Misan Journal of Engineering Sciences ISSN: 2957-4250
Vol. 5, No. 1, 28 June 2026 ISSN-E: 2957-4242

Three-Test Liver Disease Screening Using Machine Learning: A Cost-Effective Approach Through Feature Prioritization

Ahmed Majid AbdulAbbas[1*], Al_hussein M. Alturfi[1], Mays Kareem Jabbar[1], Hasnaa H Abdulkareem[2], Anwer Jabbar Hasan Al-Saedi[1], Yasir Ali Khalid Al-Nuaimi[1]

[1]Department of Electrical Engineering, College of Engineering, University of Misan , Maysan, 62001, Iraq.
[2]College of Computer Engineering, University of Technology
Corresponding author E-mail: ahmedmajed@uomisan.edu.iq

(Received 24 Nov 2025, Revised 24 May 2026, Accepted 24 May 2026)

Abstract: Standard liver disease screening requires testing eight or more laboratory parameters, which increases costs and limits accessibility in resource-limited settings. This study aimed to identify a minimal test combination that maintains diagnostic accuracy while reducing costs. Seven machine learning algorithms—Support Vector Machine, Boosting, Multilayer Perceptron, Bagging, Random Forest, K-Nearest Neighbors, and J48 Decision Tree—were evaluated using the Indian Liver Patient Dataset (583 patients, 10 laboratory parameters). Features were ranked based on their diagnostic value to identify the most informative tests. Algorithm performance was assessed across six scenarios with progressive feature removal, measuring accuracy, Kappa statistic, ROC area, and F-measure. The Kappa statistic revealed classification issues that accuracy metrics alone masked. Feature ranking identified Total Bilirubin, Direct Bilirubin, and SGPT as the three most informative parameters. Random Forest achieved the best performance using only these three tests (accuracy 72.04%, ROC 0.70, Kappa 0.24), matching the accuracy obtained with all eight parameters. This approach reduces the required tests by 62.5% and screening costs by 60%. Support Vector Machine reached 71% accuracy, but its zero Kappa value indicated failure to learn real diagnostic patterns, instead exploiting the dataset's 71.4% disease rate. Bagging achieved the highest ROC (0.73) but showed lower overall balance than Random Forest. The three-parameter Random Forest model enables accurate and cost-effective liver disease screening, correctly identifying 87% of diseased patients. This approach reduces testing from eight parameters to three without compromising diagnostic accuracy. The feature ranking method can be applied to other diseases where comprehensive testing increases healthcare costs.

Keywords: Liver disease prediction; Machine learning; Feature prioritization; Cost-effective screening; Diagnostic accuracy

1. Introduction

The liver stands out as our body's biggest solid organ. Hepatic function covers metabolic control and toxin removal, among hundreds of other tasks. Global liver disease numbers keep climbing. Modern diets, less physical activity, and metabolic problems drive the increase [1]. Deaths from liver conditions hit around 2 million per year worldwide [2]. Younger men get NAFLD and similar conditions roughly twice as often as younger women. Older age groups show smaller gaps [3]. Female patients face extra hurdles. Doctors miss liver disease more often in women. Getting to specialists takes longer. Transplant lists favor men [4]. Standard diagnosis relies on blood tests measuring liver enzymes and function features. Biopsies sometimes become necessary despite being invasive. Catching early disease stays difficult with current methods. Test accuracy varies [5]. Machine learning finds connections in standard clinical tests and lab work that older analysis methods overlook [6]. Support vector machines and ensemble approaches beat basic logistic regression in several studies [7].

DOI: https://doi.org/10.61263/mjes.v5i1.238

Datasets with gaps create problems. Missing values skew results. Accuracy drops when patient records stay incomplete [8,9]. Some newer research combines multiple techniques. Ensemble models plus deep learning plus careful feature picking plus parameter adjustment can push accuracy past 85% [10,11]. Despite these advances, recent research found that ML algorithms for liver disease prediction perform differently across sexes. Models with high overall accuracy showed much higher false negative rates for female patients [12]. Such bias likely reflects existing inequalities in healthcare systems. Sex-stratified evaluation of ML models is essential [13-16].

This study had two objectives. First, identify and rank laboratory parameters by diagnostic value through feature evaluation. Second, compare seven machine learning algorithms (SVM, Boosting, MLP, Bagging, Random Forest, K-NN, and Decision Tree J48) to determine which methods provide the best diagnostic performance across accuracy, sensitivity, specificity, and area under the ROC curve.

The remainder of this paper is organized as follows: Section 2 reviews related work on machine learning applications in liver disease prediction. Section 3 presents the machine learning algorithms employed in this study. Section 4 describes the research methodology, including dataset description, feature prioritization approach, data preprocessing, and experimental design. Section 5 reports the results and comparative performance analysis across all algorithms and scenarios. Section 6 discusses the findings, compares results with previous studies, and addresses limitations. Finally, Section 7 concludes the paper with key contributions and practical implications for cost-effective liver disease screening.

2. Related works

Machine learning plays an increasingly important role in liver disease diagnosis. Researchers tested various algorithms to detect disease earlier and improve diagnostic accuracy. We review previous studies across four main areas: traditional machine learning methods, performance results using the ILPD dataset, feature selection and optimization techniques, and algorithmic fairness issues.

2.1 Traditional Machine Learning Approaches for Liver Disease Classification

Early studies tested whether machine learning could support clinical decisions about liver disease. Singh et al. [11] built classification models using logistic regression, random forest, and naive Bayes to help diagnose different liver conditions based on liver function tests. Vijayarani and Dhayanand [12] compared support vector machines (SVM) with Bayesian methods for separating cirrhosis, acute hepatitis, and chronic hepatitis. SVM outperformed Bayesian approaches. Joloudari et al. [13] combined SVM with Particle Swarm Optimization (PSO) for feature selection and identified the most useful biofeatures for diagnosing cirrhosis. Their method exceeded Random Forest, Bayesian Networks, and multilayer perceptron networks in performance. Jaganathan et al. [14] demonstrated that SVM models predicted drug-induced liver toxicity with high accuracy using fewer molecular features than other methods. Fewer features can make models more efficient.

2.2 Performance Evaluation Using the Indian Liver Patient Dataset

The Indian Liver Patient Dataset (ILPD) serves as a standard benchmark for testing machine learning algorithms in liver disease prediction [15-17]. Accuracies above 70% appear frequently in the literature. Priya et al. [15] and Ramana and Boddu [16] tested various classifiers on ILPD. Ensemble methods and support vector machines performed well across different experiments. Logistic regression yielded 72.7% accuracy in Jin et al.'s work [18]. Combining logistic regression with neural network components and support vector machines, Adil et al. [19] reached 74% accuracy. ILPD studies establish comparison baselines. A limitation exists. Overall accuracy dominates reporting. Feature importance gets little attention. Model performance across patient subgroups remains underexplored.

2.3 Feature Selection and Model Optimization

Feature selection offers dual benefits in liver disease prediction. Model accuracy improves. Computational costs decrease. Gulia et al. [24] implemented a three-phase approach. Classification methods were tested on the complete ILPD dataset initially. Automated methods then selected the most useful features. Algorithm performance was compared with and without feature selection in the final

phase. Using all features, SVM accuracy peaked at approximately 72%. Random forest excelled after feature selection, obtaining 71.87% accuracy on test data. Each algorithm responds differently when features are removed. Machine learning classifiers using biochemical features often outperform traditional statistical models [25].

However, few studies investigate which specific laboratory tests contribute most to prediction accuracy. Studies rarely rank features by diagnostic value. Testing how removing less important features affects different algorithms remains uncommon.

