A Report on

Heart Disease Prediction Using Ensemble Learning

Submitted to Manipal University, Jaipur

Towards the partial fulfillment for the Award of the Degree of

BACHELORS OF TECHNOLOGY

In Computers Science and Engineering

2024-2025

By

Eshan Sud

229301214 (CSE 6H)

MANIPAL UNIVERSITY
JAIPUR

Under the guidance of

Dr. Shikha Mundra

Department of Computer Science and Engineering

School of Computing and Information Technology

Manipal University Jaipur

Jaipur, Rajasthan

1. Introduction of the Problem Statement

Heart disease is a major global health concern, making early detection essential for effective treatment. Conventional diagnostic techniques depend on medical tests and expert evaluation, which can be time-intensive and susceptible to human error. This project focuses on creating a predictive model using machine learning to assess the possibility of heart disease based on key medical factors, including age, blood pressure, cholesterol levels, and other health indicators.

The primary challenge lies in the complexity and variability of heart disease symptoms, which differ among individuals. By leveraging machine learning algorithms, we can analyse patterns in large datasets and improve predictive accuracy. The proposed model will utilise supervised learning techniques, trained on historical medical data, to provide reliable risk assessments. This approach can enhance decision-making for healthcare professionals and enable proactive measures to prevent severe complications.

Furthermore, the integration of such predictive models into healthcare systems can reduce the burden on medical professionals by streamlining the diagnostic process. With the rising availability of electronic health records and advancements in computational power, machine learning-based heart disease prediction has the potential to significantly impact public health by facilitating early detection and personalised treatment strategies.

2. Contribution Highlights

- Optimised Feature Engineering: The approach incorporates scaling and dimensionality reduction to enhance learning efficiency while minimizing noise and redundancy in the dataset.

- Balanced Training with Oversampling: To address class imbalance, SMOTE-based oversampling was applied before training ensemble models, improving prediction stability.

- Comprehensive Model Evaluation: Model performance is rigorously assessed using Accuracy, Precision, Recall, F1-score, MCC, and Confusion Matrices, ensuring robust generalizability.

- Advanced Ensemble Learning for Enhanced Prediction: Soft Voting, Hard Voting, and Stacking Classifiers were implemented, integrating Logistic Regression, Random Forest, SVM, and XGBoost, with and without Hyperparameter Tuning, to improve classification accuracy.

3. Dataset Description & Visualisation

The dataset used in this project contains relevant medical attributes such as age, cholesterol levels, blood pressure, and ECG readings. A summary of the dataset is presented in the table below:

- With 303 entries & a total 14 columns

Table 1 Features and their description

Feature	Description	Unit / Categories
age	Patient's Age.	In Years
sex	Patient's Gender.	1 = Male, 0 = Female
cp	Type of Chest pain.	0-3
trestbps	B.P calculated when patient is at rest.	In mm Hg
chol	Serum cholesterol level.	In Mg/dL
fbs	Blood sugar when patient is fasting. (>120 mg/dL)	1 = True, 0 = False
restecg	ECG results calculated when patient is at rest.	0-2
thalach	The highest heart rate reached by the patient.	Continuous
exang	Angina triggered by physical exertion.	1 = Yes, 0 = No
oldpeak	ST depression caused by physical activity.	Continuous
slope	Slope of the ST segment at peak exercise.	0-2
ca	Count of major vessels highlighted by fluoroscopy.	0-4
thal	Thalassemia type.	0-3
target	Whether heart disease is present or absent.	1 = Presence, 0 = Absence

Figure 1 Dataset Visualisation:

```
   age  sex  cp  trestbps  chol  fbs  restecg  thalach  exang  oldpeak  slope  \
0   63    1   3       145   233    1        0      150      0      2.3      0
1   37    1   2       130   250    0        1      187      0      3.5      0
2   41    0   1       130   204    0        0      172      0      1.4      2
3   56    1   1       120   236    0        1      178      0      0.8      2
4   57    0   0       120   354    0        1      163      1      0.6      2

   ca  thal  target
0   0     1       1
1   0     2       1
2   0     2       1
3   0     2       1
4   0     2       1
```

Figure 2 Missing values found:

```
No missing values found.
```

Figure 3 Feature distributions:

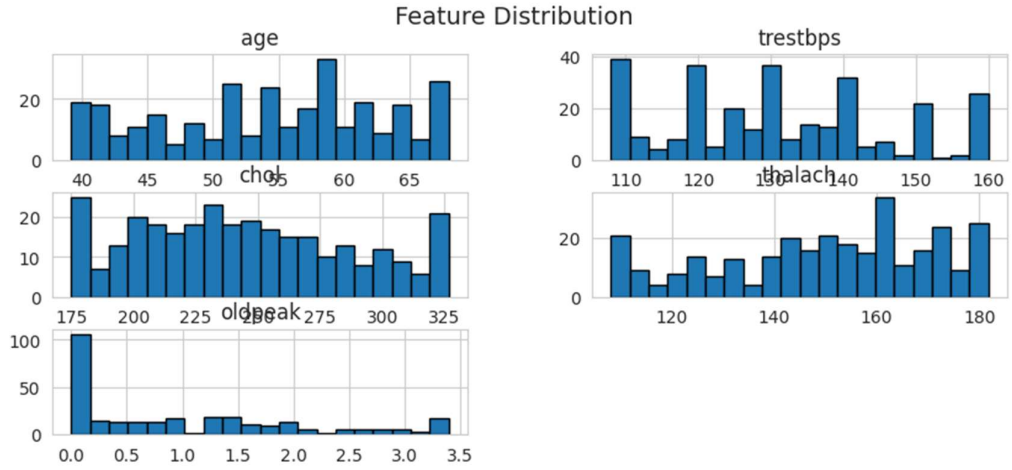


Figure 4 Target distributions:

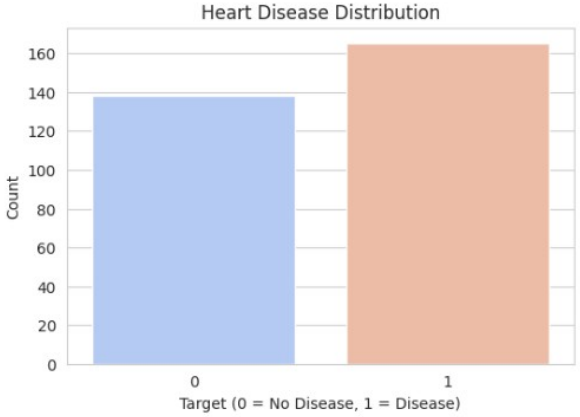


Figure 5 Boxplots:

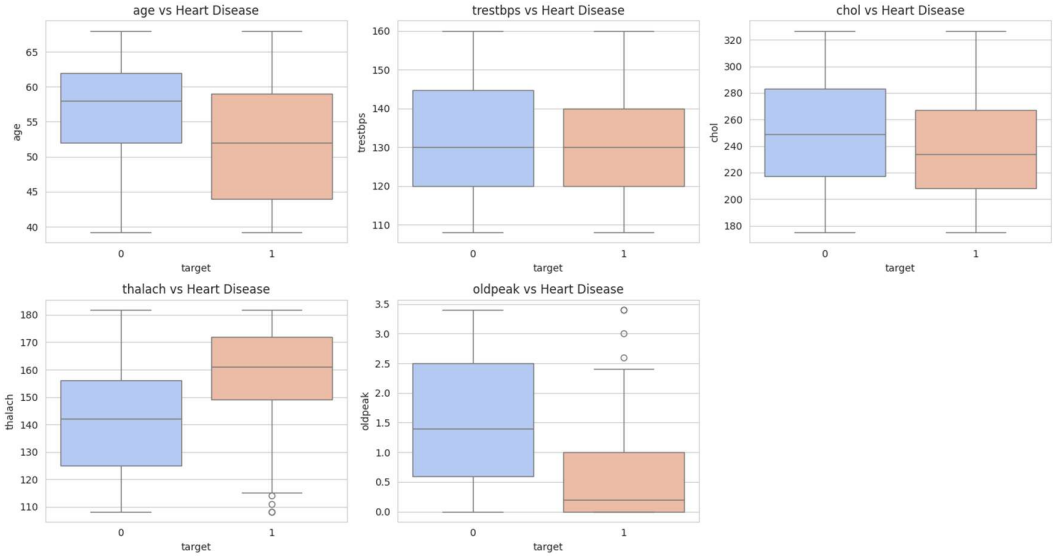


Figure 6 PCA plot:

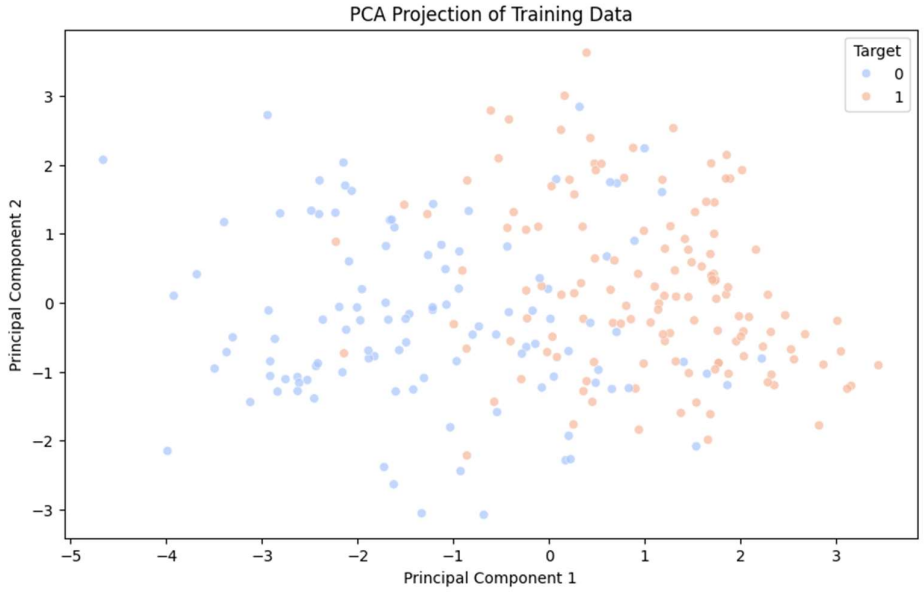
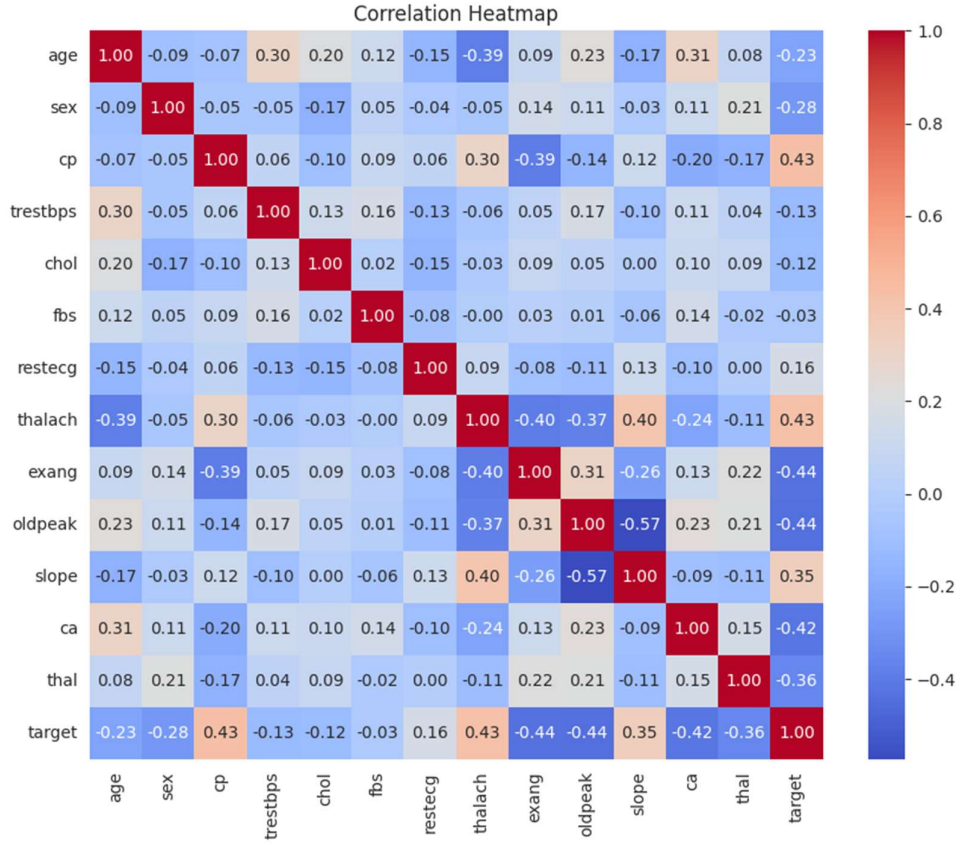


Figure 6 Correlation heatmap:



4. Algorithms Used

- The ensemble learning approach integrates multiple classifiers to improve prediction reliability. We use soft voting to combine predictions from different models.
- Mathematically, soft voting is expressed as: $\mathbf{P\ (y = k) = 1/n * \sum_{i=1}^{n} (P_i\ (y = k))}$
  Where, P_i (y = k) represents the probability prediction of classifier 'I' for class 'k'.

4.1. Train-Test Split & Normalisation

- The dataset is split into 80 and 20% respectively of training and testing, ensuring class balance using stratification.
- Data normalization is performed using StandardScaler to improve model performance and prevent feature dominance.
- Mathematically, standardization is applied as:
  $\mathbf{X' = (X - \mu) / \sigma}$
  Where, μ = Mean & σ = Standard Deviation.

4.2. Models Used

4.2.1. Logistic Regression

- Effective for linearly separable data.
- Provides probabilistic interpretations of predictions.
- Working:
  - Computes the probability (sigmoid function):
    $\mathbf{P\ (y) = 1 / (1 + e^{(-\ (\beta 0 + \beta 1 * X1 + \ldots + \beta n * Xn))})}$
  - Uses maximum likelihood estimation to find optimal coefficients.

4.2.2. Random Forest

- Reduces overfitting through bagging.
- Handles missing data efficiently.
- Working:
  - Constructs multiple decision trees from bootstrap samples.
  - Uses majority voting for final prediction.

