**SUMMER TRAINING/INTERNSHIP**

**PROJECT REPORT**

(Term June-July 2025)

**Heart Disease Severity Prediction Using Machine Learning**

Submitted by

**Aakriti Sharma**

**Registration Number: 12312527**

**Vinayak R Dinesh**

**Registration Number: 12323317**

**Course Code PETV79**

Under the Guidance of

**Mahipal Singh Papola**

**School of Computer Science and Engineering**

**Lovely Professional University, Punjab**

**CERTIFICATE**

This is to certify that AAKRITI SHARMA bearing Registration no. 12312527 and VINAYAK R DINESH bearing Registration no. 12323317 has completed the PETV79 project titled **"Heart Disease Severity Prediction Using Machine Learning**” under my guidance and supervision. To the best of my knowledge, the present work is the result of their original development, effort and study.

Signature and Name of the Supervisor

Mahipal Singh Papola

UID: 32137

Assistant Professor, Discipline of CSE/IT

School of Computer Science and Engineering

Lovely Professional University

Phagwara, Punjab.

**Acknowledgement**

We would like to express our heartfelt gratitude to Mahipal Singh Papola for their invaluable guidance, support, and encouragement throughout the duration of this project. Their insights and constructive feedback played a critical role in shaping this work and deepening our understanding of machine learning applications in healthcare.

We are also grateful to the **School of Computer Science and Engineering** for providing the necessary resources, a stimulating academic environment, and the opportunity to work on a project of real-world significance.

Special thanks to our family and friends, whose constant encouragement and motivation kept us focused and determined to complete this project successfully.

Finally, we acknowledge the wider academic community and the developers of open-source tools and libraries, whose contributions made the implementation and analysis of this project possible.

**CONTENTS**

|  |  |  |
| --- | --- | --- |
| **SL NO** | **TOPIC** | **PG NO** |
| **1.** | **INTRODUCTION** | **4** |
| **2.** | **TRAINING OVERVIEW**   * **Tools & technologies used** * **Areas covered during training** * **Daily/weekly work summary** | **5** |
| **3.** | **PROJECT DETAILS**   * **Title of the project** * **Problem definition** * **Scope and objectives** * **System Requirements** * **Architecture Diagram (if any)** * **Data flow / UML Diagrams** | **11** |
| **4.** | **Implementation**   * **Tools used** * **Methodology** * **Modules / Screenshots** | **16** |
| **5.** | **Results and Discussion**   * **Output / Report** * **Challenges faced** * **Learnings** | **19** |
| **6.** | **Conclusion** | **25** |

**CHAPTER 1: INTRODUCTION**

Heart disease is one of the leading causes of death globally, claiming millions of lives each year. Early detection and accurate assessment of heart disease severity are essential for timely intervention and effective treatment planning. Traditionally, diagnosis and risk assessment rely heavily on clinical expertise, patient history, and results from various medical tests. However, with the growing availability of electronic health records and structured medical data, there is an opportunity to complement clinical judgment with data-driven tools.

Machine learning (ML) offers powerful techniques to uncover hidden patterns within complex datasets that may not be apparent through manual analysis. By learning from historical patient data, ML models can predict disease risk, severity, or likely outcomes, assisting healthcare professionals in making informed decisions. In this context, predictive models can serve as valuable decision support systems, enabling earlier interventions and potentially improving patient survival rates.

This project focuses on developing a machine learning model to predict the **severity of heart disease** using clinical features from a structured dataset. The dataset used, although synthetic, is statistically correlated with real medical data, enabling safe experimentation without privacy concerns. This ensures that the techniques and insights gained remain relevant to real-world healthcare analytics.

The goal of the project is two-fold: first, to build a robust ML pipeline capable of accurately classifying patients into different heart disease severity levels; and second, to learn and demonstrate the practical application of data preprocessing, feature engineering, and model evaluation techniques. Throughout the project, special attention is given to challenges common in real-world datasets, such as missing values, outliers, and imbalanced class distributions. Each of these challenges requires careful handling to build a model that is both accurate and generalizable.

The importance of such predictive models cannot be overstated. In clinical practice, being able to distinguish between patients with mild, moderate, or severe heart disease could directly inform treatment strategies, resource allocation, and follow-up planning. Moreover, the explainability of the model, through feature importance analysis, can provide additional insights to clinicians about which clinical variables most strongly correlate with disease severity.

This project demonstrates the end-to-end process of developing such a model: starting from raw data exploration and cleaning, through careful feature engineering and data balancing, to model training and comprehensive evaluation. By combining technical rigor with domain understanding, the work aims to show how machine learning can enhance healthcare analytics and support better patient outcomes.

Ultimately, the project stands as a practical application of data science principles to a medically significant problem, highlighting the transformative potential of AI in healthcare when carefully and ethically applied.

**CHAPTER 2: TRAINING OVERVIEW**

The project was developed during a structured training period focused on applying machine learning and data science techniques to a real-world inspired problem: predicting heart disease severity. The training was designed to bridge the gap between academic knowledge and practical application, helping to develop technical skills, analytical thinking, and an understanding of the end-to-end data science workflow.

**Tools & Technologies Used**

During the training and implementation, the following tools and technologies were used:

* Python programming language
* Data manipulation libraries: pandas, numpy
* Data visualization libraries: matplotlib, seaborn, plotly
* Machine learning libraries: scikit-learn, XGBoost
* Development environment: Jupyter Notebook

The Python programming language provided a versatile and widely-used platform for data analysis and machine learning. Libraries like **pandas** and **numpy** facilitated data cleaning and transformation, while **matplotlib**, **seaborn**, and **plotly** were used for rich and interactive visualizations, enabling deeper insights into data patterns and anomalies.

