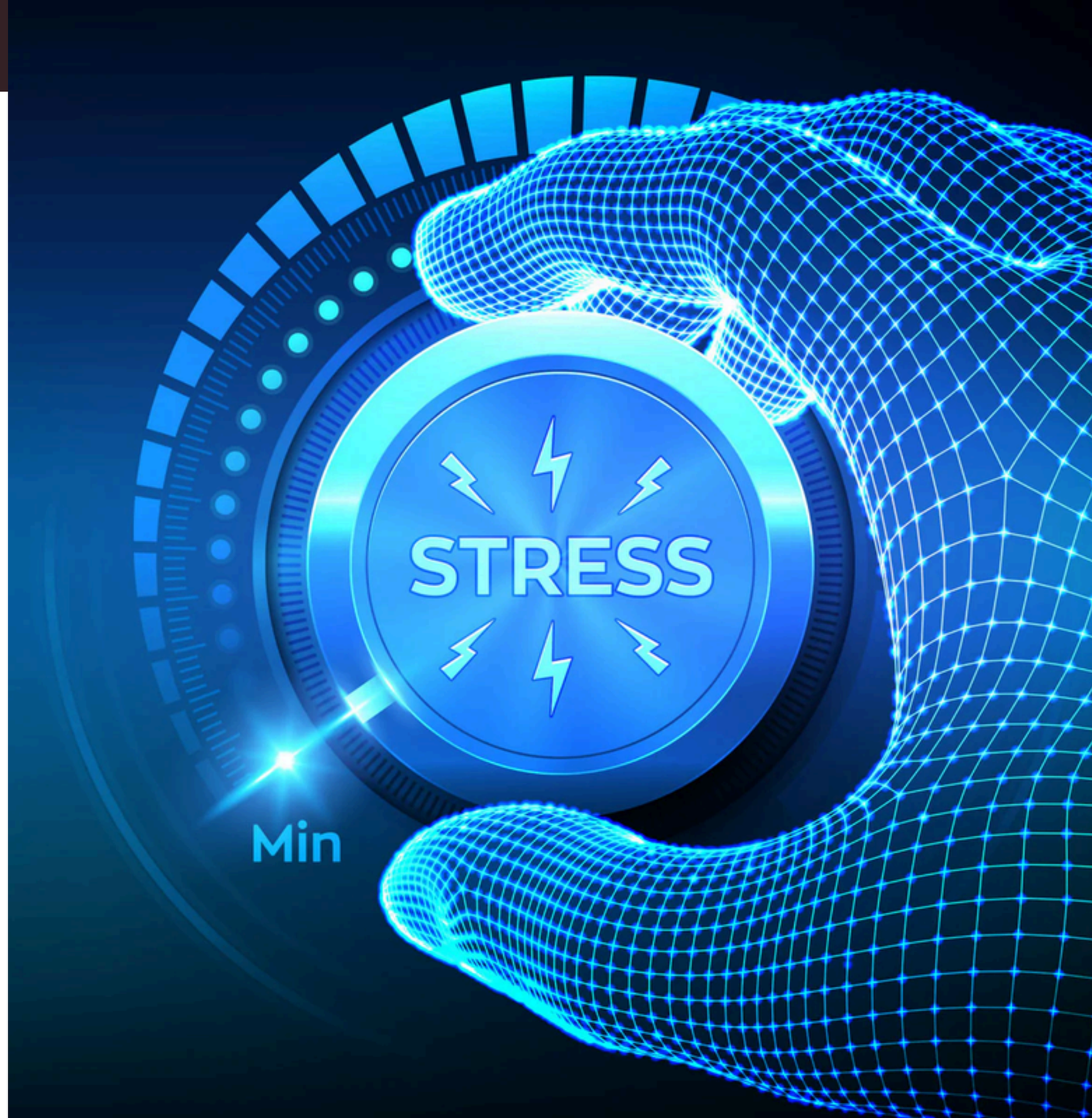
Cardiovascular Diease Prediction among MBA students using lifestyle factors

