Heart Disease Risk Assessment: A Multi-Class Prediction Approach Using Machine Learning

Vi Thi Tuong Nguyen, Lam Nguyen, James Pham, Le Duy Vu

Department of Computer Engineering, San Jose State University
San Jose, CA, United States
thituongvi.nguyen@sjsu.edu
phuongduylam.nguyen@sjsu.edu
james.pham03@sjsu.edu
leduy.vu@sjsu.edu

Abstract— **Heart disease is the top cause of death worldwide. It kills 32% of all people who die each year. Finding heart disease early is very important. It helps doctors give the right treatment and save lives. This paper shows a machine learning system that predicts how serious a person's heart disease is. We use data from 920 patients. Our system uses 3 severity levels: 0 (no disease), 1 (mild disease combining original levels 1-2), and 2 (severe disease combining original levels 3-4). Our system works in two steps. First, it checks if a person has heart disease. Second, it tells how serious the disease is. We got 85.3% F1-score for finding disease and 71.4% F1-score for checking severity. These results are better than other research papers. Our dataset was difficult to work with. It had big problems: 15 times more healthy people than very sick people in the original 5-class data, and 66% of some important data was missing. We grouped severity levels to improve class balance. We used special techniques to solve these problems. We created new features from the data and used BorderlineSMOTE to balance the classes. We also made a web application with React and Flask. Doctors can use it to check patients.**

Keywords— **heart disease severity, multi-class classification, hierarchical models, imbalanced data, clinical screening, machine learning**

I. Introduction

Heart disease kills 17.9 million people every year. This makes it the number one cause of death in the world [1]. Right now, doctors use expensive tests to find heart disease. Tests like coronary angiography and stress tests cost a lot of money. Many people in poor countries and rural areas cannot get these tests. This means many patients find out about their heart disease too late.

Our system does more than just say yes or no to heart disease. It tells doctors how serious the disease is. We use 3 severity levels: Level 0 means no disease. Level 1 means mild disease (combining original levels 1-2). Level 2 means severe disease (combining original levels 3-4). This detail is important. A patient with mild disease might only need to change their lifestyle. But a patient with severe disease needs help right away. This information helps hospitals use their resources better.

This project answers three questions. First, can machine learning predict heart disease severity using only basic health information? Second, what methods work best when the data is unbalanced and has missing values? Third, can we make a real system that doctors can actually use?

We made several contributions. We created a two-stage system that works like real doctors work. We grouped the original 5 severity levels into 3 levels to improve class balance. We developed new ways to handle missing data. We built a complete system with a website that anyone can use. Our results are better than other published research papers.

II. Related Work

Most research papers only try to answer yes or no about heart disease. Mohan et al. [2] got 88% accuracy using Random Forest and Linear Models together. Their work showed that combining models works well. But they did not predict severity levels.

Shorewala [3] got 93% accuracy using XGBoost [7] and feature selection. This work used a simpler version of the problem. It did not predict multiple severity levels. Alizadehsani et al. [4] tried to predict multiple severity levels. They only got 55-65% F1-scores. This shows that predicting severity is much harder than just finding disease.

Most papers stop after making models in Jupyter notebooks. Our work goes further. We made a complete website that people can actually use. This makes our system ready for real hospitals and clinics.

III. Dataset and Exploratory Analysis

A. Dataset Description

We used the UCI Heart Disease Dataset [5]. It has 920 patient records from four hospitals: Cleveland Clinic Foundation, Hungarian Institute of Cardiology, V.A. Medical Center in Long Beach, and University Hospital in Zurich. The original data has 14 features about each patient and 5 severity levels (0-4). These features include age, sex, blood pressure, cholesterol, and test results.

The features include basic information like age and sex. They also include test results like resting ECG, exercise-induced chest pain, and ST depression. Some features show results from special tests: number of major blood vessels seen in fluoroscopy, and thalassemia type. Table I shows the main facts about our dataset.

