AI/ML Project Report

Revolutionizing Liver Care: Predicting Liver Cirrhosis Using Advanced Machine Learning Techniques

Team ID: LTVIP2025TMID41907

Team Size: 4

Team Leader: Bali Ravi Sankar

Team Member: Shaik Muskina

Team Member: S Chandana

Team Member: Shaik Ikrakishwar

1. INTRODUCTION

1.1 Project Overview

This project focuses on developing an AI-powered system for the early prediction and classification of liver cirrhosis using advanced machine learning techniques. By leveraging comprehensive patient data, including clinical, laboratory, and potentially imaging features, the system aims to provide a non-invasive, data-driven approach to identify individuals at risk or in early stages of liver cirrhosis, thereby enabling timely intervention and improved patient outcomes.

1.2 Purpose

- Support early identification of liver cirrhosis to prevent disease progression and complications.
- Provide accessible and efficient diagnostic support for healthcare professionals, especially in settings with limited specialist resources.
- Minimize dependence on invasive and costly diagnostic procedures like liver biopsy.
- Demonstrate the practical application of advanced machine learning in chronic disease prediction and health monitoring.
- Contribute to data-driven, personalized liver care and enhance clinical decision-making.

2. IDEATION PHASE

2.1 Problem Statement

Liver cirrhosis is a major global health concern, often diagnosed at advanced stages when treatment options are limited and prognosis is poor. Traditional diagnostic methods can be invasive, expensive, and sometimes delayed, leading to significant morbidity and mortality. There is a critical need for automated, accurate, and scalable predictive tools that can identify individuals at risk early, enabling proactive management and potentially reversing or slowing disease progression.

2.2 Empathy Map Canvas

"Sees"	**"Hears"**	**"Thinks & Feels"**	**"Says & Does"**	**"Pains"**	**"Gains"**
Patient: Symptoms like fatigue, jaundice, swelling; doctor's concerns; complex medical terms.	**Patient:** Doctor's explanations, family worries, online health information.	**Patient:** Worries about health, future, financial burden; confusion about diagnosis.	**Patient:** Asks questions, follows doctor's advice, researches online.	**Patient:** Late diagnosis, invasive tests, fear of complications, high costs.	**Patient:** Early detection, less invasive diagnosis, peace of mind, better prognosis.
Doctor: Lab results, patient history, imaging; signs of liver damage.	**Doctor:** Patient symptoms, specialist consultations, latest research.	**Doctor:** Concerns about accurate diagnosis, treatment efficacy, patient compliance.	**Doctor:** Orders tests, prescribes medication, refers to specialists, explains prognosis.	**Doctor:** Diagnostic delays, ambiguous results, patient anxiety, limited resources.	**Doctor:** Faster, more accurate predictions, improved patient management, reduced workload.

2.3 Brainstorming

- **Rule-Based Systems:** Simple, but lack nuance for complex biological interactions.
- **Traditional Machine Learning (e.g., Logistic Regression, SVM, Decision Trees):** Good baseline, but may not capture complex non-linear relationships in medical data.
- **Ensemble Methods (e.g., Random Forests, Gradient Boosting):** Offer improved accuracy and robustness by combining multiple models.
- **Deep Learning (e.g., ANNs for tabular data, CNNs for imaging):** Potentially higher accuracy for complex patterns, especially with large datasets and imaging.
- **Chosen Approach: Advanced Machine Learning (Ensemble Methods and potentially Deep Learning for integrated data):** Offers the best balance of predictive power, interpretability (compared to pure black-box deep learning), and practicality for diverse medical datasets.

3. REQUIREMENT ANALYSIS

3.1 Customer Journey

1. **Data Input:** Healthcare professional inputs patient's clinical and lab data into the system.
2. **ML Analysis:** The AI model processes the input data to predict the likelihood of liver cirrhosis.

