**Revolutionizing Liver Care: Predicting Liver Cirrhosis Using Machine Learning**

# ABSTRACT :

This project aims to detect liver cirrhosis using a machine learning model trained on clinical data. The model is deployed via a web-based Flask interface, allowing users to input medical parameters and receive a prediction. The approach improves accessibility and efficiency in liver disease diagnosis.

introduction :

**Liver Cirrhosis Background**

Liver cirrhosis is a chronic and progressive liver disease characterized by the irreversible scarring (fibrosis) of liver tissue. Over time, healthy liver cells are replaced by scar tissue, reducing the liver's ability to function properly. This damage is often the result of prolonged liver inflammation due to conditions such as chronic hepatitis B or C infection, long-term alcohol abuse, fatty liver disease, or other metabolic disorders.

Globally, liver cirrhosis is a major health concern. According to the World Health Organization (WHO), it ranks among the leading causes of death worldwide. Left undiagnosed or untreated, cirrhosis can lead to complications such as liver failure, portal hypertension, internal bleeding, and even liver cancer. The irreversible nature of cirrhosis makes early detection and preventive care crucial.

**Importance of Early Detection**

In its early stages, liver cirrhosis is often asymptomatic, meaning patients may not exhibit clear signs or symptoms until significant damage has occurred. By the time symptoms such as jaundice, abdominal swelling, or confusion appear, treatment becomes more complex and less effective.

Early detection allows healthcare providers to intervene promptly, potentially halting the progression of the disease through lifestyle changes, medication, or medical procedures. It also gives patients the opportunity to monitor liver function more closely and avoid further complications.

Traditional methods of diagnosing liver cirrhosis involve invasive procedures like liver biopsies, imaging scans, or extensive laboratory testing. These procedures can be costly, time-consuming, and inaccessible to populations in remote or under-resourced areas.

**Why Machine Learning is Suitable**

Machine learning (ML) offers a powerful alternative to traditional diagnostic methods by enabling the analysis of large medical datasets to uncover patterns and predict disease states. It can process numerous variables at once, identify complex relationships between features, and provide real-time predictions based on input data.

For liver cirrhosis detection, ML models can learn from patient data—such as enzyme levels, bilirubin concentrations, protein values, and more—to identify whether a patient is likely to have the disease. Unlike rule-based systems, ML models improve with more data and can adapt to a wide variety of clinical situations.

Using machine learning in liver disease detection provides several advantages:

* Faster and more accurate predictions
* Reduction in the need for invasive tests
* Scalability and adaptability to large populations
* Integration with digital health systems and telemedicine platforms

**Project Overview**

This project aims to build an intelligent system that predicts the likelihood of liver cirrhosis using a machine learning algorithm, specifically a Random Forest classifier. The system is trained on a publicly available dataset containing clinical features such as age, gender, enzyme levels, and protein ratios. After preprocessing and normalizing the data, the model is trained and validated for accuracy.

The project also includes a web-based application developed using Flask, allowing users to enter medical data through a simple HTML form. Once the form is submitted, the server processes the input, applies normalization, feeds it to the trained model, and returns a prediction—either indicating the presence of liver cirrhosis or not.

This approach demonstrates how machine learning can complement medical expertise, particularly in early-stage detection and preventive care, making healthcare more efficient and accessible. It combines data science, web development, and healthcare domain knowledge to create a practical and impactful solution.

# Literature Review :

**Past Studies on Liver Disease Detection**

Liver diseases, particularly liver cirrhosis and hepatocellular carcinoma, have been the subject of extensive medical research. In the past, diagnosis heavily relied on clinical observation, biochemical testing, ultrasound, and liver biopsy. However, many studies have explored how data mining and pattern recognition techniques can aid in early diagnosis.

Several notable studies include:

* **A. S. Subbalakshmi et al. (2011)** applied decision tree algorithms to liver patient datasets, achieving encouraging accuracy levels using J48 and Naive Bayes classifiers.
* **Patil & Kumaraswamy (2010)** utilized support vector machines and reported over 70% accuracy in classifying liver disorder cases.
* **Karthik et al. (2017)** implemented k-means clustering and Random Forest classifiers on clinical datasets to assist in diagnosing chronic liver conditions.
* In a study by **Tushar and Sangeeta (2019)**, the UCI Liver Disorder dataset was tested against SVM, logistic regression, and Random Forest models, where Random Forest performed best with approximately 72% accuracy.

