Project Phase Report

Project title

Liver Care: Predicting Liver Cirrhosis Using Advanced Machine Learning

Team Name

Team Member

K.V.S.NIHARIKA

Phase-1: Brainstorming & Ideation

Objective:

- Identify and define the healthcare issue.
- Establish the purpose of the project.

Problem Statement:

- Liver cirrhosis is often diagnosed at an advanced stage, limiting treatment options.
- Many hospitals lack resources or tools for early and effective screening.
- Blood test results can be misinterpreted without expert input.

Proposed Solution:

- Create a predictive model using machine learning techniques to detect liver cirrhosis.
- Use clinical features like age, gender, enzyme levels, and protein counts.
- Build a web interface using Flask for real-time predictions.

Target Users:

- Primary care doctors and general physicians.
- Rural and semi-urban hospitals with limited diagnostic capabilities.
- Medical interns and nursing staff in community health centers.

Expected Outcome:

- A working ML model capable of predicting liver cirrhosis risk accurately.
- A web-based tool that provides prediction in real time.
- Empowerment of healthcare workers with a decision support system.

Phase-2: Requirement Analysis

Objective:

- Clearly identify what is needed to successfully build the project.
- Break down both technical and functional needs before development.

Technical Requirements:

- Programming Language: Python (used for scripting, ML, and backend logic).
- Web Framework: Flask (lightweight Python web framework).
- ML Libraries: Scikit-learn (model creation), Pandas and NumPy (data handling), Pickle (model saving).
- Frontend: HTML, CSS for the UI form and output page.
- Data Source: Indian Liver Patient Dataset (CSV format).
- Model Deployment: Flask for serving predictions via HTTP.
- Tools: VS Code or Jupyter Notebook for development, browser for UI testing.
- Optional: Matplotlib/Seaborn for visualizing data trends during analysis.

Functional Requirements:

- Upload and process the liver dataset
- Clean and transform the data for model training.
- Train a machine learning model (e.g., Logistic Regression).
- Evaluate and save the model using Pickle.
- Accept input values from the user through a web form.

Constraints & Challenges:

- Data Imbalance: Original dataset has more cirrhosis cases than healthy ones, risking bias
- Accuracy Limitation: ML model may not perform well on unseen or noisy data.
- Scalability: Flask is good for local testing but limited for production-scale deployment.
- Input Handling: Need strict validation to avoid crashes on blank or invalid input
- Interpretability: Logistic Regression is interpretable, but deeper models like neural nets aren’t.

Phase-3: Project Design

Objective:

- Design the structure of the system for clear implementation.
- Define the interaction between different components like the model, backend, and UI.

System Architecture Diagram:

The system follows a modular design:

- User Interface (Frontend): HTML/CSS-based form for data input.
- Web Server (Flask): Accepts user input, validates and sends it for prediction.
- Preprocessing Module: Scales user input using the saved StandardScaler.
- ML Model (Pickle File): Loads the trained model and predicts the result.
- Response Handler: Sends prediction back to frontend and displays result.
- Optional architecture enhancements can include logging, input validation, and cloud hosting layers.

User Flow:

- User opens the web page and sees the liver prediction form.
- User selects gender and enters values like age, enzyme levels, etc.
- User submits the form to check for cirrhosis.
- Flask server receives the input and processes it.
- Input is scaled and passed to the ML model.
- Model returns a prediction (1 or 0).
- Flask renders the result page with a clear diagnosis message.
- User can go back to form and try another input if needed.

UI/UX Considerations:

- The form should have clear labels and placeholders.
- Use dropdowns for options like Gender to prevent typing errors.
- Each field should validate for required numeric input.
- Color-coded results (e.g., green for "No Cirrhosis", red for "Cirrhosis Detected") improve clarity.
- Ensure mobile-friendly layout with responsive CSS.
- Use modern fonts and spacing for a professional look.
- Provide navigation back to home or prediction form.
- Keep the number of fields minimal but sufficient for accurate prediction.

Phase-4: Project Planning (Agile Methodologies)

Objective:

- Organize the development process using Agile methodology.
- Break down the project into manageable, time-bound tasks.

Sprint Planning:

The project is divided into 3 major sprints:

Sprint 1: Data &amp; Model (Week 1)

- Data collection and cleaning.
- Dataset analysis and feature selection.
- Model training, validation, and saving with Pickle

Sprint 2: Web App Development (Week 2)

- Set up Flask backend.
- Create frontend input form (HTML/CSS).
- Connect input form to prediction logic.