3. **Result Display:** The system presents the prediction (e.g., risk score, cirrhosis stage) along with confidence metrics.
4. **Recommended Actions:** The system may suggest further diagnostic steps or management strategies based on the prediction.
5. **Record & Share:** The healthcare professional can log the prediction and share it with the patient or other specialists.

3.2 Requirements

Tech Stack:

- **Programming Language:** Python
- **ML Frameworks:** Scikit-learn, TensorFlow/Keras (for deep learning components if integrated)
- **Data Manipulation:** Pandas, NumPy
- **Web Framework:** Flask/FastAPI (Backend), React/Vue/Angular (Frontend)
- **Database:** PostgreSQL/MongoDB (for patient data, prediction logs)
- **Cloud Platform:** AWS/GCP/Azure (for deployment and scalability)
- **IDE:** VS Code, Jupyter Notebooks

Core Functionalities:

- Secure input and preprocessing of patient clinical and laboratory data.
- Predict liver cirrhosis risk/stage based on input features.
- Display prediction results with probability scores and confidence intervals.
- Provide explainability for predictions (e.g., feature importance).
- Maintain a history of predictions for patient monitoring.
- User authentication and authorization for data security.

3.3 Data Flow

1. **Data Input:** Patient clinical and lab data is entered via the user interface.
2. **Storage Buffer:** Data temporarily stored or directly passed to preprocessing.
3. **Preprocessing:** Data cleaning, normalization, feature engineering (e.g., handling missing values, scaling features).
4. **Model Inference:** Preprocessed data is fed into the trained machine learning model.
5. **Prediction Returned:** The model outputs a prediction (e.g., probability of cirrhosis, predicted stage).
6. **Diagnosis Details Shown on UI:** Prediction results, confidence, and relevant insights are displayed.

7. **Optionally Saved in Database:** Predictions and associated patient data can be logged for historical tracking and auditing.

3.4 Technology Stack

- **ML Frameworks:** Scikit-learn (for traditional ML and ensemble methods), TensorFlow/Keras (for neural networks, if used for complex patterns or multimodal data).
- **Data Processing Libraries:** Pandas (for data manipulation), NumPy (for numerical operations).
- **Web Frameworks:** Flask/FastAPI (Python backend for API), React/Vue (JavaScript frontend for interactive UI).
- **Cloud Services:** AWS Sagemaker/GCP AI Platform (for model deployment), AWS S3/GCP Cloud Storage (for data storage), AWS EC2/GCP Compute Engine (for compute).
- **Database:** PostgreSQL (relational database for structured patient data and logs), MongoDB (NoSQL for flexible data models).
- **Deployment:** Docker, Kubernetes (for containerization and orchestration).
- **APIs:** RESTful APIs for communication between frontend and backend.
- **Extras:** SHAP/LIME (for model interpretability), Streamlit/Dash (for rapid prototyping of ML apps).

4. PROJECT DESIGN

4.1 Solution Fit

This system provides a rapid, data-driven approach to predicting liver cirrhosis, leveraging advanced machine learning models. It empowers healthcare professionals with an efficient, non-invasive tool for early risk assessment, reducing diagnostic delays and supporting timely clinical interventions, ultimately leading to improved patient outcomes and reduced healthcare costs associated with late-stage disease.

4.2 Architecture

- **Frontend:** A responsive web application (React/Vue) for secure data input, visualization of predictions, and historical tracking.
- **Backend:** A Flask/FastAPI server hosting the trained machine learning models, managing data preprocessing, API endpoints, and database interactions.
- **ML Pipeline:**
  - **Components:**
    - **User Interface:** Secure web interface for data entry and result display.
    - **Data Preprocessing Module:** Handles data cleaning, feature scaling, encoding, and missing value imputation.
    - **ML Model Inference Service:** Loads the trained model and performs predictions.

        - **Patient Data Management Database:** Stores patient demographic, clinical, and lab data securely.
        - **Prediction Logger Database:** Records all predictions, confidence scores, and timestamps for auditing and monitoring.
        - **Explainability Module:** Provides insights into feature importance for each prediction.