Course :-

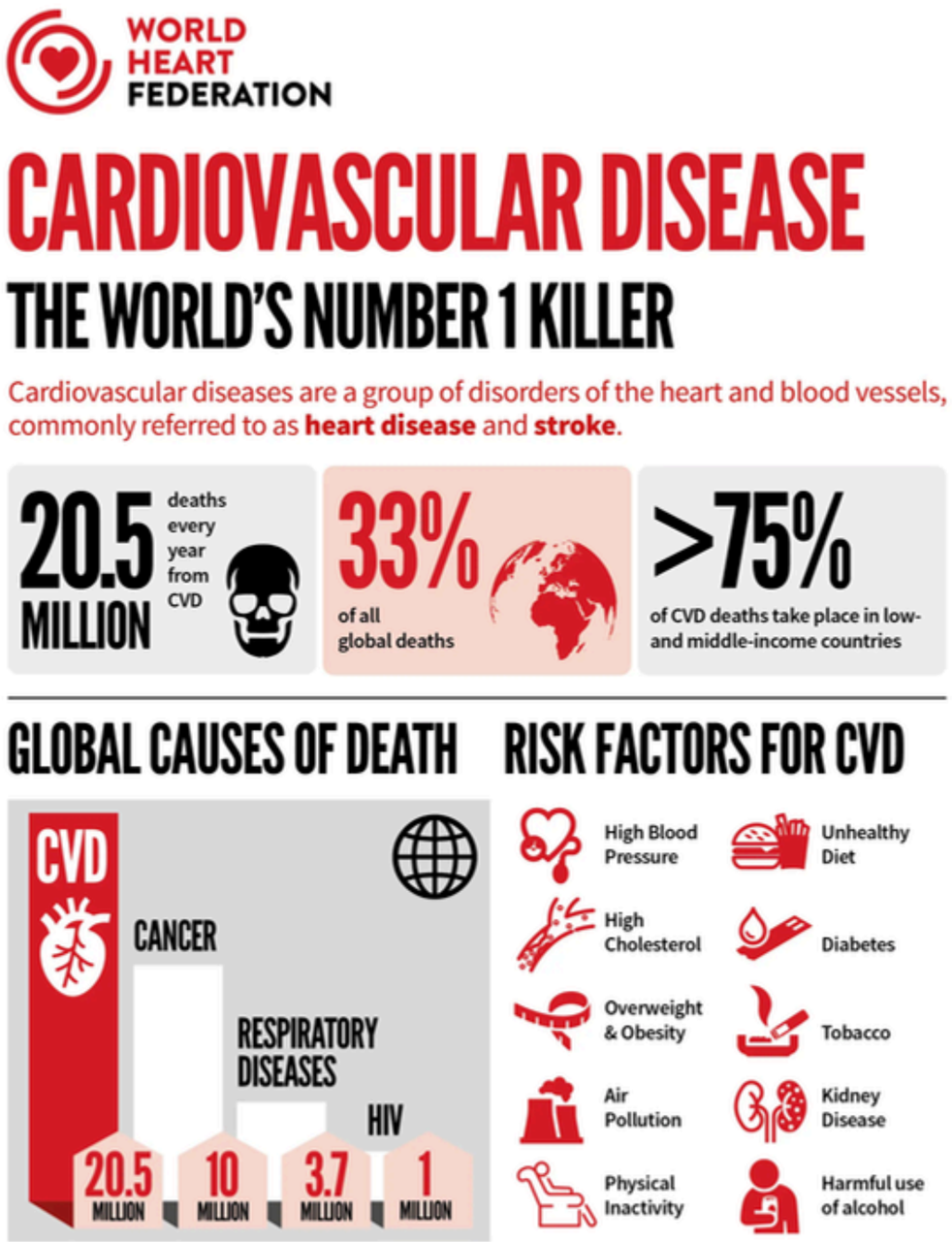
Foundation of Big Data Analytics using Python

Presented By:-

- Akshit Kansal - 065008
- Mayank Jha - 065036
- Parv - 065040
- Awantika Kholia - 065060



Problem Statement



Why This Project?

Heart disease is a leading cause of death globally.
Early prediction allows timely intervention & reduced healthcare burden.
Traditional screening often fails to identify high-risk patients early.

Business objective

Build a machine learning model to predict heart disease risk using lifestyle, demographic, and medical data to help insurance company and hospitals for better preventive care

Key Success Criteria

Recall > 50% for heart disease cases
Interpretable, clinically meaningful model
Balanced trade-off between recall & precision

Dataset Overview

Category	Details
Total Records	**308,854**
Number of Features	**19**
Target Variable	**Heart_Disease (Yes/No)**
Missing Values	**None (Clean Dataset)**
Data Source	Kaggle – Cardiovascular Disease Risk Dataset

Class Distribution (Imbalanced Data)

Class	Count	Percentage
Heart Disease – No	**283,883**	**91.90%**
Heart Disease – Yes	**24,971**	**8.10%**

Insight: Severe class imbalance → default models tend to classify most cases as No.

Feature Categories

Feature Group	Examples
Demographics	Age Category, Sex
Existing Health Conditions	Diabetes, Arthritis, Cancer, Kidney Disease, Depression
Lifestyle & Behavior	Smoking History, Exercise, Alcohol Intake, Food Consumption
Physical Measurements	Height, Weight, BMI

