CORONARY ARTERY DISEASE DETECTION USING AI

Minor project report submitted in partial fulfilment of the requirement for the

degree of Bachelor of Technology

in

Computer Science and Engineering

By

Harsh Kumar (221031040)

Mansi Salar (221030167)

Kshitij (221030311)

Badal Singh (221030450)

UNDER THE SUPERVISION OF

Dr. Deepak Gupta

Department of Computer Science & Engineering and Information Technology

Jaypee University of Information Technology, Waknaghat,

173234, Himachal Pradesh, INDIA

TABLE OF CONTENT

DECLARATION

We hereby declare that this project has been carried out by us under the supervision of Dr. Deepak Gupta, Assistant Professor (SG), Department of Computer Science & Engineering and Information Technology, Jaypee University of Information Technology. We further declare that this project or any part of it has not been submitted elsewhere for the award of any degree or diploma.

Supervised by:

Dr. Deepak Gupta

Assistant Professor (SG)

Department of Computer Science & Engineering and Information Technology

Jaypee University of Information Technology

Submitted by:

Harsh Kumar (221031040)

Mansi Salar (221030167)

Kshitij (221030311)

Badal Singh (221030450)

Computer Science & Engineering Department

Jaypee University of Information Technology

CERTIFICATE

This is to certify that the work presented in the project report titled “Coronary Artery Disease Detection using AI”, in partial fulfilment of the requirements for the award of the degree of B.Tech in Computer Science and Engineering, and submitted to the Department of Computer Science and Engineering, Jaypee University of Information Technology, Waknaghat, is an authentic record of work carried out by: Harsh Kumar (221031040), Mansi Salar (221030167), Kshitij (221030311), and Badal Singh (221030450) during the period from January 2025 to May 2025, under the supervision of Dr. Deepak Gupta, Assistant Professor (SG), Department of Computer Science and Engineering, Jaypee University of Information Technology, Waknaghat.

Harsh Kumar (221031040)
Mansi Salar (221030167)
Kshitij (221030311)
Badal Singh (221030450)

The above statement is correct to the best of my knowledge.

Dr. Deepak Gupta
Assistant Professor (SG)
Computer Science & Engineering and Information Technology
Jaypee University of Information Technology, Waknaghat

ACKNOWLEDGEMENT

Firstly, we express our heartfelt gratitude to the Almighty for His divine blessings, which enabled us to complete this project successfully.

We are profoundly grateful to our supervisor, Dr. Deepak Gupta, Assistant Professor (SG), Department of Computer Science & Engineering, Jaypee University of Information Technology, Waknaghat, for his constant support, expert guidance, and motivation throughout the project. His extensive knowledge, valuable suggestions, and constructive feedback at every stage have been instrumental in the successful completion of our project.

We also extend our sincere thanks to all the faculty members and staff of the Department of Computer Science & Engineering for their kind cooperation and assistance during the course of this project.

Finally, we are deeply thankful to our families for their unwavering support, patience, and encouragement throughout this journey.

Harsh Kumar (221031040)
Mansi Salar (221030167)
Kshitij (221030311)
Badal Singh (221030450)

ABSTRACT

Coronary Artery Disease (CAD) is among the most prevalent and potentially life-threatening cardiovascular diseases in the world. It is imperative that CAD is diagnosed early in order to decrease mortality and enhance the quality of life. For this project, we created a CAD prediction system using machine learning from the Z-Alizadeh Sani dataset, with 303 samples and 55 features comprising clinical, demographic, and laboratory information. The dataset was thoroughly preprocessed—missing values treated, categorical data label-encoded, and numerical features scaled via StandardScaler.

There was a robust feature engineering procedure undertaken to shortlist the most critical inputs. Correlation analysis, as well as model-based feature importance via Random Forest and XGBoost, were implemented in order to limit redundancy as well as enhance learning performance. Once this had been done, the feature list had been condensed from 54 down to about 23–31 effective features. We trained and tested 10 machine learning algorithms including Logistic Regression, Decision Tree, Random Forest, Naive Bayes, K-Nearest Neighbors, SVM, XGBoost, LightGBM, CatBoost, AdaBoost, and Gradient Boosting. Accuracy, precision, recall, F1-score, and ROC-AUC were some of the evaluation metrics employed. Among them, ensemble-based algorithms such as XGBoost and AdaBoost were found to perform outstandingly with an accuracy of up to 94% and recall values as high as 97%, which is extremely important when identifying high-risk CAD patients.

To solve for privacy issues and support distributed model training, we added Federated Learning (FL) to the project. FL enables several clients (e.g., hospitals or clinics) to train models on their local private data and only exchange model updates with a central server. This decentralized solution ensures patient confidentiality but still leverages collaborative learning. We continued testing ensemble methods in FL, such as clients adopting disparate voting processes—hard voting, soft voting, and weighted voting—each of which combines predictions from a collection of base classifiers. Findings indicated that high model performance was possible even in a federated context, proving the viability of secure, privacy-aware AI-based CAD detection systems for healthcare settings.

CHAPTER 01: INTRODUCTION

1.1 Introduction

Coronary Artery Disease (CAD) is a life-threatening condition [1] caused by the narrowing or blockage of coronary arteries, primarily due to plaque buildup. It is one of the leading causes of mortality globally. Early detection and diagnosis [2,9,15,16] of CAD can significantly reduce risks and improve patient outcomes. Traditional diagnostic [3,4,5] methods are often time-consuming and require advanced clinical infrastructure. This project addresses these challenges by building an intelligent system that leverages machine learning and federated learning [5,11,15] techniques to detect CAD from patient health records.

Our solution incorporates a hybrid approach to feature engineering, data preprocessing, and model training [3,8,14] using diverse machine learning algorithms. We used SMOTE [10,12] to balance class distributions, followed by extensive experimentation with ten classifiers. Advanced ensemble methods like stacking and voting (both majority and weighted) [6,14] were implemented to improve accuracy and robustness. Finally, we explored federated learning [11,15] to enable privacy-preserving collaborative model training.

1.2 Objective

The primary objective of this project is to develop a cost-effective and time-efficient alternative to traditional Coronary Artery Disease (CAD) detection methods like ECG or angiography. These conventional diagnostic techniques, while effective, are often expensive, resource-intensive, and time-consuming.

To overcome these limitations, we aim to:

- Build a machine learning-based diagnostic [3,4,5,9] system that can detect CAD using readily available clinical data (non-imaging, tabular health parameters).
- Replace the dependency on high-cost instruments by leveraging data-driven methods that require only routine health measurements.
- Improve the detection accuracy through advanced ensemble techniques such as stacking and voting [6,14].

- Implement a hybrid feature engineering [3,8,14] approach to enhance model interpretability and reduce computational overhead.
- Introduce federated learning [11,15] to allow collaborative model training across multiple systems without sharing sensitive patient data, thus maintaining privacy and scalability.

1.3 Motivation

Coronary Artery Disease (CAD) is one of the leading causes of death around the world, and detecting it early is really important. But the usual diagnostic methods like ECGs, angiography [17,18], or stress tests are expensive, time-consuming, and often not available in rural or low-resource areas. These tests also need special equipment and trained experts, which makes them hard to use for large-scale screening.

This project was inspired by the idea of creating a more affordable, faster, and easily scalable way to detect CAD early using machine learning. By using basic clinical data that's usually collected in regular health checkups (tabular form), and applying feature engineering to pick out the most useful information, we hope to make CAD screening more accessible and less dependent on costly tools.

We also included ensemble learning techniques and federated learning [3,8,11,14,15] in our approach. This not only helped improve the accuracy of our predictions but also ensured that patient data stays private and secure. Overall, this project shows how AI can be used responsibly and effectively to support healthcare.

1.4 Languages Used

The implementation of this project was primarily carried out using Python, owing to its simplicity, vast ecosystem of machine learning libraries, and suitability for data science applications. The following key Python libraries and tools were used:

- Pandas and NumPy: For data manipulation and numerical operations.
- Scikit-learn: For implementing machine learning models, preprocessing, and feature selection techniques.
- XGBoost, CatBoost, LightGBM: For advanced gradient boosting-based classifiers.

- Matplotlib and Seaborn: For visualizing data distributions, feature importance, and model performance metrics.
- Imbalanced-learn (SMOTE): For handling imbalanced [12] datasets.
- Google Colab: As the cloud-based development environment to write, run, and share notebooks.
- Flask is used to implement and simulate the Federated Learning [11,15] setup, enabling communication between clients and the central server.

1.5 Technical Requirements

This project was developed and executed on Google Colab, allowing us to run experiments in the cloud and avoid relying heavily on personal hardware. For anyone looking to run the project locally or in a collaborative setting—especially for federated learning experiments—the following technical specifications are recommended:

Hardware Requirements:

- Minimum: Intel Core i5 (8th Gen), 8 GB RAM, 256 GB SSD, Windows 10, Linux, or macOS, with internet access
- Recommended: Intel Core i7 / AMD Ryzen 7 or higher, 16 GB RAM, NVIDIA GTX 1650 or better, 512 GB SSD; Google Colab Pro (optional) for better GPU and memory

Software Requirements:

- Python (including packages like scikit-learn, XGBoost, LightGBM, CatBoost)
- Jupyter Notebook or Google Colab
- For federated learning: optional libraries such as PySyft or Flower

1.6 Deliverables/Outcomes

The following are the major deliverables and outcomes of the project:

Deliverables:

- Cleaned and Preprocessed Dataset: Handled missing values, encoded categorical data, applied scaling (Min-Max), and addressed class imbalance using SMOTE.
- Feature Engineering and Selection Module: Implemented a hybrid approach using XGBoost feature importance, Recursive Feature Elimination (RFE), and Mutual Information (MI) to reduce features from 54 to 23.

- Model Training and Evaluation: Applied and evaluated 10 machine learning classifiers on both full and reduced datasets. Developed ensemble models using Stacking and Voting (Majority & Weighted) strategies for improved performance.
- Federated Learning Setup (Prototype): Laid groundwork for a distributed learning model across multiple devices/systems ensuring data privacy and scalability.
- Google Colab Notebooks: Modular notebooks for each step (Preprocessing, Feature Selection, Model Training, Ensemble) with shared links.

Outcomes:

- Achieved an accuracy of over 94% using AdaBoost on the optimized feature set.
- Improved F1 Score and AUROC through ensemble techniques.
- Demonstrated a cost-effective and efficient alternative to ECG-based CAD detection.
- Enabled potential for privacy-preserving collaborative learning through federated learning.