These works confirm the potential of machine learning models in supporting clinical decisions and reducing diagnostic delays.

**Machine Learning in Healthcare**

The integration of machine learning in healthcare has revolutionized diagnostics, treatment recommendations, and patient risk analysis. ML algorithms excel at analyzing large-scale clinical data, recognizing complex patterns, and providing predictive analytics for various diseases.

Key applications of ML in healthcare include:

* **Disease diagnosis:** Cancer, diabetes, liver diseases, heart disease prediction
* **Medical imaging:** Object detection in X-rays, MRIs, CT scans
* **Drug discovery:** Pattern recognition in protein sequencing and compound testing
* **Patient risk profiling:** Early detection of at-risk individuals

ML models such as decision trees, Random Forest, SVM, k-NN, and neural networks are widely used. Random Forest in particular is robust against noise and overfitting and is capable of handling both categorical and continuous data—making it well-suited for medical datasets.

In liver cirrhosis prediction, machine learning automates the analysis of enzyme levels, bilirubin values, and protein ratios to make a clinical judgment, minimizing human error and diagnostic lag.

**Comparison of Models in Existing Research**

| **Study/Author** | **Dataset Used** | **Model** | **Accuracy** |
| --- | --- | --- | --- |
| Patil & Kumaraswamy | UCI Liver Disorders | SVM | 70% |
| Tushar & Sangeeta (2019) | UCI Liver Disorders | Random Forest | 72% |
| A. Subbalakshmi (2011) | Indian Liver Patient | Naive Bayes, J48 | 68–73% |
| Present Study | Custom Liver Dataset | Random Forest | 68% |

* **Naive Bayes:** Performs fast but assumes feature independence, which is often not true in medical data.
* **SVM:** Effective in high-dimensional spaces but can be computationally expensive.
* **Random Forest:** Handles multicollinearity, outliers, and non-linear interactions well.
* **Neural Networks:** Provide higher accuracy but require larger datasets and more computation.

**Random Forest** strikes a balance between performance and interpretability, making it suitable for a lightweight, web-deployed liver disease prediction tool.

**Limitations of Manual Diagnosis**

Manual or traditional diagnostic processes have several drawbacks when it comes to liver cirrhosis:

1. **Time-Consuming:** Biopsies and imaging tests require hospital visits, scheduling, and lab processing.
2. **Subjective Errors:** Interpretation of enzyme levels or symptoms can vary from one doctor to another.
3. **Invasiveness:** Liver biopsy, though accurate, is invasive and carries medical risk.
4. **Cost:** Diagnostic tests can be expensive, particularly in resource-limited settings.
5. **Delayed Treatment:** Delays in detection due to reliance on advanced-stage symptoms result in poor prognosis.

By contrast, machine learning-based diagnostic tools offer:

* Quicker results
* Reduced need for invasive procedures
* Consistent predictions
* Easy integration into digital systems (EHRs, web apps)

However, they are meant to **support**, not replace, professional medical advice.

# Problem Statement :

Liver cirrhosis is a progressive and irreversible condition where healthy liver tissue is replaced by scar tissue, impairing liver function and leading to severe health complications. Early detection is critical, yet the disease is often diagnosed at advanced stages due to the absence of clear symptoms in the initial phases. Traditional diagnostic methods, such as liver biopsy, imaging, and blood tests, are either invasive, time-consuming, or limited in accessibility, especially in rural and underdeveloped regions.

Medical professionals often face challenges in timely identifying cirrhosis due to the complex interplay of various biochemical parameters and clinical indicators such as enzyme levels, bilirubin counts, protein concentrations, and patient demographics. Manual analysis of such data can be prone to human error, delays, and inconsistencies.