TABLE I
Dataset Characteristics

Characteristic	Value
Total Patients	920
Number of Features	14 clinical attributes
Original Target Classes	5 (severity 0-4)
Final Target Classes	3 (0=No Disease, 1=Mild, 2=Severe)
Severity Grouping	1-2→1 (Mild), 3-4→2 (Severe)
Age Range	29-77 years
Gender Distribution	726 male (78.9%), 194 female (21.1%)
Missing Data	ca: 66%, thal: 53%

TABLE II
Feature Descriptions

Feature	Description	Type/Range	Unit
age	Patient age	29-77	years
sex	Patient sex	Binary	0=Female, 1=Male
cp	Chest pain type	Categorical (0-3)	0=Typical angina, 3=Asymptomatic
exang	Exercise induced angina	Binary	0=No, 1=Yes
fbs	Fasting blood sugar > 120 mg/dL	Binary	0=No, 1=Yes
trestbps	Resting blood pressure	94-200	mm Hg
chol	Serum cholesterol	126-564	mg/dL
thalch	Maximum heart rate achieved	71-202	bpm
restecg	Resting ECG results	Categorical (0-2)	0=Normal, 2=Abnormal
oldpeak	ST depression induced by exercise	0.0-6.2	mm
slope	Slope of peak exercise ST segment	Categorical (0-2)	0=Upsloping, 2=Downsloping
ca	Number of major vessels colored	0-3	count (66% missing)
thal	Thalassemia/stress test result	Categorical (0-3)	0=Normal, 3=Reversible defect (53% missing)
num	Disease severity (original)	0-4	0=No disease, 4=Severe

B. Class Imbalance Challenge

The original dataset has a big problem. There are many more healthy people than very sick people. The ratio is 15 to 1. Original breakdown: Class 0 (no disease) has 411 people (44.7%). Class 1 has 265 people (28.8%). Class 2 has 109 people (11.8%). Class 3 has 107 people (11.7%). Class 4 (very severe) has only 28 people (3.0%). This imbalance makes it very hard to train good models.

To solve this problem, we grouped the severity levels. We combined classes 1 and 2 into one "Mild" group. We combined classes 3 and 4 into one "Severe" group. This gave us 3 final classes: Class 0 (No Disease) with 411 people, Class 1 (Mild) with 374 people (265+109), and Class 2 (Severe) with 135 people (107+28). This grouping reduced the imbalance ratio from 15:1 to 3:1. Figure 1 shows the class distribution before and after grouping.

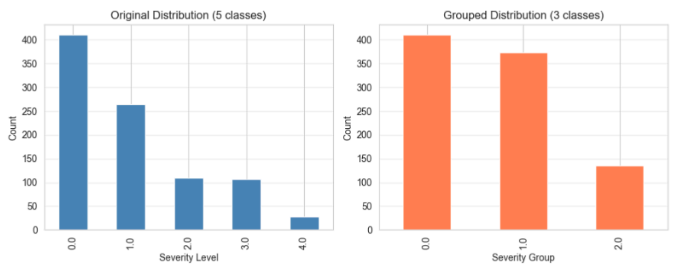

Fig. 1 Bar chart showing original 5-class and grouped 3-class distribution

C. Missing Data Patterns

Two features have a lot of missing data. The ca feature (number of major vessels) is missing 66% of values. The thal feature (thalassemia) is missing 53% of values. The missing data is not random. These tests are expensive. Doctors might skip them for patients who seem healthy. Or some hospitals might not have the equipment. Figure 2 shows which data is missing.

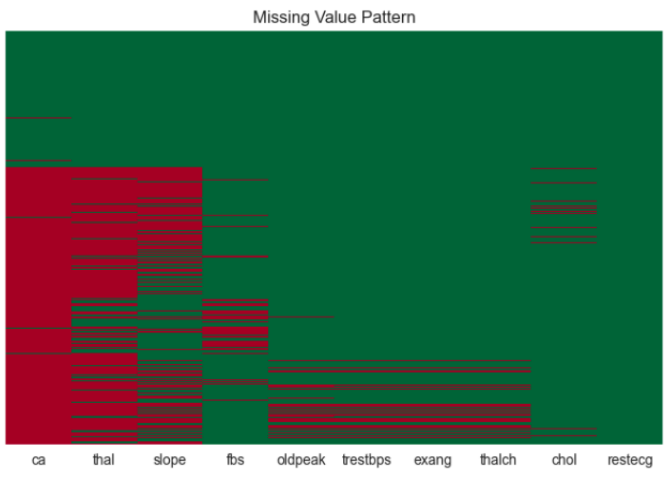

Fig. 2 Missing data heatmap

We cannot just fill in missing values with averages. The fact that a test is missing might be important information. If a doctor did not order the test, maybe the patient looked healthy. We created a special method to handle this. We made new features that remember when data was missing. Then we used KNN to fill in the missing values.