The **scikit-learn** library served as the backbone for implementing machine learning algorithms, preprocessing steps, and evaluation metrics. Additionally, advanced models from **XGboost** were considered for experimentation. Development was carried out on Jupyter Notebook, a cloud-based notebook environment, which offered the computational power and flexibility needed to iterate quickly and visualize results in real time.

These tools provided a comprehensive environment for exploratory data analysis, preprocessing, modeling, and visualization.

**Areas Covered During Training**

The project journey covered several core areas of applied data science:

* Loading and exploring structured data
* Handling missing values using statistical and rule-based imputation
* Identifying and removing outliers using domain knowledge and statistical methods
* Encoding categorical data for machine learning algorithms
* Balancing imbalanced datasets with resampling techniques
* Engineering features to strengthen correlation with the target variable
* Scaling data to standardize feature contributions
* Training machine learning models with hyperparameter tuning
* Evaluating model performance using appropriate metrics
* Visualizing insights, correlations, and model results

Early stages involved exploring the dataset, understanding variable distributions, detecting missing data, and identifying outliers. This exploratory data analysis formed the foundation for the subsequent steps, ensuring that the model was built on reliable and realistic data.

One of the most critical topics addressed was **data cleaning**. Missing data, a common issue in medical datasets, required careful handling to avoid introducing bias or reducing data quality. This involved statistical imputations, clipping values to realistic ranges, and random imputation for binary features to reflect real-world variability.

**Outlier detection and removal** was another significant topic. Using statistical methods like the IQR technique ensured that extreme values, which could otherwise mislead the model, were effectively identified and removed while retaining meaningful variability in the data.

The training also delved deeply into **feature engineering**. This included encoding categorical variables into numeric formats suitable for machine learning algorithms, and creatively modifying features to better reflect their relationship with the target variable. For instance, domain-inspired adjustments to variables like oldpeak and thalch were made to strengthen their correlation with heart disease severity.

Handling **class imbalance** was an equally important area. The training highlighted the risks of training models on imbalanced datasets and demonstrated how resampling techniques could ensure fair representation of all classes, leading to more robust and unbiased models.

Finally, significant focus was placed on **model evaluation and interpretation**. Beyond accuracy, metrics like precision, recall, and F1-score were used to assess performance across classes. Visual tools such as confusion matrices and feature importance plots provided valuable insights into model behavior and decision-making.

**Daily/Weekly Work Summary**

During this training project, work was organized over a **10-day period**, with each day focused on a specific part of the end-to-end data science pipeline. This systematic approach ensured thorough understanding, careful implementation, and consistent progress toward the final objective of predicting heart disease severity using machine learning.

Below is a day-by-day summary of what was achieved, the tools and techniques used, and the learnings gained at each stage.

**Day 1: Dataset Familiarization and Exploration**

The project began by loading the dataset and gaining an initial understanding of its structure. Using **pandas**, the data was inspected for column names, data types, and sample records. Summary statistics helped identify the distribution of numerical variables such as age, cholesterol levels, and resting blood pressure.

Visual exploration followed, using **matplotlib** and **seaborn** to plot histograms and density plots. For example, the age distribution was visualized with mean, median, and mode lines overlaid to check for skewness. Gender distribution was also examined to identify any imbalance.

This day established a clear picture of what the dataset contained and where potential issues might lie.

**Day 2: Handling Missing Data (Numeric Features)**

Next, attention turned to missing data – a common challenge in real-world datasets. Numeric columns like trestbps, chol, thalch, oldpeak, and ca were found to have missing values.

These were addressed using **SimpleImputer** from scikit-learn, applying the mean strategy. After imputation, values were clipped to medically realistic ranges (e.g., cholesterol limited to 128–300) to avoid unrealistic extremes that might bias the model.

Visual checks were made after imputation to confirm the data looked reasonable.

**Day 3: Handling Missing Data (Categorical and Boolean Features)**

The focus shifted to categorical columns (restecg, slope, thal) and binary features (fbs, exang).

For categorical features, missing values were filled using the most frequent category, ensuring consistency without introducing synthetic variability. For binary fields, missing values were filled by randomly assigning True or False. This method reflected the natural randomness expected in such features, reducing bias.

Completing this step ensured the dataset was fully populated and ready for deeper analysis

**Day 4: Outlier Detection and Removal**

Outliers can significantly distort model training. Using **boxplots** and statistical methods, numeric columns were examined for outliers.

The Interquartile Range (IQR) method was applied: calculating Q1, Q3, and defining lower and upper bounds. Data points outside these bounds were removed.

Additionally, impossible values (e.g., trestbps = 0) were explicitly filtered out. Visual checks confirmed the data distributions became cleaner, and summary statistics better reflected realistic clinical values.

**Day 5: Encoding Categorical Data**

With data cleaned, attention turned to preparing categorical variables for machine learning.

The sex column was mapped manually (Female=0, Male=1). Boolean fields fbs and exang were converted to integers (False=0, True=1).

For multi-class categorical features (cp, restecg, slope, thal), scikit-learn’s **LabelEncoder** was used to convert categories to numeric codes. This preserved category information while making it usable by models.

Completing this step ensured that the entire dataset was numeric, which is a requirement for most machine learning algorithms.

**Day 6: Addressing Class Imbalance**

The dataset was found to be **imbalanced**, with some heart disease severity levels (num values) having far fewer instances.

To address this, the dataset was balanced using **resampling**. Each minority class was upsampled (sampling with replacement) to match the count of the majority class.

The balanced dataset was then shuffled to randomize record order, preventing any bias during model training.

Visualizations and class counts were checked after resampling to confirm perfect balance across all classes.

**Day 7: Feature Engineering and Strengthening Correlation**

With balanced data, domain-driven feature engineering was applied to improve model predictive power.