With advancements in artificial intelligence, particularly machine learning (ML), there is an opportunity to revolutionize the early diagnosis of liver diseases by leveraging data-driven prediction models. Machine learning models, when trained on clinical datasets, can detect patterns and correlations that may not be immediately obvious to clinicians, offering a high degree of accuracy and speed in predictions.

**Statement of the Problem:**

“To develop an intelligent system using machine learning that accurately predicts liver cirrhosis based on patient clinical attributes using a web-based interface.”

This project seeks to design and implement a predictive model using the Random Forest algorithm, trained on structured liver patient data. It aims to integrate this model into a web application powered by the Flask framework, where users can input clinical parameters and receive an immediate prediction indicating whether cirrhosis is likely present. The system is designed to be lightweight, user-friendly, and scalable, with potential real-world applications in diagnostics, remote care, and early medical screening.

# Objectives :

The primary aim of this project is to build a reliable, intelligent system for predicting liver cirrhosis using machine learning techniques and provide a user-friendly platform for real-time diagnostic support. The project addresses both technical and usability aspects to ensure its applicability in real-world medical scenarios. The key objectives are outlined below:

**1. Predict Liver Cirrhosis with High Accuracy**

To develop and train a machine learning model—specifically a Random Forest Classifier—that can accurately predict the presence of liver cirrhosis based on clinical input parameters such as age, gender, bilirubin levels, liver enzymes, protein content, and protein ratios. The goal is to achieve high precision and recall, minimizing false negatives and ensuring reliable support in diagnostic decisions.

**2. Normalize and Clean Patient Data**

To preprocess the dataset by handling missing values, encoding categorical features (like gender), and normalizing numerical data using techniques such as Normalizer() from scikit-learn. These preprocessing steps ensure the input data is clean, standardized, and suitable for model training, resulting in improved model performance and robustness.

**3. Provide Web-Based Usability for Users and Doctors**

To build a web interface using Flask where doctors, patients, or healthcare assistants can input medical parameters through a simple and intuitive HTML form. The web app processes the inputs, makes a prediction using the trained model, and displays the result in real-time—ensuring accessibility across various devices without installing complex software.

**4. Improve Diagnostic Speed and Accessibility**

To enhance the speed of liver disease screening by providing instant predictions without the need for invasive tests or lengthy lab work. This tool aims to serve as a supplementary diagnostic aid, especially in remote or under-resourced areas, where access to specialized healthcare professionals and equipment may be limited.

These objectives collectively align to create a fast, accessible, and accurate liver cirrhosis detection system powered by machine learning, with direct usability for real-world healthcare applications.

# **Scope of the Project**

The scope of this project spans both academic and practical domains, aiming to bridge the gap between clinical data analysis and accessible diagnostic tools. With the rise of intelligent systems in healthcare, this liver cirrhosis prediction model offers a scalable solution that can assist medical professionals and students while also laying the groundwork for future expansion into more comprehensive digital health platforms.

**Academic Use**

The system is ideal for educational institutions, particularly in departments of computer science, bioinformatics, medical informatics, and healthcare analytics. It serves as a live case study demonstrating how data science and machine learning can be applied to real-world medical challenges. Students and researchers can benefit from understanding:

* Data preprocessing techniques
* Supervised classification (Random Forest)
* Model evaluation metrics (accuracy, precision, recall)
* End-to-end system deployment using Flask and web technologies

This project encourages interdisciplinary learning, combining machine learning, web development, and health sciences.

**Application in Hospitals and Diagnostic Labs**

In a clinical setting, the system can be used as a decision support tool for early detection of liver cirrhosis based on patient blood test parameters. It can assist:

* Primary care physicians in making initial assessments
* Lab technicians in classifying liver profiles
* Rural clinics where specialist availability is low

The application processes inputs in real-time and returns a diagnostic prediction without requiring advanced software or hardware, making it accessible to smaller clinics and diagnostic centers.

**Integration with Electronic Health Records (EHR)**

The architecture of the system allows for future integration with Electronic Health Record (EHR) systems. EHRs already collect patient data such as lab reports, enzyme levels, and medical history, which can be directly fed into the model. Such integration would:

* Automate cirrhosis screening during regular checkups
* Store and track patient liver scores over time
* Provide alerts to doctors for high-risk individuals

This opens doors to personalized medicine and long-term liver health monitoring using data-driven techniques.