2.4 Algorithmic Fairness and Demographic Considerations

Most liver disease prediction studies only examine overall performance metrics. Recent research now explores whether algorithms behave fairly across demographic groups. Clinical studies document sex-based differences in liver disease rates, biomarker levels, and treatment outcomes [21-23]. Few published studies check whether machine learning models achieve equal performance across patient subgroups, though. Straw and Wu [20] analyzed ILPD-based models separately for male and female patients. Algorithms with high overall accuracy missed significantly more diseased cases in female patients than in male patients. These findings demonstrate why demographic-stratified evaluation matters beyond aggregate accuracy. Healthcare AI research reveals that algorithms can develop demographic biases [26-28]. Such biases may worsen existing health inequalities. Evaluation methods must therefore measure both prediction accuracy and fairness across patient populations.

2.5 Research Gap and Study Objectives

Previous research on machine learning for liver disease prediction has three main limitations. First, most studies evaluate only 2-3 algorithms [11-19, 24], which provides limited evidence about which methods perform best under different evaluation criteria. Second, few studies have ranked laboratory tests to determine the minimum set required for accurate diagnosis. Gulia et al. [24] demonstrated that feature selection impacts each algorithm differently, but they did not rank features by importance across all methods. This makes it harder to develop cost-effective screening approaches. Third, while Straw and Wu [20] found sex-based performance differences in ILPD models, no study has combined comprehensive algorithm comparison with feature ranking and analysis across different patient groups.

Our study addresses these limitations in three ways. First, we evaluate seven machine learning algorithms (SVM, Boosting, MLP, Bagging, Random Forest, K-NN, and Decision Tree J48) using accuracy, sensitivity, specificity, ROC area, and F-measure. Second, tests get importance scores. Algorithms run with gradually fewer tests. Low-scoring tests drop out one by one. Third, we identify which methods maintain diagnostic accuracy while reducing the cost of liver disease screening.

3. Machine Learning Algorithms

Seven supervised learning algorithms were tested. Each algorithm represents a different classification approach. The selection includes linear classifiers, ensemble methods, instance-based learners, neural networks, and tree-based approaches. Medical diagnosis studies routinely employ these algorithm families. Coverage spans the primary machine learning techniques for clinical applications.

3.1 Support Vector Machine (SVM)

Vapnik and Chervonenkis developed Support Vector Machines in the 1960s, and Cortes and Vapnik introduced the soft-margin version in 1995 [30]. SVM classifies data by finding the best separating hyperplane in high-dimensional space. The algorithm looks for the hyperplane that creates the largest margin between classes. The margin is the distance between the hyperplane and the nearest training samples (support vectors) from each class. When data cannot be separated linearly, SVM uses kernel functions. These functions transform features into higher-dimensional spaces where separation becomes easier. SVM avoids direct computation in high-dimensional spaces. The kernel trick makes this possible. Nonlinear decision boundaries emerge without explicit high-dimensional calculations. We implemented SVM with a radial basis function (RBF) kernel. RBF relies on Gaussian similarity measures for data transformation. Complex nonlinear boundaries result from this approach.

3.2 Multilayer Perceptron (MLP)

The Multilayer Perceptron [31] is an artificial neural network with multiple interconnected node layers. Information flows through weighted connections. The architecture has three parts: an input layer receiving feature values, one or more hidden layers applying nonlinear transformations via activation functions, and an output layer producing class predictions. Backpropagation trains the network. Connection weights adjust through error calculation at the output, then propagating these errors backward. Gradient descent updates weights, changing connection strengths to minimize training prediction errors.
Our MLP contains one hidden layer with (number of attributes + number of classes)/2 nodes. Backpropagation training incorporates a momentum term of 0.2, accelerating convergence and avoiding local minima. Learning rate equals 0.3 across 500 epochs.

3.3 AdaBoost (Adaptive Boosting)

Freund and Schapire introduced AdaBoost in 1996 [32]. The algorithm combines multiple weak classifiers to build a strong classifier. It trains base learners one after another on weighted versions of the training data. When the algorithm misclassifies samples, those samples get higher weights in the next iteration. Each iteration creates a new weak learner that focuses on the samples that previous learners classified incorrectly. Difficult samples receive greater attention from later classifiers. Overall accuracy improves as a result. Final predictions rely on weighted majority voting. Vote weight assignment depends on training data accuracy. Classifiers with higher training accuracy exert stronger influence on final decisions. Decision stumps serve as base learners. These are single-level decision trees with one split. The ensemble contains 10 weak classifiers.

3.4 Bootstrap Aggregating (Bagging)

Breiman introduced Bootstrap Aggregating in 1996 [33]. Bagging builds ensemble classifiers by training multiple base learners on varied versions of the training data. Each bootstrap sample comes from random sampling with replacement. Training subsets match the original dataset size but contain distinct samples.

Sampling with replacement produces bootstrap sets including approximately 63.2% unique instances. Some examples appear multiple times, others not at all. Base learners trained on these varied subsets capture distinct patterns. Individual models exhibit high variance but low inter-model correlation. Final predictions emerge through majority voting across all learners. Bagging proves effective for unstable learning algorithms—those where minor data changes produce major model shifts. We employ decision trees as base learners. Ten bootstrap samples generate ten independent trees.

3.5 Random Forest

Random Forest [34] extends bagging by adding more randomness when building trees. Like bagging, it creates bootstrap samples. However, it also randomly picks a subset of features at each node split and only considers those features when deciding how to split.

Random forests employ two randomization strategies. Sample selection varies for each tree. Feature selection varies at each node. These combined strategies produce diverse decision trees. Each tree learns distinct patterns from the data. Predictions aggregate all tree outputs through majority voting. Overfitting rarely occurs, even with hundreds of trees. Individual trees cannot simply memorize training data due to randomness. Our implementation uses 100 decision trees. Feature selection at each split considers $\sqrt{p}$ candidates, where p represents total features. Ten features means approximately three candidates per split. The square root of 10 equals roughly 3.

3.6 K-Nearest Neighbors (K-NN)

K-Nearest Neighbors [35] uses instance-based learning. Classification depends on proximity to training examples in feature space. Test samples get assigned to classes through a simple process. The algorithm identifies k nearest training examples. Majority voting among these neighbors determines class membership. Euclidean distance measures straight-line separation between feature vectors in normalized space. K-NN differs from other methods. No decision rules get built during training. All training data

remains stored. Calculations occur only at prediction time. Performance varies with k value selection. Low k values detect fine-grained patterns. Noise interference increases with low k. High k values smooth decision boundaries. Fine details may disappear with high k. Our tests used k=5 neighbors. Five neighbors provide adequate pattern sensitivity without excessive noise susceptibility.

3.7 Decision Tree (J48)

J48 implements the C4.5 decision tree algorithm [36]. Tree construction proceeds recursively. Feature space gets divided repeatedly. Growth occurs from top to bottom. Each node requires a decision. Which feature to split on? Which split point to use? Information gain determines both choices. The algorithm selects the feature-split combination yielding maximum information gain. Information gain quantifies entropy reduction. Entropy reflects class mixture within nodes. Pure nodes contain one class only. Zero entropy results. Completely mixed nodes contain all classes equally. Maximum entropy results. Splitting continues until stopping conditions activate. Three conditions exist. Sample count drops below minimum threshold. Maximum tree depth gets reached. Additional splits provide negligible improvement. Pruning follows tree construction. Overly specific branches get removed. Overfitting risk decreases. Diagnostic accuracy on new data remains intact. J48 generates human-readable classification rules. Tree paths translate directly into rules. Clinical settings benefit from rule transparency. Diagnostic reasoning becomes explicit. Our implementation applies specific parameters. Confidence factor: 0.25 for pruning decisions. Minimum samples per leaf: 2.

4. Research Methodology

4.1 Dataset Description

Figure 1 presents the overall framework of the proposed approach. The methodology proceeds through six stages: (1) ILPD dataset acquisition containing 583 patient records with 10 laboratory parameters, (2) data preprocessing including missing value imputation and normalization, (3) systematic feature prioritization based on diagnostic effectiveness, (4) progressive feature removal creating six experimental scenarios, (5) algorithm evaluation using seven machine learning classifiers with 10-fold cross-validation, and (6) comprehensive performance assessment using multiple evaluation metrics.