4.2.3. Support Vector Machine (SVM)

- Useful for high-dimensional data.
- Employs kernels for non-linearly separable cases.
- Working:
  - Finds the optimal hyperplane that maximises margin between classes.
  - Uses the following optimization function:
    $\mathbf{min\ (1/2)\ ||w||^2}$
    Such that, $\mathbf{y_i\ (w * x_i + b) \geq 1}$.

4.2.4. XGBoost

- Gradient boosting enhances prediction accuracy.
- Built-in regularization reduces overfitting.
- Working:
    - Utilises weak learners (decision trees) in sequential boosting.
    - Optimises using gradient descent to minimise residuals.

4.2.5. Soft Voting Classifier

- Combines multiple models by averaging their predicted probabilities.
- More stable than individual models as it considers overall confidence levels.
- Working:
    - Each base model generates probability scores for each class.
    - The final prediction is determined by the average of these probabilities.
    - The class with the highest average probability is chosen as the output.
- Mathematically, for N models, the final probability for class c is:
  $\mathbf{P\ (y = c) = 1/N * \sum_{i=1}^{N} P_i\ (y=c)}$
  Where, P_i (y=c) is the probability predicted by the i-th model.

4.2.6. Hard Voting Classifier

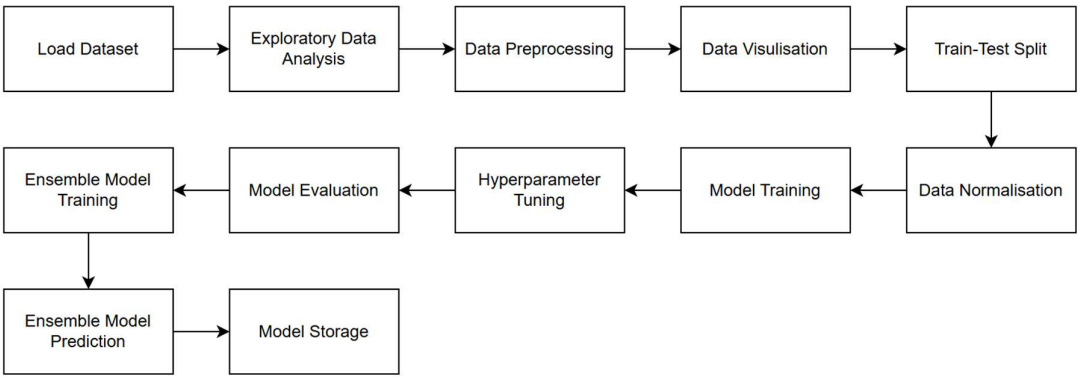
- Uses a majority rule to determine the final class.
- Works best when base models are diverse yet accurate.
- Working:
    - Each model makes an independent class prediction.
    - The final prediction is the most frequently predicted class.
- In case of a tie, a predefined strategy (random selection or weighted votes) is used.
- Mathematically, the final class is determined by:
  $\mathbf{\hat{y} = mode(\hat{y}_1, \hat{y}_2, \ldots, \hat{y}_N)}$
  Where, $\hat{y}_i$ is the predicted class by model 'I'.

4.2.7. Stacking Classifier

- Uses a meta-model to combine predictions from multiple base models.
- Can capture complex relationships between model outputs.
- Working:
    - Base models generate individual predictions on training data.
    - These predictions are used as new features for a meta-classifier (e.g., Logistic Regression).
    - The meta-classifier learns to combine base models optimally for final predictions.
- Mathematically, if base models $M_1, M_2, \ldots, M_N$ predict outputs $O_1, O_2, \ldots, O_N$, the meta-model learns:
  $\mathbf{\hat{y} = f(O_1, O_2, \ldots, O_N)}$
  Where, f is the final estimator that refines predictions.

4.3. Pipeline

Figure 7 Pipeline:



This pipeline ensures a structured approach, from data preprocessing to evaluation, optimising prediction accuracy and reliability.

5. Hyperparameter Description & Training Visualization

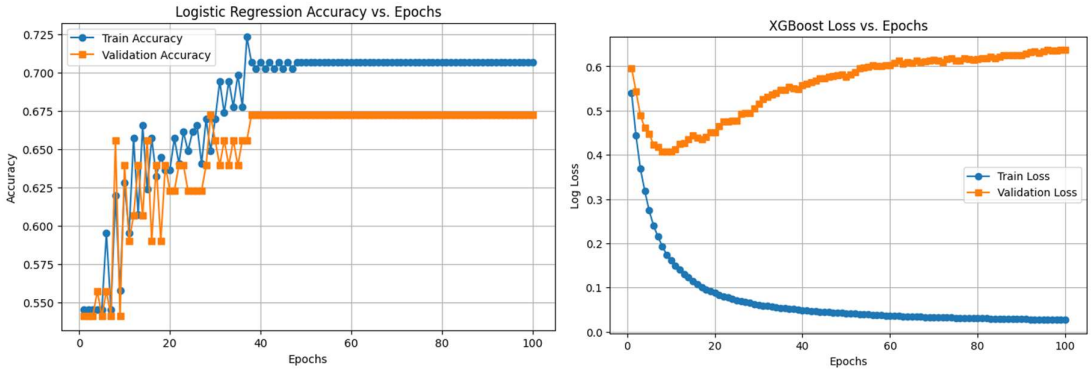
Hyperparameters control model learning behaviour and performance. Unlike model parameters (like weights in neural networks), hyperparameters must be set before training. Below are key hyperparameters for each model:

Table 2 Models used and their hyperparameters

Model	Hyperparameter	Description
Logistic Regression	c	Inverse of the regularization parameter (a lower value indicates stronger regularization).
	solver	Optimisation algorithm (eg, 'lbfgs', 'saga').
	max_iter	Max. no. of iterations allowed for the algorithm to converge.
Random Forest (Used for further tuning)	**n_estimators**	**No. of trees included in the forest.**
	max_depth	**Max. depth of each tree.**
	min_samples_split	**Min. no. of samples needed to split each node.**
Support Vector Machine (SVM)	c	Regularization parameter (higher = stricter margins).
	kernel	Specifies kernel type (linear, polynomial, RBF, etc).
	gamma	Controls the influence of a single training example (for RBF/poly kernels).
XGBoost (Used for further tuning)	**n_estimators**	**No. of boosting iterations.**
	learning_rate	**Learning rate used to prevent overfitting.**
	max_depth	**Max. depth of the trees, which controls their complexity.**
	subsample	**Fraction of training data used per boosting iteration.**

5.1. Visualisation of training process

Figure 8 Accuracy vs. Epochs Curve (For Logistic Regression) and Loss vs. Epochs Curve (For XGBoost):



6. Experimental Results & Evaluation Measures

To evaluate the effectiveness of the Machine Learning algorithms basic measures like Accuracy, Precision, Recall and F1-Measure (Han & Kamber, 2012) were adopted. Squared error-based cost functions are inconsistent for solving classification problems. Also, these measures are widely used in domains such as information retrieval, machine learning and other domains that involve classification (Olson & & Delen, 2008). A confusion matrix is a base for the determination of these measures.

6.1. Evaluation Metrics

Confusion Matrix:

		Predicted/Classified	
		Negative	Positive
Actual	Negative	TN	FP
	Positive	FN	TP

Accuracy = (TP + TN) / (TP + TN + FP + FN)

Precision = TP / (FP + TP)

Recall = TP / (FN + TP)

F1-Measure = 2 * (Precision * Recall) / (Precision + Recall)

Sensitivity = TP / P

Specificity = TN / N

FPR = FP / N

FNR = FN / P

NPV = TN / (TN + FN)

FDR = FP / (FP + TP)

MCC = (TP * TN) - (FP * FN) / √((TP + FP) * (TP + FN) * (TN + FP) * (TN + FN))

Where,
True Positive (TP) refers to the no. of positive instances correctly identified as positive.
False Positive (FP) refers to the no. of positive instances incorrectly identified as negative.
True Negative (TN) refers to the no. of negative instances correctly identified as negative.
False Negative (FN) refers to the no. of negative instances incorrectly identified as positive.

6.2. Experimental Results

Table 3 Confusion Matrix: Logistic Regression

		Predicted/Classified	
		0	1
Actual	0	TN = 19	FP = 9
	1	FN = 3	TP = 30

Table 4 Confusion Matrix: Random Forest

		Predicted/Classified	
		0	1
Actual	0	TN = 19	FP = 9
	1	FN = 1	TP = 32

Table 5 Confusion Matrix: SVM

		Predicted/Classified	
		0	1
Actual	0	TN = 19	FP = 9
	1	FN = 2	TP = 31

Table 6 Confusion Matrix: XGBoost

		Predicted/Classified	
		0	1
Actual	0	TN = 18	FP = 10
	1	FN = 2	TP = 31

Table 7 Confusion Matrix: Random Forest (Hyperparameter Tuned)

		Predicted/Classified	
		0	1
Actual	0	TN = 20	FP = 8
	1	FN = 1	TP = 32

Table 8 Confusion Matrix: XGBoost (Hyperparameter Tuned)

