PROJECT REPORT

Qandil Fatima | Bahadur Ali

M24F0049AI012 M24F00DS003

Project Report: Heart Disease Prediction

Objective

The goal of this project is to develop a robust predictive model for assessing the risk of heart disease using the Framingham Heart Study dataset. The project focuses on preprocessing the data, implementing and optimizing machine learning models like Weighted SVM and Random Forest, and evaluating model performance. An interactive Gradio application is also developed to allow end users to predict heart disease risk.

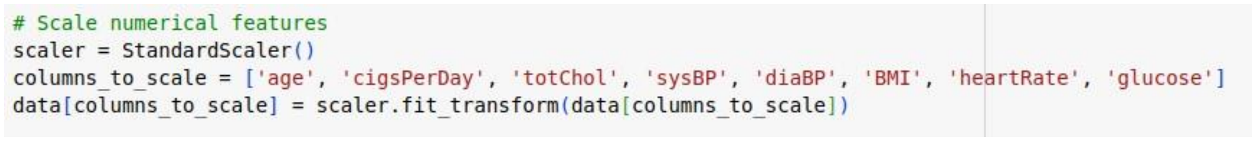
Data Preprocessing

Handling Missing Values:

There are two types of variables, continuous and categorical. Continuous variables (like cigsPerDay, totChol, BMI, heartRate, glucose) are imputed using the mean to ensure no information is lost during modeling. Whereas Categorical variables (like education, BPMeds) are imputed using the mode, as these represent discrete groupings.

Feature Scaling:

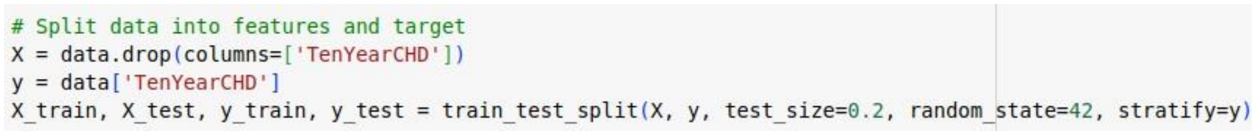
Numerical features (e.g., age, sysBP, diaBP) were normalized using StandardScaler to standardize the data and improve model performance by ensuring uniform feature importance and avoiding model bias toward features with larger ranges.



```
# Scale numerical features
scaler = StandardScaler()
columns_to_scale = ['age', 'cigsPerDay', 'totChol', 'sysBP', 'diaBP', 'BMI', 'heartRate', 'glucose']
data[columns_to_scale] = scaler.fit_transform(data[columns_to_scale])
```

Train-Test Split:

The dataset was split into training (80%) and testing (20%) sets, with stratification to preserve the original class distribution for better model validation and preventing overfitting or under-representation of minority classes.



```
# Split data into features and target
X = data.drop(columns=['TenYearCHD'])
y = data['TenYearCHD']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42, stratify=y)
```

Machine Learning Models

We have used Weighted support vector machine and random forest algorithms and have compared their results to evaluate the performance of each model on the framingham dataset.

Weighted Support Vector Machine (SVM)

Purpose: Weighted SVM addresses class imbalance by assigning weights to classes during training.

Working:

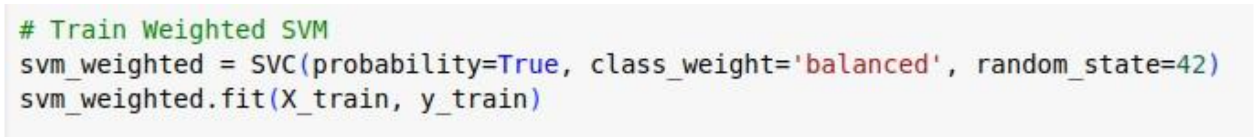
SVM is a supervised machine learning algorithm that works by finding an optimal hyperplane to separate data points into classes. For nonlinear relationships, kernels like the RBF kernel are used to map the input data into a higher-dimensional space where a linear decision boundary can be applied. When data points are not linearly separable, SVM uses a kernel function to transform the data into a higher-dimensional space where the classes become linearly separable.

The RBF kernel is effective for complex, nonlinear relationships by computing similarities between data points based on their distances.The class_weight='balanced' parameter adjusts the importance of each class during training, ensuring that the minority class (heart disease cases) gets higher weight, thereby reducing the bias toward the majority class.The probability=True setting allows SVM to estimate probabilities for predictions, which is useful for interpreting model confidence in classifications.

Implementation:

The Radial Basis Function (RBF) kernel in the SVM was explicitly chosen because of its flexibility in capturing nonlinear relationships between features. This kernel allows the model to map input data into a higher-dimensional space, making it effective for datasets where the decision boundary is not linear.

The class_weight='balanced' parameter in the SVC ensures that the model adjusts weights for each class inversely proportional to their frequency. This makes the model more sensitive to the minority class (e.g., heart disease cases).