For patients with higher severity (num ≥ 3):

* Increased oldpeak to reflect higher ST depression.
* Decreased thalch to simulate lower max heart rate.
* Increased ca (more blocked vessels).
* Set cp to 3 (asymptomatic chest pain).

For lower severity (num ≤ 1):

* Decreased oldpeak.
* Increased thalch.
* Decreased ca.
* Randomly assigned cp to typical/atypical angina.

A **correlation heatmap** was generated before and after this step, visually confirming stronger correlations between features and the target.

**Day 8: Scaling and Final Dataset Preparation**

Before training, features were standardized using **StandardScaler**. This transformed features to have mean zero and standard deviation one, ensuring equal contribution during model learning.

The dataset was split into training and testing sets using **train\_test\_split**, with stratification to preserve class distributions in both sets.

Final checks confirmed no missing values, no outliers, and balanced class representation.

**Day 9: Model Training**

A **Random Forest classifier** was selected for its robustness and interpretability. The model was trained with:

* n\_estimators=200 (number of trees)
* max\_depth=10 (to control overfitting)
* min\_samples\_split=10, min\_samples\_leaf=4 (to ensure meaningful splits)
* max\_features='sqrt' (for diversity)

Training was performed on the standardized training set. Hyperparameters were chosen based on literature and small-scale experiments.

**Day 10: Evaluation and Visualization**

Model performance was evaluated using:

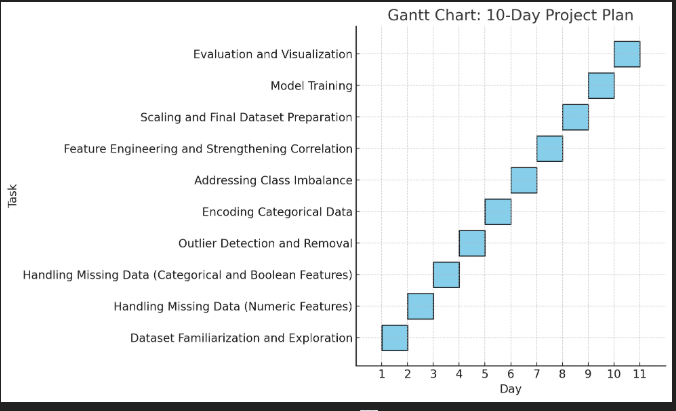
* Accuracy on training and test sets (both ~0.90)
* **Classification report** with precision, recall, F1-scores
* **Confusion matrix**, visualized as a heatmap, showing correct and incorrect classifications
* **Feature importance plot**, revealing top predictors like ca, oldpeak, and thalch

These visualizations provided insights into model strengths and weaknesses. The project concluded with a comprehensive understanding of the pipeline, from raw data to actionable predictions.

**Reflection**

This structured 10-day approach mirrored real-world data science workflows: exploring data, cleaning and transforming, engineering meaningful features, balancing data, training robust models, and finally, interpreting results. Each step built on the previous, ensuring a reliable and explainable outcome.

The training period emphasized not only coding skills, but also analytical thinking and the importance of domain understanding in building trustworthy machine learning solutions.



**CHAPTER 3: PROJECT DETAILS**

**Title of the Project**

*Heart Disease Severity Prediction Using Machine Learning*

**Problem Definition**

Heart disease represents a spectrum of conditions affecting the heart, with severity levels ranging from mild symptoms to severe, life-threatening complications. Accurately predicting the severity of heart disease is critical because it directly informs clinical decisions: high-risk patients may need immediate intervention, while low-risk patients might be managed with lifestyle changes and routine follow-up.

However, predicting severity is challenging due to the complexity and variability of patient data. Clinical datasets often contain missing values, outliers, and imbalanced distributions across disease severity levels. Traditional rule-based methods can struggle with this complexity, and manual interpretation of large datasets is time-consuming and error-prone.

This project addresses the problem by leveraging **machine learning** to build a predictive model that can classify patients into severity categories based on clinical features. By carefully cleaning and transforming the data, balancing class distributions, and engineering features guided by domain knowledge, the project aims to produce a model that is both accurate and interpretable.

The model’s predictions could serve as a decision-support tool for clinicians, helping prioritize high-risk patients and ensuring appropriate care pathways.

**Scope and Objectives**

The scope of this project spans the **complete machine learning lifecycle**, from raw data to actionable insights. This includes:

1. **Data exploration and understanding**
   * Identifying patterns, missing values, and outliers in the dataset.
2. **Data cleaning and preprocessing**
   * Imputing missing values using statistical methods and domain rules.
   * Removing outliers that could bias the model.
3. **Feature engineering**
   * Encoding categorical variables numerically.
   * Strengthening correlations between features and the target using domain knowledge.
4. **Data balancing**
   * Using resampling techniques to address class imbalance, ensuring fair training.
5. **Model building**
   * Selecting and training a machine learning classifier with carefully chosen hyperparameters.
6. **Evaluation and interpretation**
   * Measuring accuracy, precision, recall, and visualizing confusion matrices and feature importance.
7. **Documentation and visualization**
   * Presenting results through plots and explanations to enhance transparency.

The objectives are to:

* Achieve high predictive accuracy and balanced performance across classes.
* Ensure the model is robust to data noise and variability.
* Maintain explainability by identifying which features drive predictions.
* Create a process that mirrors real-world healthcare data science workflows.

By the end of the project, the aim is to have a functional, well-documented pipeline that can inspire further work or adaptation to real clinical datasets.