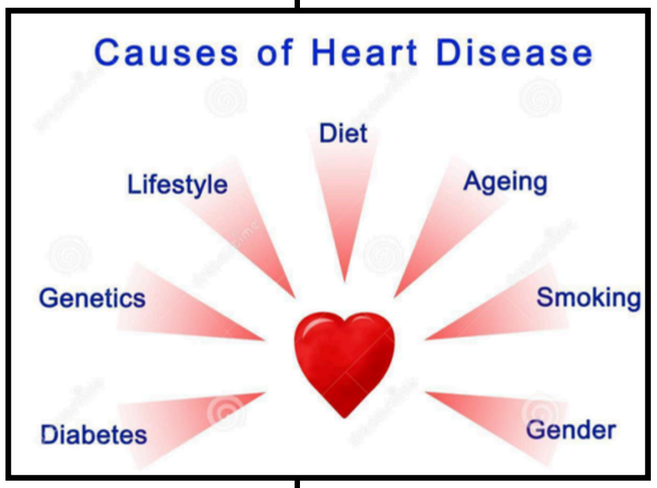
Data Preprocessing Workflow

Step	Description
1. Data Inspection	Checked data types, distributions, outliers, and validated dataset completeness.
2. Categorical Encoding	Applied One-Hot Encoding to 11 categorical variables, expanding features from 18 → 48.
3. Target Encoding	Converted target labels “Yes/No” → 1/0 for model compatibility.
4. Train–Test Split	Split dataset into 80% training and 20% testing using stratified sampling to maintain class imbalance.
5. Class Imbalance Handling	Applied SMOTE on training data → achieved balanced classes: 197,666 Yes & 197,666 No.
6. Threshold Optimization	Tested probability thresholds from 0.1 to 0.9 to improve recall for heart disease cases.

Key Predictive Features

Demographic Features

- Age (strongest natural risk factor)
- Biological sex differences



Medical Condition Features

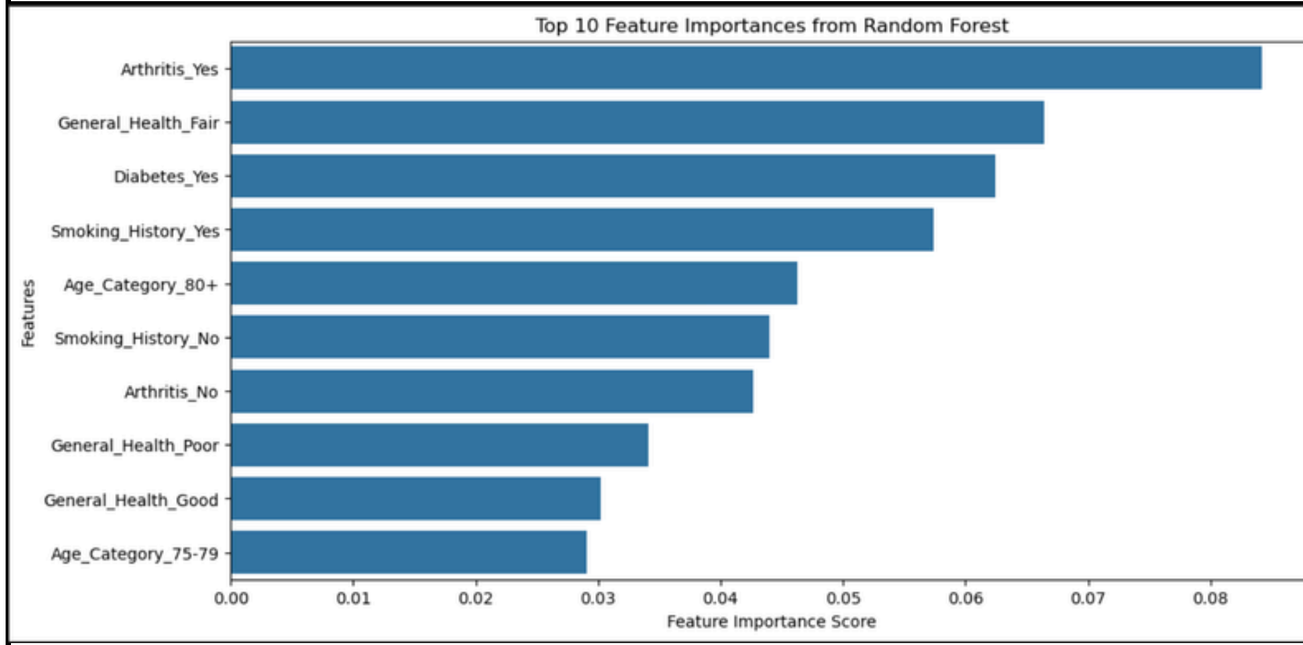
- Diabetes — increases risk
- Arthritis — chronic inflammation
- Kidney disease — linked to cardiovascular stress
- Depression — associated with heart health

Lifestyle & Behavioral Factors

- Smoking history
- Physical exercise
- Dietary habits (fruits, vegetables, fried foods)
- Alcohol consumption