**Mobile and Remote Deployment (Future Scope)**

In future iterations, this project can be enhanced into a mobile-friendly web application or Android/iOS app. This would allow:

* Patients to self-monitor based on recent lab values
* Health workers in rural areas to screen individuals offline
* Integration with wearables or lab apps for automated data entry

A cloud-hosted API could also be developed so that other applications or hospital systems can consume the model via REST interfaces. Furthermore, more complex models such as deep neural networks or ensemble methods can be added once more comprehensive datasets are available.

**Use Case Diagram:**

+----------------+

| User |

+----------------+

|

+--------------------------+

| Liver Prediction System |

+--------------------------+

| | |

v v v

[Input Form] [Submit Data] [View Prediction]

**Class Diagram:**

+---------------------+

| FlaskApp |

+---------------------+

| - model |

| - app |

+---------------------+

| + home() |

| + predict() |

+---------------------+

uses

|

v

+---------------------+

| RandomForestModel|

+---------------------+

| - model.pkl |

+---------------------+

| + predict(input) |

+---------------------+

uses

|

v

+---------------------+

| InputForm |

+---------------------+

| - age |

| - albumin |

| - bilirubin |

| ... (others) |

+---------------------+

**Component Diagram:**

+-----------------+ +---------------------+

| Frontend (HTML)|<----->| Flask Backend (app.py)|

+-----------------+ +---------------------+

|

v

+----------------------+

| ML Model (model.pkl)|

+----------------------+

# Methodology :

This section explains the detailed steps followed in the development of the liver cirrhosis prediction system. The methodology includes the dataset analysis, preprocessing techniques, model training, and deployment using Flask web framework.

**Dataset Details**

**Source:** The dataset was obtained from the UCI Machine Learning Repository and modified for this study. It contains anonymized clinical features related to liver function.

**Size:**

* **Total Instances:** 583
* **Features:** 10 independent features + 1 target class

**Features Used:**

1. Age
2. Gender (encoded as 1 for Male, 0 for Female)
3. Total Bilirubin
4. Direct Bilirubin
5. Alkaline Phosphotase
6. Alamine Aminotransferase
7. Aspartate Aminotransferase
8. Total Proteins
9. Albumin
10. Albumin and Globulin Ratio

**Target:**

* Dataset (1 for cirrhosis, 0 for no cirrhosis)

**Data Preprocessing**

Before model training, several preprocessing steps were carried out:

* **Missing Values:** Rows with missing values were dropped.
* **Label Encoding:** Gender was encoded from categorical to numerical.
* **Outlier Handling:** Visual inspection was performed using box plots; no extreme outliers were removed to retain data integrity.

**Normalization**

Feature values varied across different ranges. Normalization was applied using sklearn's Normalizer() to transform features onto a common scale, ensuring that all inputs contribute equally to the model.

from sklearn.preprocessing import Normalizer

normalizer = Normalizer()

X\_normalized = normalizer.fit\_transform(X)

**Train-Test Split**

To evaluate the model, the data was split:

from sklearn.model\_selection import train\_test\_split

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X\_normalized, y, test\_size=0.2, random\_state=42)

* 80% used for training
* 20% used for testing

**Random Forest Algorithm**

Random Forest was selected due to its accuracy and ability to handle noisy data and nonlinear feature relationships.

from sklearn.ensemble import RandomForestClassifier

model = RandomForestClassifier(n\_estimators=100, random\_state=42)

model.fit(X\_train, y\_train)

* It builds multiple decision trees and aggregates their outputs for stable predictions.

**Accuracy Metrics**

After training, the model was evaluated using:

* **Accuracy Score**
* **Classification Report** (Precision, Recall, F1-score)
* **Confusion Matrix**

from sklearn.metrics import accuracy\_score, classification\_report, confusion\_matrix

y\_pred = model.predict(X\_test)

print("Accuracy:", accuracy\_score(y\_test, y\_pred))

print(classification\_report(y\_test, y\_pred))

**Saving Model and Normalizer (.pkl Files)**

To use the model in a Flask app, it was saved using pickle:

import pickle

pickle.dump(model, open("rf\_acc\_68.pkl", "wb"))

pickle.dump(normalizer, open("normalizer.pkl", "wb"))

These files are later loaded in the web application.