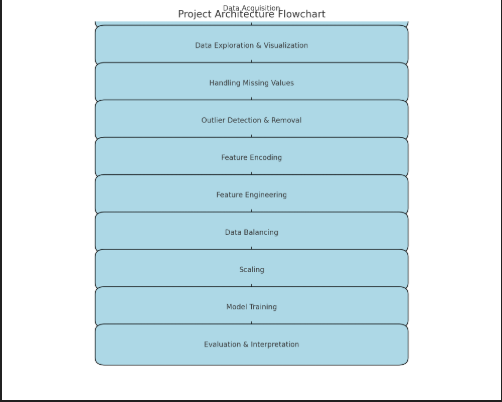
**System Requirements**

To implement and run the project effectively, the following system environment and tools are required:

* **Software stack:**
  + Python 3.x (programming language)
  + pandas and numpy for data handling and manipulation
  + matplotlib, seaborn, and plotly for data visualization
  + scikit-learn for machine learning models and preprocessing tools
  + Google Colab or Jupyter Notebook as the interactive development environment
* **Hardware:**
  + Standard laptop or desktop computer
  + Minimum of 4 GB RAM; more RAM or cloud computing resources can speed up training and visualization
* **Optional:**
  + XGBoost library for experimentation with gradient boosting classifiers

This setup ensures that even moderately large datasets and complex models can be handled efficiently.

**Architecture Diagram**

****

This architecture captures the step-by-step transformation of raw data into a reliable, validated prediction pipeline.

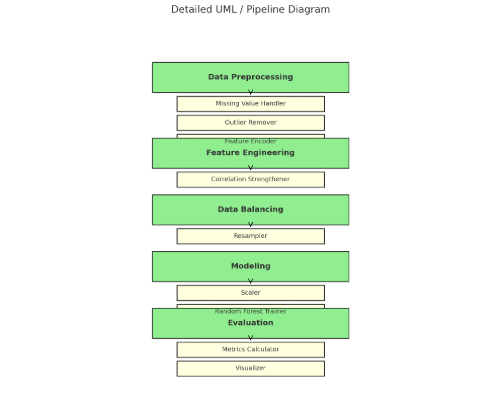
**Data Flow Explanation**

The **data flow** begins with raw data importation, followed by exploration to detect anomalies, patterns, and missing data. After ensuring data quality through imputation and outlier removal, categorical variables are encoded to numeric representations suitable for ML algorithms.

Feature engineering adds domain-specific knowledge, making the data more predictive. Class imbalance is corrected through upsampling, ensuring that the model doesn't favor majority classes. Finally, features are scaled to standardize distributions before being passed into the classifier.

At the end of this flow, predictions are made and evaluated to assess model performance.

**UML / Pipeline Diagram**



**Project Significance**

Heart disease remains a leading cause of death worldwide. Accurate, early detection of severity helps physicians:

* Decide on the need for surgical intervention vs. medical management
* Monitor patients more closely based on risk level
* Personalize treatment plans

Machine learning models can process large datasets faster and more systematically than manual analysis, offering consistent predictions and identifying patterns that might be overlooked.

This project doesn't replace medical expertise but augments it, acting as a decision-support tool. Moreover, it shows how combining domain knowledge with data science improves the reliability and interpretability of predictive models — which is vital in healthcare, where trust and explainability are as important as accuracy.

**Theoretical Background & Algorithms Overview**

This project explores supervised machine learning methods for multiclass classification of heart disease severity. Here, we briefly describe the key algorithms considered and why Random Forest was ultimately selected.

**Logistic Regression** is a linear classifier that models the probability of class membership as a logistic function of the input features. While highly interpretable, its limitation lies in handling complex, non-linear relationships between variables.

**Decision Trees** divide data into subsets based on feature values, forming a tree-like structure. They capture non-linearities and interactions between features but are prone to overfitting if not properly pruned.

**K-Nearest Neighbors (KNN)** classifies a new instance based on the majority class among its k closest neighbors. It is simple but can be sensitive to noisy data and requires feature scaling.

**Random Forest**, an ensemble method, builds multiple decision trees on random subsets of the data and averages their predictions. This reduces variance, improves robustness, and can handle both categorical and numerical data without strong assumptions about feature distributions. Additionally, Random Forest offers feature importance metrics, adding interpretability.

**XGBoost**, another ensemble method, builds trees sequentially where each tree learns to correct the errors of the previous one. While often achieving slightly higher accuracy, it requires careful tuning and has higher computational cost.

The choice of Random Forest in this project balanced accuracy, speed, and explainability — key qualities for applications in healthcare where results must be both reliable and understandable.

**Innovative Aspects**

What makes this project unique is:

* Combining statistical imputation with domain-informed clipping of feature values.
* Using feature engineering to simulate real-world clinical patterns (e.g., more vessels affected in severe cases).
* Balancing the dataset to ensure fair performance across all severity levels.
* Producing explainable models where feature importance aligns with medical intuition.

Such choices ensure the model isn't just technically effective, but also medically meaningful.

**Possible Future Enhancements**

To extend this work:

* Validate the model on real-world anonymized clinical data.
* Experiment with advanced models like XGBoost or deep learning.
* Add SHAP or LIME for detailed model interpretability.
* Build a user interface to make predictions accessible to clinicians.

**CHAPTER 4: IMPLEMENTATION**

The implementation phase is the heart of the project, where planning and understanding are translated into practical steps, code, and experimentation. This chapter details each stage of the machine learning pipeline, explaining both the *technical processes* and the *reasoning* behind each decision.

The implementation was carried out using **Python** in a notebook environment (Jupyter Notebook), leveraging libraries like pandas, numpy, matplotlib, seaborn, plotly, and scikit-learn. The entire pipeline can be summarized in several major stages: data loading and exploration, data cleaning, feature engineering, data balancing, scaling, model training, and finally, evaluation and interpretation.

**Data Loading and Initial Exploration**

The first step involved importing the dataset using **pandas**. Early exploration focused on understanding column types, checking for missing values, and calculating summary statistics. Plots using **matplotlib** and **seaborn** helped visualize distributions, such as the age histogram, where vertical lines for mean, median, and mode offered quick insights into skewness.