		Predicted/Classified	
		0	1
Actual	0	TN = 19	FP = 9
	1	FN = 1	TP = 32

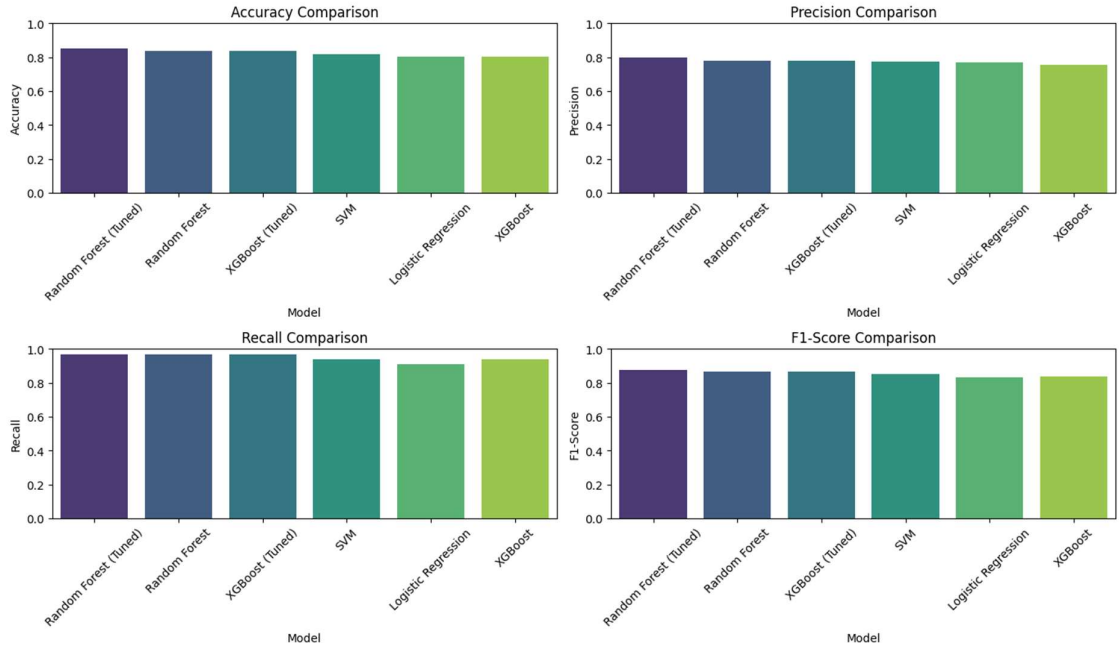


Figure 9 Comparison of Models: Accuracy, Precision, Recall, F1_Score

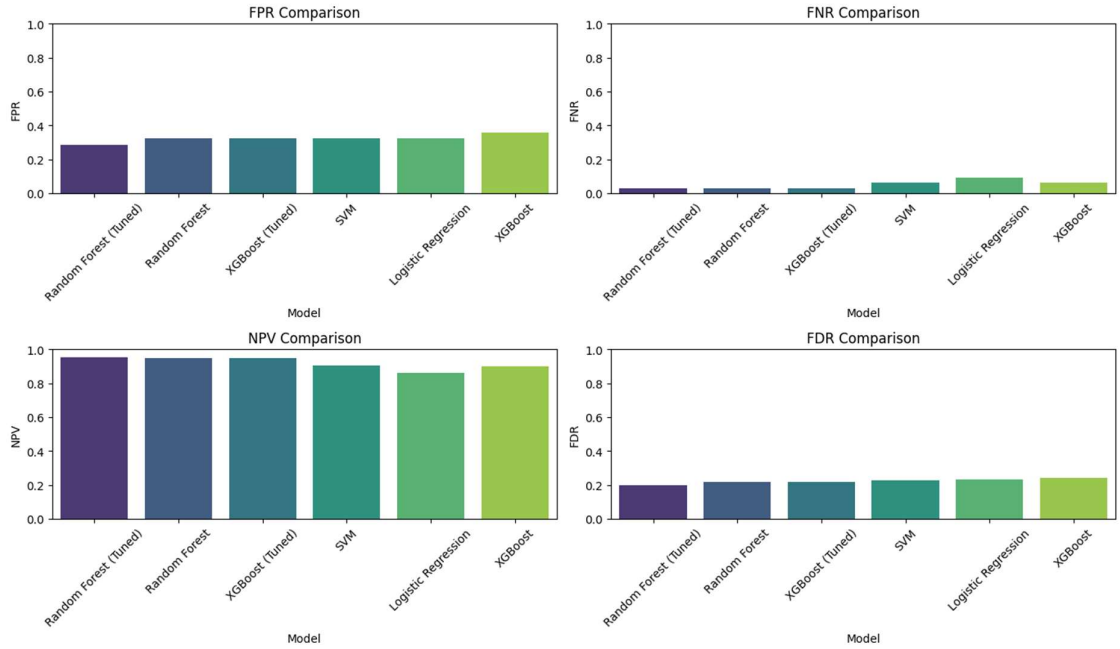


Figure 10 Comparison of Models: FPR, FNR, NPV, and FDR

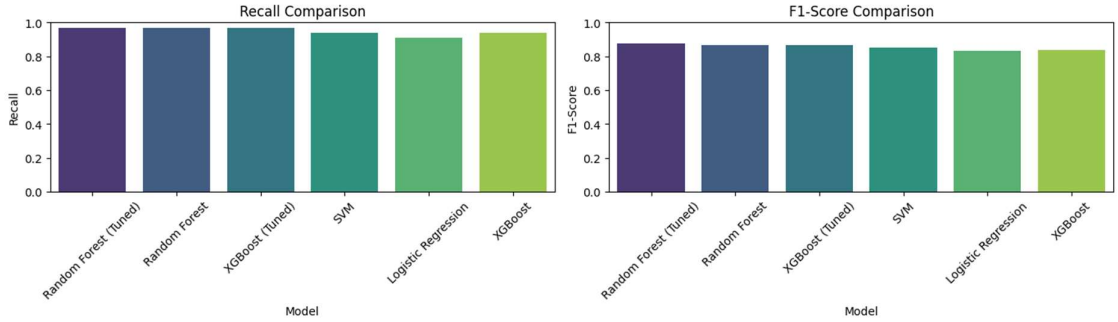


Figure 11 Comparison of Models: MCC, Specificity

```
------- Model Comparison Table -------
                   Model  Accuracy  Precision  Recall  F1-Score     MCC  \
4  Random Forest (Tuned)    0.8525     0.8000  0.9697    0.8767  0.7174
1          Random Forest    0.8361     0.7805  0.9697    0.8649  0.6882
5        XGBoost (Tuned)    0.8361     0.7805  0.9697    0.8649  0.6882
2                    SVM    0.8197     0.7750  0.9394    0.8493  0.6481
0    Logistic Regression    0.8033     0.7692  0.9091    0.8333  0.6098
3                XGBoost    0.8033     0.7561  0.9394    0.8378  0.6181

   Specificity     FPR     FNR     NPV     FDR
4       0.7143  0.2857  0.0303  0.9524  0.2000
1       0.6786  0.3214  0.0303  0.9500  0.2195
5       0.6786  0.3214  0.0303  0.9500  0.2195
2       0.6786  0.3214  0.0606  0.9048  0.2250
0       0.6786  0.3214  0.0909  0.8636  0.2308
3       0.6429  0.3571  0.0606  0.9000  0.2439
```

Figure 12 Model Comparison Table

6.3. Results

The results for various evaluation measures for the various training percentages are indicated in table 7, figure 12, 13 and 14

Table 8 Confusion Matrix: Hybrid Model (Hard Voting - Before SMOTE)

		Predicted/Classified	
		0	1
Actual	0	TN = 19	FP = 9
	1	FN = 1	TP = 32

```
------ Model Performance Comparison ------

+---+-----------------------------------+----------+-----------+--------+----------+--------+-------------+--------+--------+--------+--------+