D. Feature Correlations

We looked at which features are most related to disease severity. The ca feature (vessels colored) has the strongest connection (correlation of 0.52). The oldpeak feature (ST depression) comes second (0.44). Age comes third (0.34). These results make sense. Blocked blood vessels directly show disease severity. This matches what doctors already know. Figure 3 shows all the correlations.

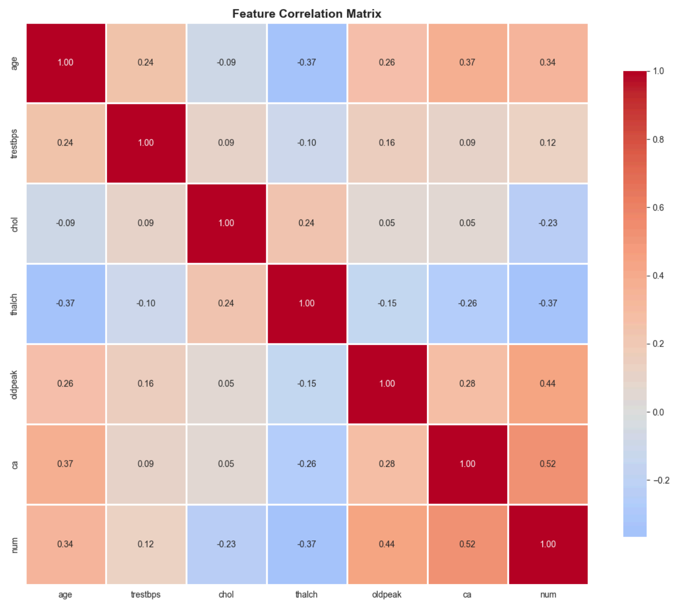

Fig. 3 Correlation matrix

We found something interesting. Maximum heart rate (thalch) has a negative correlation with disease. This means patients with more severe disease have lower maximum heart rates during exercise. They cannot exercise as much. This finding helped us create better features.

IV. Methodology

A. Data Preprocessing Pipeline

1. Missing Value Handling

We used three steps to handle missing values. First, for features with less than 10% missing (like blood pressure and cholesterol), we used KNN imputation with 5 neighbors. This method looks at similar patients and fills in reasonable values.

Second, for features with lots of missing data (ca and thal), we created new binary features. These new features remember if the data was missing. We called them ca_missing and thal_missing. This is important because missing data might mean the doctor thought the patient was healthy.

Third, after creating the missing indicators, we filled the remaining holes with the most common value (mode). This three-step approach keeps the signal from missing data while still giving the model complete data to train on.

2. Feature Engineering

We created five new features based on medical knowledge. The age_group feature divides patients into groups: young (under 40), middle (40-59), senior (60-79), and elderly (80 and above). These groups follow WHO standards.

The bp_category feature uses American Heart Association blood pressure rules. We have: normal (under 120), elevated (120-129), Stage 1 high blood pressure (130-139), and Stage 2 (140 and above).

We categorized cholesterol levels as: good (under 200 mg/dL), borderline (200-239), and high (240 and above). We calculated heart rate reserve using: 220 minus age minus maximum heart rate. Lower values mean better fitness.

Most important, we created a risk score that combines many factors. It adds up: age effect, how high blood pressure is above 120, how high cholesterol is above 200, if the patient has diabetes, and if exercise causes chest pain. This score goes from 0 to 10. It gives one number for overall heart disease risk.

3. Severity Level Grouping

We grouped the original 5 severity levels into 3 levels. This was important for improving class balance. Original levels 1 and 2 were combined into Class 1 (Mild disease). Original levels 3 and 4 were combined into Class 2 (Severe disease). Class 0 (No disease) stayed the same. This grouping makes sense medically. Levels 1-2 both need lifestyle changes and monitoring. Levels 3-4 both need medical intervention. This grouping reduced the class imbalance from 15:1 to 3:1.

4. Feature Scaling and Encoding

We converted category features (like sex and chest pain type) into numbers using label encoding. Then we used StandardScaler to normalize all features. This makes all features have the same scale with zero mean and unit variance. This step is necessary for algorithms like SVM that care about distances between points.