Physical Metrics

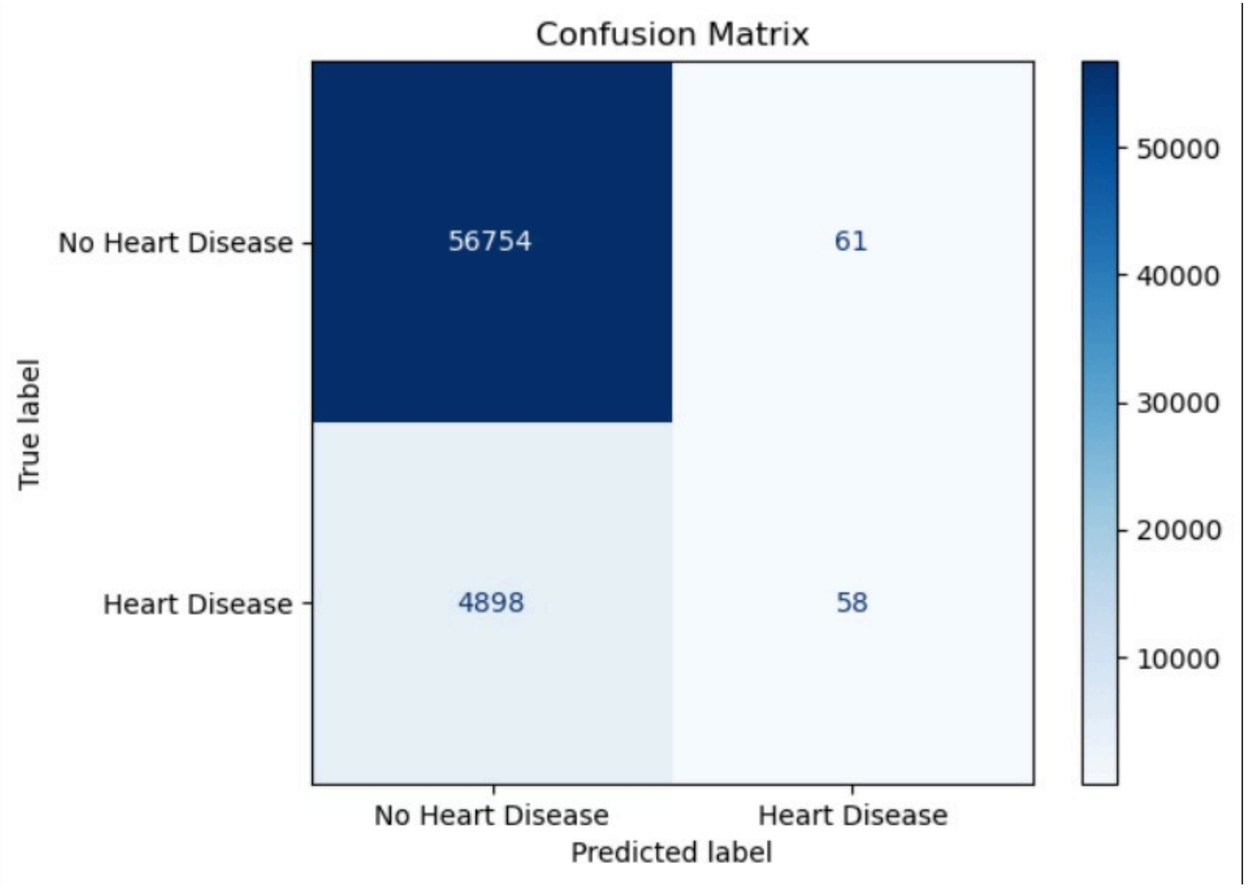
- BMI
- Height/weight combination



Logistic Regression

Logistic Regression (Baseline Performance)

- Accuracy: 91.93%
- Recall (Heart Disease): 6%
- Observation:
- Model predicts almost all cases as “No Heart Disease”.
- Fails to detect minority class → unsuitable for medical screening.



The model barely detects heart-disease cases, showing extremely low recall.

Why the Models Performed Poorly

- *Extreme class imbalance hides heart disease cases.*
- *Accuracy is not meaningful when 92% of data belongs to one class.*
- *Recall is the priority in medical prediction — missing a sick patient is very costly.*