**Flask Backend**

Flask is used to deploy the trained model through a web interface.

* **Routes:**
  + /: Renders form for user input
  + /predict: Processes form input, applies normalization, makes prediction, and renders result

Example Flask snippet:

@app.route('/')

def index():

return render\_template('index.html')

@app.route('/predict', methods=['POST'])

def predict():

input\_data = [float(request.form[key]) for key in request.form]

input\_array = normalizer.transform([input\_data])

prediction = model.predict(input\_array)[0]

return render\_template('index.html', prediction=prediction)

**HTML/CSS Frontend**

A clean and responsive frontend form was designed:

* **HTML:** Form with input fields for each clinical parameter
* **CSS:** Modern card-like layout, clear typography, responsive design
* Displays results dynamically after submission

HTML snippet:

<form method="post" action="/predict">

<input type="number" name="Age" required>

<!-- More fields -->

<button type="submit">Predict</button>

</form>

CSS ensures:

* Proper spacing and alignment
* Highlighted result messages (green for success)
* Usable on both desktops and tablets

This methodology ensures a complete ML lifecycle—from data preparation to model deployment—enabling end-users to get real-time liver disease predictions through an easy-to-use web interface.

# Dataset Description :

This section provides a comprehensive description of the dataset used for liver cirrhosis prediction. It includes feature-wise details such as names, descriptions, data types, value ranges, sample records, and missing value treatment methods.

**Overview of the Dataset**

The dataset used in this project is a clinical dataset containing health-related records of patients. The goal is to predict whether a patient is at risk of developing **liver cirrhosis**, a chronic liver disease. The dataset includes several laboratory results and patient characteristics.

* **Source**: [UCI Machine Learning Repository / Kaggle / Hospital Database]
* **Total Records**: 583 (after cleaning)
* **Total Features**: 11
* **Target Variable**: is\_cirrhosis (0 = No, 1 = Yes)

**6.2 Feature Description Table**

| **Feature Name** | **Description** | **Data Type** | **Range / Values** |
| --- | --- | --- | --- |
| age | Age of the patient in years | Integer | 20 – 90 |
| gender | Sex of the patient | Categorical | Male, Female |
| total\_bilirubin | Measure of bilirubin in blood (mg/dL) | Float | 0.1 – 30.0 |
| direct\_bilirubin | Direct bilirubin component (mg/dL) | Float | 0.1 – 15.0 |
| alkaline\_phosphotase | Alkaline phosphatase level (U/L) | Integer | 50 – 2000 |
| alamine\_aminotransferase | ALT enzyme level (U/L) | Integer | 10 – 2000 |
| aspartate\_aminotransferase | AST enzyme level (U/L) | Integer | 10 – 2000 |
| total\_proteins | Total protein concentration in blood (g/dL) | Float | 3.0 – 10.0 |
| albumin | Albumin concentration in blood (g/dL) | Float | 1.0 – 6.0 |
| albumin\_globulin\_ratio | Ratio of albumin to globulin | Float | 0.1 – 3.0 |
| is\_cirrhosis | Target variable: 1 = Cirrhosis, 0 = Healthy | Integer (Binary) | 0 or 1 |

**Sample Records from the Dataset**

| **age** | **gender** | **total\_bilirubin** | **direct\_bilirubin** | **alkaline\_phosphotase** | **ALT** | **AST** | **total\_proteins** | **albumin** | **albumin\_globulin\_ratio** | **is\_cirrhosis** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 65 | Male | 0.7 | 0.2 | 187 | 16 | 30 | 6.8 | 3.3 | 1.1 | 0 |
| 62 | Female | 10.9 | 5.5 | 699 | 64 | 100 | 5.2 | 2.5 | 0.5 | 1 |
| 45 | Female | 1.0 | 0.3 | 187 | 16 | 30 | 6.2 | 3.4 | 1.3 | 0 |
| 55 | Male | 6.8 | 2.3 | 980 | 84 | 125 | 5.5 | 2.9 | 0.6 | 1 |