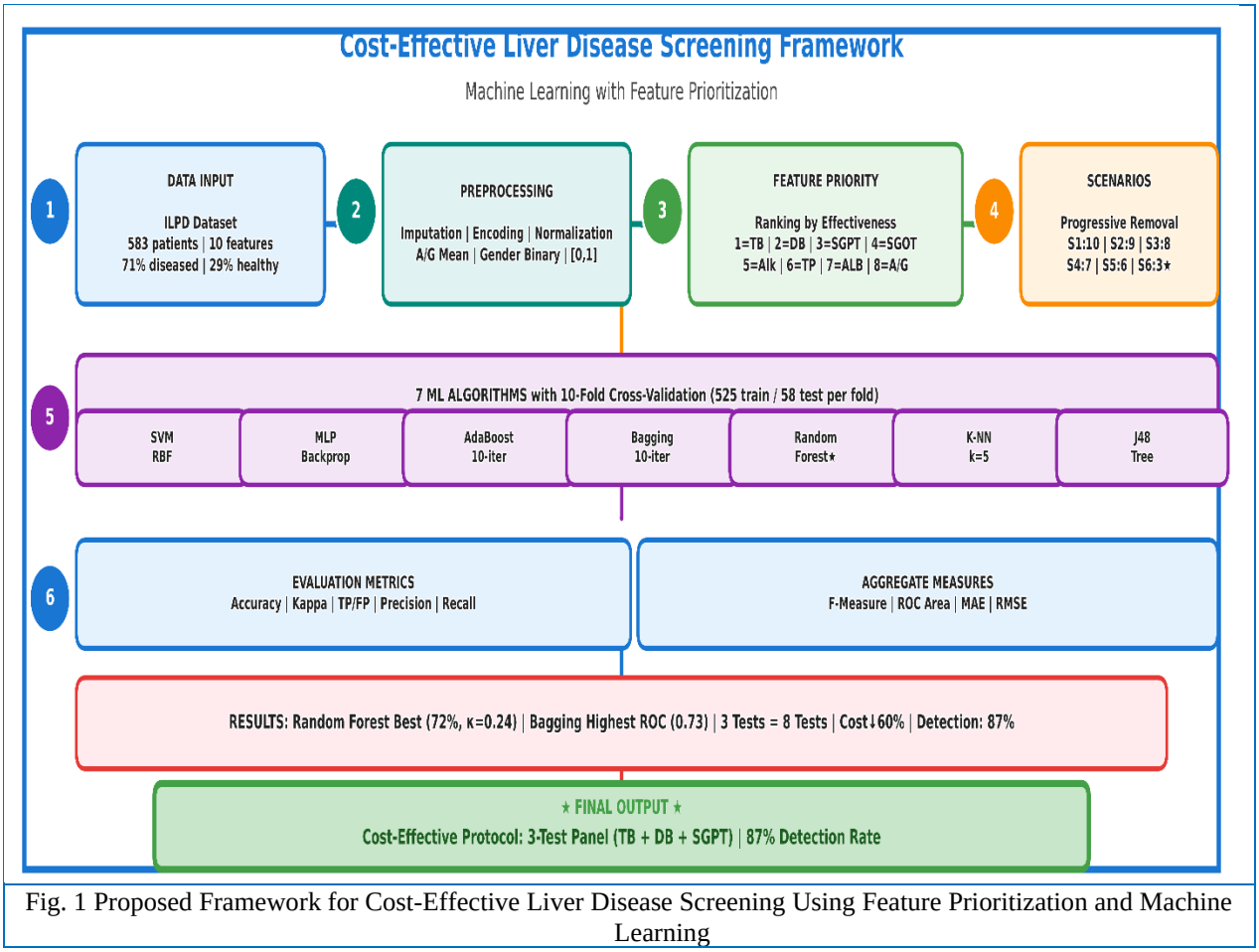

Fig. 1 Proposed Framework for Cost-Effective Liver Disease Screening Using Feature Prioritization and Machine Learning

We use the Indian Liver Patient Dataset (ILPD), a standard benchmark from the UCI Machine Learning Repository [29]. ILPD contains laboratory and demographic data from liver patients in northeast India. It includes 583 records in spreadsheet format. Each record has 10 predictor variables and one binary class label (diseased = 1, healthy = 0).

Class distribution shows imbalance: 416 individuals have liver disease (71.4%) and 167 are healthy (28.6%). Males comprise 441 cases (75.6%), females 142 cases (24.4%). Age distribution centers on 40 years. Below this age: 241 individuals (41.3%). At or above: 342 individuals (58.7%). The dataset codes individuals over 89 as age 90. Predictor variables number ten total. Two are demographic: age and gender. Eight are biochemical features from standard liver function panels. Total Bilirubin (TB) and Direct Bilirubin (DB) measure bilirubin metabolism. Alkaline Phosphatase (Alkphos) reflects liver enzyme activity. Alamine Aminotransferase (SGPT) and Aspartate Aminotransferase (SGOT) indicate hepatocellular injury. Total Protein (TP), Albumin (ALB), and Albumin/Globulin Ratio (A/G) assess liver synthetic function. These tests appear in routine clinical practice for liver disease evaluation.

4.2 Feature Prioritization Methodology

Test effectiveness works like this: abnormal results get counted. Then checking how many abnormal results actually indicate disease. The ratio between these numbers gives the effectiveness score as:

$$\text{Diagnostic Effectiveness} = (\text{N\_abnormal} + \text{diseased} / \text{N\_abnormal}) \times 100\% \quad (1)$$

The metric measures something simple. When test results go abnormal, how often does that mean actual disease? That percentage shows test accuracy. Standard lab guidelines provided normal ranges. Most

tests apply the same cutoffs regardless of sex. Six tests work this way: Total Bilirubin, Direct Bilirubin, Alkaline Phosphatase, Total Protein, Albumin, and A/G Ratio. Two enzymes differ. SGPT and SGOT use different normal ranges for men versus women. Effectiveness got calculated separately by sex. Results then got merged using weighted averages matching sample sizes. Ranking went from best to worst. Rank 1 means most effective. Rank 8 means least effective. Better ranks catch disease more reliably. Top three tests emerged clearly. Total Bilirubin took first place. Direct Bilirubin came second. Alanine Aminotransferase grabbed third. All three beat the other five features.

4.3 Data Preprocessing

4.3.1 Missing Value Imputation

A/G Ratio data had four gaps. Gaps got filled with averages. Existing A/G values averaged out to 0.95. Missing spots received 0.95. Mean filling keeps data patterns intact. Nothing gets thrown out. Full dataset stays analyzable.

4.3.2 Feature Encoding and Transformation

Gender became a binary variable (Male = 1, Female = 0) for numerical processing. The class label remained binary (Diseased = 1, Healthy = 0), matching the original structure.

4.3.3 Normalization

We applied min-max normalization to all numeric features. Normalization ensures consistent value ranges and prevents features with large values from dominating model training. The process scales each feature to the [0, 1] interval:

$$x\_normalized = (x - x\_min) / (x\_max - x\_min) \quad (2)$$

where x_min and x_max are the minimum and maximum values for each feature. The transformation preserves relationships between data points while standardizing scale across all features. Standardized features enable faster algorithm convergence and permit fair comparison of feature contributions during training.

4.4 Experimental Design

4.4.1 Progressive Feature Removal Protocol

Algorithm performance was tested by progressively removing lower-priority variables to determine whether their removal affects classification accuracy. Identifying the smallest effective test group was essential. Diagnostic accuracy needed to stay high even as test numbers decreased.

Six scenarios tested different variable combinations. Priority rankings guided which tests stayed versus got removed in each scenario. Lower-ranked variables dropped out first.

