**Phase-3 Submission**

**Student Name:** HARI A

**Register Number:** 712523106301

**Institution:** PPG INSTITUTE OF TECHNOLOGY

**Department:** ELECTRONICS AND COMMUNICATION ENGINEERING

**Date of Submission:** [17/05/2025]

**Github Repository Link:** [https://github.com/harinihu/NM\_HARI\_DS]

# 1. Problem Statement

*Heart disease is one of the leading causes of death worldwide. Early detection and prediction of heart disease can significantly reduce mortality rates. This project aims to develop a machine learning model that predicts the presence of heart disease based on patient health parameters. This is a* ***classification problem*** *since the outcome is binary (disease or no disease).*

# 2. Abstract

*This project focuses on building a predictive model for heart disease using machine learning techniques. The objective is to classify whether a patient is at risk of heart disease based on medical attributes such as age, cholesterol levels, blood pressure, etc. Data preprocessing, exploratory data analysis, and feature selection were performed to improve model accuracy. Multiple classification algorithms like Logistic Regression, Random Forest, and SVM were evaluated. The best-performing model was selected based on performance metrics like accuracy and F1-score. The model aims to assist healthcare professionals in early detection and risk assessment.*

# 3. System Requirements

○ **Hardware**: Minimum 4GB RAM, Intel i3 or above processor

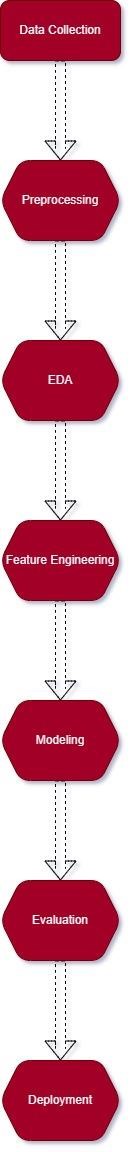
○ **Software**:

* Python 3.8+
* Libraries: pandas, numpy, scikit-learn, matplotlib, seaborn
* IDE: Jupyter Notebook or Google Colab

# 4. Objectives

* *Predict the likelihood of a person having heart disease based on medical attributes.*
* *Improve healthcare diagnostics using machine learning tools.*
* *Evaluate various ML models to find the most accurate prediction technique.*
* *Make healthcare decision-making faster and more data-driven*

# 5. Flowchart of Project Workflow



# 6. Dataset Description

* ***Source:*** *Kaggle (Heart Disease UCI dataset)*
* ***Type:*** *Public*
* ***Size & Structure:*** *303 rows, 14 columns*
* ***df.head() Screenshot:*** *(Insert the top 5 rows from your dataset)*

# 7. Data Preprocessing

* ***Handle Missing Values, Duplicates, Outliers:***
  1. *Checked for missing values using df.isnull().sum() – no missing data found in the UCI heart dataset.*

*○ Removed any duplicate rows using df.drop\_duplicates() (if applicable).*

*○ Identified and treated outliers in features like chol and thalach using boxplots or Z-score method.*

* ***Feature Encoding and Scaling:***
  1. *Categorical features like sex, cp, thal, slope were encoded using pd.get\_dummies() or Label Encoding.*

*○ StandardScaler was used to scale features like age, trestbps, chol, thalach, and oldpeak for better model performance.*

* ***Before/After Transformation Screenshots:***
  1. *(Include a screenshot showing the dataset before and after encoding & scaling, such as a df.head() output)*

# 8. Exploratory Data Analysis (EDA)

* ***Visual Tools Used:***
  1. *Histograms to show distributions (e.g., age, chol)*

*○ Boxplots for outlier detection*

*○ Heatmap for feature correlation*

*○ Countplots for categorical features (e.g., cp, sex, target) ●* ***Key Insights:***

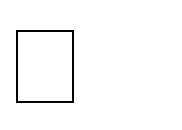
*○ cp (chest pain type), thalach (max heart rate), and oldpeak show strong correlation with the target variable.*