CHAPTER 02: FEASIBILITY STUDY, REQUIREMENTS ANALYSIS AND DESIGN

2.1 Feasibility Study

2.1.1 Problem Definition

Coronary Artery Disease (CAD) is a leading cause of death globally. Traditional diagnostic methods [17,18] like ECG, angiography, or stress tests are often expensive, invasive, time-consuming, and require specialized infrastructure. There is a need for a non-invasive, cost-effective, and efficient method for early CAD detection using readily available data. The Federated Learning approach [11,15] enabled us to create a predictive model without sharing real patient data, thereby protecting data privacy while supporting collaborative learning.

2.1.2 Problem Analysis

Current Limitations:

- Dependence on hospital infrastructure and skilled personnel.
- Inaccessibility in rural or underdeveloped regions.
- Limited dataset

Literature Survey Insights:

- Recent research explores using machine learning [13] on clinical data (age, cholesterol, blood pressure, etc.) for prediction.
- Ensemble models like XGBoost, Random Forest, SVM, and Stacking/Voting have demonstrated good accuracy.
- Federated learning is an emerging solution that allows decentralized model training while preserving privacy.

Table 2.1: Literature review

Author	Title	Work Done	Pros	Cons
Vardhan Shorewala	Early Detection of Coronary Heart Disease using Ensemble Techniques	Base (KNN, LR, SVM, DT, NB) vs Ensemble (Bagging, Boosting, Stacking) for CAD	Stacking (KNN, RF, SVM) improved accuracy (75.1%).Ensembles outperformed base models.	Accuracy remains moderate (75.1%).Limited Kaggle dataset. No clinical validation.
Kaveh Hosseini, Seyedeh Hamideh Mortazavi,Tehran Heart Center Study	Prevalence and Trends of CAD Risk Factors in Patients with Documented CAD	Analysed data from 90,094 CAD patients to identify risk factor trends and their effect on diagnosis age	Studied 90,094 CAD cases over 11 years; opium use (2.2 yrs) and male gender (~3 yrs) linked to earlier onset.	Lacked data on opium/cigarette use amounts, full BP values, and clinical validation of interventions.
Dibakar Sinha and Ashish Sharma	Automated Detection of Coronary Artery Disease Using Machine Learning Algorithm	Used modified K-means + SVM for heart disease prediction; achieved high accuracy with fewer attributes.	This method provides high prediction accuracy of 89% with fewer attributes; suitable for clinical use.	The approach is limited by its reliance on a specific dataset and may require additional medical attributes for broader applicability.
Can Eyupoglu Oktay Karakuş	Novel CAD Diagnosis Method Based on Search, PCA, and AdaBoostM1 Techniques	Boosts CAD with PCA + AdaBoost, achieving 91.80% accuracy using 5 features on the Z-Alizadeh Sani	Achieves 91.8% accuracy using 5 features reducing complexity and cost.	Data quality, bias, temporal ambiguity, confounding variables, and validity of measurements/causality inference.

Shasha Zhang Yuyu Yuan Zhonghua Yao Xinyan Wang Zhen Lei	Improving Coronary Artery Disease prediction performance using XGBoost and feature processing.	XGBoost and Random Forest used with four feature processing techniques and SMOTE for improved accuracy.	Balancing methods improve stability; feature processing boosts CAD accuracy and reduces redundancy for improved performance.	Needs validation due to small dataset. May be less reliable than angiography in complex cases. Future work to explore more ensemble techniques.
Angela Koloi, Vasileios S Loukas, Cillian Hourican, Antonis I Sakellarios, Rick Quax, Pashupati P Mishra	Predicting early-stage CAD using machine learning and routine clinical biomarkers improved by augmented virtual data	This study applies machine learning, using Gradient Boosting and Random Forests, to detect early-stage CAD from routine lab tests.	Non-invasive method using routine clinical data, offering high accuracy with GB models and potential for early CAD identification.	Dependence on specific biomarkers that may not be readily available in all clinical settings, limiting the method's general applicability and potential .
Multidisciplinary Digital Publishing Institute (MDPI), Mohammad Javad Sayadi, Vijayakumar Varadarajan.	A Machine Learning Model for Early Diagnosis of Coronary Artery Disease Using Clinical Data.	Developed a machine learning model for early CAD diagnosis using clinical data and Pearson feature selection.	The model provides a non-invasive, high-accuracy CAD diagnosis with 95.45% accuracy, high sensitivity, and specificity.	The model's performance is dataset-dependent and may not generalize to all populations or include all relevant CAD features.
Stephen D. Cagle Jr., MD, and Noah Cooperstein, MD	Coronary Artery Disease: Diagnosis and Management	Reviewed CAD diagnosis and management, prevention, risk stratification, and clinical guidelines.	Covers both symptomatic and asymptomatic CAD, integrates guideline-based screening and lifestyle interventions.	Relies on existing guidelines without proposing new diagnostic methods; lacks novel data or advanced computational approaches.

Afzal Hussain Shahid, M.P. Singh	A Novel Approach for CAD Diagnosis using Hybrid Particle Swarm Optimization based Emotional Neural Network	Proposed PSO-EmNN model for CAD using 3 datasets, 4 feature selectors, and 19 tested feature subsets.	Achieved 88.34% accuracy, precision (92.37%), and F1-score (92.12%), fast learning, robust to imprecision, and strong performance across datasets.	Works offline only, lacks interpretability (black-box), and may underperform on small datasets; needs online support and larger datasets for improvement.
Mohammad M. Ghiasi, Sohrab Zendehboudi, Ali Asghar Mohsenipour	Decision tree-based diagnosis of coronary artery disease: CART model	Developed CART-based models for CAD diagnosis using the Z-Alizadeh Sani dataset with 10-fold cross-validation .	High precision, simple and robust model, performs well with selected features, reliable for clinical use on Z-Alizadeh Sani dataset.	Performance sensitive to data quality and size; limited features may reduce accuracy; larger datasets needed for real-world deployment.
Olfa Hrizi, Karim Gasmi, Abdulrahman Alyami, Adel Alkhalil, Ibrahim Alrashdi, Ali Alqazzaz	Federated and Ensemble Learning Framework with Optimized Feature Selection for Heart Disease Detection	Proposed a privacy-preserving heart disease detection model using federated learning.	Achieved 95% precision with a scalable, robust FL-based model by voting and stacking for privacy, performance.	Computational complexity of ensemble and FL setup; dependent on data quality and communication infrastructure in federated settings.
Sultan Alasmari, Rayed AlGhamdi, Ghanshyam G. Tejani, Sunil Kumar Sharma, Seyed Jalaleddin	Federated Learning-Based Multimodal Approach for Early Detection and Personalized Care in Cardiac Disease	Proposed CardioNet federated learning-based framework for CAD using images, ECG	Accuracy (99.12%), precision (98.76%), sensitivity (97.65%), privacy-preserving, enables personalized care.	Multimodal integration, data quality dependency, real-time challenges, interpretability, high federated training resources.

Key Challenge:

- Ensuring balanced datasets, optimal feature selection, and robust generalization of models in real-world settings.

2.1.3 Solution

We propose a machine learning-based Coronary Artery Disease (CAD) detection system using structured health data, with an emphasis on:

- Feature engineering (hybrid selection from 54 to 23 features),
- Evaluation of multiple classifiers,
- Ensemble methods for better prediction accuracy,
- Prototype federated learning setup for privacy-preserving distributed training.

The project also shows how a prototype federated learning setup can make AI training safer and more private across different hospitals or research centers. Normally, machine learning needs all the patient data to be gathered in one place for training, which can raise serious privacy and security risks. But with federated learning, each hospital keeps its patient data locally. Instead of sharing the actual data, they only send model updates—like learned patterns or adjustments—to a central system. This way, sensitive patient information stays protected at its source, helping to prevent data leaks and stay in line with important privacy laws like HIPAA or GDPR.

By using federated learning for coronary artery disease detection, the system can learn from a wider variety of patients across different regions, without anyone having to give up their raw data. This not only keeps data safer but also encourages collaboration between hospitals and clinics that might otherwise hesitate to share sensitive information. As a prototype, it sets the stage for building bigger, safer AI systems in healthcare—where privacy, trust, and teamwork are all crucial.

2.2 Requirements

The project was implemented using Google Colab, enabling cloud-based execution and reducing local hardware dependency. Below are the technical requirements for local or collaborative deployment, especially for federated learning experiments:

Hardware Requirements:

- Minimum: Intel Core i5 (8th Gen), 8 GB RAM, 256 GB SSD, Windows 10/Linux/macOS, Internet access
- Recommended: Intel Core i7 / Ryzen 7+, 16 GB RAM, NVIDIA GTX 1650+, 512 GB SSD, optional Google Colab Pro for enhanced GPU/memory

Software Requirements:

- Python (with scikit-learn, XGBoost, LightGBM, CatBoost, etc.)
- Jupyter Notebook / Google Colab & optional libraries for federated learning: PySyft, Flower