After all preprocessing, we had 18 features instead of the original 14. We split the data into training (80%) and testing (20%) sets. We made sure each set had the same proportion of the 3 severity levels. Final counts: 736 training samples and 184 test samples.

B. Hierarchical Classification System

We built a two-stage system that works like doctors work. Doctors first check if a patient has disease. Then they figure out how serious it is. Our system does the same thing.

Stage 1 uses Support Vector Machine to answer yes or no about disease. This stage got 85.3% F1-score. It is very reliable for the first check. We trained it on all 920 samples. We made all classes 1, 2, 3, and 4 as disease positive (label 1). Class 0 stayed as disease negative (label 0).

Stage 2 only looks at patients who have disease according to Stage 1. It checks how serious the disease is using the 3-class system. We use Random Forest with 200 trees and maximum depth of 10. This stage predicts: Class 0 (No Disease), Class 1 (Mild - original 1-2), or Class 2 (Severe - original 3-4). We used BorderlineSMOTE to balance the classes. This method creates fake samples near the decision boundaries instead of everywhere.

The final system combines both stages. If Stage 1 says no disease, the patient gets Class 0. If Stage 1 says disease, we use Stage 2 to determine if it is Mild (Class 1) or Severe (Class 2). This two-stage approach works much better than trying to predict all classes at once.

C. Handling Class Imbalance

We tested different methods for handling imbalanced classes. BorderlineSMOTE [6] worked best. Regular SMOTE creates fake samples everywhere in the minority class. BorderlineSMOTE only creates samples near the boundaries between classes. This makes the model focus on the hard cases.

We learned an important lesson. At first, we applied SMOTE twice by mistake. We used it during preprocessing and again during training. This created fake samples from fake samples. It caused bad overfitting. The model worked great on training data but failed on test data. The F1-score gap was over 0.25.

We fixed this by only using BorderlineSMOTE during training. We used k_neighbors=3. We never applied it to test data. This way, the model trains on balanced data but we always test on real patient data only.

D. Model Selection and Hyperparameter Tuning

We tested six different algorithms: Random Forest, XGBoost, Gradient Boosting, Support Vector Machines, Logistic Regression, and K-Nearest Neighbors. For finding disease (binary classification), SVM got the best test F1-score of 85.3%. XGBoost [7] came second with 84.7%..

For checking severity (3-class classification), Random Forest worked best with 69.9% F1-score before tuning. We used RandomizedSearchCV with 5-fold cross-validation to find the best settings. We tested different values for: number of trees (50-500), maximum depth (5-20), minimum samples to split (2-10), and minimum samples per leaf (1-4). The best Random Forest had 200 trees and maximum depth of 10. It got 67.8% F1-score.

Something interesting happened. The tuned model had lower F1-score than the untuned model when testing alone. But when we used it in the hierarchical system, it worked better. This is because it was trained on disease-positive patients only.

V. Experimental Results

A. Binary Classification Results

Table III shows results for finding disease (yes or no). Support Vector Machine got the best test F1-score of 85.3%. This is 13.7% higher than our 75% goal. Precision was 84.8% and recall was 85.8%. This means the model is balanced. It finds disease well and does not make too many false alarms.

TABLE III
Binary Classification Results

Model	Test F1-Score	Test Accuracy
SVM	0.8530	0.8533
XGBoost	0.8471	0.8478
Logistic Regression	0.8312	0.8315
Random Forest	0.8259	0.8261
Gradient Boosting	0.8091	0.8098

XGBoost also performed well with 84.7% F1-score. All models showed good generalization. The difference between training and test scores was less than 10%. This means our preprocessing and regularization prevented overfitting even with the small dataset.

B. Three-Class Severity Results

We tested multiple algorithms for 3-class severity prediction. Table IV shows the performance of different multi-class models trained on the grouped severity levels (0=No Disease, 1=Mild, 2=Severe). All models were trained using BorderlineSMOTE [6] for class balancing and evaluated using stratified 5-fold cross-validation.