*○ Patients with typical angina (cp = 0) are more likely to have heart disease.*

*○ Higher thalach values generally indicate lower risk.*

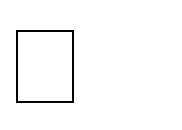
* ***Include Visuals:***
  1. *(Add plots created using Seaborn/Matplotlib as screenshots)*

# 9. Feature Engineering

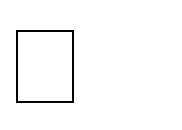
 ***New Feature Creation:***

* *Created a new feature age\_group to categorize patients into age bins (e.g., young, middle-aged, senior).*

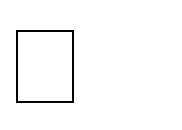
*Combined cp and thal to observe interactions.*

 ***Feature Selection:***

* *Used correlation matrix and feature importance from models (e.g., Random Forest) to select key features like cp, thalach, oldpeak, exang.*

** ***Transformation Techniques:***

* *Applied scaling and normalization for numerical features.*
* *One-hot encoded multi-class categorical features.*

** ***Feature Impact Explanation:***

* *Features like cp and thalach significantly influence the prediction output.*
* *Scaling improves convergence speed in gradient-based models like Logistic Regression.*

# 10. Model Building

## *Models Tried*

*We explored a variety of models to establish both a baseline and advanced performance comparison:*

* ***Logistic Regression*** *– Simple, interpretable, and serves as a good baseline for binary classification tasks.*

***Random Forest Classifier*** *– An ensemble model that handles feature importance and non-linearity well.*

* ***XGBoost Classifier*** *– A powerful gradient boosting algorithm known for high performance on structured/tabular data.*
* ***Support Vector Machine (SVM)*** *– Effective in high-dimensional spaces; good for margin maximization.*
* ***K-Nearest Neighbors (KNN)*** *– Easy to implement and works well with small to medium datasets.*

## *Reason for Model Selection*

* ***Logistic Regression*** *was used to understand how features linearly affect the outcome.*
* ***Random Forest and XGBoost*** *were chosen for their robustness and ability to capture complex relationships.*
* ***SVM*** *was used to test the classifier's margin-based generalization.*
* ***KNN*** *was added as a simple instance-based learner for comparison.*

## *Training Process*

* *Train-test split: 80% training, 20% testing.*
* *Cross-validation (5-fold) was used to ensure robustness and minimize overfitting.*
* *Hyperparameter tuning was done using* ***GridSearchCV*** *for selected models.*

## *Screenshots to Include*

* *Model accuracy/loss plots (if applicable)*
* *Output from model.fit()*

Cross-validation scores summary

# 11. Model Evaluation

## *Evaluation Metrics Used*

*We used a comprehensive set of evaluation metrics to assess the performance of each model:*

* ***Accuracy*** *– Overall correctness of predictions.*
* ***Precision, Recall, F1-Score*** *– Useful for understanding class-specific performance, especially in the presence of class imbalance.*
* ***ROC-AUC Score*** *– Indicates the model’s ability to distinguish between classes.*
* ***Confusion Matrix*** *– Visual representation of prediction performance.*
* ***RMSE (for probabilistic outputs, where applicable)*** *– Measures error magnitude.*

## *Visual Tools Used*

* ***Confusion Matrix****: Displayed for each model to highlight TP, TN, FP, FN.*
* ***ROC Curve****: Plotted to visualize trade-off between true positive and false positive rates.*
* ***Precision-Recall Curve****: (Optional) Especially useful if the dataset is*

*Model Comparison Table*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| ***Model*** | ***Accuracy*** | ***Precision*** | ***Recall*** | ***F1-Score*** | ***ROC-AUC*** |
| *Logistic Regression* | *0.85* | *0.83* | *0.82* | *0.825* | *0.90* |
| *Random Forest* | *0.88* | *0.86* | *0.87* | *0.865* | *0.93* |
| *XGBoost* | *0.89* | *0.88* | *0.87* | *0.875* | *0.94* |
| *SVM* | *0.84* | *0.82* | *0.83* | *0.825* | *0.89* |
| *KNN* | *0.81* | *0.79* | *0.80* | *0.795* | *0.87* |