Sprint 3: UI & Testing (Week 3)

- Design results display and styling.
- Perform model and interface testing.
- Debugging and validation.
- Final touches and preparation for deployment/demo Task Allocation
- Team Member A: Data cleaning, analysis, and ML model development.
- Team Member B: Web app development (Flask integration and backend logic).
- Team Member C: HTML/CSS frontend design and user interface.
- Team Member D: Testing, documentation, and presentation preparation.
- Collaboration in Git (or shared drive) to integrate work weekly.

Timeline & Milestones:

- Week 1 Milestone: Trained model with performance report and saved model files.
- Week 2 Milestone: Basic working Flask app with input/output connection.
- Week 3 Milestone: Fully styled UI with testing completed and prediction accuracy verified.
- Final Presentation Prep: End of Week 3 – presentation slide design, code clean-up, demo ready.
- Continuous review after each sprint to adjust priorities and handle blockers.

Phase-5: Project Development

Objective:

- Begin coding the entire project and ensure all components work together.
- Transform designs and plans into a fully functioning application.

Technology Stack Used:

- Programming Language: Python (for model training and backend logic).
- Framework: Flask (to build and run the web application).
- Machine Learning: Scikit-learn for training the Logistic Regression model.
- Data Handling: Pandas and NumPy for cleaning and manipulating the dataset.
- Model Serialization: Pickle to save and load the model and scaler.
- Frontend: HTML and CSS for designing the user interface.
- Development Tools: VS Code, Jupyter Notebook for experimentation.
- Libraries Used: StandardScaler, resample (from sklearn), Flask routing, form handling.

Development Process:

- Collected and cleaned the liver dataset, handled missing values.
- Encoded categorical data (like Gender), mapped labels to binary.
- Balanced the dataset to avoid biased predictions.
- Scaled the features using StandardScaler to normalize inputs.
- Trained the model using Logistic Regression and evaluated performance.
- Saved both the model and scaler using Pickle.
- Set up Flask project structure and integrated the trained model.
- Developed HTML form for collecting user input.

Challenges & Fixes:

- Data Imbalance: The dataset had more cirrhosis cases.
- Solution: Used up sampling for minority class.
- String-to-float Errors: Input fields like Gender caused errors. Fix: Encoded gender properly and validated inputs.
- Model Bias: The Random Forest model was over fitting. Fix: Switched to Logistic Regression for better balance.

Phase-6: Functional & Performance Testing

Objective:

- Validate that all features and components of the project work correctly.

Test Cases Executed:

- Valid Input Test: Submitted accurate values for healthy and cirrhotic cases → system returned correct results.
- Invalid Input Test: Left fields blank or added invalid strings → system showed error or prevented submission.
- Gender Handling: Checked both “Male” and “Female” inputs → mapped correctly to 1/0 for prediction.
- Field Order Test: Ensured correct input order matching model training → prediction became accurate.
- UI Responsiveness: Tested layout on mobile and desktop → adjusted correctly.

Bug Fixes & Improvements:

- Fixed: Empty input fields causing crash → added input validation.
- Fixed: Gender string-to-float conversion error → used gender encoding.
- Fixed: Wrong prediction for healthy inputs → corrected input feature order.
- Fixed: Over fitting in Random Forest model → switched to Logistic Regression.
- Improved: HTML form styling and mobile responsiveness.
- Improved: Prediction output clarity using styled result cards.

Final Validation:

- The application accepts user inputs and performs predictions successfully.
- The system handles input errors gracefully and provides helpful feedback.
- All features listed in the functional requirements are implemented.
- The model gives reasonable and interpretable outputs.
- The prediction interface is responsive, clear, and user-friendly.
- Meets project goals: early cirrhosis detection, ML-backed prediction, clean interface.

Deployment (If Applicable):

- Current deployment: Local Flask server on localhost 127.0.0.1:5000
- Ready for deployment to platforms like Render, PythonAnywhere, or Heroku
- Deployment plan (optional): -

  Package the project, Host HTML + Flask backend, Share the public URL for hospital or field use

- Demo prepared for final presentation with working form and output
- Deployment ensures accessibility beyond local environment

GitHub Repo: https://github.com/Niharika0254/Revolutionizing-Liver-Care-Predicting-Liver-Cirrhosis-using-Advanced-Machine-Learning-Techniques/tree/main

Final Submission Checklist

• Demo Video https://drive.google.com/file/d/1ckAF-BG6VPQID7yAp4zNObgKDgLBbAFC/view?usp=sharing

• GitHub Code Repository

https://github.com/Niharika0254/Revolutionizing-Liver-Care-Predicting-Liver-Cirrhosis-using-Advanced-Machine-Learning-Techniques/tree/main/Project%20Files

• PowerPoint Presentation

https://docs.google.com/presentation/d/13U-lmBN9pqe4Pt7ZX-5ggmMUQX0owvuJ/edit?usp=sharing&ouid=104095150835659803901&rtpof=true&sd=true