2.3 Data-Flow Diagram (DFD)

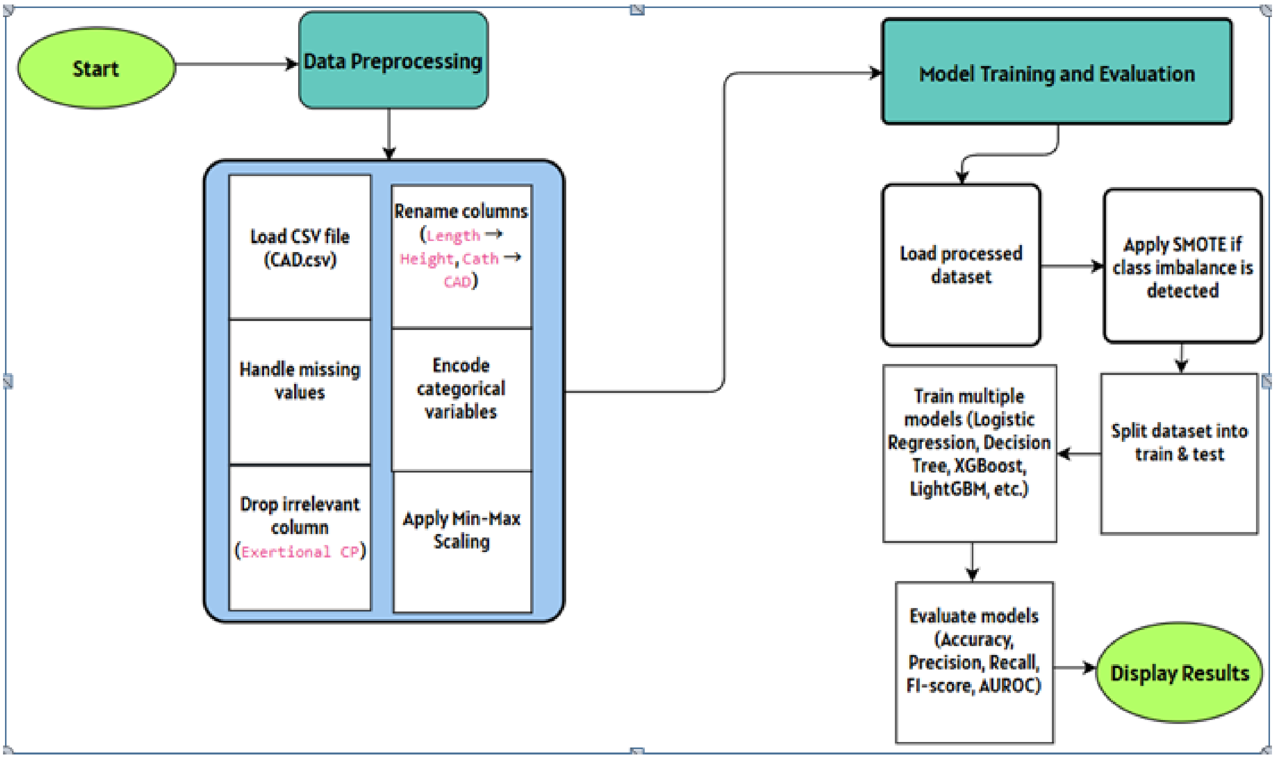

Start
Data Preprocessing
Load CSV file (CAD.csv)
Handle missing values
Drop irrelevant column (Exertional CP)
Rename columns (Length → Height, Cath → CAD)
Encode categorical variables
Apply Min-Max Scaling
Model Training and Evaluation
Load processed dataset
Apply SMOTE if class imbalance is detected
Split dataset into train & test
Train multiple models (Logistic Regression, Decision Tree, XGBoost, LightGBM, etc.)
Evaluate models (Accuracy, Precision, Recall, FI-score, AUROC)
Display Results

Figure 2.1: Project flow diagram illustrating the sequential process

CHAPTER 03: IMPLEMENTATION

3.1 Data Set Used in the Minor Project

CAD.csv: This is the original dataset used for detecting Coronary Artery Disease (CAD). Z-Alizadeh Sani dataset [19] (publicly available in the UCI Machine Learning repository)

- Description: 303 samples and 55 features
- Numerical (34 columns): Age, BMI, BP, PR, various blood test values, etc.
- Categorical (21 columns): Sex, Obesity, CHF, Dyspnea, etc.
- Target Variable (Cath): Indicates whether a patient has CAD (Cad) or is normal (Normal).

PreProcessed_Dataset_MinMax.csv: This version of the dataset was created after applying Min-Max Scaling, which transforms features to a common scale without distorting differences in the range of values. It is particularly useful for algorithms that rely on distance-based metrics.

PreProcessed_Dataset_StdScaler.csv: This dataset was preprocessed using Standard Scaler, which standardizes features by removing the mean and scaling to unit variance. It ensures that features contribute equally during model training, especially in models sensitive to feature scales.

Reduced_Dataset_Hybrid_Approach.csv: This is the final, feature-engineered dataset created using a hybrid feature selection approach. Top features were selected using methods like Mutual Information, RFE, and XGBoost Importance, reducing the original feature set significantly while retaining key predictive power.

3.2 Data Set Features

3.2.1 Types of Dataset

The primary dataset[19] used for this project is tabular in nature and consists of structured data. It includes both categorical and numerical features relevant to medical diagnosis for Coronary Artery Disease (CAD). The dataset is supervised, meaning it contains a labeled target variable indicating whether a patient is diagnosed with CAD or not.

The target variable is binary, with classes such as:

- 1 indicating presence of CAD (positive diagnosis),
- 0 indicating absence of CAD (negative diagnosis).

3.2.2 Number of Attributes, Fields, Description of the Dataset

Original Dataset (CAD.csv):

- Instances (Rows): 303
- Attributes (Columns): 54
- Target Attribute: CAD diagnosis (binary classification)

The dataset comprises a mix of patient demographics and clinical measurements such as:

- Age
- Sex
- Chest Pain Type (cp)
- Resting Blood Pressure (trestbps)
- Serum Cholesterol (chol)
- Fasting Blood Sugar (fbs)
- Resting Electrocardiographic Results (restecg)
- Maximum Heart Rate Achieved (thalach)
- Exercise Induced Angina (exang)
- Oldpeak (ST depression)
- Slope of ST segment (slope)
- And many engineered features extracted from raw clinical data

Final Dataset (Reduced_Dataset_Hybrid_Approach.csv):

- Number of Selected Features: 23 (reduced using hybrid feature selection)
- These features were chosen based on their high contribution to model performance using methods like Mutual Information, Recursive Feature Elimination, and XGBoost Feature Importance.

3.3 Design of Problem Statement

Coronary Artery Disease (CAD) continues to be one of the leading causes of death globally. The early and accurate detection of CAD [2,9,15] is critical in reducing mortality and enabling timely treatment. However, current diagnostic methods such as [17,18] ECG, stress tests, and angiography are either expensive, invasive, or time-consuming. Additionally, access to these diagnostic tools may be limited in under-resourced regions.

The goal of this project is to design and implement a cost-effective and time-efficient machine learning-based system to assist in the early detection of CAD using readily available patient data. This system aims to process clinical attributes, perform intelligent feature selection using a hybrid engineering approach, and apply advanced classification algorithms and ensemble techniques [4,5,6,7,14] to achieve high predictive accuracy.

Key objectives of the problem design include:

- Pre-processing and balancing the dataset using Min-Max Scaling and SMOTE.
- Applying and comparing multiple machine learning models.
- Using ensemble techniques (stacking, weighted and majority voting) to boost performance.
- Reducing dimensionality without compromising accuracy through hybrid feature selection methods.
- Ensuring robust generalization using stratified k-fold validation.
- Preparing the model for potential deployment in federated learning environments, thus preserving data privacy across institutions.

This framework ultimately aims to support healthcare professionals in making faster, data-driven diagnostic decisions, especially in primary care and rural settings, where access to specialists or high-end equipment is limited.

3.4 Algorithm/Pseudo Code of the Project Problem

Input: CAD.csv

Output: Trained model, evaluation metrics

1. Preprocessing: Handle missing values, encode categorical, scale numerics.
2. Balancing: If imbalance < 0.6 → apply SMOTE.
3. Feature Selection (Hybrid): Combine XGBoost, RFE, MI scores → select top 23 features.
4. Model Training:
5. Train: RF, XGBoost, LGBM, CatBoost, SVM, LR, KNN, DT, NB, AdaBoost.
6. Evaluate: Accuracy, Precision, Recall, F1, AUROC.
7. Ensemble:
8. Stacking: XGBoost, RF, LGBM, CatBoost, ExtraTrees, SVM → RF meta-model (5-fold CV).
9. Voting: Majority & weighted (all & top 3 models).
10. Evaluation: Compare models; report & visualize metrics.