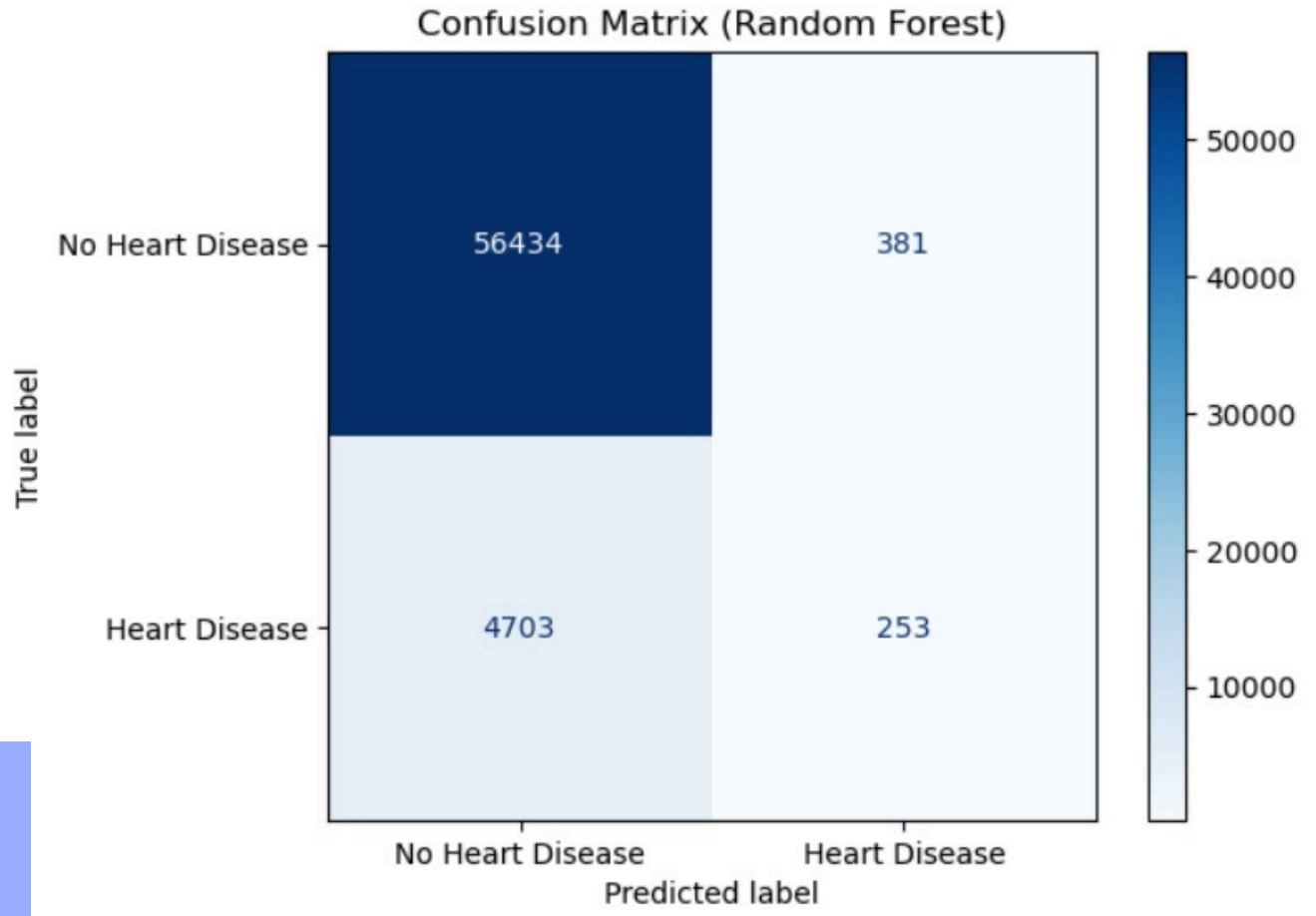
Model Evaluation (Before Threshold Tuning)

Performance Overview

- Both Logistic Regression and Random Forest achieved high overall accuracy, but this is misleading due to severe class imbalance.
- Models struggled to identify positive heart-disease cases, resulting in very low recall.

Random Forest (Default Setting)

- Accuracy: 91.73%
- Recall (Heart Disease): 6%
- Key Issue:
- Even with SMOTE applied during training, the test set imbalance still causes poor detection of true positives.



True Positives = extremely low (253)
False Negatives = very high (4703)

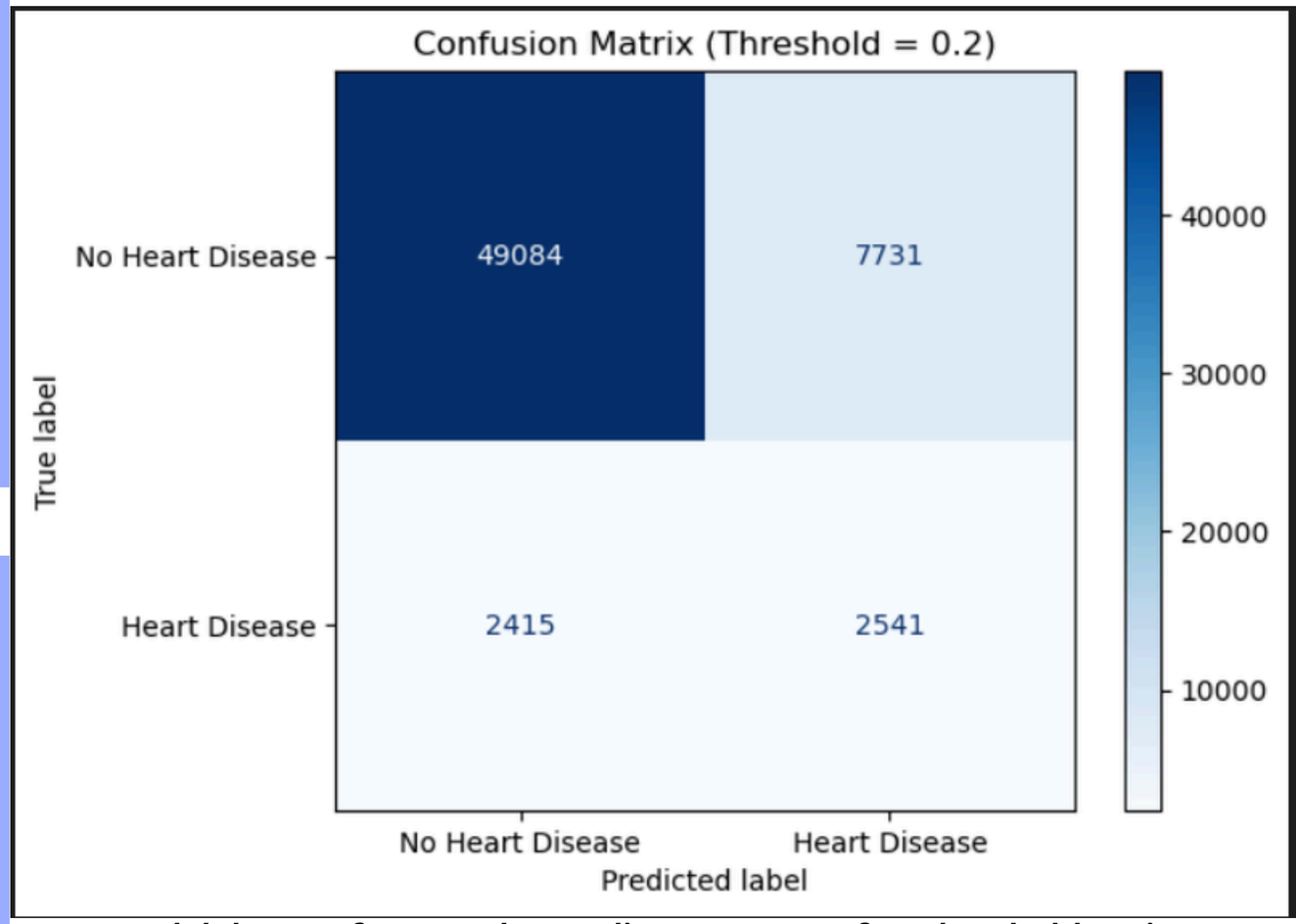
Model Evaluation (After Threshold Tuning)