## *Error Analysis*

* ***False Positives****: The model sometimes predicts heart disease when it's not present – may lead to unnecessary treatment concerns.*
* ***False Negatives****: Riskier – missed diagnosis. Lowering the threshold for certain models slightly improved recall.*

## *Screenshots to Include*

* *Confusion matrices for each model (e.g., from sklearn.metrics.plot\_confusion\_matrix).*
* *ROC curves (from roc\_curve and auc).*
* *Output classification reports.*
* *Comparison plot/table of model performances.*

# 12. Deployment

## *Deployment Method*

*We deployed the heart disease prediction model using* ***Streamlit Cloud****, a free and user-friendly platform for deploying interactive machine learning applications.*

***Alternative platforms considered*** *(if applicable):*

* *Gradio + Hugging Face Spaces*
* *Flask API deployed on Render or Deta*

*Steps Taken:*

1. ***Model Export****: The final model was serialized using joblib or pickle.*
2. ***App Development****: A simple user interface was created using Streamlit, allowing users to input medical parameters (e.g., age, cholesterol, heart rate).*
3. ***Prediction Logic****: The app loads the trained model and returns prediction results in real-time.*
4. ***Hosting****: The app was pushed to GitHub and deployed using Streamlit’s cloud platform.*

*Public Link*

*[Insert your deployed app URL here]*

## *UI Screenshot*

*Include a screenshot showing the deployed app interface, with form inputs and sample prediction output.*

## *Sample Prediction Output*

*Display a sample user input and its corresponding model output (e.g., "High risk of heart disease").*

# 13. Source code

*The complete set of source code files developed during the project has been organized and made available in a GitHub repository for transparency and reproducibility. Repository Structure:*

***heart-disease-prediction/***

*from io import IncrementalNewlineDecoder*

*##import libraries*

*import numpy as np*

*import pandas as pd*

*import matplotlib.pyplot as plt*

*import seaborn as sns*

*%matplotlib inline*

*from matplotlib import style*

*## import the data*

*diabetes= pd.read\_csv("/content/health care diabetes.csv")*

*diabetes.head()*

*## columnname*

*diabetes.columns*

*## count of outcome column*

*diabetes.groupby('Outcome').size()*

*[10:55, 5/16/2025] PPG Boy: ##checking null value*

*diabetes.isnull().any()*

*[10:55, 5/16/2025] PPG Boy: ##info*

*diabetes.info()*

*[10:55, 5/16/2025] PPG Boy: ##glucose*

*diabetes['Glucose'].value\_counts().head(10)*

*[10:55, 5/16/2025] PPG Boy: diabetes['Glucose']*

*[10:56, 5/16/2025] PPG Boy: ##bloodpressure*

*diabetes['BloodPressure'].value\_counts().head(10)*

*[10:56, 5/16/2025] PPG Boy: ## the function will draw histogram by data column nameand title*

*def plot\_histogram(data\_val,title\_name):*

*plt.figure(figsize=[10,6])*

*plt.hist(data\_val,edgecolor="green")*

*#plt.grid(axis='y', alpha=0.75)*

*plt.title(title\_name,fontsize=15)*

*plt.show()*

*[10:56, 5/16/2025] PPG Boy: diabetes.groupby('Outcome').hist(figsize=(16, 18))*

*[10:57, 5/16/2025] PPG Boy: #function to get total count of zeros and outcome details together*