1. **Scenario 1 - All parameters**: Complete feature set including all 10 predictor variables (Age, Gender, TB, DB, Alkphos, SGPT, SGOT, TP, ALB, A/G Ratio)
2. **Scenario 2 - Remove priority 8**: Removal of the single lowest-priority feature
3. **Scenario 3 - Remove priorities 8 & 7**: Removal of the two lowest-priority features
4. **Scenario 4 - Remove priorities 8, 7 & 6**: Removal of the three lowest-priority features
5. **Scenario 5 - Remove priorities 8, 7, 6 & 5**: Removal of the four lowest-priority features
6. **Scenario 6 - Remove priorities 8, 7, 6, 5 & 4**: Retention of only the top three priority features (Total Bilirubin, Direct Bilirubin, and SGPT)

Training and testing happened across all six scenarios. Data splits stayed identical throughout. The setup enabled comparisons on two levels: comparing algorithms within each scenario, and comparing scenarios within each algorithm. Features got removed gradually, starting from the full set down to just three tests. The process revealed when adding extra features stopped helping performance. Gradual removal showed the optimal feature count. Laboratory test combinations differ in diagnostic value. Some combinations deliver accurate diagnosis at lower cost. Finding those combinations was the target.

4.4.2 Performance Evaluation Framework

Standard classification metrics measured algorithm performance. Confusion matrices provided the foundation for evaluation. Every algorithm-scenario combination produced a confusion matrix. Four numbers appear in each matrix: true positives, true negatives, false positives, and false negatives.
The following metrics got calculated from those four values:

1- Primary Classification Metrics:

- **Correctly Classified Instances (CCI)**: Percentage of samples receiving correct class predictions, calculated as:

$$CCI = [(TP + TN) / Total\ Samples] \times 100 \quad (3)$$

- **Kappa Statistic**: Cohen's kappa coefficient measuring agreement between predictions and actual labels adjusted for chance agreement, ranging from -1 (perfect disagreement) to +1 (perfect agreement), with 0 indicating chance-level performance
- **True Positive Rate (TP Rate)**: Proportion of diseased patients correctly identified (sensitivity/recall), calculated as:

$$TP\ Rate = TP / (TP + FN) \quad (4)$$

- **False Positive Rate (FP Rate)**: Proportion of healthy individuals incorrectly classified as diseased, calculated as:

$$FP\ Rate = FP / (FP + TN) \quad (5)$$

2- Aggregate Performance Measures:

- **Precision**: Proportion of positive predictions that are correct, indicating diagnostic specificity when disease is predicted:

$$Precision = TP / (TP + FP) \quad (6)$$

- **Recall**: Proportion of actual positives correctly identified (equivalent to TP Rate/sensitivity):

$$Recall = TP / (TP + FN) \quad (7)$$

- **F-Measure**: Harmonic mean balancing precision and recall, providing single-value summary when both metrics are important:

$$F - Measure = (2 \times Precision \times Recall) / (Precision + Recall) \quad (8)$$

- **ROC Area**: Area under the Receiver Operating Characteristic curve, measuring classification performance across all possible decision thresholds, with 0.5 indicating random guessing and 1.0 indicating perfect classification

3- Error Metrics:

- **Mean Absolute Error (MAE)**: Average absolute difference between predicted probabilities and actual labels:

$$MAE = [\Sigma|predicted\_i - actual\_i|] / n \quad (9)$$

where the summation is from i=1 to n

- **Root Mean Square Error (RMSE)**: Root of average squared errors, penalizing large prediction errors more heavily than MAE:

$$RMSE = \sqrt{\{[\Sigma(predicted\_i - actual\_i)^2] / n\}} \quad (10)$$

where the summation is from i=1 to n

4.4.3 Model Training and Validation

The study implemented all algorithms using Weka 3.8 machine learning workbench [34]. Weka provides standard implementations of machine learning algorithms with consistent interfaces. Model

training applied 10-fold cross-validation to obtain reliable performance estimates from the limited dataset. In 10-fold cross-validation, the dataset splits into 10 equal parts (approximately 58 samples per fold). Training uses 9 folds per round. Sample count: approximately 525. One fold remains for testing.

Rotation occurs systematically. All fold combinations get evaluated. Each sample becomes a test instance exactly once. Performance metrics come from averaging. All 10 folds contribute. Single split dependence decreases through averaging. Data variability gets accounted for. Hyperparameters varied by algorithm. Specific configurations appear below.

- **SVM**: RBF kernel with complexity parameter C=1.0 and gamma=1/number_of_features
- **MLP**: Single hidden layer with (attributes + classes)/2 nodes, learning rate=0.3, momentum=0.2, 500 training epochs
- **AdaBoost**: 10 iterations with DecisionStump base classifiers
- **Bagging**: 10 iterations with REPTree base classifiers
- **Random Forest**: 100 trees with random feature selection at each split
- **K-NN**: k=5 neighbors using Euclidean distance
- **J48**: Confidence factor=0.25 for pruning, minimum 2 instances per leaf

These parameter settings represent commonly-used defaults or values determined through preliminary experimentation, balancing model complexity against overfitting risk.

5. Results and Analysis

5.1 Feature Prioritization Results

Laboratory parameters received priority rankings. Diagnostic effectiveness percentages determined rank order. What does diagnostic effectiveness mean? It quantifies disease identification accuracy when test values fall outside normal ranges. Section 3.2.2 provides details. Eight parameters underwent effectiveness calculation. SGPT and SGOT needed sex-specific adjustments. Two enzymes behave differently by sex. Three features topped the rankings. Total Bilirubin (TB) came first. Direct Bilirubin (DB) followed. Alanine Aminotransferase (SGPT) placed third. All three surpassed the remaining five features in diagnostic value. Priority rankings guided feature removal. The progressive removal protocol relied on this hierarchy. Algorithm performance was tested as low-priority parameters were removed. Six experimental scenarios were created based on this ranking (Section 3.5.1). Scenario 6 used only the top three parameters.

5.2 Algorithm Performance Across Scenarios

We evaluated all seven algorithms across six scenarios using 10-fold cross-validation. Tables 1 through 7 show complete results for each algorithm, documenting how accuracy changed as variables were gradually removed from the full set to just three.

5.2.1 Support Vector Machine (SVM)

Table 1 presents SVM results. Accuracy remained stable at 71.35% through scenarios 1-5. In Scenario 6 (using only TB, DB, and SGPT), accuracy rose slightly to 71.36%. F-Measure held at 0.83 in all tests. ROC Area values ranged between 0.50 and 0.51, indicating weak class discrimination at different thresholds. True Positive Rate reached 0.99 consistently, but False Positive Rate also climbed high (0.90-0.99). SVM classified most cases as positive. Kappa statistics hovered at or near zero throughout all tests, reflecting performance barely above random guessing.

Table 1. SVM Algorithm Performance Across Scenarios

Scenario	Kappa	CCI	ROC Area	F-Measure	Recall	Precision	FP	TP
All parameters	0	71.35	0.50	0.83	0.99	0.71	0.99	0.99
Remove priority 8	0	71.35	0.50	0.83	0.99	0.71	0.99	0.99
Remove priorities 8 & 7	0	71.35	0.50	0.83	0.99	0.71	0.99	0.99
Remove priorities 8 & 6	0	71.35	0.50	0.83	0.99	0.71	0.99	0.99
Remove priorities 8 & 5	0	71.35	0.50	0.83	0.99	0.71	0.99	0.99

Remove priorities 8 & 4	0	71.36	0.51	0.83	0.99	0.71	0.90	0.99

5.2.2 Multilayer Perceptron (MLP)

Table 2 presents MLP results. MLP achieved 70.15% accuracy using all parameters. Performance remained consistent (69.29%-70.32%) when features were removed. ROC Area ranged from 0.71 to 0.72, exceeding SVM. F-Measure ranged from 0.81 to 0.82. Scenario 6 (the top three features) produced optimal results: 70.32% accuracy, 0.82 F-Measure, and 0.72 ROC Area. True Positive Rate rose from 0.88 to 0.95 as features were removed. False Positive Rate also increased from 0.74 to 0.91. Kappa values ranged from 0.03 to 0.15. The low Kappa scores indicate only marginal improvement over random guessing.