Improved Detection of Heart Disease

- Lowering the threshold to 0.2 greatly increased the model's sensitivity.
- The model now correctly identifies 2,541 true heart-disease cases (vs only 253 before tuning).

Confusion Matrix Interpretation

- True Negatives (49,084): Most healthy individuals correctly classified
- False Positives (7,731): More healthy people flagged as at-risk (acceptable in screening)
- False Negatives (2,415): Reduced significantly — fewer missed heart-disease cases
- True Positives (2,541): Major improvement in catching actual heart-disease cases



Model detects far more heart-disease cases after threshold tuning, improving recall to 53%.

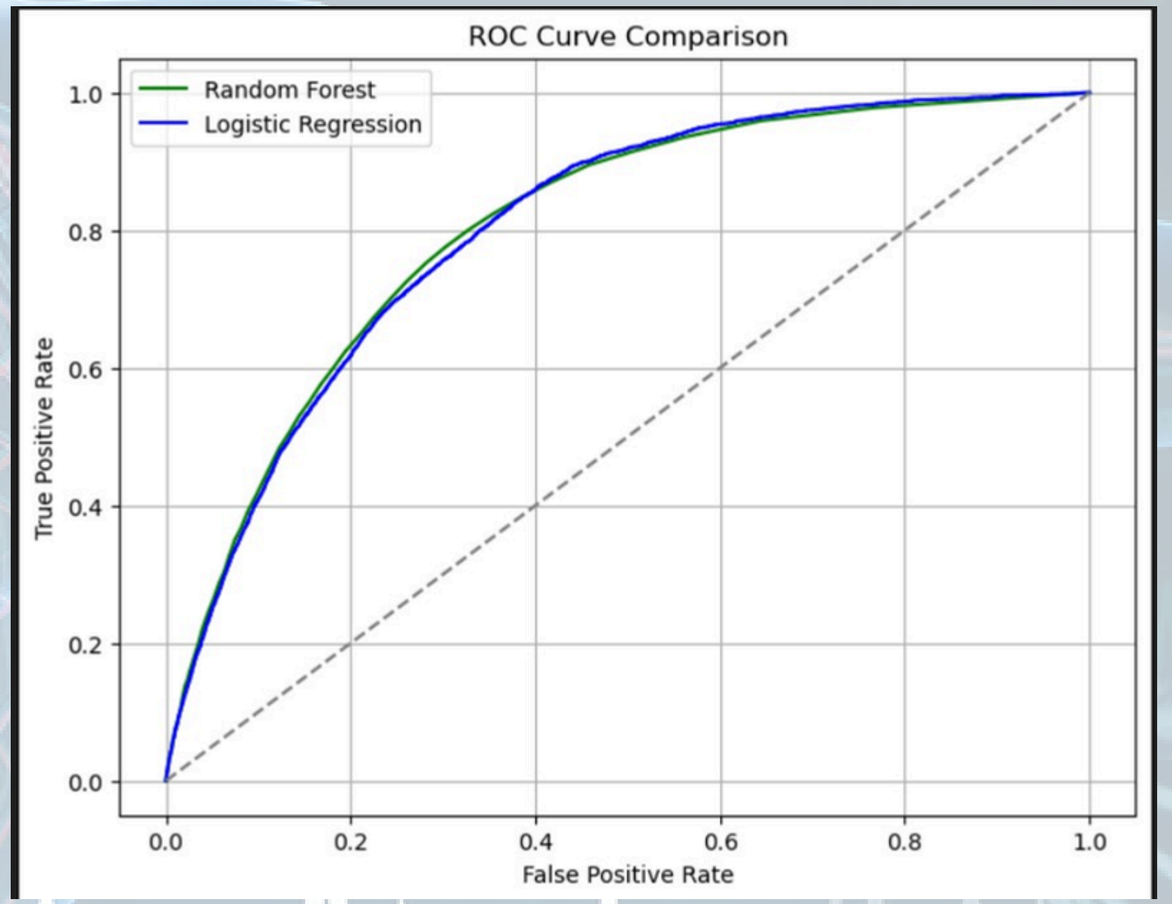
Random Forest VS Logistic Regression

ROC Curve Insights

- Both models show good discriminative power, with AUC values close to each other.
- Random Forest performs slightly better across most thresholds.
- Higher AUC indicates the model is effective at distinguishing between
- Heart Disease vs No Heart Disease