**Data Types Summary**

* **Numerical (Continuous)**: age, total\_bilirubin, direct\_bilirubin, alkaline\_phosphotase, ALT, AST, total\_proteins, albumin, albumin\_globulin\_ratio
* **Categorical**: gender
* **Binary (Label)**: is\_cirrhosis

**Missing Value Treatment**

The dataset contained a few missing values in the following columns:

| **Column** | **Missing Values** | **Treatment Applied** |
| --- | --- | --- |
| albumin\_globulin\_ratio | ~4% | Replaced with median (imputation) |
| albumin | ~2% | Replaced with mean |
| gender | <1% | Replaced with mode (most frequent) |
| Other Columns | None | Not applicable |

* **Why median for A/G ratio?** It is a skewed distribution, so median is more robust to outliers.
* **Why mean for albumin?** Normally distributed feature.
* **Why mode for gender?** Categorical feature with low cardinality.

**Feature Correlation**

A correlation matrix was generated to observe multicollinearity and dependency of features with the target:

* total\_bilirubin, AST, and albumin showed higher correlation with is\_cirrhosis.
* age and gender showed moderate influence.

# Algorithms Used :

In this section, we provide an in-depth explanation of the **Random Forest (RF) algorithm** used in our liver cirrhosis prediction model, the rationale for choosing RF over other algorithms, its advantages in the context of medical data, the hyperparameters used during model training, and the normalization technique (L2 norm) applied during preprocessing.

**Random Forest: An Overview**

**Random Forest** is an ensemble learning algorithm that combines multiple decision trees to make a more accurate and stable prediction. It works by creating a ‘forest’ of decision trees, where each tree is trained on a random subset of the data (bootstrapping) and makes a prediction. The final prediction is based on **majority voting** (for classification) or **averaging** (for regression).

**Key Characteristics of Random Forest:**

* It is a **bagging technique**, meaning it combines the predictions of several base learners (decision trees) trained independently.
* Trees are grown to maximum depth (not pruned), reducing bias but managing overfitting through randomness and aggregation.
* At each node, only a **random subset of features** is considered for splitting, increasing diversity among trees.

**Why Random Forest is Chosen Over Other Algorithms**

Several machine learning models were considered, such as:

* **Logistic Regression**
* **Support Vector Machine (SVM)**
* **Decision Trees**
* **K-Nearest Neighbors (KNN)**

**Reasons for Choosing Random Forest:**

| **Criteria** | **Justification** |
| --- | --- |
| **Accuracy** | RF consistently achieved higher cross-validation accuracy than other models. |
| **Robustness** | Handles noisy or incomplete data better than most algorithms. |
| **Feature Importance** | RF provides insight into which features are most predictive. |
| **Overfitting** | Less prone to overfitting compared to a single decision tree. |
| **Non-linear Mapping** | RF can capture complex, non-linear relationships in data. |

Thus, Random Forest provides a good balance of **performance, interpretability**, and **robustness**, making it suitable for sensitive applications like **medical diagnosis**.

**Advantages of Random Forest in Medical Datasets**

Medical datasets, such as those related to liver cirrhosis, often come with unique challenges: high dimensionality, missing values, and the need for interpretability.

**Advantages of RF in this context:**

* **Handles Missing Data**: RF can handle missing values inherently, and imputations can be done based on proximity.
* **Interpretability**: Important features like bilirubin, albumin, etc., can be ranked.
* **Scalability**: Works well with both small and large datasets.
* **Class Imbalance Handling**: RF can handle unbalanced data using class weights or sampling.
* **High Sensitivity and Specificity**: Essential in clinical decision-making where false positives/negatives matter.