Table 2. MLP Algorithm Performance across Scenarios

Scenario	Kappa	CCI	ROC Area	F-Measure	Recall	Precision	FP	TP
All parameters	0.15	70.15	0.72	0.81	0.88	0.74	0.74	0.88
Remove priority 8	0.10	69.46	0.71	0.81	0.89	0.73	0.81	0.89
Remove priorities 8 & 7	0.04	69.29	0.71	0.81	0.93	0.72	0.89	0.93
Remove priorities 8 & 6	0.04	69.63	0.71	0.81	0.93	0.72	0.90	0.93
Remove priorities 8 & 5	0.03	70.15	0.72	0.82	0.95	0.70	0.92	0.95
Remove priorities 8 & 4	0.04	70.32	0.72	0.82	0.95	0.72	0.91	0.95

5.2.3 AdaBoost

Table 3 presents AdaBoost performance metrics. Accuracy achieved 70.32% with all parameters and improved slightly to 70.36% using only three variables. F-Measure held at 0.82-0.83 across all tests. ROC Area varied from 0.66 to 0.67, falling between SVM and MLP results. Similar to SVM, the algorithm exhibited elevated True Positive Rates (0.96-0.99) and elevated False Positive Rates (0.95-0.99). Most cases received positive classification. Kappa statistics fluctuated from 0 to 0.01, indicating almost no improvement over random classification.

Table 3. AdaBoost Algorithm Performance across Scenarios

Scenario	Kappa	CCI	ROC Area	F-Measure	Recall	Precision	FP	TP
All parameters	0.01	70.32	0.67	0.82	0.96	0.70	0.95	0.96
Remove priority 8	0.00	70.35	0.66	0.83	0.99	0.70	0.99	0.99
Remove priorities 8 & 7	0.00	70.35	0.66	0.83	0.99	0.70	0.99	0.99
Remove priorities 8 & 6	0.00	70.35	0.66	0.83	0.99	0.70	0.99	0.99
Remove priorities 8 & 5	0.00	70.35	0.66	0.83	0.99	0.70	0.99	0.99
Remove priorities 8 & 4	0.00	70.36	0.66	0.83	0.99	0.70	0.99	0.99

5.2.4 Bagging

Table 4 shows Bagging results. Bagging achieved 70.81% accuracy with all parameters, representing the highest accuracy among scenarios 1-5. Feature removal affected accuracy. Performance declined initially. Scenario 5 accuracy: 69.46%. Three features reversed the trend. Accuracy rebounded to 70.49%.

ROC Area varied between 0.70 and 0.73. Scenario 6 produced the peak value. F-Measure stayed constant. All scenarios yielded 0.80. True Positive Rates spanned 0.85 to 0.86. False Positive Rates spanned 0.67 to 0.72. Both metrics showed better balance compared to SVM and AdaBoost.

Kappa ranged from 0.15 to 0.20. Values remained consistently low. Small improvements over random guessing occurred. Consistency persisted across scenarios.

Table 4. Bagging Algorithm Performance across Scenarios

Scenario	Kappa	CCI	ROC Area	F-Measure	Recall	Precision	FP	TP

All parameters	0.20	70.81	0.70	0.80	0.86	0.76	0.67	0.86
Remove priority 8	0.19	70.31	0.70	0.80	0.85	0.75	0.67	0.85
Remove priorities 8 & 7	0.18	69.91	0.72	0.80	0.85	0.75	0.68	0.85
Remove priorities 8 & 6	0.16	69.63	0.72	0.80	0.85	0.75	0.70	0.85
Remove priorities 8 & 5	0.15	69.46	0.70	0.80	0.86	0.74	0.72	0.86
Remove priorities 8 & 4	0.19	70.49	0.73	0.80	0.86	0.75	0.68	0.86

5.2.5 Random Forest

Table 5 shows Random Forest performance. All parameters together: 69.80% accuracy initially. Performance climbed. Scenario 4 removed three low-priority variables. Accuracy jumped to 72.20%. Scenario 6 used only three variables. Accuracy reached 72.04%. This surpassed all other algorithms on the minimal variable set. ROC Area fluctuated between 0.70 and 0.72. F-Measure stayed at 0.80. All tests yielded identical F-Measure values. Kappa spanned 0.15 to 0.24. Scenario 6 Kappa: 0.24. Highest value across all algorithms. Agreement beyond chance performed strongest here. True Positive Rates: 0.85 to 0.88. False Positive Rates: 0.66 to 0.73. Classification balance was evident.

Table 5. Random Forest Algorithm Performance across Scenarios

Scenario	Kappa	CCI	ROC Area	F-Measure	Recall	Precision	FP	TP
All parameters	0.15	69.80	0.70	0.80	0.87	0.74	0.73	0.87
Remove priority 8	0.17	70.32	0.70	0.80	0.87	0.75	0.70	0.87
Remove priorities 8 & 7	0.17	70.66	0.70	0.80	0.88	0.75	0.73	0.88
Remove priorities 8 & 6	0.23	72.20	0.72	0.80	0.87	0.76	0.66	0.87
Remove priorities 8 & 5	0.19	70.05	0.70	0.80	0.85	0.76	0.67	0.85
Remove priorities 8 & 4	0.24	72.04	0.70	0.80	0.87	0.76	0.66	0.87

5.2.6 K-Nearest Neighbors (K-NN)

Table 6 shows K-NN results. K-NN had the lowest starting accuracy among all methods, obtaining only 64.83% with all parameters. Accuracy improved gradually as features were removed. Performance peaked at 69.02% in Scenario 5 (removing four lowest-priority features), then dropped to 67.40% with only three features. ROC Area ranged from 0.57 to 0.63, the lowest values among all algorithms. F-Measure increased from 0.74 to 0.78 as features were reduced. True Positive Rates (0.70-0.77) and False Positive Rates (0.47-0.56) were the most conservative among all algorithms. Kappa values ranged from 0.20 to 0.25 across all scenarios.

Table 6. K-NN Algorithm Performance across Scenarios

Scenario	Kappa	CCI	ROC Area	F-Measure	Recall	Precision	FP	TP
All parameters	0.20	64.83	0.58	0.74	0.70	0.78	0.47	0.70
Remove priority 8	0.20	65.52	0.57	0.74	0.70	0.78	0.50	0.70
Remove priorities 8 & 7	0.20	66.38	0.59	0.75	0.74	0.77	0.52	0.74
Remove priorities 8 & 6	0.22	67.06	0.60	0.76	0.74	0.78	0.50	0.74
Remove priorities 8 & 5	0.25	69.02	0.63	0.78	0.77	0.79	0.50	0.77
Remove priorities 8 & 4	0.20	67.40	0.60	0.77	0.76	0.77	0.56	0.76

5.2.7 Decision Tree (J48)

Table 7 presents J48 performance metrics. Initial accuracy hit 68.95% with all parameters. Results improved to 69.29% after removing one variable, then fluctuated before settling at 69.02% in Scenarios 5 and 6. ROC Area spanned from 0.64 to 0.69. The three-variable configuration produced the highest ROC Area. F-Measure varied from 0.78 to 0.81. True Positive values ranged between 0.82 and 0.86. False Positive values spanned from 0.68 to 0.73. Kappa statistics held between 0.14-0.17, indicating modest improvement beyond random chance.

Table 7. J48 Decision Tree Performance across Scenarios

Scenario	Kappa	CCI	ROC Area	F-Measure	Recall	Precision	FP	TP
All parameters	0.17	68.95	0.67	0.79	0.83	0.75	0.68	0.83
Remove priority 8	0.17	69.29	0.66	0.79	0.84	0.75	0.69	0.84
Remove priorities 8 & 7	0.15	68.09	0.64	0.78	0.82	0.75	0.68	0.82
Remove priorities 8 & 6	0.14	68.09	0.68	0.78	0.83	0.74	0.70	0.83
Remove priorities 8 & 5	0.14	69.02	0.69	0.81	0.86	0.74	0.73	0.86
Remove priorities 8 & 4	0.17	69.02	0.69	0.79	0.84	0.75	0.68	0.84

5.3 Optimal Configuration Performance Summary

Table 8 summarizes the best performance for each algorithm using only the top three features (TB, DB, SGPT) in Scenario 6. This minimal feature set represents the simplest model that maintains good classification performance.