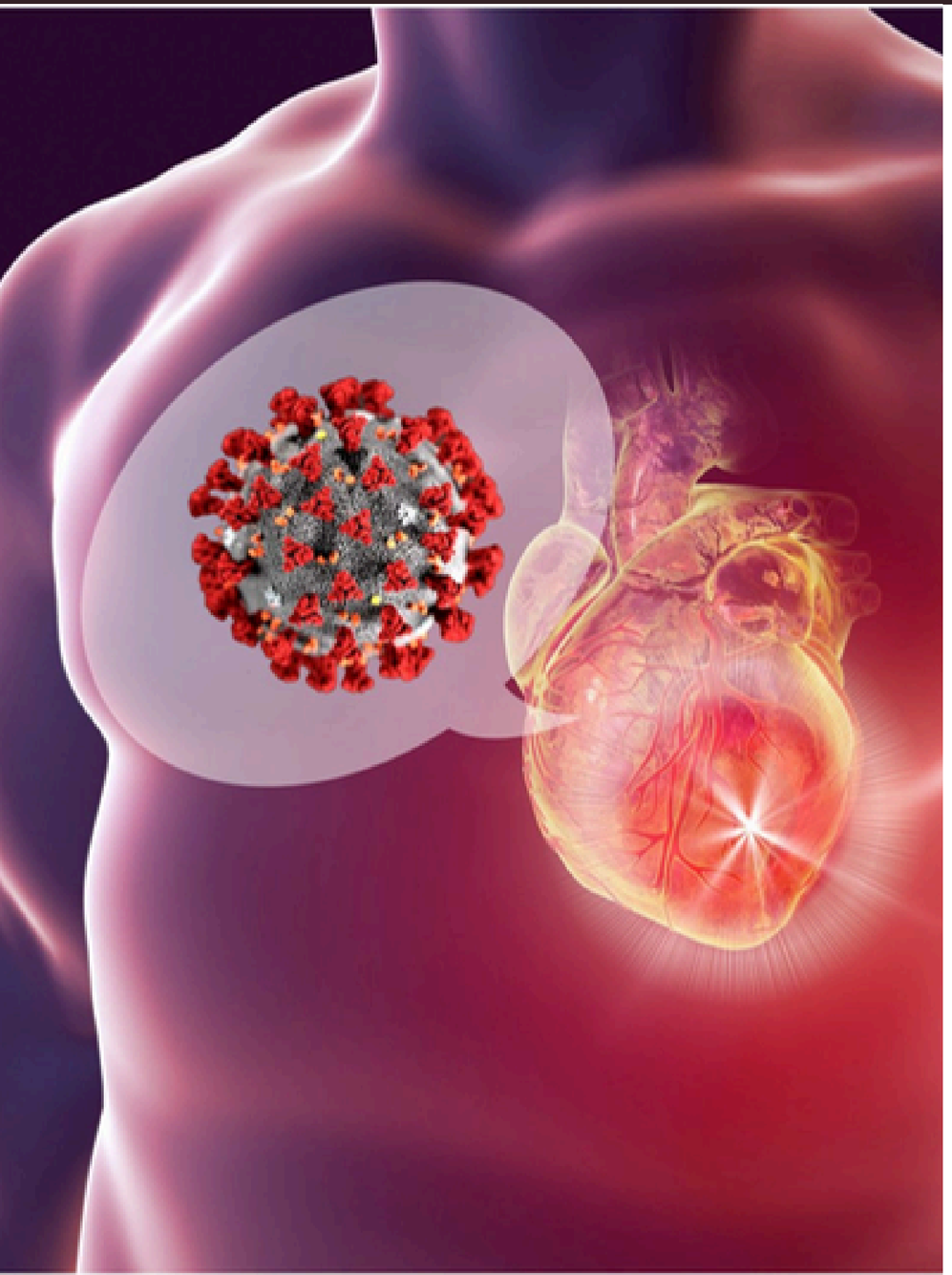
Why It Matters?

- ROC helps identify which model is more suitable for threshold tuning.
- Confirms that Random Forest is the stronger candidate for improving recall.
- Supports the decision to apply threshold = 0.2 later.



