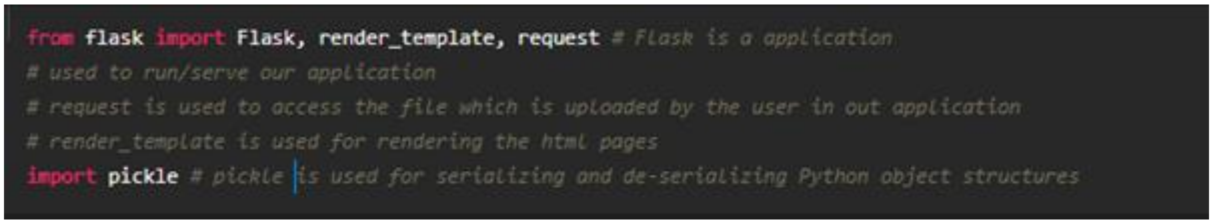
```
0.6495726495726496
```

```
# showing the confusion matrix
KNNcm=confusion_matrix(KNNpred, ytest)
KNNcm
```

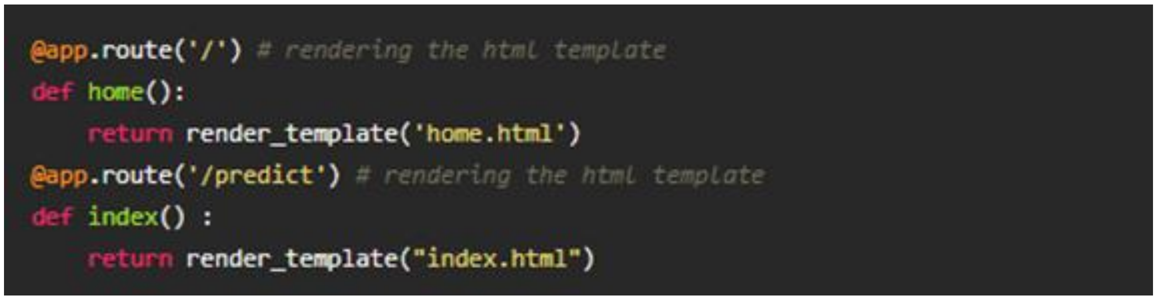
```
array([[70, 22],
       [19,  6]], dtype=int64)
```



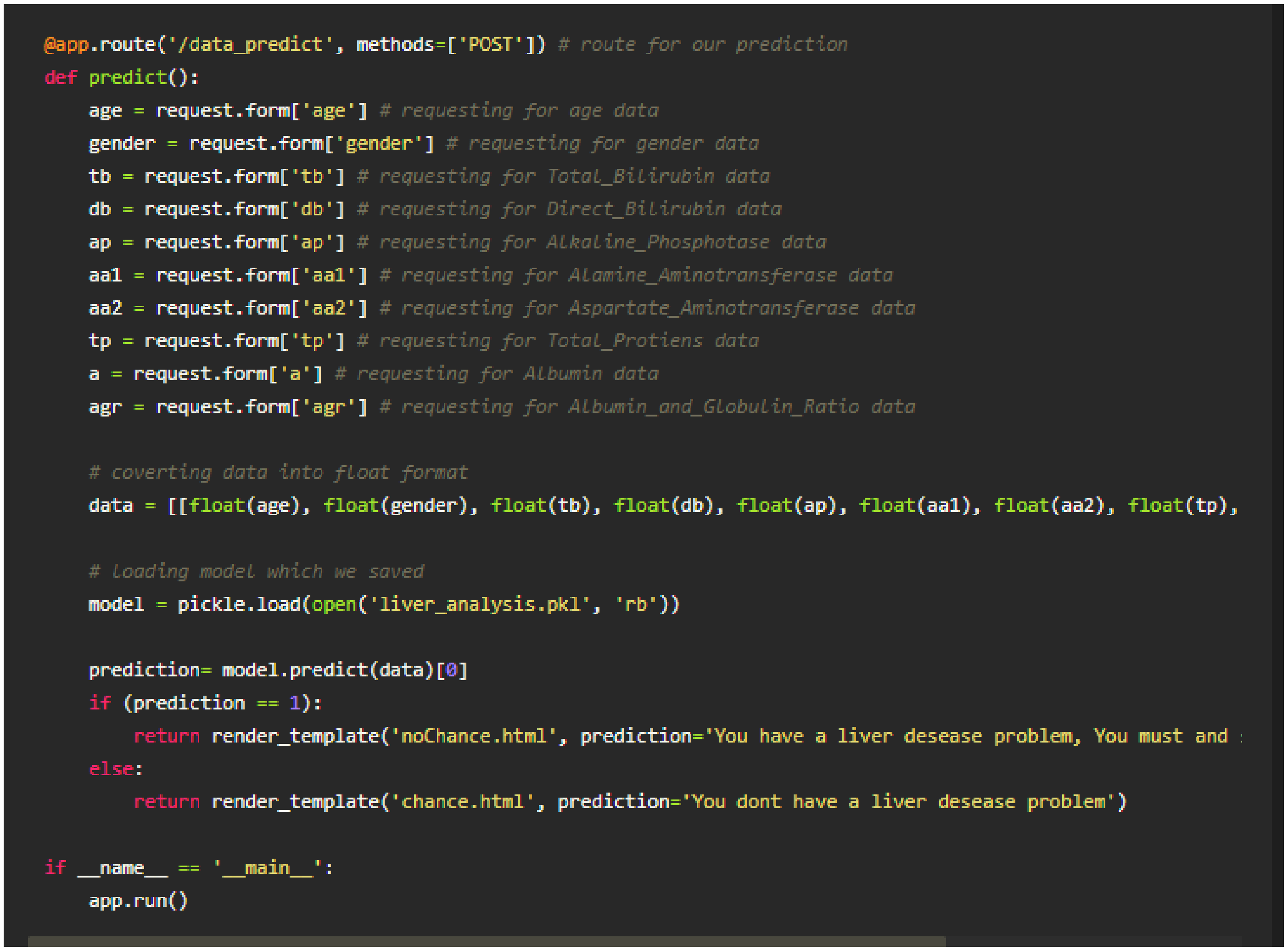
```
# saving the model
import pickle
pickle.dump(svm, open('liver_analysis.pkl','wb'))
```



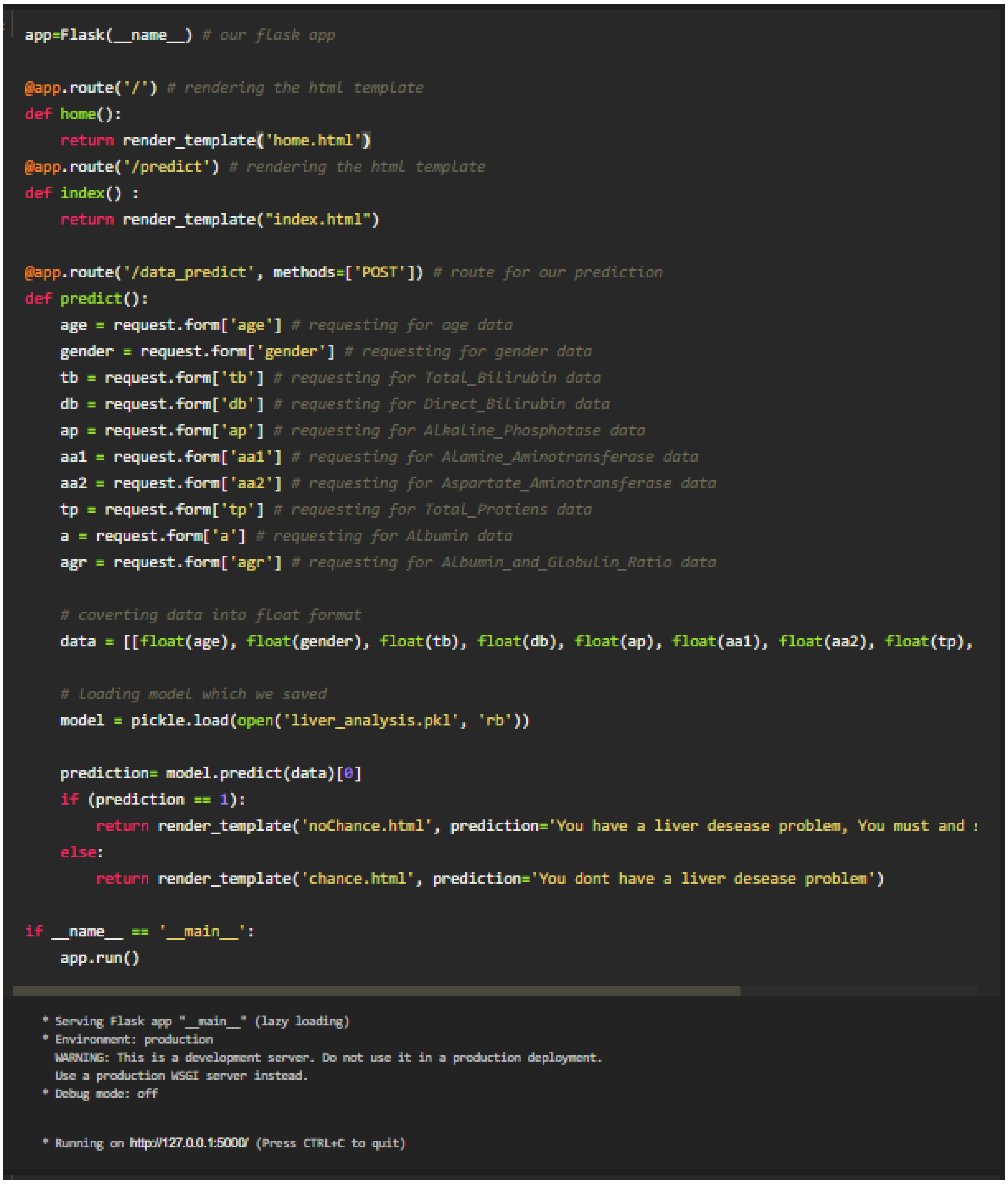
```
from flask import Flask, render_template, request # Flask is a application
# used to run/serve our application
# request is used to access the file which is uploaded by the user in out application
# render_template is used for rendering the html pages
import pickle # pickle is used for serializing and de-serializing Python object structures
```



```
@app.route('/') # rendering the html template
def home():
    return render_template('home.html')
@app.route('/predict') # rendering the html template
def index() :
    return render_template("index.html")
```



```
@app.route('/data_predict', methods=['POST']) # route for our prediction
def predict():
    age = request.form['age'] # requesting for age data
    gender = request.form['gender'] # requesting for gender data
    tb = request.form['tb'] # requesting for Total_Bilirubin data
    db = request.form['db'] # requesting for Direct_Bilirubin data
    ap = request.form['ap'] # requesting for Alkaline_Phosphotase data