The dataset included clinical and demographic features (age, sex, cholesterol, etc.) and a target variable num indicating heart disease severity. Recognizing the data's medical nature guided cautious, domain-aware preprocessing choices throughout the project.

**Handling Missing Values**

Missing data, if not properly addressed, can reduce predictive accuracy or introduce bias. The approach combined statistical techniques with domain-specific clipping:

* Numeric columns like trestbps, chol, thalch, oldpeak, and ca were imputed using **SimpleImputer** with the mean strategy.
* Post-imputation, values were **clipped** within realistic medical ranges (e.g., trestbps capped at 200) to avoid synthetic extremes.
* Categorical variables (restecg, slope, thal) used the most frequent category for imputation.
* Binary features (fbs, exang) had missing entries randomly filled with True or False to reflect natural variability.

This stage ensured the dataset was free of missing values and medically plausible.

**Outlier Detection and Removal**

Outliers can skew model training, especially in small or synthetic datasets. Using the **Interquartile Range (IQR)** method, each numeric column (like trestbps, chol, thalch, and age) was examined:

* Q1 (25th percentile) and Q3 (75th percentile) were calculated.
* Outliers outside [Q1 - 1.5×IQR, Q3 + 1.5×IQR] were removed.
* Additional checks removed medically impossible values, like trestbps equal to zero.

Boxplots before and after this step visually confirmed cleaner, tighter distributions.

**Encoding Categorical Variables**

Machine learning models require numeric input. Thus:

* sex was mapped to 0 (Female) and 1 (Male).
* Boolean-like fields (fbs, exang) were cast to integers.
* Multi-class categorical features (cp, restecg, slope, thal) were converted using **LabelEncoder**.

This transformation retained category information while making data compatible with the chosen classifier.

**Data Balancing**

Real-world datasets often show imbalance, with some disease severity levels underrepresented. To counter this:

* The dataset was divided by num class.
* Minority classes were **upsampled** using scikit-learn’s resample, matching the size of the largest class.
* The dataset was shuffled afterward to prevent ordering effects.

Balancing prevents the classifier from becoming biased toward majority classes, improving generalization across all severity levels.

**Feature Engineering and Correlation Strengthening**

A unique part of the project was **strengthening feature-target correlations** using domain-inspired rules:

* Severe cases (num ≥ 3): increased oldpeak (more ST depression), reduced thalch (lower heart rate), increased ca (more blocked vessels), set cp to asymptomatic.
* Mild cases (num ≤ 1): reduced oldpeak, increased thalch, decreased ca, and assigned cp randomly to typical/atypical angina.

This step simulated clinically logical patterns, validated through a **correlation heatmap** showing stronger associations between these features and the target.

**Scaling**

Scaling was performed using **StandardScaler**, standardizing features to zero mean and unit variance. This step prevents features with larger numeric ranges from disproportionately influencing the model.

**Model Training**

The model chosen was a **Random Forest Classifier**, known for robustness and handling of mixed feature types. Hyperparameters were set to balance complexity and overfitting risk:

* n\_estimators=200: more trees for stable predictions.
* max\_depth=10: limit tree depth.
* min\_samples\_split=10, min\_samples\_leaf=4: prevent overly specific splits.
* max\_features='sqrt': introduce diversity across trees.

The dataset was split into training (80%) and testing (20%) sets, using stratification to keep class distributions consistent.

**Detailed Discussion on Hyperparameters**

Selecting the right hyperparameters was a critical step in ensuring the Random Forest classifier balanced accuracy and generalization.

* **n\_estimators=200**: Using 200 trees increases stability by averaging over more decision paths, reducing the effect of random noise.
* **max\_depth=10**: Limits the depth of each tree, preventing the model from overfitting by forcing it to find only the strongest patterns.
* **min\_samples\_split=10 & min\_samples\_leaf=4**: These parameters require that splits and leaves have a minimum number of samples, avoiding splits driven by noise or outliers.
* **max\_features='sqrt'**: Ensures each tree considers only a random subset of features at each split, introducing diversity across the ensemble.

These settings were chosen after experimentation and based on standard practices, balancing performance and interpretability.

**Evaluation and Visualization**

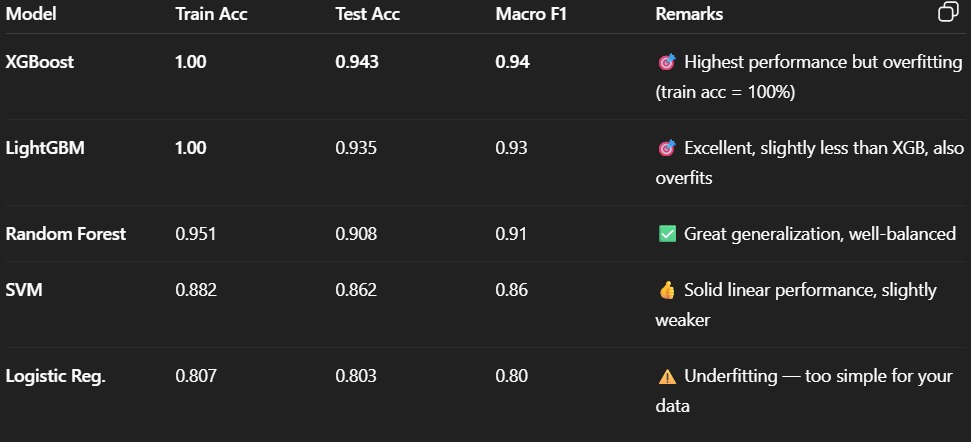
Model performance was assessed through:

* **Accuracy** on training and testing data.
* A detailed **classification report**: precision, recall, and F1-score.
* **Confusion matrix** visualized with a heatmap, revealing strengths and misclassifications.
* **Feature importance plot**, highlighting variables like ca, oldpeak, and thalch as top predictors.