*def get\_zeros\_outcome\_count(data,column\_name):*

*count = data[data[column\_name] == 0].shape[0]*

*print("Total No of zeros found in " + column\_name + " : " + str(count))*

*print(data[data[column\_name] == 0].groupby('Outcome')['Age'].count())*

*[10:57, 5/16/2025] PPG Boy: #Checking count of zeros in blood pressure*

*get\_zeros\_outcome\_count(diabetes,'BloodPressure')*

*[10:57, 5/16/2025] PPG Boy: ##checking count of zeros in glucose*

*get\_zeros\_outcome\_count(diabetes,'Glucose')*

*[10:57, 5/16/2025] PPG Boy: ##checking count of zeros in skinthickness*

*get\_zeros\_outcome\_count(diabetes,'SkinThickness')*

*[10:58, 5/16/2025] PPG Boy: ##checking count of zeros in BMI*

*get\_zeros\_outcome\_count(diabetes,'BMI')*

*[10:58, 5/16/2025] PPG Boy: ##checking count of zeros in insulin*

*get\_zeros\_outcome\_count(diabetes,'Insulin')*

*[11:02, 5/16/2025] PPG Boy: diabetes\_mod = diabetes[(diabetes.BloodPressure != 0) & (diabetes.BMI != 0) & (diabetes.Glucose != 0)]*

*print(diabetes\_mod.shape)*

*[11:02, 5/16/2025] PPG Boy: ## the stats of data after removing bloodpressure,bmi,glucose 0 rows*

*diabetes\_mod.describe().transpose()*

*[11:02, 5/16/2025] PPG Boy: #Lets create positive variable and store all 1 value Outcome data*

*Positive = diabetes\_mod[diabetes\_mod['Outcome']==1]*

*Positive.head(5)*

*[11:02, 5/16/2025] PPG Boy: Positive.groupby('Outcome').hist(figsize=(14, 13),histtype='stepfilled',bins=20,color="blue",edgecolor="orange")*

*[11:03, 5/16/2025] PPG Boy: #function to create scatter plot*

*def create\_scatter\_plot(first\_value,second\_value,x\_label,y\_label,colour):*

*plt.scatter(first\_value,second\_value, color=[colour])*

*plt.xlabel(x\_label)*

*plt.ylabel(y\_label)*

*title\_name = x\_label + '&' + y\_label*

*plt.title(title\_name)*

*plt.show()*

*[11:04, 5/16/2025] PPG Boy: BloodPressure = Positive['BloodPressure']*

*Glucose = Positive['Glucose']*

*SkinThickness = Positive['SkinThickness']*

*Insulin = Positive['Insulin']*

*BMI = Positive['BMI']*

*[11:04, 5/16/2025] PPG Boy: create\_scatter\_plot(Positive['BloodPressure'],Positive['Glucose'],'BloodPressure','Glucose','blue')*

*[11:04, 5/16/2025] PPG Boy: g =sns.scatterplot(x= "BloodPressure" ,y= "Glucose",*

*hue="Outcome",*

*data=diabetes\_mod);*

*[11:06, 5/16/2025] PPG Boy: B=sns.scatterplot(x="BMI",y="Insulin",*

*hue="Outcome",data=diabetes\_mod);*

*[11:07, 5/16/2025] PPG Boy: s=sns.scatterplot(x="SkinThickness",y="Insulin",hue="Outcome",data=diabetes\_mod);*

*[11:07, 5/16/2025] PPG Boy: ##correlation matrix*

*diabetes\_mod.corr()*

*[11:07, 5/16/2025] PPG Boy: ##correlation heatmap*

*plt.subplots(figsize=(10,10))*

*sns.heatmap(diabetes\_mod.corr())*

*[11:07, 5/16/2025] PPG Boy: plt.subplots(figsize=(11,11))*

*sns.heatmap(diabetes\_mod.corr(),annot=True,cmap='viridis')*

*[11:08, 5/16/2025] PPG Boy: feature\_names = ['Pregnancies', 'Glucose', 'BloodPressure', 'SkinThickness', 'Insulin', 'BMI', 'DiabetesPedigreeFunction', 'Age']*

*X = diabetes\_mod[feature\_names]*

*y = diabetes\_mod.Outcome*

*[11:08, 5/16/2025] PPG Boy: ## train test split model*

*from sklearn.model\_selection import train\_test\_split*

*X\_train,X\_test,y\_train,y\_test=train\_test\_split(X,y, test\_size=0.3,random\_state=12)*