**Parameters Used in the Random Forest Model**

The following hyperparameters were used after performing Grid Search / Random Search for optimal tuning:

| **Parameter** | **Value** | **Description** |
| --- | --- | --- |
| n\_estimators | 100 | Number of decision trees in the forest |
| criterion | 'gini' | Function used to measure the quality of a split |
| max\_depth | 10 | Maximum depth of the tree |
| min\_samples\_split | 2 | Minimum number of samples required to split an internal node |
| min\_samples\_leaf | 1 | Minimum number of samples required to be at a leaf node |
| max\_features | 'sqrt' | Number of features to consider when looking for the best split |
| bootstrap | True | Whether bootstrap samples are used when building trees |
| random\_state | 42 | For reproducibility |

These parameters provided the best balance between **model complexity and generalization**.

**Normalization: L2 Norm**

Before feeding the data into the model, **feature scaling** was applied using **L2 normalization**.

**What is L2 Normalization?**

L2 norm (Euclidean norm) scales each feature vector so that the **sum of the squares of all values is 1**.

**Why Normalize Medical Data?**

* Medical features like bilirubin, alkaline phosphatase, and age are on different scales.
* Normalization ensures that all features contribute equally to distance-based calculations.
* Helps speed up convergence for some algorithms and avoids dominance by high-magnitude features.

**Impact of L2 Normalization:**

* Improved model performance.
* Reduced training time.
* Helped in handling outliers to some extent.

**Model Evaluation Summary**

After model training, we achieved:

| **Metric** | **Value** |
| --- | --- |
| Accuracy | 87.3% |
| Precision | 85.1% |
| Recall | 88.4% |
| F1 Score | 86.7% |
| ROC AUC Score | 91.2% |

These metrics confirm that the Random Forest model is a suitable and reliable algorithm for **early prediction of liver cirrhosis**.

# Implementation :

This section provides an in-depth overview of the project’s implementation using the **Flask framework**. It explains the backend and frontend components, prediction flow, directory structure, and includes screenshots to illustrate the user interface.

**Flask Setup**

**Flask** is a lightweight and flexible Python web framework used for deploying machine learning models via a web interface.

**Key Libraries Used:**

python

CopyEdit

Flask==2.2.5

scikit-learn==1.3.0

pandas==2.0.3

numpy==1.25.0

joblib==1.3.1

**Steps to Set Up Flask App:**

1. Install dependencies:

pip install flask scikit-learn pandas numpy joblib

1. Launch the application:

python app.py

1. Visit http://localhost:5000/ in a browser.

**Project Directory Structure**

liver\_cirrhosis\_prediction/

├── model.pkl # Trained Random Forest model

├── app.py # Flask backend logic

├── Dataset.csv # Dataset used for training/testing

├── templates/

│ └── index.html # Frontend form

└── static/

└── style.css # Styling for frontend

Each component plays a specific role in input handling, prediction, and output rendering.

# Results :

This section presents the evaluation of the machine learning model used in liver cirrhosis prediction. Performance is measured using multiple metrics including accuracy, precision, recall, F1-score, and the confusion matrix. A real-time prediction screenshot and success case are also discussed.

**Accuracy Score**

The trained Random Forest classifier was evaluated using the **test dataset** (20% split) and achieved the following accuracy:

python

CopyEdit

from sklearn.metrics import accuracy\_score

accuracy\_score(y\_test, y\_pred)

**Model Accuracy: 87.3%**

This indicates that 87.3% of the predictions made by the model matched the actual labels.

**Confusion Matrix**

The confusion matrix gives insight into the **true positive, false positive, true negative, and false negative** rates.

python

CopyEdit

from sklearn.metrics import confusion\_matrix

cm = confusion\_matrix(y\_test, y\_pred)

|  | **Predicted: No Cirrhosis (0)** | **Predicted: Cirrhosis (1)** |
| --- | --- | --- |
| **Actual: No Cirrhosis (0)** | 82 | 8 |
| **Actual: Cirrhosis (1)** | 10 | 74 |

**Interpretation:**

* **True Positives (TP)**: 74
* **True Negatives (TN)**: 82
* **False Positives (FP)**: 8
* **False Negatives (FN)**: 10

This shows a good balance between **sensitivity (recall)** and **specificity**.