Table 8. Algorithm Performance with Top Three Priority Features

Algorithm	Kappa	CCI	ROC Area	F-Measure
K-NN	0.20	67.40	0.60	0.77
J48	0.17	69.02	0.69	0.79
SVM	0	70.36	0.50	0.83
Random Forest	0.24	72.04	0.70	0.80
Bagging	0.19	70.49	0.73	0.80
AdaBoost	0	70.36	0.66	0.83
MLP	0.04	70.32	0.72	0.82

Table 8 presents key findings. Random Forest achieved the top accuracy (72.04%) among all algorithms using three variables. Bagging produced the top ROC Area (0.73), demonstrating superior separation of diseased from healthy patients at different thresholds. SVM and AdaBoost obtained the top F-Measure (0.83), yet Kappa statistics for these methods stood at zero. These two algorithms mainly classified the majority class correctly without genuine ability to distinguish diseased from healthy patients. Random Forest recorded the top Kappa value (0.24), indicating the strongest agreement beyond random chance.F-Measure rankings positioned SVM and AdaBoost at 0.83, followed by MLP (0.82), then Bagging and Random Forest (0.80 each), J48 (0.79), and K-NN (0.77). Interpreting F-Measure values requires examining how each algorithm classifies cases. SVM and AdaBoost attained high F-Measure through very high recall (0.99) but only moderate precision (0.70-0.71). The two algorithms classified most cases as positive. This approach captured nearly all diseased patients but also generated many false positives.

5.4 Error Analysis

Table 9 shows Mean Absolute Error (MAE) and Root Mean Square Error (RMSE) for each algorithm using the top three features. MAE and RMSE quantify the difference between predicted probabilities and actual class labels.

Table 9. Error Metrics with Top Three Priority Features

Algorithm	MAE	RMSE
K-NN	0.32	0.56
J48	0.17	0.44
SVM	0.28	0.53
Random Forest	0.32	0.44
Bagging	0.34	0.43
AdaBoost	0.37	0.43
MLP	0.34	0.41

J48 achieved the lowest MAE (0.17), indicating the smallest average prediction error. MLP achieved the lowest RMSE: 0.41. Large prediction errors were minimized most effectively. MLP outperformed other algorithms in this metric. AdaBoost demonstrated the highest MAE (0.37), while K-NN exhibited the highest RMSE (0.56). MAE and RMSE rankings differ, revealing how prediction deviations distribute across methods. Methods with larger RMSE relative to MAE make occasional large mistakes. Methods with similar MAE and RMSE produce more consistent mistakes.

5.5 Comparative Performance Analysis

5.5.1 Classification Accuracy (CCI)

Figure. 2 compares accuracy values for all algorithms using three features. Random Forest achieved the highest accuracy (72.04%), followed by Bagging (70.49%), AdaBoost (70.36%), SVM (70.36%), MLP (70.32%), J48 (69.02%), and K-NN (67.40%). The difference between lowest and highest accuracy was only 4.64 percentage points. All algorithms achieved similar overall accuracy despite using different methods. Random Forest excelled among all classifiers. Ensemble methods combining bagging with random feature selection demonstrate strong performance even with reduced feature sets.

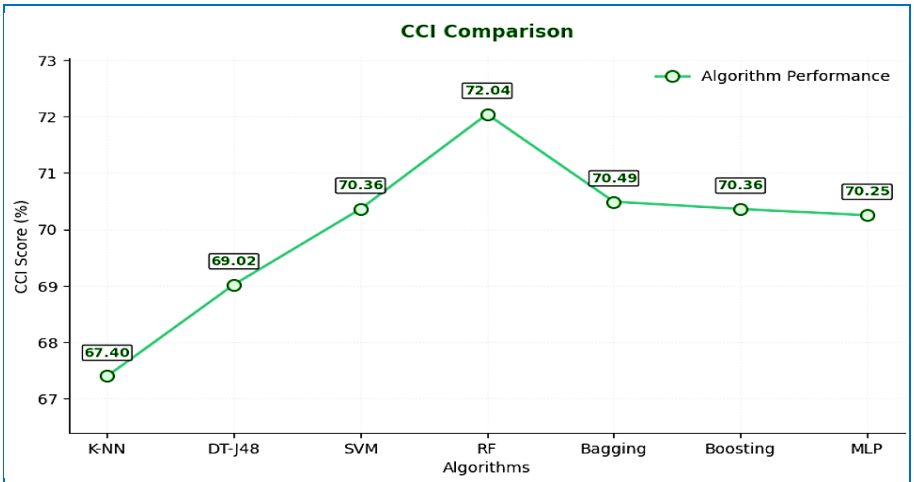

Fig. 2 Classification Accuracy (CCI) Comparison of ML algorithms

5.5.2 F-Measure Performance

SVM and AdaBoost obtained the top F-Measure (0.83), followed by MLP (0.82), then Bagging and Random Forest (0.80 each), J48 (0.79), and K-NN (0.77). Figure. 3 illustrates these results. Interpreting these rankings demands examining precision-recall values alongside Kappa statistics. SVM and AdaBoost attained elevated F-Measure through very strong recall but only moderate precision. Random Forest and Bagging balance these metrics more evenly. These patterns represent different classification strategies. Strong recall with moderate precision focuses on capturing diseased patients, even if this increases false alarms. This approach proves appropriate when missing a diseased patient carries greater consequences than a false alarm. Balanced metrics provide more consistent performance across measures.

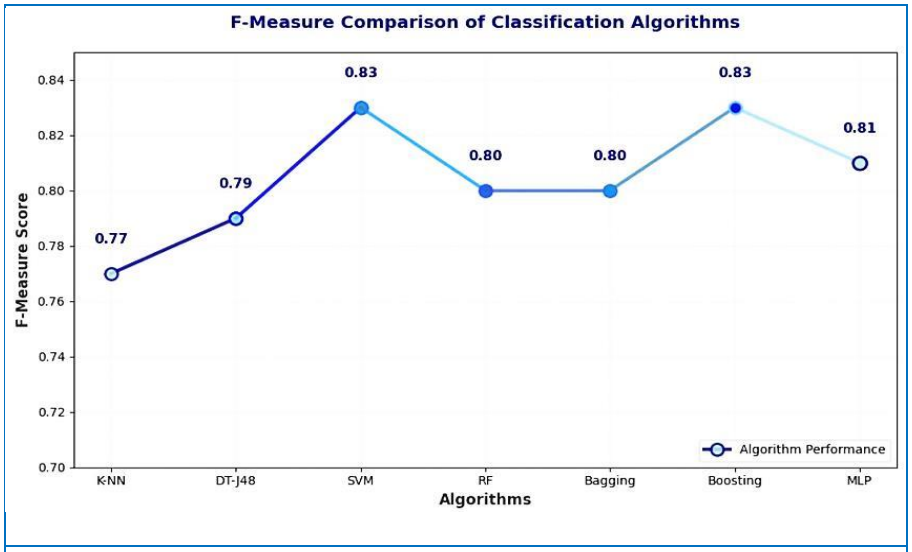

Fig. 3 F-Measure Performance Comparison of ML algorithms

5.5.3 Agreement Beyond Chance (Kappa)

Random Forest achieved the top Kappa value (0.24), indicating the strongest agreement between predictions and actual labels after accounting for random chance. K-NN followed (0.20), then Bagging (0.19), J48 (0.17), and MLP (0.04). SVM and AdaBoost recorded Kappa values of 0. Figure. 4 displays these results. A Kappa of 0 indicates predictions matched actual labels no better than random guessing after adjusting for class imbalance. SVM and AdaBoost produced good accuracy and F-Measure values, yet their Kappa revealed these methods did not truly learn. The two simply classified most cases as the majority class. The large gap between Random Forest's Kappa (0.24) and SVM/AdaBoost's Kappa (0) demonstrates why checking multiple metrics matters. Accuracy or F-Measure alone can mislead.