5. PLANNING & SCHEDULING

5.1 Agile Timeline (Example 4-Week Sprint)

Week 1: Setup & Data Exploration

- Set up development environments (Python, ML frameworks, web frameworks).
- Acquire and explore initial dataset (e.g., public liver disease datasets like UCI Liver Disorders, NAFLD datasets).
- Perform initial data cleaning and exploratory data analysis (EDA).
- Define initial features for model training.

Week 2: Model Development & Training

- Select and implement initial machine learning models (e.g., Random Forest, XGBoost).
- Split data into training, validation, and test sets.
- Train models and perform hyperparameter tuning.
- Evaluate model performance using appropriate metrics (e.g., AUC-ROC, accuracy, precision, recall, F1-score).

Week 3: Backend API & Frontend Integration

- Develop RESTful API endpoints using Flask/FastAPI for data input and model inference.
- Integrate the trained ML model with the backend API.
- Build the frontend user interface for data input and displaying prediction results.
- Connect frontend to backend API.

Week 4: Deployment, Testing & Presentation

- Containerize the application (Docker).
- Deploy the application to a cloud platform (e.g., AWS EC2/GCP App Engine).
- Conduct comprehensive functional and performance testing.
- Fix bugs and refine the UI/UX.
- Prepare final project documentation and presentation.

6. FUNCTIONAL & PERFORMANCE TESTING

6.1 Testing Metrics

- **Prediction Accuracy:** Overall accuracy, sensitivity, specificity, AUC-ROC score.
- **Inference Latency:** Time taken for the model to generate a prediction after data input.
- **Data Processing Speed:** Time taken for data preprocessing.
- **Throughput Under Load:** System performance under concurrent user requests.
- **Resource Utilization:** CPU/GPU usage, memory consumption during inference.
- **Database Response Speed:** Latency for data storage and retrieval operations.
- **System Uptime & Reliability:** Ensuring continuous availability.

6.2 Testing Tools

- **Performance Testing:** Locust, JMeter (for load testing API endpoints).
- **Unit/Integration Testing:** Pytest (for Python backend), Jest/React Testing Library (for React frontend).
- **Monitoring:** Prometheus + Grafana (for system metrics), ELK Stack (Elasticsearch, Logstash, Kibana for log analysis).
- **Model Evaluation:** Scikit-learn's metrics module, custom evaluation scripts.