|   |               Model               | Accuracy | Precision | Recall | F1-Score |  MCC   | Specificity |  FPR   |  FNR   |  NPV   |  FDR   |
+---+-----------------------------------+----------+-----------+--------+----------+--------+-------------+--------+--------+--------+--------+
| 0 | Soft Voting (Before SMOTE, Tuned) |  0.8197  |  0.7619   | 0.9697 |  0.8533  | 0.6591 |   0.6429    | 0.3571 | 0.0303 | 0.9474 | 0.2381 |
| 1 | Hard Voting (Before SMOTE, Tuned) |  0.8525  |    0.8    | 0.9697 |  0.8767  | 0.7174 |   0.7143    | 0.2857 | 0.0303 | 0.9524 |  0.2   |
| 2 |  Stacking (Before SMOTE, Tuned)   |  0.8197  |   0.775   | 0.9394 |  0.8493  | 0.6481 |   0.6786    | 0.3214 | 0.0606 | 0.9048 | 0.225  |
| 3 | Soft Voting (After SMOTE, Tuned)  |  0.7869  |  0.7632   | 0.8788 |  0.8169  | 0.5731 |   0.6786    | 0.3214 | 0.1212 | 0.8261 | 0.2368 |
| 4 | Hard Voting (After SMOTE, Tuned)  |  0.8197  |  0.7895   | 0.9091 |  0.8451  | 0.641  |   0.7143    | 0.2857 | 0.0909 | 0.8696 | 0.2105 |
| 5 |   Stacking (After SMOTE, Tuned)   |  0.8197  |  0.7895   | 0.9091 |  0.8451  | 0.641  |   0.7143    | 0.2857 | 0.0909 | 0.8696 | 0.2105 |
+---+-----------------------------------+----------+-----------+--------+----------+--------+-------------+--------+--------+--------+--------+
```

Figure 13 Comparison of various Ensemble Models (Soft Voting Classifier, Hard Voting Classifier, Stacking Classifier, all before and after SMOTE): Accuracy, Precision, Recall, F1_Score, MCC, Specificity, FPR, FNR, NPV, and FDR

```
Confusion Matrix: [[20  8]
 [ 1 32]]
Accuracy: 0.8525
Precision: 0.8000
Recall: 0.9697
F1-Score: 0.8767
MCC: 0.7174
Specificity: 0.7143
FPR: 0.2857
FNR: 0.0303
NPV: 0.9524
FDR: 0.2000
```