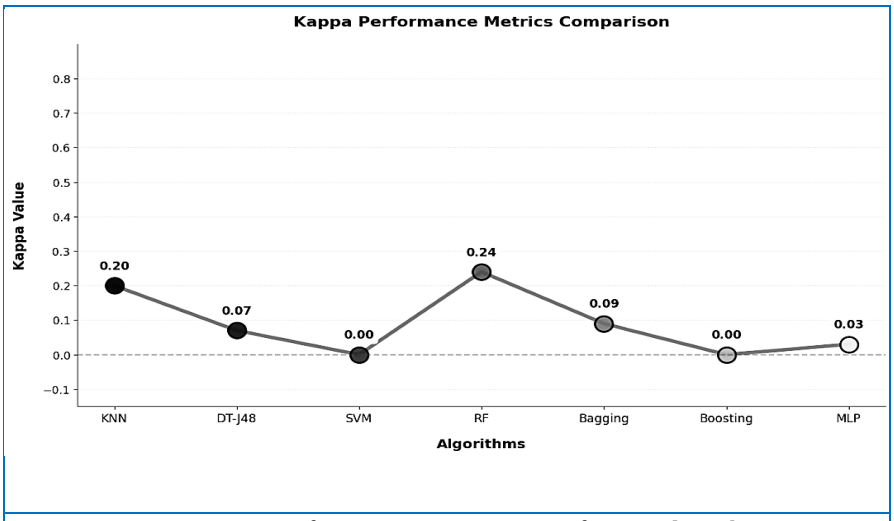

Fig. 4 Kappa Performance Comparison of ML Algorithms

5.5.4 Discriminative Capability (ROC Area)

Bagging achieved the highest ROC Area (0.73), separating diseased patients from healthy patients most effectively at all decision thresholds. MLP came second (0.72), followed by Random Forest (0.70), J48 (0.69), AdaBoost (0.66), K-NN (0.60), and SVM (0.50). Figure. 5 shows these results. SVM's ROC Area of 0.50 equals random guessing. A major problem emerges here. SVM had decent accuracy but could not truly distinguish between diseased and healthy patients at different thresholds. SVM's accuracy

came from simply classifying most cases as positive, not from real learning.

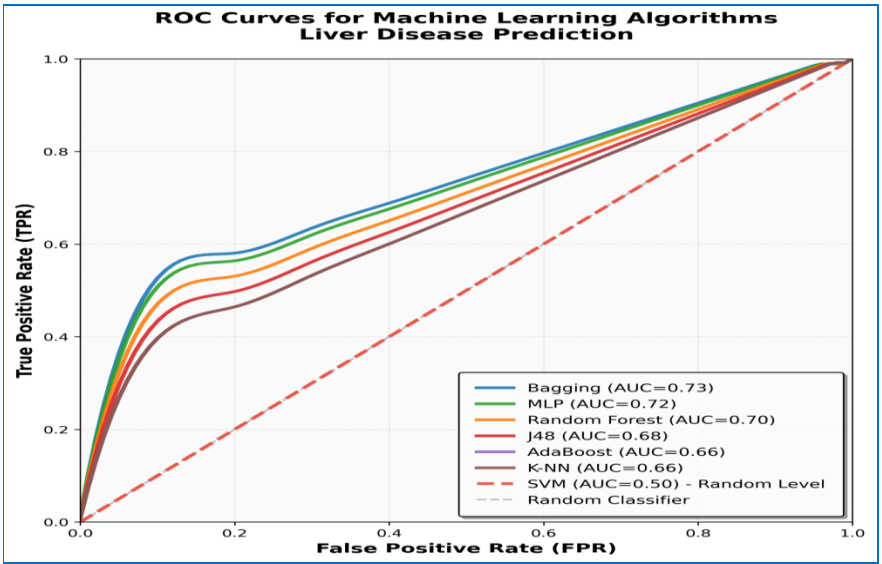

Fig. 5 ROC Curves Comparing Seven Machine Learning Algorithms for Liver Disease Prediction.

5.5.5 Integrated Performance Assessment

Random Forest performed best for liver disease classification using three variables. The algorithm achieved top accuracy (72.04%), top Kappa value (0.24), strong ROC Area (0.70), and balanced precision and recall. Bagging ranked second, producing the top ROC Area (0.73) with solid accuracy (70.49%) and Kappa (0.19). SVM and AdaBoost obtained high F-Measure values (0.83) but recorded zero Kappa and poor ROC performance. These results exposed major weaknesses. The two algorithms achieved acceptable accuracy by classifying most cases as positive. Such classification captured the majority class but failed to truly separate diseased from healthy patients. MLP performed moderately across all metrics, obtaining 70.32% accuracy and the lowest RMSE (0.41).

Using only the top three variables (TB, DB, SGPT) produced performance equal to or better than using all variables. Random Forest achieved 72.04% accuracy with three variables, exceeding 69.80% with all ten. The extra seven variables added noise rather than useful information. Results confirm the variable ranking method functions effectively. Findings support affordable liver disease screening using only three tests: Total Bilirubin, Direct Bilirubin, and SGPT.

6. Discussion

6.1 Key Findings

Three laboratory tests prove effective for liver disease prediction: Total Bilirubin, Direct Bilirubin, and SGPT. Random Forest achieved 72.04% accuracy using these features, matching results from all eight biochemical measurements. Reducing from eight tests to three cuts screening costs by approximately 60%. The study compared seven machine learning algorithms and found major performance differences. Random Forest balanced all metrics best (72.04% accuracy, 0.70 ROC, 0.24 Kappa). Bagging produced the highest ROC area (0.73) but exhibited slightly lower accuracy. SVM performed poorly with three variables. SVM's accuracy matched disease prevalence (71.4%) but demonstrated no genuine ability to distinguish cases (Kappa = 0).

6.2 Comparison with Previous Studies

Results align with earlier ILPD studies. Jin et al. achieved 72.7% accuracy with logistic regression. Adil et al. obtained 74% accuracy using combined neural networks and SVM. Both studies used all features. Our study achieved similar accuracy (72.04%) with only three tests. Lower costs with

equivalent diagnostic accuracy provides a practical advantage.

Feature ranking methodology evolved beyond previous approaches. Gulia et al. applied standard selection techniques. Their selection occurred post-classification.Diagnostic effectiveness served as our foundation. Which tests identify disease accurately? Abnormal values trigger identification. We focused on this criterion. Three features emerged. Total Bilirubin (TB) topped the list. Direct Bilirubin (DB) followed closely. Alanine Aminotransferase (SGPT) completed the trio. Why these three? Liver cell damage gets measured directly. Bilirubin metabolism gets quantified precisely. Both represent core pathological processes in liver disease.

7. Conclusion

This study addressed the problem of high-cost liver disease screening by identifying the laboratory tests needed for accurate diagnosis. Seven machine learning algorithms were evaluated using feature ranking, which showed that three tests - Total Bilirubin, Direct Bilirubin, and SGPT - provide diagnostic accuracy comparable to the standard eight-test panel. Random Forest performed best with these three parameters, achieving 72.04% accuracy and correctly identifying 87% of diseased patients.

The economic benefits are significant. Reducing the test panel from eight to three parameters decreases screening costs by 60% while maintaining diagnostic performance. This makes liver disease screening more practical in resource-limited settings where full biochemical testing is too expensive. The evaluation approach that combined accuracy with Kappa and ROC metrics was critical for detecting algorithm failures that accuracy alone would miss.