3.5 Flowchart of the Minor Project Problem

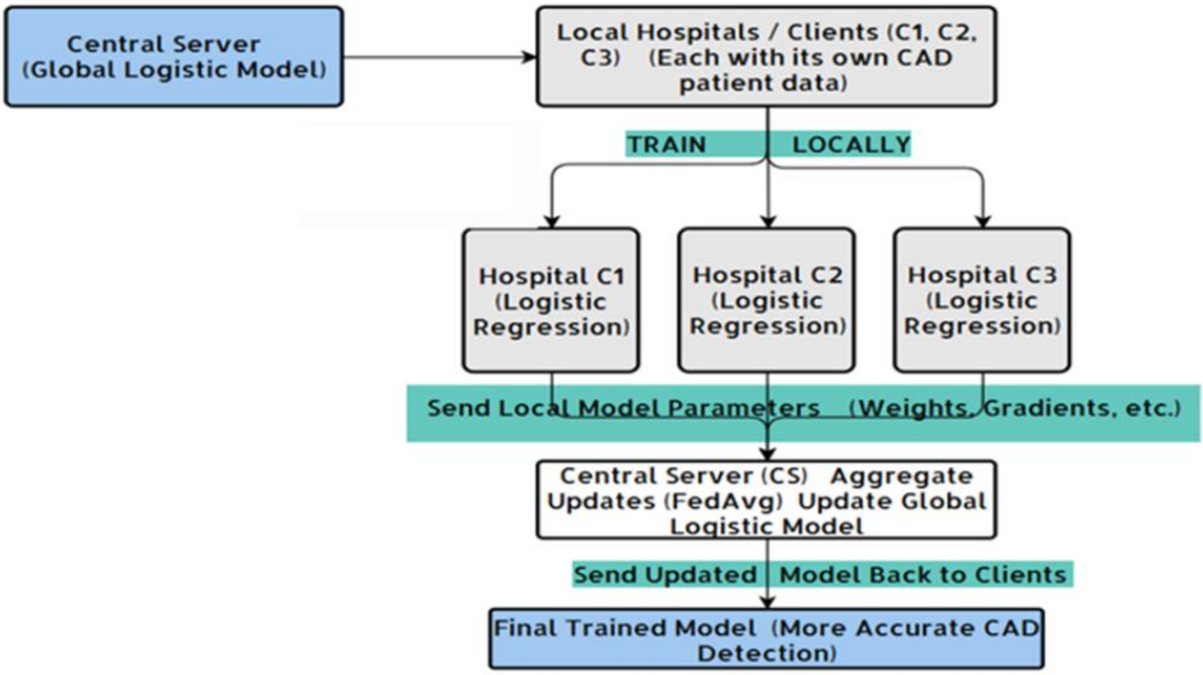
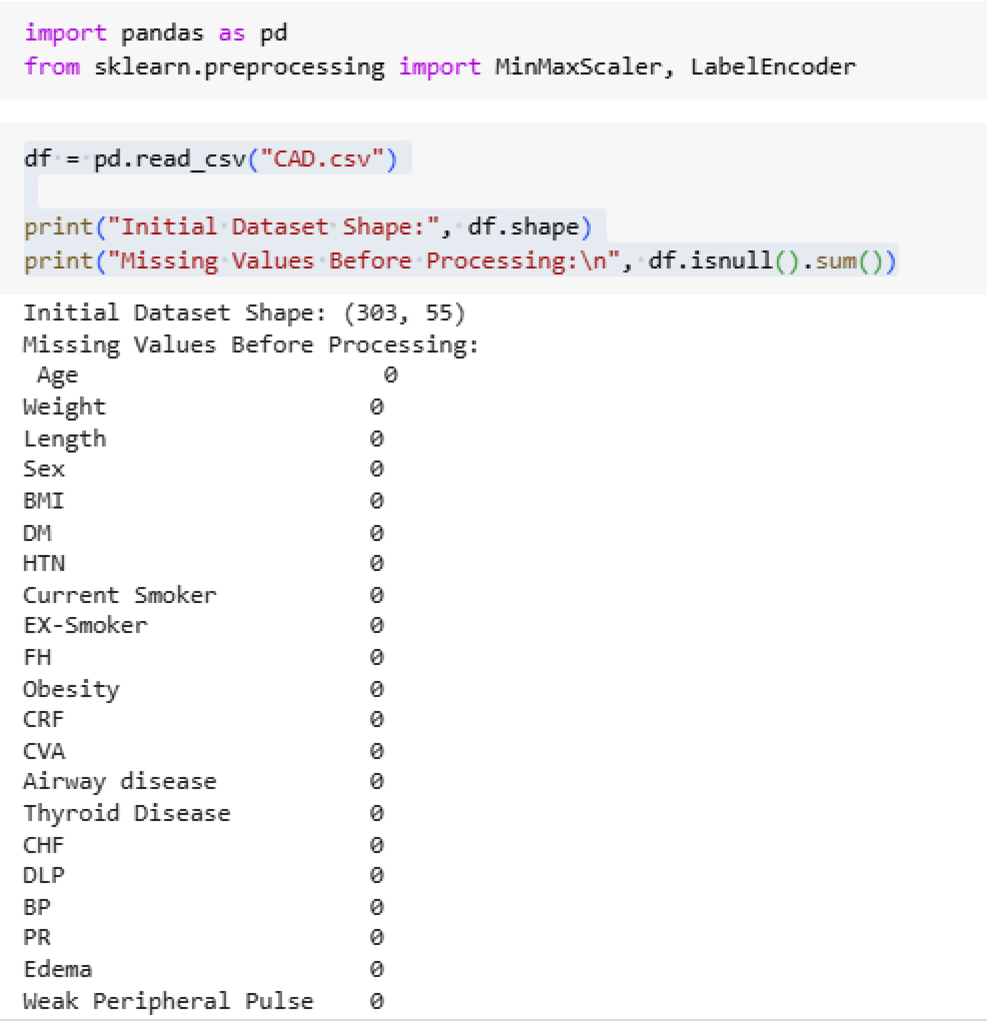

Central Server (Global Logistic Model)
Local Hospitals / Clients (C1, C2, C3) (Each with its own CAD patient data)
TRAIN LOCALLY
Hospital C1 (Logistic Regression)
Hospital C2 (Logistic Regression)
Hospital C3 (Logistic Regression)
Send Local Model Parameters (Weights, Gradients, etc.)
Central Server (CS) Aggregate Updates (FedAvg) Update Global Logistic Model
Send Updated Model Back to Clients
Final Trained Model (More Accurate CAD Detection)

Figure 3.1: Federated learning approach flowchart

3.6 Screenshots of the Various Stages of the Project



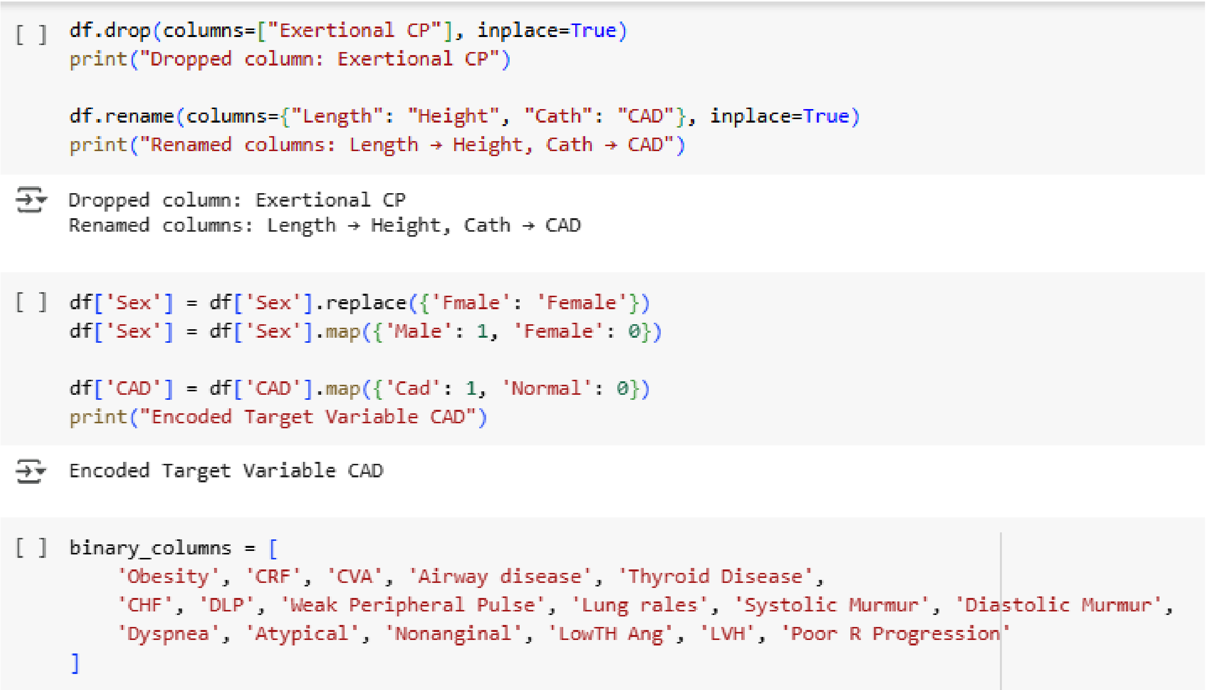
```
import pandas as pd
from sklearn.preprocessing import MinMaxScaler, LabelEncoder
```

```
df = pd.read_csv("CAD.csv")

print("Initial Dataset Shape:", df.shape)
print("Missing Values Before Processing:\n", df.isnull().sum())
```

```
Initial Dataset Shape: (303, 55)
Missing Values Before Processing:
 Age                     0
Weight                  0
Length                  0
Sex                     0
BMI                     0
DM                      0
HTN                     0
Current Smoker          0
EX-Smoker               0
FH                      0
Obesity                 0
CRF                     0
CVA                     0
Airway disease          0
Thyroid Disease         0
CHF                     0
DLP                     0
BP                      0
PR                      0
Edema                   0
Weak Peripheral Pulse   0
```

Figure 3.2: Preprocessing the raw dataset

The shape of the dataset is (303, 55), indicating 303 patient records with 55 features each. The isnull().sum() function confirms that there are no missing values in any column, which means the dataset is complete and ready for further cleaning and transformation.



```python
df.drop(columns=["Exertional CP"], inplace=True)
print("Dropped column: Exertional CP")

df.rename(columns={"Length": "Height", "Cath": "CAD"}, inplace=True)
print("Renamed columns: Length → Height, Cath → CAD")
```

```
Dropped column: Exertional CP
Renamed columns: Length → Height, Cath → CAD
```

```python
df['Sex'] = df['Sex'].replace({'Fmale': 'Female'})
df['Sex'] = df['Sex'].map({'Male': 1, 'Female': 0})

df['CAD'] = df['CAD'].map({'Cad': 1, 'Normal': 0})
print("Encoded Target Variable CAD")
```

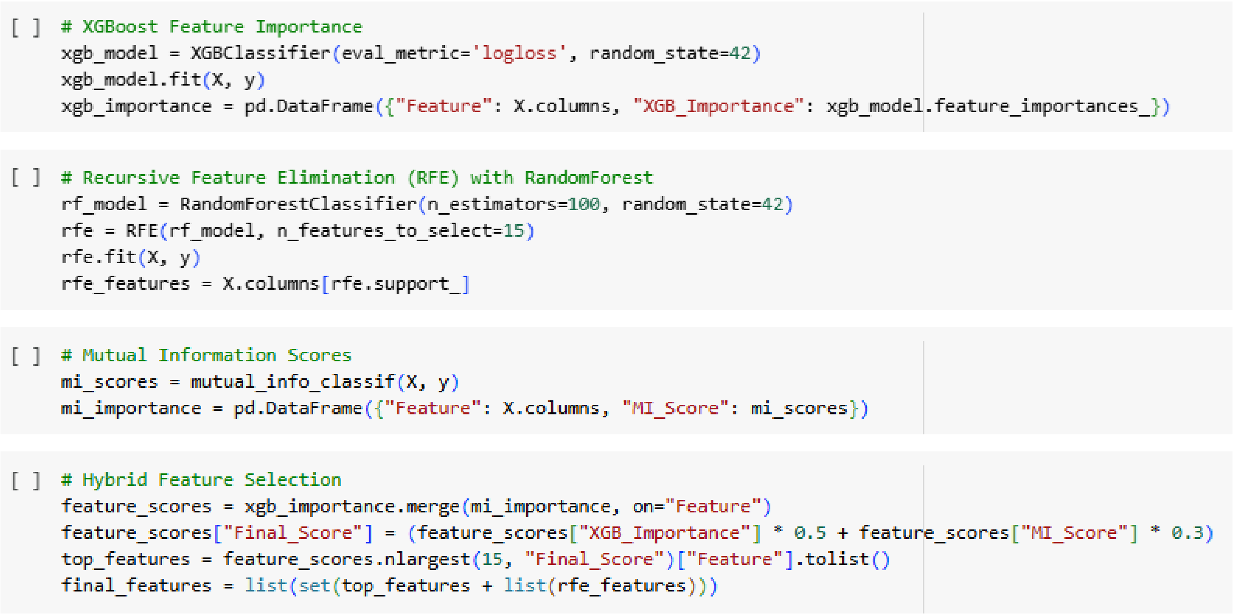
```
Encoded Target Variable CAD
```

```python
binary_columns = [
    'Obesity', 'CRF', 'CVA', 'Airway disease', 'Thyroid Disease',
    'CHF', 'DLP', 'Weak Peripheral Pulse', 'Lung rales', 'Systolic Murmur', 'Diastolic Murmur',
    'Dyspnea', 'Atypical', 'Nonanginal', 'LowTH Ang', 'LVH', 'Poor R Progression'
]
```

Figure 3.3: Encoding applied to nominal data

This code snippet performs preprocessing on a DataFrame df by:

1. Dropping and Renaming Columns:
    - Removes the "Exertional CP" column.
    - Renames "Length" to "Height" and "Cath" to "CAD".
2. Encoding Categorical Variables:
    - Fixes a typo in the "Sex" column ('Fmale' → 'Female') and maps 'Male' to 1, 'Female' to 0.
    - Converts the "CAD" column values ('Cad' → 1, 'Normal' → 0).
3. Preparing for Binary Encoding:
    - Lists binary categorical columns in binary_columns for further processing.