Figure 14 Results for Hybrid Model (using the Hard Voting Classifier): Confusion Matrix, Accuracy, Precision, Recall, F1_Score, MCC, Specificity, FPR, FNR, NPV, and FDR

The specificity of the best-performing model, Random Forest (Tuned), is 0.71, which is slightly higher than other models such as SVM and XGBoost but still within a similar range. The recall (sensitivity) score is 0.97, making it the most effective model in correctly identifying positive cases. The precision is 0.80, showing a good balance between identifying true positives while minimizing false positives.

The False Positive Rate (FPR) for the top-performing model is 0.29, which is relatively lower compared to some models but slightly higher than the ideal threshold. The False Negative Rate (FNR) is minimal at 0.03, demonstrating that the model effectively reduces false negatives, making it a strong candidate for situations where missing positive cases is critical. The Negative Predictive Value (NPV) is high at 0.95, meaning the model reliably predicts negative cases. The False Discovery Rate (FDR) is 0.20, which is lower than most models, indicating a strong precision balance.

The accuracy of the best model, Random Forest (Tuned), is 0.85, outperforming all other models, including standard Random Forest (0.83), SVM (0.82), and XGBoost (0.80). This makes it the most

reliable model for the given dataset. The F1-score is 0.88, confirming a consistent balance between precision and recall, making it ideal for classification tasks with imbalanced data.

The Matthews Correlation Coefficient (MCC) score for the Random Forest (Tuned) model is 0.72, highlighting a strong correlation between actual and predicted classifications. Although it marginally outperforms standard Random Forest (0.69) and XGBoost (0.69), it remains superior to SVM (0.65) and Logistic Regression (0.61). The high MCC score validates the robustness of the model, making it a dependable choice for this classification problem.

Comparing evaluation metrics, Random Forest (Tuned) emerges as the best model due to its superior accuracy, strong sensitivity, and balanced precision-recall trade-off. While minor trade-offs exist in specificity and false positives, the model's ability to accurately classify positive cases makes it the most suitable choice for this dataset.

Table 9 Overall Performance of hybrid classification model over other methods

Algorithms → Measures ↓	**Random Forest (Tuned)**	**Random Forest**	**XGBoost (Tuned)**	**SVM**	**Logistic Regression**	**XGBoost**	**Hybrid Model (Hard Voting Before SMOTE)**
Specificity	0.7143	0.6786	0.6786	0.6786	0.6786	0.6429	**0.7143**
Sensitivity/Recall	0.9697	0.9697	0.9394	0.9394	0.9091	0.9697	**0.9697**
Accuracy	0.8525	0.8361	0.8361	0.8197	0.8033	0.8033	**0.8525**
Precision	0.8000	0.7805	0.7805	0.7750	0.7692	0.7561	**0.8**
FPR	0.2857	0.3214	0.3214	0.3214	0.3214	0.3571	**0.2857**
FNR	0.0303	0.0303	0.0303	0.0606	0.0909	0.0606	**0.0303**
NPV	0.9524	0.9500	0.9500	0.9048	0.8636	0.9000	**0.9524**
FDR	0.2000	0.2195	0.2195	0.2250	0.2308	0.2439	**0.2**
F1- Score	0.8767	0.8649	0.8649	0.8493	0.8333	0.8378	**0.8767**
MCC	0.7174	0.6882	0.6882	0.6481	0.6098	0.6181	**0.7174**

Table 10 Overall Performance of ensemble models

Algorithms → Measures ↓	**Before SMOTE**			**After SMOTE**		
	Soft Voting	**Hard Voting (Hybrid Model)**	**Stacking**	**Soft Voting**	**Hard Voting**	**Stacking**
Specificity	0.6429	**0.7143**	0.6786	0.6786	0.7143	0.7143
Sensitivity/Recall	0.9697	**0.9697**	0.9394	0.8788	0.9091	0.9091
Accuracy	0.8197	**0.8525**	0.8197	0.7869	0.8197	0.8197
Precision	0.7619	**0.8**	0.775	0.7632	0.7895	0.7895
FPR	0.3571	**0.2857**	0.3214	0.3214	0.2857	0.2857
FNR	0.0303	**0.0303**	0.0606	0.1212	0.0909	0.0909
NPV	0.9474	**0.9524**	0.9048	0.8261	0.8696	0.8696
FDR	0.2381	**0.2**	0.225	0.2368	0.2105	0.2105
F1- Score	0.8533	**0.8767**	0.8493	0.8169	0.8451	0.8451
MCC	0.6591	**0.7174**	0.6481	0.5731	0.641	0.641