    aa1 = request.form['aa1'] # requesting for Alamine_Aminotransferase data
    aa2 = request.form['aa2'] # requesting for Aspartate_Aminotransferase data
    tp = request.form['tp'] # requesting for Total_Protiens data
    a = request.form['a'] # requesting for Albumin data
    agr = request.form['agr'] # requesting for Albumin_and_Globulin_Ratio data

    # coverting data into float format
    data = [[float(age), float(gender), float(tb), float(db), float(ap), float(aa1), float(aa2), float(tp),

    # loading model which we saved
    model = pickle.load(open('liver_analysis.pkl', 'rb'))

    prediction= model.predict(data)[0]
    if (prediction == 1):
        return render_template('noChance.html', prediction='You have a liver desease problem, You must and
    else:
        return render_template('chance.html', prediction='You dont have a liver desease problem')

if __name__ == '__main__':
    app.run()
```



```
app=Flask(__name__) # our flask app

@app.route('/') # rendering the html template
def home():
    return render_template('home.html')
@app.route('/predict') # rendering the html template
def index() :
    return render_template("index.html")

@app.route('/data_predict', methods=['POST']) # route for our prediction
def predict():
    age = request.form['age'] # requesting for age data
    gender = request.form['gender'] # requesting for gender data
    tb = request.form['tb'] # requesting for Total_Bilirubin data
    db = request.form['db'] # requesting for Direct_Bilirubin data
    ap = request.form['ap'] # requesting for Alkaline_Phosphotase data
    aa1 = request.form['aa1'] # requesting for Alamine_Aminotransferase data
    aa2 = request.form['aa2'] # requesting for Aspartate_Aminotransferase data
    tp = request.form['tp'] # requesting for Total_Protiens data
    a = request.form['a'] # requesting for Albumin data
    agr = request.form['agr'] # requesting for Albumin_and_Globulin_Ratio data

    # coverting data into float format
    data = [[float(age), float(gender), float(tb), float(db), float(ap), float(aa1), float(aa2), float(tp),

    # loading model which we saved
    model = pickle.load(open('liver_analysis.pkl', 'rb'))

    prediction= model.predict(data)[0]
    if (prediction == 1):
        return render_template('noChance.html', prediction='You have a liver desease problem, You must and :
    else:
        return render_template('chance.html', prediction='You dont have a liver desease problem')

if __name__ == '__main__':
    app.run()
```

```
 * Serving Flask app "__main__" (lazy loading)
 * Environment: production
   WARNING: This is a development server. Do not use it in a production deployment.
   Use a production WSGI server instead.
 * Debug mode: off

 * Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
```