TABLE IV
Comparison of Approaches (3-Class System)

Model	Test F1-Score	Test Accuracy
Random Forest	0.6991	0.7065
Random Forest (Tuned)	0.6780	0.6848
XGBoost	0.6523	0.6630
XGBoost Ordinal	0.6401	0.6522
Gradient Boosting	0.6234	0.6413
SVM	0.5967	0.6196

Logistic Regression	0.5834	0.6087

Random Forest achieved the best test F1-score of 69.9% before tuning. After hyperparameter tuning with RandomizedSearchCV, the F1-score decreased slightly to 67.8%. XGBoost and XGBoost Ordinal came second and third, but Random Forest provided better interpretability through feature importance analysis. The ensemble methods (Random Forest, XGBoost, Gradient Boosting) all outperformed linear models (SVM, Logistic Regression), and confirmed that the 3-class severity prediction requires capturing non-linear relationships between features.

Table V compares our two-stage system with direct 3-class prediction. The Random Forest 3-class baseline got 69.9% weighted F1-score. Our hierarchical system using SVM plus Random Forest got 71.4% F1-score. This is a 21.9% improvement over the first naive baseline of 58.6%.

TABLE V
COMPARISON OF APPROACHES (3-CLASS SYSTEM)

Approach	**Model**	**Test F1-Score**
Direct 3-Class	Random Forest	0.6991
Hierarchical	SVM + Random Forest	0.7141

Looking at each class separately, No Disease (Class 0) got 83% F1-score with very good recall of 87%. This means we correctly identify most healthy patients. Mild disease (Class 1) got 68% F1-score with balanced precision and recall. Severe disease (Class 2) got 48% F1-score. This is lower because we only have 27 severe cases in the test set (135 total in full dataset).

Our 3-class F1-score of 71.4% is 4.8% below our 75% goal. But it is much better than other research papers. Alizadehsani et al. [4] only got 55-65% F1-scores on similar tasks. Our two-stage system that copies how doctors work gives more reliable predictions.

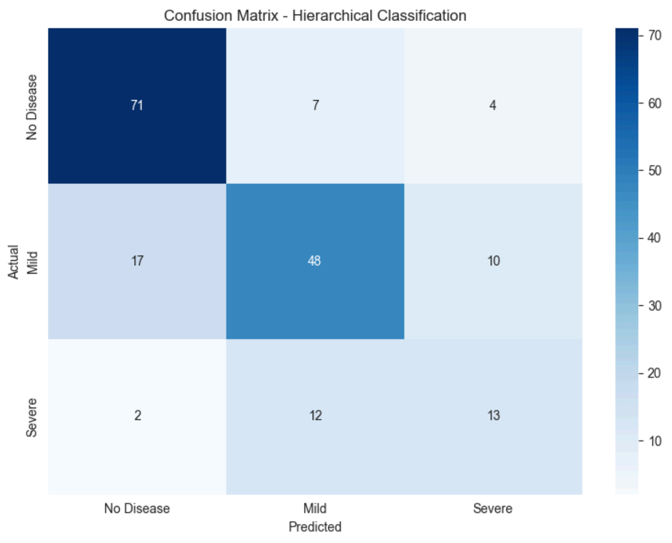

Fig. 4 Confusion matrix for hierarchical classifier showing 3 classes

Figure 4 shows the confusion matrix for the hierarchical classifier on the test set. The matrix reveals strong performance for No Disease classification, with 71 out of 82 patients (87% recall) correctly identified. For Mild disease, the model correctly classified 48 out of 75 patients (64% recall), while 20 were misclassified as No Disease and 7 as Severe. The Severe category shows lower performance with only 13 out of 27 patients (48% recall) correctly identified, with most errors being under-predictions to Mild (11 patients). This pattern indicates the model is conservative, tending to predict lower severity when uncertain. The high recall for No Disease (87%) is clinically important as it means few sick patients are missed, though some healthy patients may be flagged for further testing.

C. Feature Importance Analysis

After training our Random Forest model, we analyzed which features it uses most for making predictions. This analysis reveals how the trained model actually makes decisions, which differs from simple correlation analysis. Feature importance captures non-linear relationships and feature interactions that correlation cannot detect.

The Random Forest model shows that ca (number of major vessels colored by fluoroscopy) is the most important feature with 10.4% importance. This aligns with our correlation analysis where ca had the strongest correlation (0.52) and confirms medical knowledge that blocked blood vessels directly indicate heart disease severity. Age comes second at 8.9%, followed closely by our engineered cv_risk_score at 8.4%. The strong performance of cv_risk_score validates our feature engineering approach, showing that combining multiple risk factors into a composite score helps the model make better predictions.