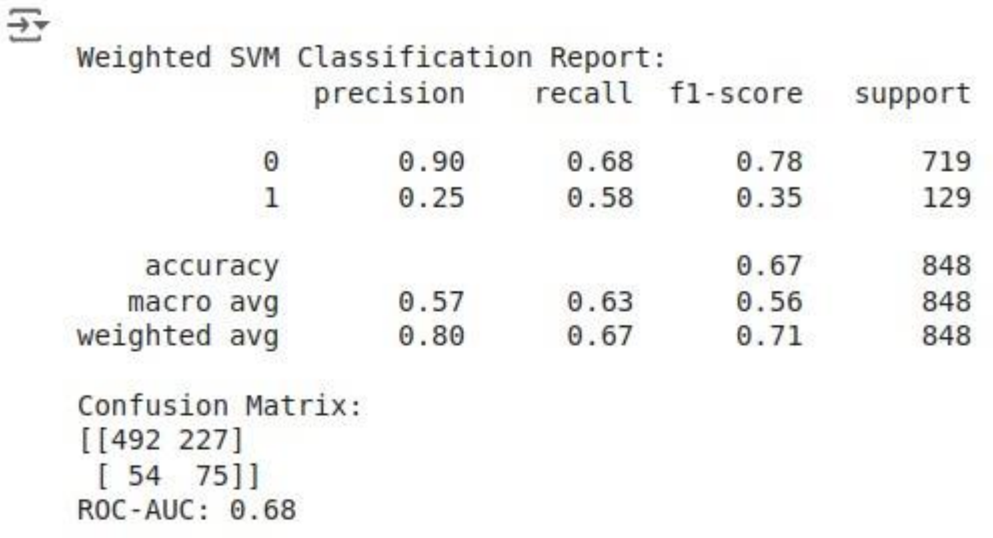


```
# Train Weighted SVM
svm_weighted = SVC(probability=True, class_weight='balanced', random_state=42)
svm_weighted.fit(X_train, y_train)
```

Performance:

The Weighted SVM model demonstrates a reasonably balanced performance between the two classes, with a precision of 90% for class 0 (No Heart Disease) and 25% for class 1 (Heart Disease). While 90% of all predicted No Heart Disease cases were correct, only 25% of Heart Disease predictions were accurate, indicating a high false positive rate for the minority class. The recall for class 0 is 68%, meaning the model identified 68% of actual No Heart Disease cases, and the recall for class 1 is 58%, showing that the model was moderately effective in detecting heart disease but still missed a significant number of cases. The F1-score for class 0 is 0.78, reflecting balanced precision and recall, while for class 1, it is 0.35, indicating the model struggles to balance precision and recall for the minority class. The model achieves an accuracy of 67%, but due to the class imbalance (719 samples for class 0 and 129 for class 1), this metric alone is not reliable. The ROC-AUC score of 0.68 indicates moderate ability to distinguish between the two classes.

The confusion matrix reveals 492 true negatives and 75 true positives, but the model also produces 227 false positives and 54 false negatives. This suggests that while the model is effective in identifying individuals without heart disease, it has a high false positive rate and struggles to achieve sufficient precision for Heart Disease.



```
Weighted SVM Classification Report:
              precision    recall  f1-score   support

           0       0.90      0.68      0.78       719
           1       0.25      0.58      0.35       129

    accuracy                           0.67       848
   macro avg       0.57      0.63      0.56       848
weighted avg       0.80      0.67      0.71       848

Confusion Matrix:
[[492 227]
 [ 54  75]]
ROC-AUC: 0.68
```

Random Forest with Optimization

Purpose: Build a flexible ensemble model capable of handling both balanced and imbalanced datasets.

Working:

Random Forest is an ensemble learning method that combines multiple decision trees to improve classification accuracy and robustness.

Random Forest creates multiple decision trees, each trained on a different bootstrap sample (randomly drawn with replacement) from the training dataset. This introduces diversity among the trees, as each tree is trained on slightly different data. At each split in a decision tree, only a random subset of features is considered for splitting. This ensures no single feature dominates the model and prevents overfitting.

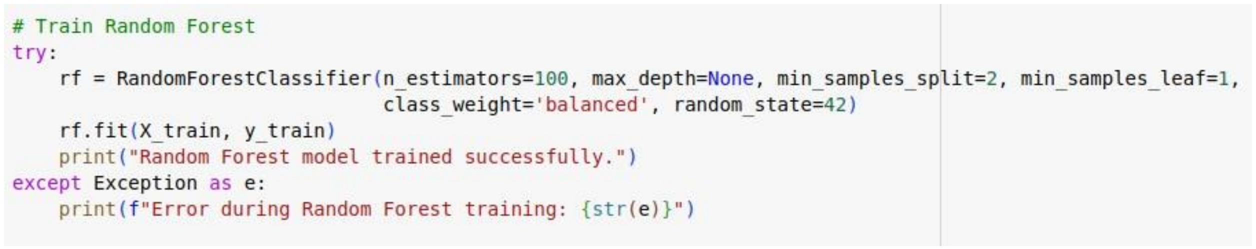
Each tree grows independently without pruning, capturing as much complexity in the data as possible within its bootstrap sample. For classification tasks, the majority vote of all decision trees determines the final class label. For regression tasks, the average prediction from all trees is taken.