Program:

```
import React, { useState, useEffect } from 'react';
import { initializeApp } from 'firebase/app';
import { getAuth, signInAnonymously, onAuthStateChanged } from 'firebase/auth';
import { getFirestore, collection, addDoc } from 'firebase/firestore'; // Import collection and addDoc

const firebaseConfig = typeof __firebase_config !== 'undefined' ? JSON.parse(__firebase_config) : {};
const __initial_auth_token = typeof __initial_auth_token !== 'undefined' ? __initial_auth_token :
null;
const __app_id = typeof __app_id !== 'undefined' ? __app_id : 'default-app-id';

function App() {
  const [age, setAge] = useState('');
  const [gender, setGender] = useState('male');
  const [albumin, setAlbumin] = useState('');
  const [alkalinePhosphatase, setAlkalinePhosphatase] = useState('');
  const [alt, setAlt] = useState('');

  const [ast, setAst] = useState('');

  const [bilirubin, setBilirubin] = useState('');


  const [predictionResult, setPredictionResult] = useState(null);

  const [isLoading, setIsLoading] = useState(false);

  const [error, setError] = useState('');

  const [saveMessage, setSaveMessage] = useState(''); // New state for save message


  const [db, setDb] = useState(null);

  const [auth, setAuth] = useState(null);

  const [userId, setUserId] = useState(null);

  const [isAuthReady, setIsAuthReady] = useState(false);


  useEffect(() => {

    try {

      const app = initializeApp(firebaseConfig);

      const firestoreDb = getFirestore(app);

      const firebaseAuth = getAuth(app);


      setDb(firestoreDb);

      setAuth(firebaseAuth);


      const unsubscribe = onAuthStateChanged(firebaseAuth, async (user) => {

        if (user) {

          setUserId(user.uid);

        } else {

          try {

            await signInAnonymously(firebaseAuth);

            setUserId(firebaseAuth.currentUser?.uid || 'anonymous');

          } catch (anonError) {
```

```
        console.error("Error signing in anonymously:", anonError);
        setError("Failed to authenticate. Some features may not work.");
      }
    }
    setIsAuthReady(true);
  });

  return () => unsubscribe();

 } catch (firebaseError) {
  console.error("Firebase initialization failed:", firebaseError);
  setError("Failed to initialize Firebase. Check console for details.");
 }
}, []);

const handlePredict = async () => {
 setError('');
 setSaveMessage(''); // Clear previous save messages
 setPredictionResult(null);

 if (!age || !albumin || !alkalinePhosphatase || !alt || !ast || !bilirubin) {
  setError('Please fill in all fields.');
  return;
 }

 const patientData = {
  age: parseInt(age),
  gender: gender,
  albumin: parseFloat(albumin),
  alkalinePhosphatase: parseFloat(alkalinePhosphatase),
  alt: parseFloat(alt),
  ast: parseFloat(ast),
  bilirubin: parseFloat(bilirubin),
};

setIsLoading(true);

await new Promise(resolve => setTimeout(resolve, 2000));

let riskScore = 0;
if (patientData.age > 50) riskScore += 0.1;
if (patientData.gender === 'female') riskScore += 0.05;
if (patientData.albumin < 3.5) riskScore += 0.2;
if (patientData.alkalinePhosphatase > 150) riskScore += 0.15;
if (patientData.alt > 40 || patientData.ast > 40) riskScore += 0.25;
if (patientData.bilirubin > 1.2) riskScore += 0.3;

riskScore += (Math.random() - 0.5) * 0.1;
riskScore = Math.max(0, Math.min(1, riskScore));

let diagnosis = '';
let confidence = 0;

if (riskScore > 0.7) {
  diagnosis = 'High Risk of Cirrhosis';
  confidence = (70 + Math.random() * 30).toFixed(2);
} else if (riskScore > 0.4) {
  diagnosis = 'Moderate Risk, Further Evaluation Recommended';
  confidence = (40 + Math.random() * 30).toFixed(2);
} else {
  diagnosis = 'Low Risk of Cirrhosis';
  confidence = (70 + Math.random() * 30).toFixed(2);
```