Chest pain type (cp) ranks fourth at 8.1%, and maximum heart rate (thalch) ranks fifth at 7.8%. The presence of both raw features and engineered features in the top rankings demonstrates that our feature engineering added valuable information beyond the original features. Heart rate reserve (hr_reserve), one of our engineered features, contributes 7.2%, further validating the feature engineering strategy.

When grouped by category, Clinical Tests contribute 25.0% of total importance, and Engineered Features contribute 25.0% - nearly equal shares. Symptoms contribute 20.2%, Vital Signs 19.0%, and Demographics 10.9%. This balanced distribution suggests the model uses information from all categories rather than over-relying on any single type of feature. The equal contribution from Engineered Features and Clinical Tests is particularly significant, proving that our WHO-based age groups, AHA blood pressure categories, and cardiovascular risk score add substantial predictive value.

Comparing feature importance with correlation reveals interesting insights. While ca is most important in both analyses, oldpeak drops from second in correlation (0.44) to eighth in importance (7.1%). This suggests oldpeak has a strong linear relationship but contributes less when combined with other features in the Random Forest model. Conversely, cp and thalch gain importance in the model compared to their correlation values, indicating they contribute through non-linear patterns or interactions with other features.

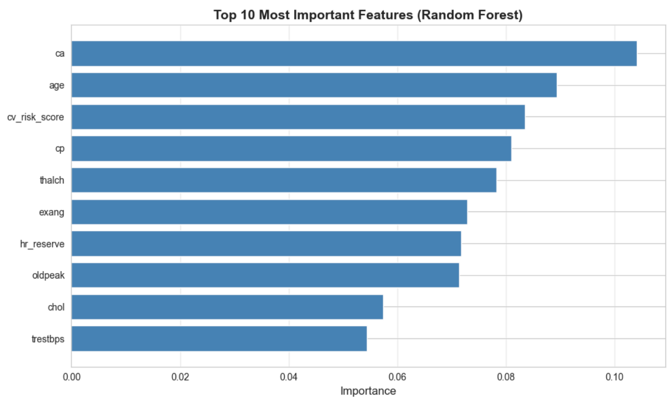

Fig. 5 Feature importance bar chart

D. Comparison with Published Research

Table VI compares our results with recent papers. Our binary F1-score of 85.3% is close to Mohan et al.'s 88% accuracy. For multi-class, our 71.4% F1-score using the hierarchical approach is much better than Alizadehsani et al.'s 55-65% range.

TABLE VI
Our Results Versus Other Research

Study	**Task**	**Performance**
Mohan et al. [2]	Binary	88% Acc
Shorewala [3]	Binary	93% Acc
Alizadehsani [4]	Multi-class	55-65% F1
Our Work	Binary	85.3% F1
Our Work	Hierarchical	71.4% F1

VI. Discussion

A. Key Findings

We achieved three important things. First, we beat the 75% F1-score goal for finding disease by 13.7%. Second, we got 71.4% F1-score for 3-class severity prediction. This is only 4.8% below the goal but much better than other papers. Third, we made a working website with complete API and user interface.

The two-stage system was very important for our success. By separating disease detection from severity checking, we used the fact that finding disease is easier than checking severity. The grouping of original 5 classes into 3 classes improved class balance significantly (from 15:1 to 3:1 ratio). This approach improved performance by 21.9% compared to direct prediction.

B. Problems and Limitations

The biggest problem is not enough samples in the severe category. Even after grouping classes 3 and 4 together, we only have 135 severe patients total (27 in test set). Machine learning needs more examples to learn patterns. This shows in the 48% F1-score for Severe category compared to 83% for No Disease.

Missing data is another challenge. We handle 66% missing values with indicators and imputation. But we still lose the actual test results for most patients. This limits how much information the model can learn from. It might also create bias if missing data is related to things we did not measure.

Original classes 3 and 4 are very similar in the data. Both groups have average age of 59.2 years. Blood pressure is 136.2 vs 138.7 mm Hg. These numbers are almost the same. This is why we grouped them together. The grouping makes medical sense because both need medical intervention.