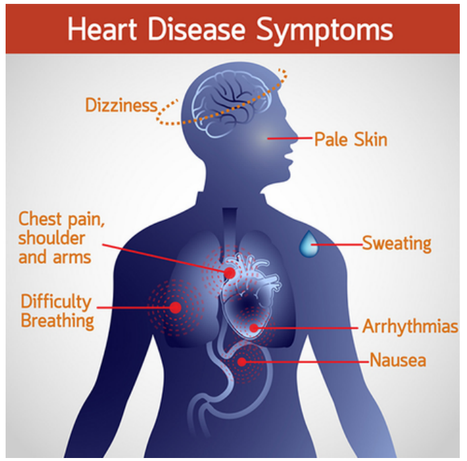
The ROC curve compares Logistic Regression and Random Forest, showing that Random Forest has slightly better overall classification performance.

Feature Importance & Key Insights



Top Predictors Identified by the Model

1. General Health (Fair, Good, Poor)
2. Age Category (70–74, 75–79, 80+)
3. Diabetes
4. BMI
5. Smoking History
6. Arthritis

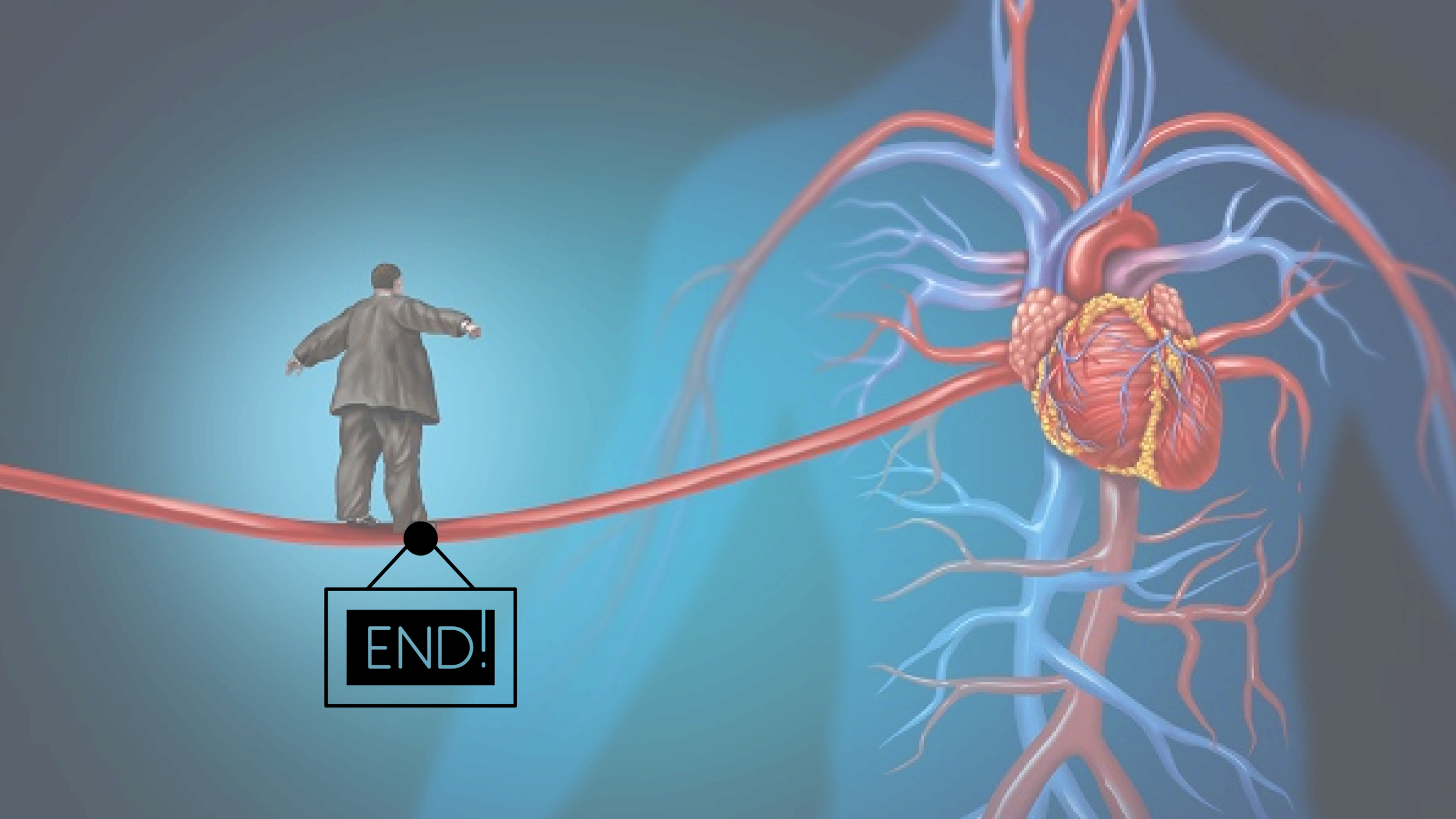


What This Means?

- Self-reported health is one of the strongest indicators of heart-disease risk.
- Older age groups show significantly higher risk levels.
- Chronic conditions like diabetes and arthritis contribute heavily to cardiovascular stress.
- Lifestyle factors (BMI, smoking) remain major modifiable risks.

Key Insight:

The model's most important features closely align with established medical research, improving interpretability and trust in predictions.


END!