Project Documentation: Predicting Liver Cirrhosis Using Machine Learning

Date: 26 June 2025

1. Introduction

Project Title: Predicting Liver Cirrhosis Using Machine Learning

Team Members:
– Kotipalli.S.S.N.V.S.S.Akshay – Machine Learning Developer
– Kotipalli.S.S.N.V.S.S.Akshay – Flask Developer
– Kotipalli.S.S.N.V.S.S.Akshay – UI/UX Designer

2. Project Overview

Purpose:
To detect liver cirrhosis early using clinical parameters and machine learning, helping doctors make faster and more accurate decisions.

Features:
- User input form for medical test values
- Model predicts if liver cirrhosis is likely
- Clean UI using HTML/CSS
- Flask backend serving predictions
- Trained model integration (Random Forest)

3. Architecture

Frontend:
Simple web UI using HTML, CSS (with responsive styling)

Backend:
Flask server with endpoints for form input and prediction

Model & Data:
Trained using ILPD dataset with Random Forest + Standard Scaler (scikit-learn)

4. Setup Instructions

Prerequisites:
Python 3.9+, pip, Flask, joblib, scikit-learn, pandas, numpy

Installation:
1. Clone or extract the project folder
2. Place rf_model.pkl and scaler.pkl in the project root
3. Run: pip install -r requirements.txt
4. Start: python app.py

5. Folder Structure

/templates/index.html → UI form
/static/style.css → UI design
app.py → Flask backend
rf_model.pkl → Trained model
scaler.pkl → Data scaler
train_model.ipynb → Model training notebook

6. Running the Application

Start the backend:
python app.py

Open browser and go to:
http://127.0.0.1:5000

7. API Documentation

/ → GET → Renders index.html
/predict → POST → Accepts form input, returns prediction

Example Input:
{ "Age": 45, "Gender": 0, "TB": 1.2, ... }

Example Output:
Liver Disease Detected

8. Authentication

No authentication is used in this academic project. Future versions may include doctor logins or admin dashboard.

9. User Interface

Includes:
- Two-column responsive form
- Input validation for clinical values
- Responsive design with soft gradient and hover effects

10. Testing

Manual testing using known input values.
Accuracy score validated with test set in train_model.ipynb.
Evaluation metrics used: accuracy, classification report

11. Screenshots or Demo

Insert screenshots of the form or prediction result (optional).

12. Known Issues

- No input validation for incorrect formats
- Model performance dependent on dataset quality

13. Future Enhancements

- Add user login (doctor, admin)
- Store patient history in database
- Allow CSV uploads for batch predictions
- Integrate model explanation (SHAP or LIME)