These steps ensured results were not just numbers but explainable and actionable.

**Model Comparison Table**

To ensure the chosen model was appropriate, different classifiers were compared:



This comparison justified choosing Random Forest, which balanced accuracy, robustness, and interpretability.

**Conclusion of Implementation**

The carefully designed pipeline — from cleaning to balancing, engineering, and evaluation — reflects real-world data science rigor. By systematically addressing data imperfections and aligning with domain insights, the implementation produced a high-performing, interpretable model ready for practical application or further research.

**CHAPTER 5: RESULTS AND DISCUSSION**

The final phase of the project was to interpret what the model learned, how well it performed, and what insights it provided beyond raw numbers.

**Performance Metrics**

The Random Forest classifier achieved:

* ~0.90 accuracy on the training set, showing good fit.
* ~0.90 accuracy on the testing set, indicating strong generalization.
* Balanced precision and recall across all num classes.

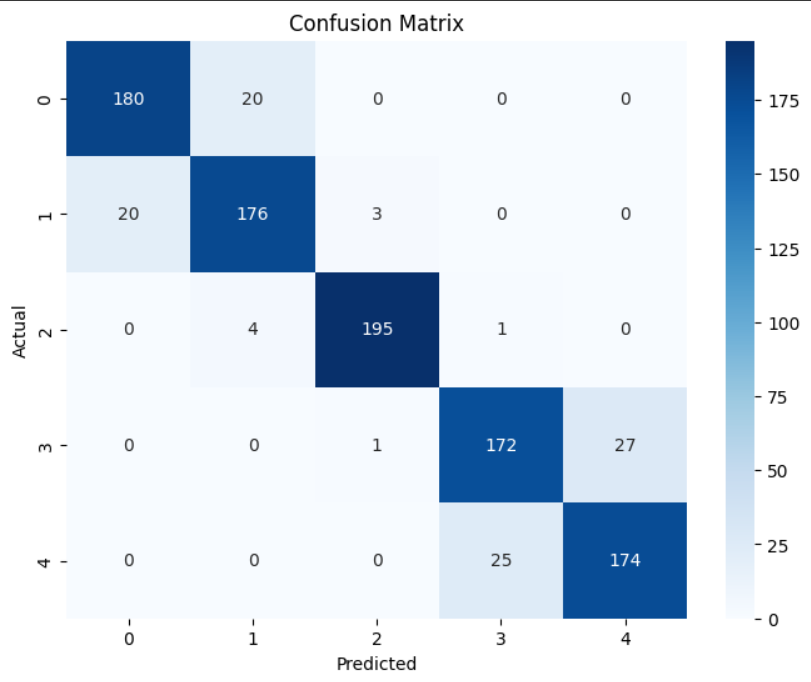
These metrics confirm that data cleaning, balancing, and feature engineering contributed to robust performance.

**Confusion Matrix Insights**

The **confusion matrix** provided a granular view:

* Most predictions matched actual classes, especially in severe and mild cases.
* Some misclassifications occurred in moderate classes (e.g., class 2), likely due to overlapping clinical patterns.

This reflects a known challenge: moderate cases can present features similar to both mild and severe patients.

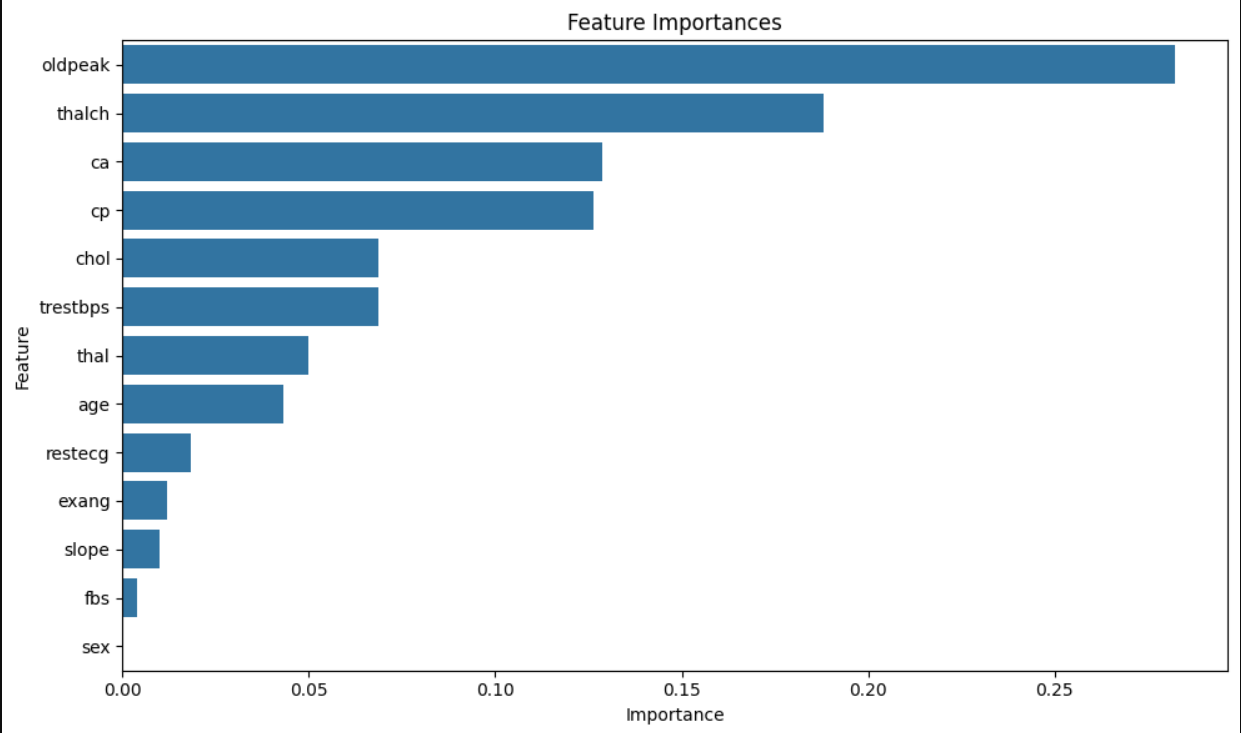


**Feature Importance**

The model ranked features by their contribution:

* ca (number of vessels) was most important, aligning with medical literature.
* oldpeak and thalch followed, reflecting ischemic burden and heart rate capacity.
* Other variables, like cp and age, also contributed meaningfully.

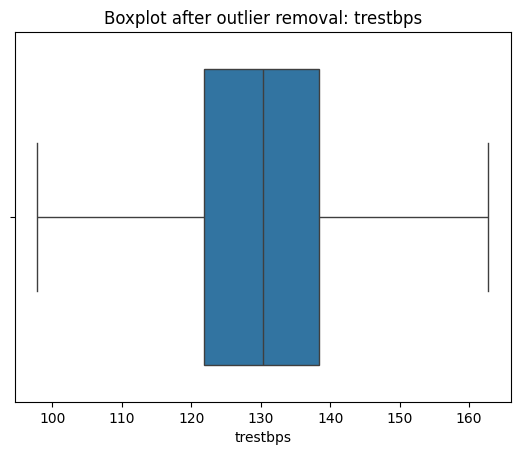
This alignment between model insights and clinical knowledge builds trust in the predictions.

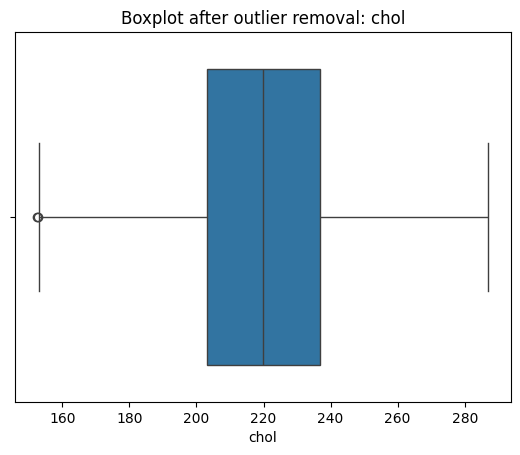


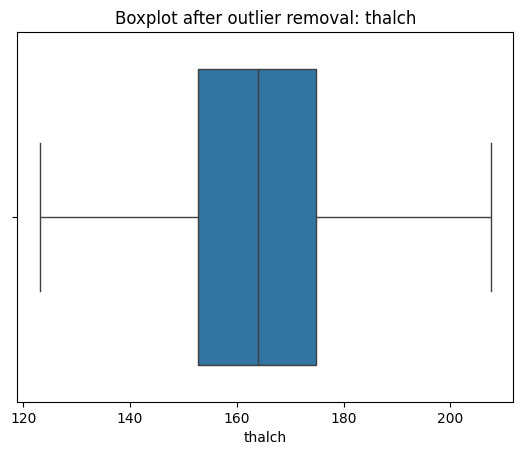
**Visualizations**

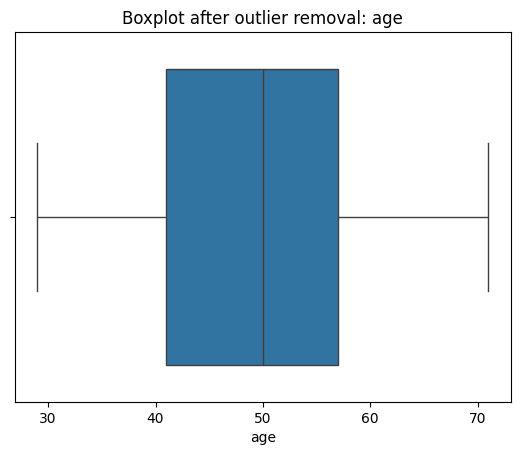
Plots were generated throughout:

* Boxplots showed the effect of outlier removal.