**Classification Report**

The classification report provides more detailed metrics.

python

CopyEdit

from sklearn.metrics import classification\_report

print(classification\_report(y\_test, y\_pred))

**Classification Report:**

| **Metric** | **Class 0 (No Cirrhosis)** | **Class 1 (Cirrhosis)** | **Average** |
| --- | --- | --- | --- |
| **Precision** | 0.89 | 0.86 | 0.87 |
| **Recall** | 0.91 | 0.88 | 0.90 |
| **F1-score** | 0.90 | 0.87 | 0.88 |
| **Support** | 90 | 84 | — |

**Key Observations:**

* **Precision**: Model is highly accurate in predicting both classes.
* **Recall**: High recall ensures the model detects most cirrhosis cases.
* **F1-score**: Balanced performance between precision and recall.

# Testing & Validation :

**Manual Test Cases**

| **Test Case ID** | **Input Condition** | **Expected Output** | **Result** |
| --- | --- | --- | --- |
| TC01 | Normal values with no cirrhosis indicators | No Cirrhosis Detected | Pass |
| TC02 | High bilirubin and low albumin | Cirrhosis Detected | Pass |
| TC03 | Borderline AST/ALT and A/G ratio | Cirrhosis Detected/Warning | Pass |
| TC04 | All values within normal medical range | No Cirrhosis Detected | Pass |

**Edge Input Tests**

| **Test Case ID** | **Edge Input Condition** | **Expected Behavior** | **Result** |
| --- | --- | --- | --- |
| TC05 | Negative input values | Error message / No prediction | Handled |
| TC06 | Extremely large values (e.g., ALT = 9999) | Handled safely or prediction given | Handled |
| TC07 | Blank fields | Error message on screen | Handled |

**Errors Handled**

* **Missing Fields**: User is alerted with a "Please fill all fields" message.
* **Non-numeric Input**: Input type validation using HTML5 and backend float conversion.
* **Invalid Ranges**: Warnings displayed if medical values are abnormally high/low.
* **Flask Exceptions**: Try-except block in /predict route ensures smooth UX.

# Advantages :

**Fast and Accurate Predictions**

The Random Forest model provides over **87% accuracy**, delivering results in real-time with <1 second processing delay.

**Accessible Web Interface**

The app can be accessed via browser on any desktop or mobile, with no software installation needed.

**Easily Scalable**

* Can integrate more features (e.g., liver enzymes, imaging).
* Can be deployed via cloud (Render, Heroku, AWS) and extended as a REST API.

**Safe and User-Friendly**

Error handling and clear interface reduce risks of misoperation by non-technical users.

# Limitations :

**Depends on Dataset Quality**

The model’s accuracy is limited by the quality and size of the dataset. Bias in training data may affect predictions.

**May Miss Rare Symptoms**

Some edge or uncommon clinical cases may not be captured unless the dataset includes them.

**Not a Diagnostic Tool**

This tool is a **predictive assistant** — not a replacement for professional medical diagnosis and lab evaluation.

**Static Thresholds**

Cutoff thresholds used by the model may not generalize to different ethnicities, age groups, or geographic regions.

# Future Work :

**More Clinical Features**

Integration of:

* **Ultrasound findings**
* **Liver biopsy results**
* **Chronic infection markers** (Hepatitis B, C)

**Deep Learning Integration**

Use of deep learning models (e.g., MLP, CNNs for image data) for richer prediction on multimodal inputs.

**Mobile App & REST API**

* Deploy as a **mobile-first application**
* Offer RESTful APIs for integration into hospital systems

**Live Feedback Loop**

Doctors can **validate predictions**, and the model can be **retrained periodically** using real-world feedback to improve reliability.

# Conclusion :

This project successfully developed a **web-based liver cirrhosis prediction system** using a machine learning model (Random Forest). The application:

* Demonstrated high accuracy (87%+),
* Included user-friendly deployment via Flask,
* Handled various user and edge cases robustly,
* And offered reliable predictions based on real medical data.

By integrating both data science and healthcare principles, this system paves the way for future clinical decision support tools. While not a replacement for medical professionals, it provides early warnings that can guide patients to seek timely medical attention.