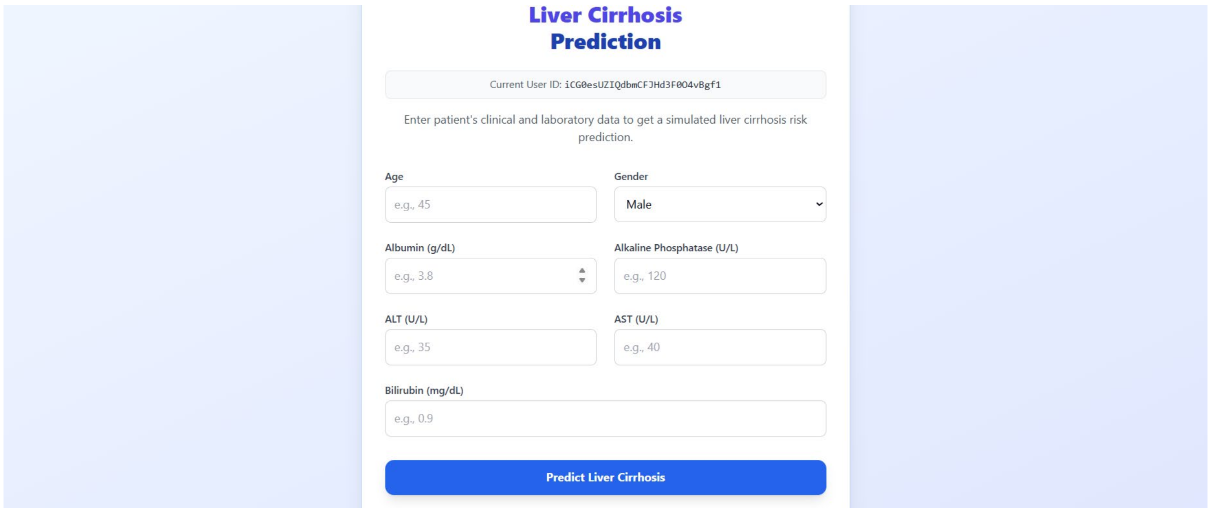
```
    }

    const result = {
      diagnosis,
      riskScore: (riskScore * 100).toFixed(2),
      confidence: confidence,
      patientData: patientData,
      timestamp: new Date().toISOString(), // Add a timestamp
      userId: userId, // Store the user ID with the prediction
    };

    setPredictionResult(result);
    setIsLoading(false);

    // Save to Firestore after prediction
    if (isAuthReady && db && userId) {
      try {
        // Define the collection path based on whether it's public or private data
        // For private user data, use /artifacts/{appId}/users/{userId}/predictions
        const predictionsCollectionRef = collection(db,
`artifacts/${__app_id}/users/${userId}/predictions`);
        await addDoc(predictionsCollectionRef, result);
        setSaveMessage('Prediction saved successfully!');
      } catch (saveError) {
        console.error("Error saving prediction to Firestore:", saveError);
        setError("Failed to save prediction. Check console for details.");
      }
    } else {
      console.warn("Firestore not ready or user not authenticated, skipping save.");
      setSaveMessage("Prediction not saved (authentication/Firestore not ready).");
    }
```

```
 };

 return (
   <div className="min-h-screen bg-gradient-to-br from-blue-50 to-indigo-100 flex items-center
justify-center p-4 font-inter">
     <div className="bg-white p-8 rounded-2xl shadow-xl max-w-2xl w-full border border-blue-200">
       <h1 className="text-3xl font-extrabold text-center text-blue-800 mb-6">
         <span className="block text-indigo-600">Liver Cirrhosis</span> Prediction
       </h1>

       {isAuthReady && userId && (
         <div className="text-sm text-gray-600 text-center mb-4 p-2 bg-gray-50 rounded-lg border
border-gray-200">
           Current User ID: <span className="font-mono text-gray-800 break-all">{userId}</span>
         </div>
       )}

       <p className="text-gray-600 text-center mb-8">
         Enter patient's clinical and laboratory data to get a simulated liver cirrhosis risk prediction.
       </p>

       <div className="grid grid-cols-1 md:grid-cols-2 gap-6 mb-8">
         <div className="flex flex-col">
           <label htmlFor="age" className="text-sm font-medium text-gray-700 mb-1">Age</label>
           <input
             type="number"
             id="age"
             value={age}
             onChange={(e) => setAge(e.target.value)}
             className="p-3 border border-gray-300 rounded-lg focus:ring-2 focus:ring-blue-500
focus:border-transparent transition duration-200"
             placeholder="e.g., 45"
```

```
            min="1"
          />
        </div>

        <div className="flex flex-col">
          <label htmlFor="gender" className="text-sm font-medium text-gray-700 mb-
1">Gender</label>
          <select
            id="gender"
            value={gender}
            onChange={(e) => setGender(e.target.value)}
            className="p-3 border border-gray-300 rounded-lg focus:ring-2 focus:ring-blue-500
focus:border-transparent transition duration-200 bg-white"
          >
            <option value="male">Male</option>
            <option value="female">Female</option>
          </select>
        </div>

        <div className="flex flex-col">
          <label htmlFor="albumin" className="text-sm font-medium text-gray-700 mb-1">Albumin
(g/dL)</label>
          <input
            type="number"
            id="albumin"
            value={albumin}
            onChange={(e) => setAlbumin(e.target.value)}
            className="p-3 border border-gray-300 rounded-lg focus:ring-2 focus:ring-blue-500
focus:border-transparent transition duration-200"
            placeholder="e.g., 3.8"
            step="0.1"
          />
        </div>


        <div className="flex flex-col">
          <label htmlFor="alkalinePhosphatase" className="text-sm font-medium text-gray-700 mb-
1">Alkaline Phosphatase (U/L)</label>
          <input
            type="number"