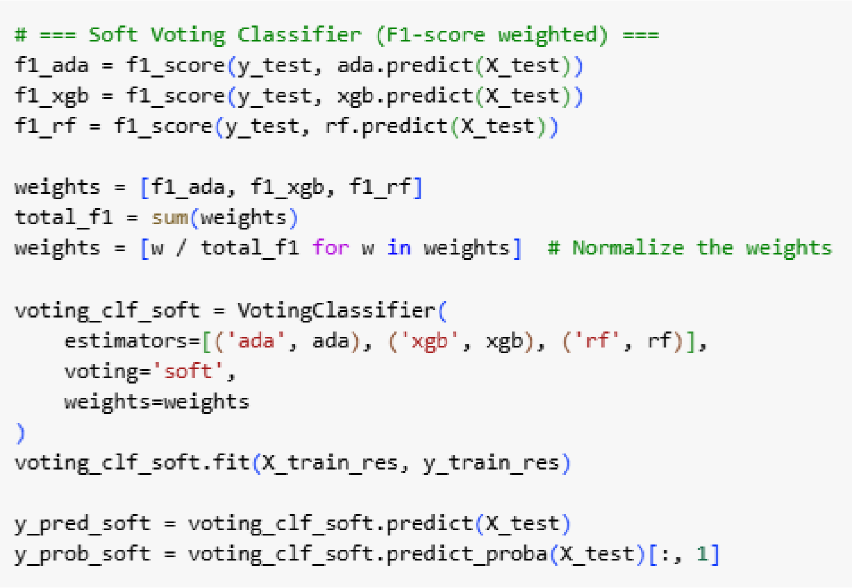
```
[ ] # XGBoost Feature Importance
    xgb_model = XGBClassifier(eval_metric='logloss', random_state=42)
    xgb_model.fit(X, y)
    xgb_importance = pd.DataFrame({"Feature": X.columns, "XGB_Importance": xgb_model.feature_importances_})
```

```
[ ] # Recursive Feature Elimination (RFE) with RandomForest
    rf_model = RandomForestClassifier(n_estimators=100, random_state=42)
    rfe = RFE(rf_model, n_features_to_select=15)
    rfe.fit(X, y)
    rfe_features = X.columns[rfe.support_]
```

```
[ ] # Mutual Information Scores
    mi_scores = mutual_info_classif(X, y)
    mi_importance = pd.DataFrame({"Feature": X.columns, "MI_Score": mi_scores})
```

```
[ ] # Hybrid Feature Selection
    feature_scores = xgb_importance.merge(mi_importance, on="Feature")
    feature_scores["Final_Score"] = (feature_scores["XGB_Importance"] * 0.5 + feature_scores["MI_Score"] * 0.3)
    top_features = feature_scores.nlargest(15, "Final_Score")["Feature"].tolist()
    final_features = list(set(top_features + list(rfe_features)))
```

Figure 3.4: Featuree engineering hybrid approach

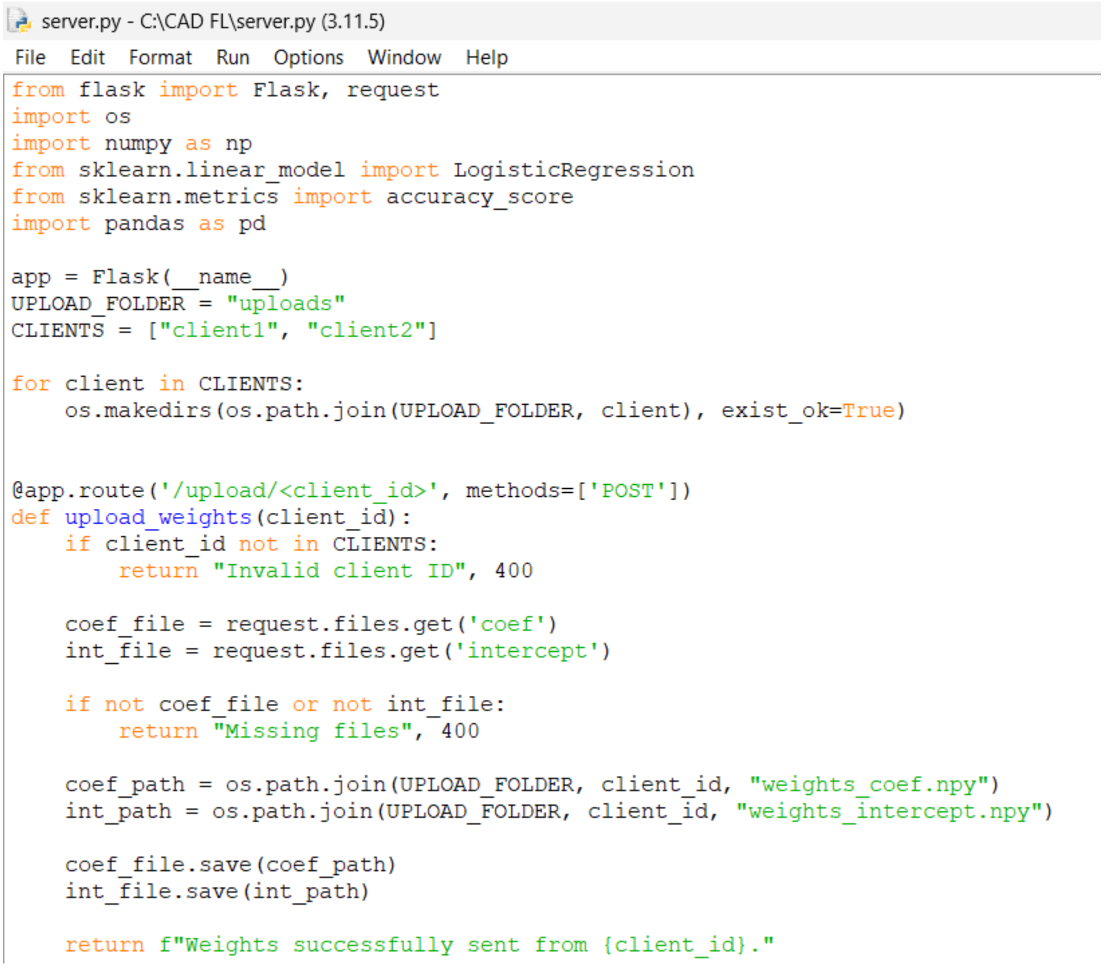


```
# === Soft Voting Classifier (F1-score weighted) ===
f1_ada = f1_score(y_test, ada.predict(X_test))
f1_xgb = f1_score(y_test, xgb.predict(X_test))
f1_rf = f1_score(y_test, rf.predict(X_test))

weights = [f1_ada, f1_xgb, f1_rf]
total_f1 = sum(weights)
weights = [w / total_f1 for w in weights]  # Normalize the weights

voting_clf_soft = VotingClassifier(
    estimators=[('ada', ada), ('xgb', xgb), ('rf', rf)],
    voting='soft',
    weights=weights
)
voting_clf_soft.fit(X_train_res, y_train_res)

y_pred_soft = voting_clf_soft.predict(X_test)
y_prob_soft = voting_clf_soft.predict_proba(X_test)[:, 1]
```

Figure 3.5: Weighted voting

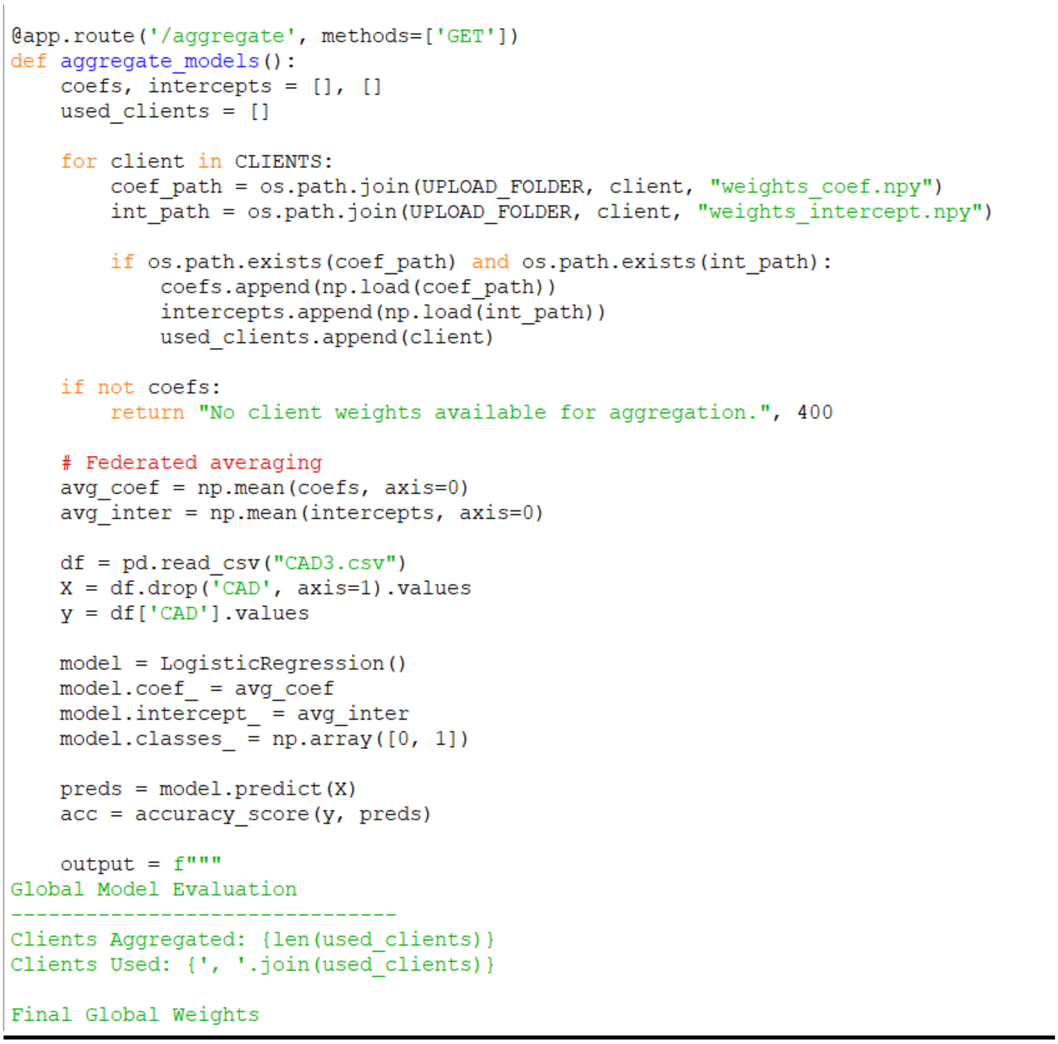
We implemented a Federated Learning approach using Logistic Regression on our dataset, splitting the data across two clients (client1 and client2), each holding 120 instances for local training. The remaining data (out of a total of 303 instances) was reserved for final evaluation on the global model. After local training, the clients shared their model updates without exchanging raw data, preserving data privacy. These updates were aggregated at the server to produce the final global model, achieving a balanced representation of both clients' data distributions.