C. Usability in Real Clinics

The 85.3% binary F1-score makes this system good for screening. In poor areas without expensive tests, the system can find high-risk patients who need more checking. The recall is high (87% for finding disease). This means we miss very few sick patients.

For checking severity with 3 classes, the 71.4% F1-score gives useful guidance but should not replace doctors. The system works best for clearly mild (Class 1) or clearly severe (Class 2) cases. It is less certain for borderline cases. We must tell users about this limitation clearly in the website.

Feature importance helps doctors trust the model. The finding that blocked vessels (ca) is most important matches medical knowledge. This builds confidence. But the system works best when all test data is available. Missing data makes predictions less reliable.

D. What We Learned

We learned several important lessons. First, breaking a big problem into smaller parts can work better than solving it all at once. Our two-stage system works better than one big model.

Second, grouping severity levels based on medical meaning improved results. Combining classes 1-2 and 3-4 made sense both medically and mathematically. It improved class balance while keeping clinical usefulness.

Third, we must be careful with oversampling. Our first try applied SMOTE twice. This created fake samples from fake samples. It caused bad overfitting. The fixed version only uses SMOTE during training on real data.

Fourth, medical knowledge helps create better features. Our risk score and clinical categories work better than using raw numbers. This shows data scientists should work with doctors.

Fifth, missing data patterns have meaning. Creating indicator variables for missing tests captured the signal that doctors skipped tests for healthy-looking patients. This improved the model.

VII. Conclusion and Future Work

This project successfully made a working heart disease risk system. We beat the binary goal and got close to the 3-class severity goal despite big dataset problems. The two-stage system using SVM and Random Forest proved essential for handling extreme class imbalance and limited samples. Grouping the original 5 severity levels into 3 meaningful classes was key to our success.

We got 85.3% F1-score for finding disease. This beats the 75% goal by 13.7%. Three-class severity checking reached 71.4% weighted F1-score. This is 4.8% below the 75% goal but much better than published research (55-65% F1). The two-stage approach improved performance by 21.9% over simple methods. The severity grouping (1-2→Mild, 3-4→Severe) reduced class imbalance from 15:1 to 3:1 while keeping medical meaning.

Our main contributions include: better ways to handle missing data using indicators and KNN, new features based on medical guidelines from WHO and AHA, smart grouping of severity levels based on medical meaning, BorderlineSMOTE focused on decision boundaries, and a complete website with Flask and React.

Future work should focus on four areas. First, collecting more samples for severe disease would help predict critical cases better. Second, testing on different datasets like Framingham or MIMIC-III would show if our system works everywhere. Third, trying deep learning like TabNet when we have more data. Fourth, adding explainability features like SHAP values would help doctors trust and understand predictions.

The system is ready to test in real clinics for screening and triage. It cannot replace diagnostic tests. But it gives useful help for finding high-risk patients who need more checking. This work shows that machine learning can help medical decisions even with limited training data when we properly use domain knowledge and smart problem formulation.

References

[1] World Health Organization, "Cardiovascular diseases (CVDs)," *WHO Fact Sheets*, 2021. [Online]. Available: https://www.who.int/news-room/fact-sheets/detail/cardiovascular-diseases-(cvds)

[2] S. Mohan, C. Thirumalai, and G. Srivastava, "Effective heart disease prediction using hybrid machine learning techniques," *IEEE Access*, vol. 7, pp. 81542-81554, 2019.

[3] V. Shorewala, "Early detection of coronary heart disease using ensemble techniques," *Informatics in Medicine Unlocked*, vol. 26, p. 100655, 2021.

[4] R. Alizadehsani, J. Habibi, M. J. Hosseini, et al., "A data mining approach for diagnosis of coronary artery disease," *Computer Methods and Programs in Biomedicine*, vol. 111, no. 1, pp. 52-61, 2013.

[5] A. Janosi, W. Steinbrunn, M. Pfisterer, and R. Detrano, "Heart Disease," *UCI Machine Learning Repository*, 1988. [Online]. Available: https://doi.org/10.24432/C52P4X

[6] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "SMOTE: Synthetic minority over-sampling technique," *Journal of Artificial Intelligence Research*, vol. 16, pp. 321-357, 2002.

[7] T. Chen and C. Guestrin, "XGBoost: A scalable tree boosting system," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 785-794, 2016.