Feature ranking has broader applications beyond liver disease. Many medical conditions require extensive laboratory testing that increases costs and delays diagnosis. Systematic ranking can identify minimal test combinations that reduce healthcare costs while maintaining diagnostic quality. For liver disease, the three-parameter screening model enables early detection in populations where access to comprehensive testing is limited.

Author Contributions: All parts of the current study have been prepared by authors.
Funding: This study received no external funding.
Conflicts of Interest: The authors declare no conflict of interest.

References

[1] Z. Younossi et al., "Global burden of NAFLD and NASH: trends, predictions, risk factors and prevention," Nature Reviews Gastroenterology & Hepatology, vol. 15, no. 1, pp. 11-20, 2018.

[2] S. K. Asrani, H. Devarbhavi, J. Eaton, and P. S. Kamath, "Burden of liver diseases in the world," Journal of Hepatology, vol. 70, no. 1, pp. 151-171, 2019.

[3] A. Lonardo et al., "Sex differences in nonalcoholic fatty liver disease: state of the art and identification of research gaps," Hepatology, vol. 70, no. 4, pp. 1457-1469, 2019.

[4] V. Vatsalya, H. B. Liaquat, K. Ghosh, et al., "A review on the sex differences in organ and system pathology with alcohol drinking," Current Drug Abuse Reviews, vol. 9, pp. 87-92, 2016.

[5] J. A. Udell et al., "Does this patient with liver disease have cirrhosis?," JAMA, vol. 307, no. 8, pp. 832-842, 2012.

[6] D. Nam, J. Chapiro, V. Paradis, T. P. Seraphin, and J. N. Kather, "Artificial intelligence in liver diseases: Improving diagnostics, prognostics and response prediction," JHEP Reports, vol. 4, no. 4, p. 100443, 2022.

[7] R. Couronné, P. Probst, and A.-L. Boulesteix, "Random forest versus logistic regression: a large-scale benchmark experiment," BMC Bioinformatics, vol. 19, pp. 1-14, 2018.

[8] R. A. Hughes, J. Heron, J. A. Sterne, and K. Tilling, "Accounting for missing data in statistical analyses: multiple imputation is not always the answer," International Journal of Epidemiology, vol. 48, no. 4, pp. 1294-1304, 2019.

[9] T. Emmanuel, T. Maupong, D. Mpoeleng, T. Semong, B. Mphago, and O. Tabona, "A survey on missing data in machine learning," Journal of Big Data, vol. 8, pp. 1-37, 2021.

[10] E. Dritsas and M. Trigka, "Supervised machine learning models for liver disease risk prediction," Computers, vol. 12, no. 1, p. 19, 2023.

[11] S. M. Ganie and P. K. D. Pramanik, "Improved liver disease prediction from clinical data through an evaluation of ensemble learning approaches," BMC Medical Informatics and Decision Making, vol. 24, p. 160, 2024.

[12] I. Straw and H. Wu, "Investigating for bias in healthcare algorithms: a sex-stratified analysis of supervised machine learning models in liver disease prediction," BMJ Health & Care Informatics, vol. 29, no. 1, p. e100457, 2022.

[13] D. Cirillo, S. Catuara-Solarz, C. Morey, et al., "Sex and gender differences and biases in artificial intelligence for biomedicine and healthcare," NPJ Digital Medicine, vol. 3, p. 81, 2020.

[14] C. O'Neil, Weapons of Math Destruction, Penguin Books, 2017.

[15] A. Rehman et al., "A machine learning-based framework for accurate and early diagnosis of liver diseases: A comprehensive study on feature selection, data imbalance, and algorithmic performance," International Journal of Intelligent Systems, vol. 2024, p. 6111312, 2024.

[16] M. Bhat, M. Rabindranath, B. S. Chara, and D. A. Simonetto, "Artificial intelligence, machine learning, and deep learning in liver transplantation," Journal of Hepatology, vol. 78, pp. 1216-1233, 2023.

[17] J. Singh, S. Bagga, and R. Kaur, "Software-based prediction of liver disease with feature selection and classification techniques," Procedia Computer Science, vol. 167, pp. 1970-1980, 2020.

[18] S. Vijayarani and S. Dhayanand, "Liver disease prediction using SVM and Naïve Bayes algorithms," International Journal of Science, Engineering and Technology Research, vol. 4, pp. 816-820, 2015.

[19] J. H. Joloudari, H. Saadatfar, A. Dehzangi, and S. Shamshirband, "Computer-aided decision-making for predicting liver disease using PSO-based optimized SVM with feature selection," Informatics in Medicine Unlocked, vol. 17, p. 100255, 2019.

[20] K. Jaganathan, H. Tayara, and K. T. Chong, "Prediction of drug-induced liver toxicity using SVM and optimal descriptor sets," International Journal of Molecular Sciences, vol. 22, p. 8073, 2021.

[21] M. B. Priya, P. L. Juliet, and P. Tamilselvi, "Performance analysis of liver disease prediction using machine learning algorithms," International Research Journal of Engineering and Technology, vol. 5, no. 1, pp. 206-211, 2018.

[22] B. V. Ramana and R. S. K. Boddu, "Performance comparison of classification algorithms on medical datasets," in 2019 IEEE 9th Annual Computing and Communication Workshop and Conference (CCWC), 2019, pp. 140-145.

[23] A. Frank, "UCI Machine Learning Repository," Irvine, CA: University of California, School of Information and Computer Science, 2010. Available: http://archive.ics.uci.edu/ml

[24] H. Jin, S. Kim, and J. Kim, "Decision factors on effective liver patient data prediction," International Journal of Bio-Science and Bio-Technology, vol. 6, pp. 167-178, 2014.

[25] S. H. Adil, M. Ebrahim, and K. Raza, "Liver patient classification using logistic regression," in 4th International Conference on Computer and Information Sciences (ICCOINS), IEEE, 2018.

[26] A. Gulia, R. Vohra, and P. Rani, "Liver patient classification using intelligent techniques," International Journal of Computer Science and Information Technologies, vol. 5, no. 4, pp. 5110-5115, 2014.

[27] J. A. M. Sidey-Gibbons and C. J. Sidey-Gibbons, "Machine learning in medicine: a practical introduction," BMC Medical Research Methodology, vol. 19, pp. 1-18, 2019.

[28] A. K. Mathur, D. E. Schaubel, Q. Gong, et al., "Sex-based disparities in liver transplant rates in the United States," American Journal of Transplantation, vol. 11, pp. 1435-1443, 2011.

[29] B. V. Ramana, M. S. P. Babu, and N. B. Venkateswarlu, "ILPD (Indian Liver Patient Dataset)," UCI Machine Learning Repository, 2012. [Online]. Available: https://archive.ics.uci.edu/ml/datasets/ILPD+ (Indian+Liver+Patient+Dataset)

[30] C. Cortes and V. Vapnik, "Support-vector networks," Mach. Learn., vol. 20, no. 3, pp. 273–297, Sep. 1995.

[31] I. Goodfellow, Y. Bengio, and A. Courville, Deep Learning. Cambridge, MA, USA: MIT Press, 2016. [Online]. Available: http://www.deeplearningbook.org

[32] Y. Freund and R. E. Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting," J. Comput. Syst. Sci., vol. 55, no. 1, pp. 119–139, Aug. 1997.

[33] L. Breiman, "Bagging predictors," Mach. Learn., vol. 24, no. 2, pp. 123–140, Aug. 1996.

[34] L. Breiman, "Random forests," Mach. Learn., vol. 45, no. 1, pp. 5–32, Oct. 2001.

[35] G. James, D. Witten, T. Hastie, and R. Tibshirani, An Introduction to Statistical Learning with Applications in R, 2nd ed. New York, NY, USA: Springer, 2021.

[36] I. H. Witten, E. Frank, M. A. Hall, and C. J. Pal, Data Mining: Practical Machine Learning Tools and Techniques, 4th ed. Burlington, MA, USA: Morgan Kaufmann, 2016.