The final global model achieved an accuracy of 81.25% when evaluated on the held-out test set, demonstrating strong generalization performance.


server.py - C:\CAD FL\server.py (3.11.5)
File Edit Format Run Options Window Help

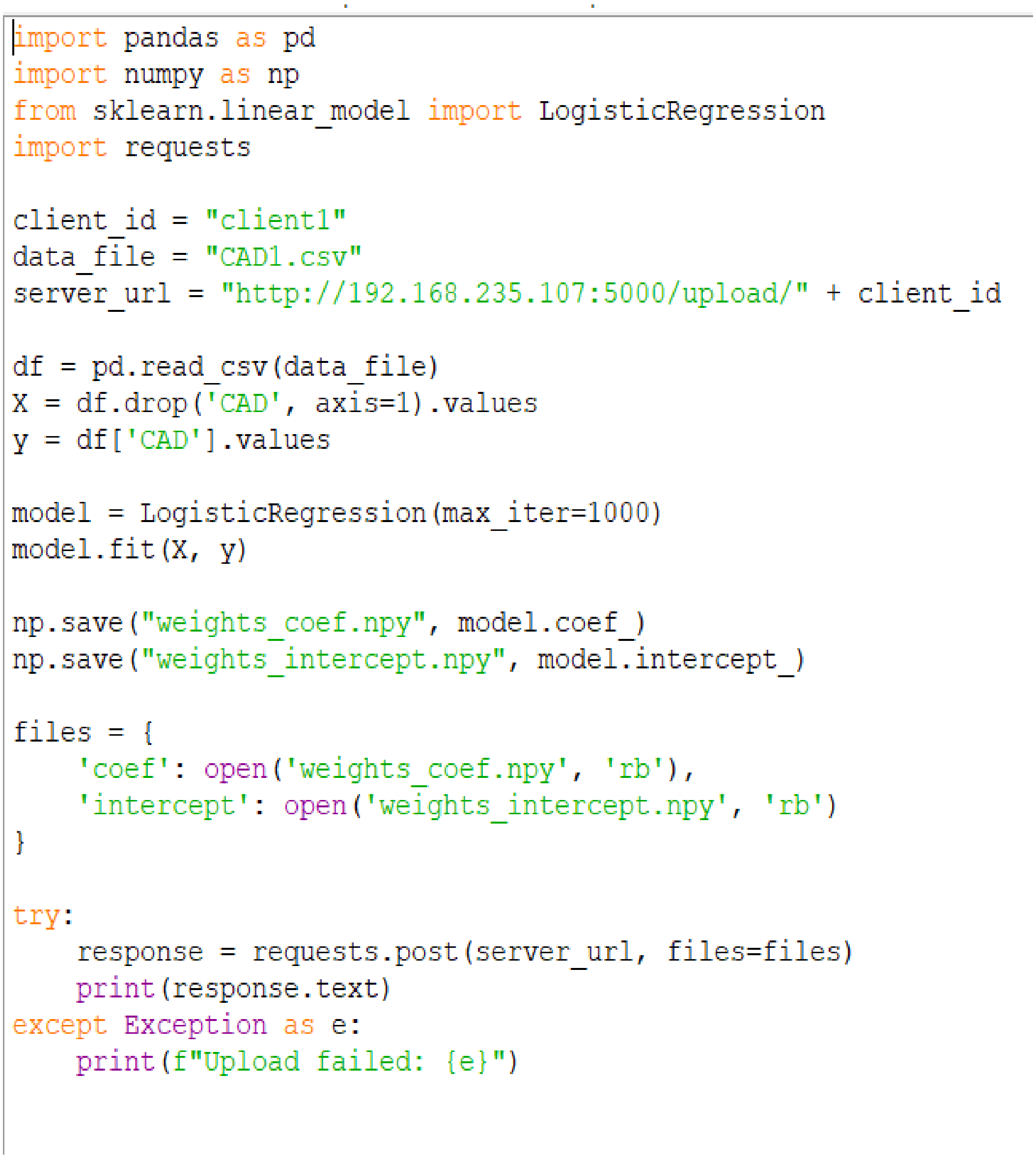
```python
from flask import Flask, request
import os
import numpy as np
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score
import pandas as pd

app = Flask(__name__)
UPLOAD_FOLDER = "uploads"
CLIENTS = ["client1", "client2"]

for client in CLIENTS:
    os.makedirs(os.path.join(UPLOAD_FOLDER, client), exist_ok=True)


@app.route('/upload/<client_id>', methods=['POST'])
def upload_weights(client_id):
    if client_id not in CLIENTS:
        return "Invalid client ID", 400

    coef_file = request.files.get('coef')
    int_file = request.files.get('intercept')

    if not coef_file or not int_file:
        return "Missing files", 400

    coef_path = os.path.join(UPLOAD_FOLDER, client_id, "weights_coef.npy")
    int_path = os.path.join(UPLOAD_FOLDER, client_id, "weights_intercept.npy")

    coef_file.save(coef_path)
    int_file.save(int_path)

    return f"Weights successfully sent from {client_id}."
```

Figure 3.6: Federated learning server side ‘upload’ route



```
@app.route('/aggregate', methods=['GET'])
def aggregate_models():
    coefs, intercepts = [], []
    used_clients = []

    for client in CLIENTS:
        coef_path = os.path.join(UPLOAD_FOLDER, client, "weights_coef.npy")
        int_path = os.path.join(UPLOAD_FOLDER, client, "weights_intercept.npy")

        if os.path.exists(coef_path) and os.path.exists(int_path):
            coefs.append(np.load(coef_path))
            intercepts.append(np.load(int_path))
            used_clients.append(client)

    if not coefs:
        return "No client weights available for aggregation.", 400

    # Federated averaging
    avg_coef = np.mean(coefs, axis=0)
    avg_inter = np.mean(intercepts, axis=0)

    df = pd.read_csv("CAD3.csv")
    X = df.drop('CAD', axis=1).values
    y = df['CAD'].values

    model = LogisticRegression()
    model.coef_ = avg_coef
    model.intercept_ = avg_inter
    model.classes_ = np.array([0, 1])

    preds = model.predict(X)
    acc = accuracy_score(y, preds)

    output = f"""
Global Model Evaluation
-------------------------------
Clients Aggregated: {len(used_clients)}
Clients Used: {', '.join(used_clients)}

Final Global Weights
```

Figure 3.7: Federated learning server side ‘aggregate’ route



```python
import pandas as pd
import numpy as np
from sklearn.linear_model import LogisticRegression
import requests

client_id = "client1"
data_file = "CAD1.csv"
server_url = "http://192.168.235.107:5000/upload/" + client_id

df = pd.read_csv(data_file)
X = df.drop('CAD', axis=1).values
y = df['CAD'].values

model = LogisticRegression(max_iter=1000)
model.fit(X, y)

np.save("weights_coef.npy", model.coef_)
np.save("weights_intercept.npy", model.intercept_)

files = {
    'coef': open('weights_coef.npy', 'rb'),
    'intercept': open('weights_intercept.npy', 'rb')
}

try:
    response = requests.post(server_url, files=files)
    print(response.text)
except Exception as e:
    print(f"Upload failed: {e}")
```

Figure 3.8: Federated learning client side

CHAPTER 04: RESULTS

The study evaluated the performance of different machine learning models for detecting coronary artery disease using both the complete dataset and a reduced version (with 23 features instead of 54). On the complete dataset, XGBoost stood out with the highest accuracy (91.95%), F1 score (91.76%), and AUROC (97.94%). CatBoost and SVM also performed well, though slightly behind. In contrast, Naïve Bayes showed high precision but struggled with recall and F1 score. After reducing the number of features, AdaBoost emerged as the top performer, achieving even better accuracy (94.25%), the highest F1 score (93.98%), and a strong AUROC (96.61%) with minimal training time (0.12s). Overall, reducing the features helped improve the performance of most models, especially boosting-based ones, while also simplifying the models and reducing training time in several cases.