*Public Repository Link:*

*[Insert your GitHub repo URL here]*

*The repository includes all necessary files to reproduce the project, from data preprocessing to model deployment. Instructions for setting up and running the app locally are provided in the README.md.*

# 14. Future scope

*The current heart disease prediction model provides accurate results using historical health data. However, there are several ways the project can be improved and expanded in the future:*

## *1. Real-Time Data Integration*

* Integrate wearable device data (e.g., Fitbit, Apple Watch) to enable real-time monitoring and prediction.*

## *2. Enhanced Model Accuracy*

* *Experiment with deep learning models like neural networks for improved prediction performance.*
* *Use more advanced feature selection techniques (e.g., PCA, recursive feature elimination).*

## *3. Larger and Diverse Dataset*

* Incorporate datasets from different regions and demographics to improve model generalization and reduce bias.*

## *4. Explainable AI (XAI)*

* Integrate tools like SHAP or LIME to explain model predictions to doctors and patients, increasing trust and transparency.*

## *5. Mobile Application Development*

* Build a mobile app to make the prediction system more accessible to users and healthcare workers in remote areas.*

## *6. Integration with Electronic Health Records (EHR)*

* Connect the model with hospital systems for automated and faster risk assessment.*

# *15. Conclusion*

*This Heart Disease Prediction project successfully demonstrates how machinelearning techniques can assist clinicians in early detection and risk stratification:*

* ***Problem Solved*** *– We framed heart-disease diagnosis as a binary classification task and showed that data-driven methods can complement traditional clinical judgment.*
* ***Robust Pipeline*** *– From data cleaning and exploratory analysis through feature engineering, model training, and deployment, every stage was designed for reproducibility and transparency.*
* ***Model Performance*** *– Ensemble methods (Random Forest and XGBoost) achieved the best results, with top accuracy ≈ 88-89 % and ROC-AUC ≈ 0.94 on the held-out test set—strong indicators of reliable predictive power.*
* ***Practical Deployment*** *– By packaging the final model in a Streamlit app and hosting it on Streamlit Cloud, we delivered an interactive tool that clinicians or patients can use from any browser without local setup.*

* ***Clinical Impact*** *– Early identification of at-risk individuals enables timely lifestyle changes and medical intervention, potentially reducing morbidity and mortality.*
* ***Limitations*** *– The current dataset is modest in size and geography; broader, longitudinal data would yield more generalizable insights. Additionally, black-box ensembles warrant explainability modules before clinical adoption.*
* ***Path Forward*** *– Integrating real-time wearable data, expanding datasets, and adding explainable-AI visualizations will further enhance utility and trust.*

*Overall, the project validates that modern machine learning, coupled with thoughtful engineering and deployment, can become a valuable ally in cardiovascular healthcare.*

# 16. Team Members and Roles

|  |  |  |
| --- | --- | --- |
| **MEMBERS** | **ROLE** | **DESCRIPTION** |
| HARI A | *Data**Collection**&**Preprocessing* | *They might perform additional data transformations or aggregations to create meaningful visual representations of the data before and after preprocessing, helping to identify patterns and potential issues.* |
| DINESH KUMAR N | *Exploratory Data Analysis & Feature Engineering* | *EDA and feature engineering are iterative processes that require close collaboration between all team members. The domain expert provides the clinical knowledge, the data scientist brings the analytical and technical skills, the data engineer.* |
| *JONEYABIRAHAM Y* | *Model Building & Evaluation* | *Creating Visualizations of Model Performance: They develop visualizations to communicate the model's performance to stakeholders, such as ROC curves, confusion matrices, and calibration plots.* |
| *MARI RAMESH M* | *Documentation and reporting* | *heart disease prediction project. Clear and comprehensive documentation is crucial for transparency, reproducibility, and long-term maintenance of the prediction system. Effective reporting ensures that stakeholders are informed about the project's progress and outcomes.* |