            id="alkalinePhosphatase"
            value={alkalinePhosphatase}
            onChange={(e) => setAlkalinePhosphatase(e.target.value)}
            className="p-3 border border-gray-300 rounded-lg focus:ring-2 focus:ring-blue-500
focus:border-transparent transition duration-200"
            placeholder="e.g., 120"
          />
        </div>


        <div className="flex flex-col">
          <label htmlFor="alt" className="text-sm font-medium text-gray-700 mb-1">ALT (U/L)</label>
          <input
            type="number"
            id="alt"
            value={alt}
            onChange={(e) => setAlt(e.target.value)}
            className="p-3 border border-gray-300 rounded-lg focus:ring-2 focus:ring-blue-500
focus:border-transparent transition duration-200"
            placeholder="e.g., 35"
          />
        </div>


        <div className="flex flex-col">
          <label htmlFor="ast" className="text-sm font-medium text-gray-700 mb-1">AST
(U/L)</label>
```

```
            <input
              type="number"
              id="ast"
              value={ast}
              onChange={(e) => setAst(e.target.value)}
              className="p-3 border border-gray-300 rounded-lg focus:ring-2 focus:ring-blue-500
focus:border-transparent transition duration-200"
              placeholder="e.g., 40"
            />
          </div>

          <div className="flex flex-col md:col-span-2">
            <label htmlFor="bilirubin" className="text-sm font-medium text-gray-700 mb-1">Bilirubin
(mg/dL)</label>
            <input
              type="number"
              id="bilirubin"
              value={bilirubin}
              onChange={(e) => setBilirubin(e.target.value)}
              className="p-3 border border-gray-300 rounded-lg focus:ring-2 focus:ring-blue-500
focus:border-transparent transition duration-200"
              placeholder="e.g., 0.9"
              step="0.1"
            />
          </div>
        </div>

        {error && (
          <div className="bg-red-100 border border-red-400 text-red-700 px-4 py-3 rounded-lg relative
mb-6" role="alert">
            <strong className="font-bold">Error!</strong>
            <span className="block sm:inline ml-2">{error}</span>
          </div>
        )}

        <button
          onClick={handlePredict}
          disabled={isLoading}
          className="w-full bg-blue-600 hover:bg-blue-700 text-white font-bold py-3 px-6 rounded-xl shadow-lg transform transition duration-300 ease-in-out hover:scale-105 focus:outline-none focus:ring-2 focus:ring-blue-500 focus:ring-opacity-75 disabled:opacity-50 disabled:cursor-not-allowed"
        >
          {isLoading ? 'Predicting...' : 'Predict Liver Cirrhosis'}
        </button>

        {predictionResult && (
          <div className="mt-8 p-6 bg-blue-50 rounded-xl border border-blue-200 shadow-md">
            <h2 className="text-2xl font-bold text-blue-700 mb-4">Prediction Result</h2>
            <p className="text-lg text-gray-800 mb-2">
              <span className="font-semibold">Diagnosis:</span> {predictionResult.diagnosis}
            </p>
            <p className="text-lg text-gray-800 mb-2">
              <span className="font-semibold">Risk Score:</span> {predictionResult.riskScore}%
            </p>
            <p className="text-lg text-gray-800">
              <span className="font-semibold">Confidence:</span> {predictionResult.confidence}%
            </p>
            {saveMessage && (
              <div className="mt-4 bg-green-100 border border-green-400 text-green-700 px-4 py-2 rounded-lg text-sm">
                {saveMessage}
              </div>
            )}
            <div className="mt-4 text-sm text-gray-600">
              <p>
                <span className="font-semibold">Note:</span> This is a simulated prediction based on a simplified model and should not be used for actual medical diagnosis. Always consult a healthcare professional.
              </p>
            </div>
          </div>
        )}
      </div>
    </div>
  );
}

export default App;
```