Table 4.1: Performance evaluation of machine learning models for complete dataset

Model	**Accuracy**	**Precision**	**Recall**	**F1 Score**	**AUROC**	**Training time**
XGBoost	**0.919540**	**0.928571**	**0.906977**	**0.917647**	**0.979387**	**0.115706**
CatBoost	0.908046	0.926829	0.883721	0.904762	0.974101	0.897274
SVM	0.908046	0.926829	0.883721	0.904762	0.956131	0.022643
Logistic Regression	0.896552	0.904762	0.883721	0.894118	0.949789	0.013163
Random Forest	0.896552	0.904762	0.883721	0.894118	0.974366	0.235903
Decision Tree	0.885057	0.851064	0.930233	0.888889	0.883192	0.012956
AdaBoost	0.873563	0.900000	0.837209	0.867470	0.958245	0.131567
LightGBM	0.862069	0.897436	0.813953	0.853659	0.969345	0.093369
KNN	0.850575	0.968750	0.720930	0.826667	0.869450	0.003061
Naïve Bayes	0.597701	0.900000	0.209302	0.339623	0.887685	0.0032288

Table 4.2: Performance evaluation of machine learning models for reduced dataset

Model	Accuracy	Precision	Recall	F1 Score	AUROC	Train Time
AdaBoost	**0.942529**	**0.975000**	**0.906977**	**0.939759**	**0.966173**	**0.116817**
Random Forest	0.919540	0.928571	0.906977	0.917647	0.970402	0.216211
XGBoost	0.919540	0.950000	0.883721	0.915663	0.975687	0.084674
CatBoost	0.896552	0.925000	0.860465	0.891566	0.968288	2.708236
Naïve Bayes	0.896552	0.925000	0.860465	0.891566	0.933932	0.002209
Logistic Regression	0.885057	0.883721	0.883721	0.883721	0.932347	0.014886
LightGBM	0.885057	0.945946	0.813953	0.875000	0.972516	0.067397
SVM	0.862069	0.860465	0.860465	0.860465	0.932347	0.018304
Decision Tree	0.839080	0.891892	0.767442	0.825000	0.838266	0.009412
KNN	0.827586	0.868421	0.767442	0.814815	0.894820	0.001947

In terms of computational efficiency, all of the models we evaluated remained remarkably fast, with most completing their training in under a second even after reducing the number of features. Both AdaBoost and Random Forest stood out in this regard, achieving an excellent balance between predictive performance and training speed. After reducing the features, XGBoost completed training in just 0.084674 seconds, and AdaBoost in 0.116817 seconds, while still maintaining high accuracy and F1 scores. This efficiency can largely be explained by the inherent design of boosting algorithms: because they train weak learners sequentially, each iteration focuses only on correcting the errors of the previous one, rather than retraining the entire ensemble from scratch.

By reducing the feature space from 54 to 23, we further simplified the learning process for each weak learner, leading to faster convergence and fewer computations per iteration.

This outcome aligns with theoretical expectations: fewer features mean fewer candidate splits for tree-based models and fewer parameters to estimate in linear models, which directly translates into faster computation. By reducing features from 54 to 23, we effectively reduced the number of computations per node in the tree, speeding up the learning process overall.

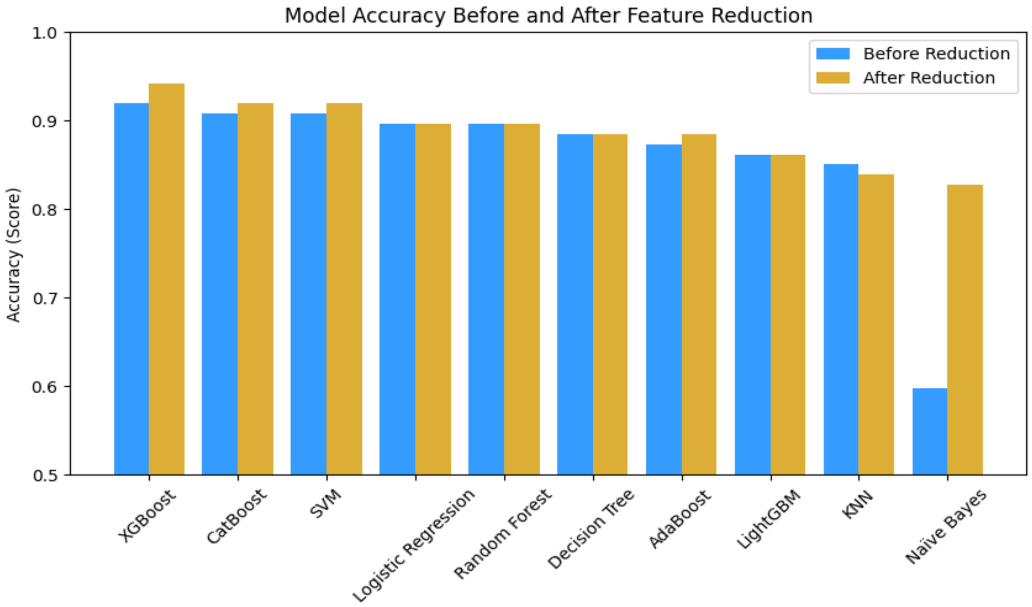


Figure 4.1: Comparison of base classifiers on the basis of accuracy

Table 4.3: Classification results using weighted and majority voting on all models

Metric	Weighted Voting (F1 Score) on all Classifiers	Majority Voting on all Classifiers
Accuracy	0.8689	0.8525
Precision	0.8889	0.8864
Recall	**0.9302**	**0.9070**
F1 Score	0.9091	0.8966

Table 4.4: Classification results using top 3 classifiers

Metric	Weighted Voting (F1 Score) on Top 3 Classifiers	Majority Voting on Top 3 Classifiers
Accuracy	0.8361	0.8524
Precision	0.8511	0.8695
Recall	**0.9302**	**0.9302**
F1 Score	0.8889	0.8988

The classification results using ensemble methods were evaluated through weighted voting and majority voting across all models and the top 3 classifiers. When combining all classifiers, weighted voting achieved slightly higher accuracy (86.89%) and F1 score (90.91%) compared to majority voting (accuracy 85.25%, F1 score 89.66%), with both methods showing strong precision and recall. When ensemble voting was limited to the top 3 classifiers, majority voting outperformed weighted voting in both accuracy (85.24% vs. 83.61%) and F1 score (89.88% vs. 88.89%), while recall remained the same (93.02%). Overall, ensemble methods provided competitive results, with majority voting on the top-performing models offering a good balance of accuracy, precision, and recall.

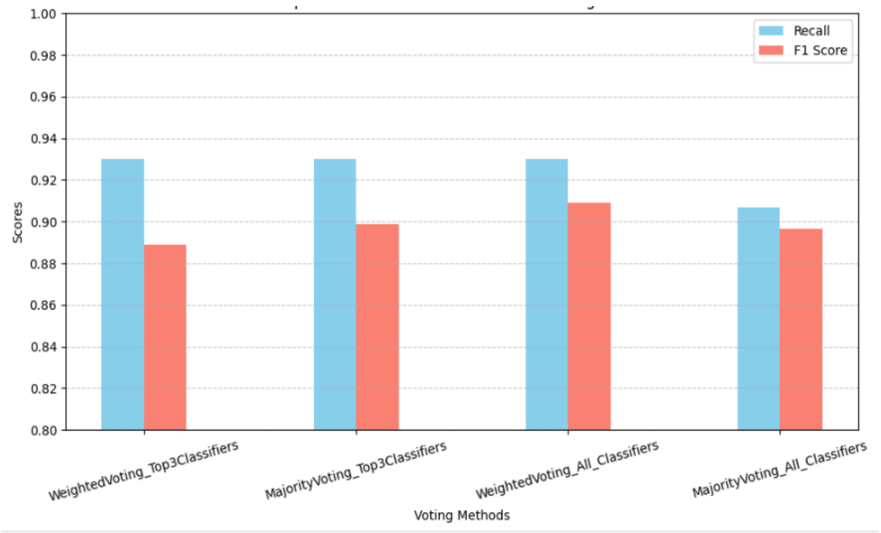
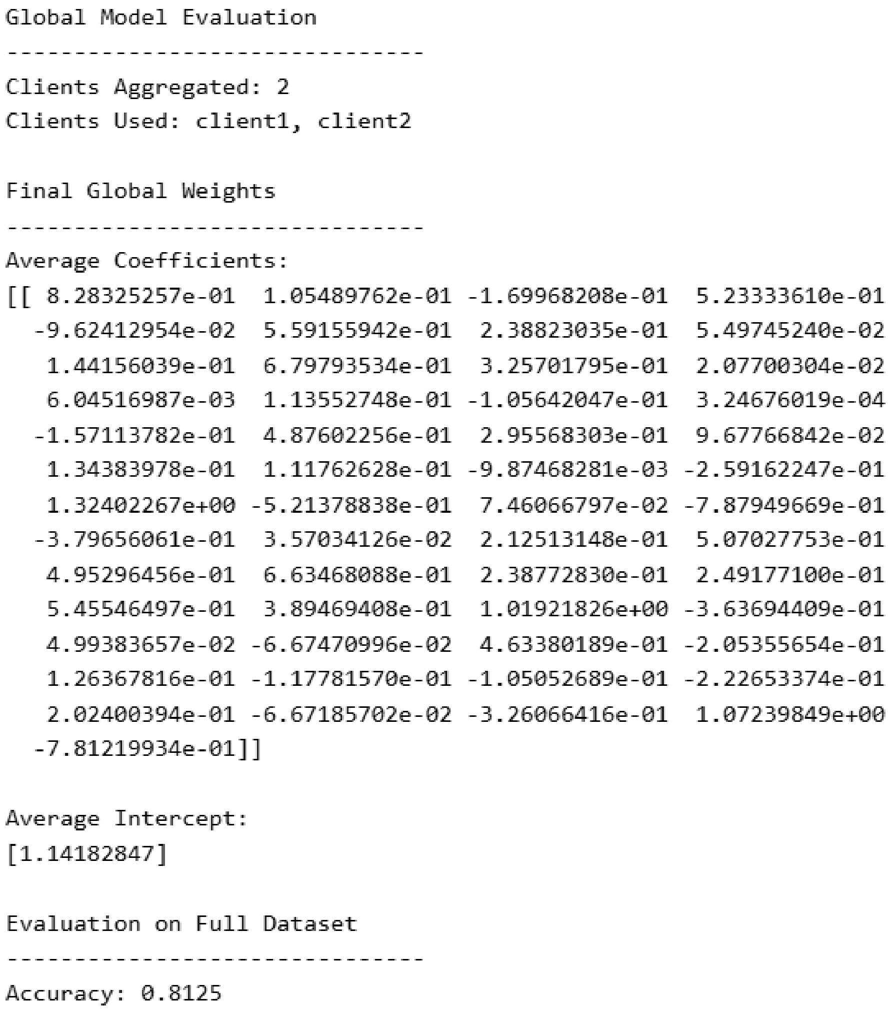


Figure 4.2: Comparison of recall and F1 score for voting methods

In our project, we built a federated learning system using Python and Flask that allowed multiple clients to work together to train a machine learning model—without ever sharing their actual data. Each client was given 120 records from the dataset and trained a logistic regression model locally on their own machine. Instead of sending their raw data to a central server, they only sent the model's learned parameters (the coefficients and intercept) to the server. The server using the /upload/<client_id> route in our Flask app, received these model updates securely and stored them separately for each client.