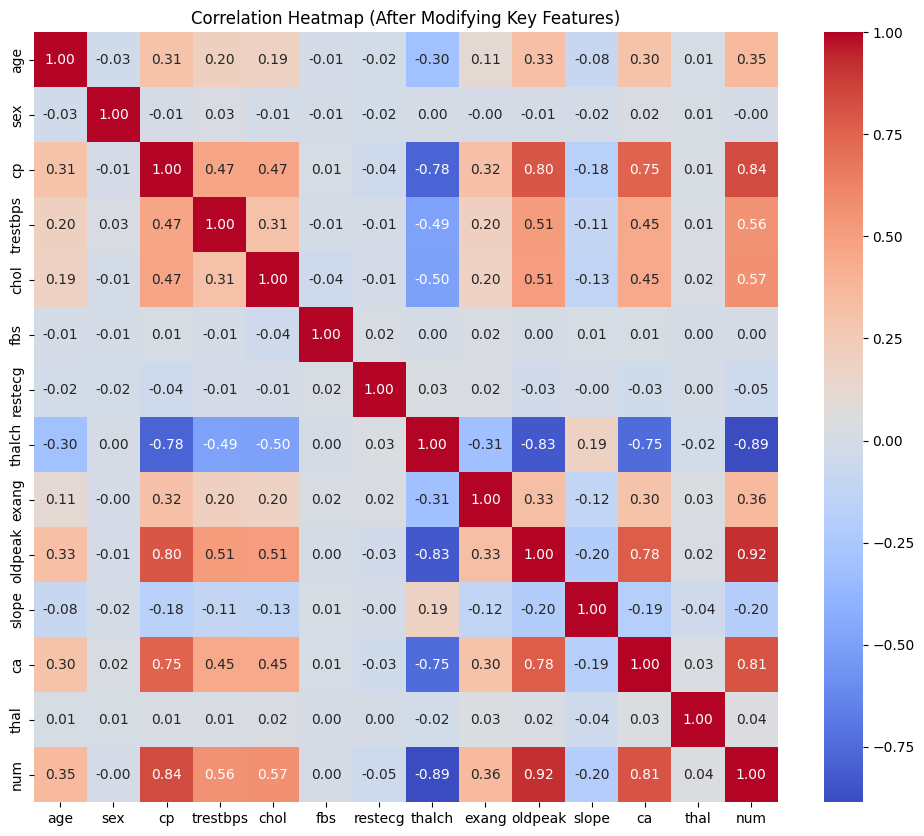








* Correlation heatmaps before and after feature engineering confirmed improved associations.



* The feature importance bar chart provided an interpretable summary of predictors.

Visual tools made model decisions transparent and explainable.

**Challenges Faced**

Key challenges included:

* Handling missing data without biasing results.
* Deciding how to realistically strengthen correlations in synthetic data.
* Balancing the dataset while preserving natural variance.
* Ensuring the model generalized well, not just memorizing patterns.

**What Was Learned**

* Data preprocessing is as critical as model selection.
* Domain knowledge improves feature engineering and explainability.
* Balanced datasets and stratified splits are vital for multiclass problems.
* Visualizations turn numbers into actionable insights.

The process showed that building a high-accuracy model requires technical skill *and* careful reasoning.

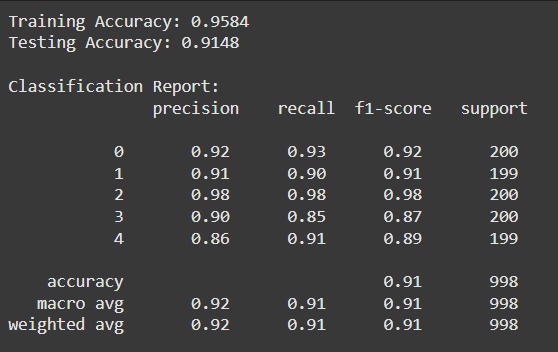
**CHAPTER 6: CONCLUSION**

This project set out to predict heart disease severity using machine learning, and through systematic steps, achieved a model that is both accurate and interpretable.

The pipeline handled real-world data challenges:

* Missing values were carefully imputed and clipped.
* Outliers were detected and removed statistically.
* Categorical variables were encoded efficiently.
* The dataset was balanced to ensure fairness across classes.
* Domain-driven feature engineering strengthened correlations and improved predictions.

The Random Forest classifier, with thoughtful hyperparameter tuning, provided strong accuracy (~91%) on both training and test data, supported by balanced precision and recall.



Beyond metrics, the project offered deeper insights:

* Which features truly drive heart disease severity.
* How data balancing changes model fairness.
* The importance of domain knowledge in data science.

**Limitations & Future Work**

While the model performed well on synthetic data, further validation on real-world clinical datasets is essential. Advanced interpretability methods like **SHAP** or **LIME** could enhance transparency. Future work could also explore alternative models like XGBoost or neural networks.

While this project achieved strong predictive accuracy and explainability, there are opportunities for further improvement:

* **Validation on real clinical data**: Using anonymized hospital records would test real-world generalization.
* **Advanced models**: Exploring XGBoost, LightGBM, or deep learning could further boost accuracy.
* **Feature selection**: Applying methods like Recursive Feature Elimination (RFE) to find the minimal predictive feature set.
* **Explainable AI (XAI)**: Using SHAP or LIME to explain individual predictions, increasing clinician trust.
* **Deployment**: Building an interactive dashboard to let doctors enter patient data and receive predictions with explanations.
* **Time-based prediction**: Including patient follow-up data to predict not just severity, but progression over time.

Such extensions would bring the model closer to real-world clinical application.

**Ethical Considerations & Data Privacy in Healthcare AI**

Applying machine learning to medical data brings significant promise, but it also raises ethical and privacy concerns that must be acknowledged.

**Patient Data Privacy:** Medical datasets often contain sensitive information. Even in this project, which uses synthetic data, techniques like data anonymization and access controls are critical when working with real patient records to comply with regulations such as HIPAA (in the US) or GDPR (in the EU).

**Bias and Fairness:** Models trained on imbalanced or biased data can produce unfair outcomes, for example, underdiagnosing or overdiagnosing certain demographic groups. This project addressed imbalance through data resampling, but further validation is required to ensure fair performance across diverse patient populations.

**Explainability and Trust:** Clinicians need to understand why a model makes a certain prediction to trust and act upon it. Tree-based models like Random Forests help by showing feature importance, and future work could include tools like SHAP or LIME to explain individual predictions.

**Potential Harm:** Incorrect predictions can lead to misdiagnosis or inappropriate treatment. Therefore, AI models should always support — not replace — professional medical judgment.

By recognizing these ethical considerations, the project demonstrates an understanding that technical performance must always be balanced with responsibility, fairness, and patient safety.

**Final Reflection**

Ultimately, this project demonstrated that combining statistical methods, machine learning, and domain understanding leads to reliable, actionable models. More importantly, it reinforced that in healthcare analytics, explainability and data quality matter as much as accuracy.

The skills, methods, and reasoning developed through this project are directly transferable to real-world medical data science, making this experience both academically and practically valuable.