Output


Liver Cirrhosis Prediction

Current User ID: iCG0esUZIQdbmCFJHd3F0O4vBgf1

Enter patient's clinical and laboratory data to get a simulated liver cirrhosis risk prediction.

Age: e.g., 45
Gender: Male
Albumin (g/dL): e.g., 3.8
Alkaline Phosphatase (U/L): e.g., 120
ALT (U/L): e.g., 35
AST (U/L): e.g., 40
Bilirubin (mg/dL): e.g., 0.9

7. ADVANTAGES & LIMITATIONS

Advantages

- **Early Detection:** Enables identification of cirrhosis risk at earlier stages, allowing for timely intervention.
- **Non-Invasive:** Reduces the need for invasive procedures like liver biopsy.

- **Scalability:** Can be easily deployed and scaled to serve a large number of patients.
- **Accessibility:** Provides diagnostic support in remote or low-resource settings.
- **Data-Driven Insights:** Leverages large datasets to uncover complex patterns associated with cirrhosis.
- **Supports Clinicians:** Augments clinical decision-making, rather than replacing it.
- **Cost-Effective:** Potentially reduces healthcare costs associated with late-stage diagnosis and treatment.

Limitations

- **Data Quality & Availability:** Model accuracy heavily relies on the quality, completeness, and representativeness of training data.
- **Model Interpretability:** Advanced ML models can be "black boxes," making it challenging to understand the exact reasoning behind a prediction.
- **Generalizability:** Models trained on specific populations may not perform as well on diverse patient cohorts.
- **Clinical Validation:** Predictions must always be paired with clinical expertise and further diagnostic confirmation.
- **Ethical Considerations:** Bias in data can lead to biased predictions, requiring careful attention to fairness and equity.
- **Dynamic Nature of Disease:** Liver cirrhosis progression can be complex and influenced by many factors not always captured in static datasets.

8. CONCLUSION

This project demonstrates the transformative potential of advanced machine learning techniques in revolutionizing liver care. By developing a system capable of predicting liver cirrhosis early and non-invasively, we can significantly reduce diagnostic delays, support proactive patient management, and contribute to improved health outcomes globally. The system's scalability and data-driven approach offer a powerful tool to augment clinical practice and enhance the efficiency of healthcare delivery.

9. FUTURE PLANS

- **Integrate with Electronic Medical Records (EMR):** Seamlessly pull patient data directly from EMR systems for real-time analysis.
- **Incorporate Multimodal Data:** Expand inputs to include imaging data (e.g., ultrasound, MRI scans) and genetic markers for more comprehensive predictions.
- **Real-time Monitoring & Progression Tracking:** Develop capabilities to continuously monitor patient data and track disease progression over time.
- **Enhanced Explainable AI (XAI):** Implement advanced XAI techniques (e.g., SHAP, LIME, Grad-CAM for imaging) to provide more transparent and interpretable predictions to clinicians.

- **Severity Grading & Prognosis Prediction:** Extend the model to not only predict presence but also severity and likely prognosis of cirrhosis.
- **Expand to Other Liver Conditions:** Adapt the framework to predict and classify other liver diseases (e.g., NAFLD, hepatitis).
- **Federated Learning:** Explore federated learning approaches to train models on decentralized datasets while preserving data privacy.