This setup helped us protect sensitive data, since the clients never had to share their original records. Once the server got the model updates from both clients, it used another endpoint (/aggregate) to combine these updates by simply averaging the parameters. This averaging gave us a global model that represented what both clients had learned.



```
Global Model Evaluation
------------------------------
Clients Aggregated: 2
Clients Used: client1, client2

Final Global Weights
------------------------------
Average Coefficients:
[[ 8.28325257e-01  1.05489762e-01 -1.69968208e-01  5.23333610e-01
  -9.62412954e-02  5.59155942e-01  2.38823035e-01  5.49745240e-02
   1.44156039e-01  6.79793534e-01  3.25701795e-01  2.07700304e-02
   6.04516987e-03  1.13552748e-01 -1.05642047e-01  3.24676019e-04
  -1.57113782e-01  4.87602256e-01  2.95568303e-01  9.67766842e-02
   1.34383978e-01  1.11762628e-01 -9.87468281e-03 -2.59162247e-01
   1.32402267e+00 -5.21378838e-01  7.46066797e-02 -7.87949669e-01
  -3.79656061e-01  3.57034126e-02  2.12513148e-01  5.07027753e-01
   4.95296456e-01  6.63468088e-01  2.38772830e-01  2.49177100e-01
   5.45546497e-01  3.89469408e-01  1.01921826e+00 -3.63694409e-01
   4.99383657e-02 -6.67470996e-02  4.63380189e-01 -2.05355654e-01
   1.26367816e-01 -1.17781570e-01 -1.05052689e-01 -2.22653374e-01
   2.02400394e-01 -6.67185702e-02 -3.26066416e-01  1.07239849e+00
  -7.81219934e-01]]

Average Intercept:
[1.14182847]

Evaluation on Full Dataset
------------------------------
Accuracy: 0.8125
```

Figure 4.3: Results of federated learning approach using logistic regression

When we tested this global model on the 63 records we had set aside (which the model hadn't seen before), it achieved an accuracy of 81.25%. This result showed that we could build an effective model even though the training data was split across different clients and never pooled together in one place. A key advantage of this approach was that the server did not need to retrain the model from the beginning; instead, it directly incorporated the averaged parameters into a new Logistic Regression instance and was able to generate predictions immediately. The process on the client side was simple: after training, each client saved their model's parameters as .npy files and uploaded them to the server with a single POST request. Overall, this approach gave us a lightweight, privacy-preserving, and collaborative way to train a shared model without needing to move sensitive data around or put a heavy load on the server.

4.2 Applications of the Minor Project

Early Detection of CAD: Helps in identifying patients at risk of Coronary Artery Disease at an early stage [2,9,15] using machine learning models.

- Clinical Decision Support: Assists doctors and healthcare professionals by providing data-driven predictions to support diagnosis.
- Integration with Hospital Systems: Can be embedded into electronic health records (EHR) or hospital management systems for real-time risk analysis.
- Cost-effective Screening Tool: Acts as a low-cost pre-screening method, especially valuable in rural or under-resourced healthcare setups.

4.3 Limitations of the Minor Project

- Limited Dataset Size: The accuracy and generalizability of the model may be constrained due to a small or imbalanced dataset.
- Data Quality Issues: Presence of missing, inconsistent, or noisy data can impact model performance.
- Lack of Real-Time Validation: The model may not have been tested in real-time clinical environments, limiting its practical reliability.
- Bias in Data: The dataset may reflect demographic or regional biases, affecting fairness and accuracy across diverse populations.

4.4 Future Work

- Larger and Diverse Dataset: Incorporate a more extensive and diverse dataset to improve generalizability and reduce bias.
- Feature Expansion: Include additional clinical parameters such as ECG results, imaging data, or genetic factors for more accurate predictions.
- Web or Mobile Application: Develop a user-friendly interface in the form of a web or mobile app to make the tool easily accessible to healthcare professionals and patients.
- Integration with Healthcare Systems: Work towards integrating the model into hospital information systems or wearable health monitoring devices.

References

[1] S. D. Cagle Jr. and N. Cooperstein, “Coronary Artery Disease: Diagnosis and Management,” Primary Care: Clinics in Office Practice, vol. 45, no. 1, pp. 45–61, Mar. 2018

[2] M. J. J. Ghrabat, S. H. Mohialdin, L. Q. Abdulrahman, M. H. Al-Yoonus, Z. A. Abduljabbar, D. G. Honi, V. O. Nyangaresi, I. Q. Abduljaleel, and H. A. Neamah, "Utilizing Machine Learning for the Early Detection of Coronary Heart Disease," *Engineering, Technology and Applied Science Research*, vol. 14, no. 5, pp. 17363–17375, Oct. 2024, doi: 10.48084/etasr.8171.

[3] Shahid, A.H.; Singh, M.P. A Novel Approach for Coronary Artery Disease Diagnosis using Hybrid Particle Swarm Optimization based Emotional Neural Network. *Biocybern. Biomed. Eng.* 2020, *40*, 1568–1585.

[4] M. M. Ghiasi, S. Zendehboudi, and A. A. Mohsenipour, "Decision tree-based diagnosis of coronary artery disease: CART model," *Computer Methods and Programs in Biomedicine*, vol. 192, p. 105400, Aug. 2020.

[5] Abdar M., Książek W., Acharya U.R., Tan R.S., Makarenkov V., Pławiak P. A new machine learning technique for an accurate diagnosis of coronary artery disease. Comput. Methods Programs Biomed. 2019;179:104992. doi: 10.1016/j.cmpb.2019.104992.

[6] Kolukisa B., Bakir-Gungor B. Ensemble feature selection and classification methods for machine learning-based coronary artery disease diagnosis. Comput. Stand. Interfaces. 2023;84:103706. doi: 10.1016/j.csi.2022.103706.

[7] S. Zhang, Y. Yuan, Z. Yao, X. Wang, and Z. Lei, "Improvement of the performance of models for predicting coronary artery disease based on XGBoost algorithm and feature processing technology," *Electronics*, vol. 11, no. 3, p. 315, Jan. 2022.

[8] Nasarian, E.; Abdar, M.; Fahami, M.A.; Alizadehsani, R.; Hussaind, S.; Basiri, M.E.; Zomorodi-Moghadam, M.; Zhou, X.J.; Pławiak, P.; Acharya, U.R.; et al. Association between work-related features and coronary artery disease: A heterogeneous hybrid feature selection integrated with balancing approach. *Pattern Recognit. Lett.* 2020, *133*, 33–40.

[9] Alizadehsani, R.; Abdar, M.; Roshanzamir, M.; Khosravi, A.; Kebria, P.M.; Khozeimeh, F.; Nahavandi, S.; Sarrafzadegan, N.; Acharya, U.R. Machine learning-based coronary artery disease diagnosis: A comprehensive review. *Comput. Biol. Med.* 2019, *111*, 103346.

[10] S. Zhang, Y. Yuan, Z. Yao, J. Yang, X. Wang, and J. Tian, "Coronary artery disease detection model based on class balancing methods and LightGBM algorithm," *Electronics*, vol. 11, no. 9, p. 1495, May 2022.

[11] O. Hrizi, K. Gasmi, A. Alyami, A. Alkhalil, I. Alrashdi, A. Alqazzaz, L. Ben Ammar, M. Mrabet, A. E. M. Abdalrahman, and S. Yahyaoui, "Federated and ensemble learning framework with optimized feature selection for heart disease detection," *AIMS Mathematics*, vol. 10, no. 3, pp. 5185–5208,Mar.2025.

[12] Ishaq A., Sadiq S., Umer M., Ullah S., Mirjalili S., Rupapara V., Nappi M. Improving the prediction of heart failure patients' survival using SMOTE and effective data mining techniques. IEEE Access. 2021;9:39707–39716. doi: 10.1109/ACCESS.2021.3064084.

[13] Ogundepo E.A., Yahya W.B. Performance analysis of supervised classification models on heart disease prediction. Innov. Syst. Softw. Eng. 2023;19:129–144. doi: 10.1007/s11334-022-00524-9.

[14] S. A. J. Zaidi, A. Ghafoor, J. Kim, Z. Abbas, and S. W. Lee, "HeartEnsembleNet: An innovative hybrid ensemble learning approach for cardiovascular risk prediction," *NPJ Digital Medicine*, vol. 7, no. 1, p. 97, 2024.

[15] S. Alasmari, R. AlGhamdi, G. G. Tejani, S. K. Sharma, and S. J. Mousavirad, "Federated Learning-Based Multimodal Approach for Early Detection and Personalized Care in Cardiac Disease," *Frontiers in Physiology*, vol. 16, Apr. 2025, Art. no. 1563185, doi: 10.3389/fphys.2025.1563185.

[16] Libby P, Bonow RO, Mann DL, Tomaselli GF, Bhatt D, Solomon SD, Braunwald E (2021) Braunwald's heart disease—E-book: a textbook of cardiovascular medicine. Accessed 6 Nov 2022.

[17] T. Chen, S. Zhao, S. Shao, and S. Zheng, "Non-invasive diagnosis methods of coronary disease based on wavelet denoising and sound analyzing," *Saudi Journal of Biological Sciences*, vol. 24, no. 3, pp. 526–536, Mar. 2017, doi: 10.1016/j.sjbs.2017.01.023.

[18] G. C. M. Siontis, D. Mavridis, J. P. Greenwood, B. Coles, A. Nikolakopoulou, P. Jüni, G. Salanti, and S. Windecker, "Outcomes of non-invasive diagnostic modalities for the detection of coronary artery disease: network meta-analysis of diagnostic randomised controlled trials," *European Heart Journal*, vol. 39, no. 26, pp. 2484–2491, 2018, doi: 10.1093/eurheartj/ehy196.

[19] S. Heydarian, "Classification of Coronary Artery Disease," *Kaggle,* 2021.