By combining multiple weak learners (decision trees), Random Forest reduces overfitting compared to standalone decision trees. The randomness in both sampling and feature selection enhances its generalizability.

Implementation:

Random Forest creates multiple decision trees during training and aggregates their outputs (majority vote for classification).

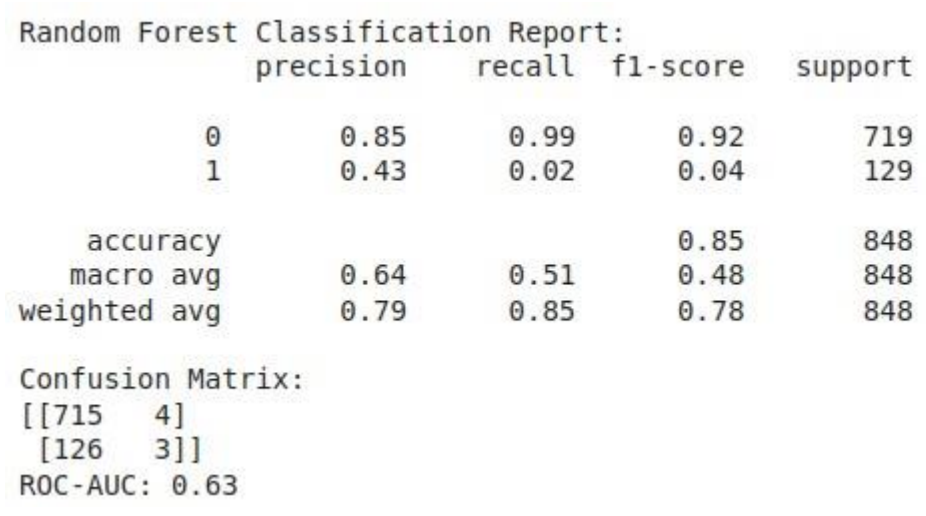
The class_weight='balanced' parameter ensures that the decision trees within the ensemble model are biased toward balancing the importance of each class during training, and other parameters were tuned manually for simplicity.



```
# Train Random Forest
try:
    rf = RandomForestClassifier(n_estimators=100, max_depth=None, min_samples_split=2, min_samples_leaf=1,
                                class_weight='balanced', random_state=42)
    rf.fit(X_train, y_train)
    print("Random Forest model trained successfully.")
except Exception as e:
    print(f"Error during Random Forest training: {str(e)}")
```

Performance:

The Random Forest model demonstrates strong performance in identifying the majority class (No Heart Disease), achieving a high precision of 85% and recall of 99%, resulting in an excellent F1-score of 0.92. This indicates that 85% of all predictions for No Heart Disease were correct, and the model successfully identified 99% of actual No Heart Disease cases. The low false positive rate (only 4 cases) further highlights its effectiveness for this class.
However, the model struggles significantly with the minority class (Heart Disease), achieving a recall of just 2% and an F1-score of 0.04, indicating a failure to identify most heart disease cases and a poor balance between precision and recall. The precision for Heart Disease predictions is 43%, showing a high false positive rate. The dataset's class imbalance (719 samples for class 0 and 129 for class 1) heavily impacts these results, with the model favoring the majority class. While the overall accuracy is 85%, it is not a reliable metric due to the imbalance (**see Figure 1**). The ROC-AUC score of 0.63 suggests moderate ability to distinguish between the two classes. In summary, Random Forest is highly effective for predicting No Heart Disease but performs poorly in detecting Heart Disease cases, highlighting the need for strategies to address class imbalance.



```
Random Forest Classification Report:
              precision    recall  f1-score   support

           0       0.85      0.99      0.92       719
           1       0.43      0.02      0.04       129

    accuracy                           0.85       848
   macro avg       0.64      0.51      0.48       848
weighted avg       0.79      0.85      0.78       848

Confusion Matrix:
[[715   4]
 [126   3]]
ROC-AUC: 0.63
```

Evaluation and Key Findings

The performance of the Weighted SVM and Random Forest models demonstrates distinct strengths and weaknesses based on the provided evaluation metrics. The Weighted SVM achieves a balanced performance between the two classes. It has a recall of 58% for class 1 (Heart Disease), significantly higher than the Random Forest's recall of only 2%. This indicates that the Weighted SVM is much better at identifying heart disease cases, despite a lower precision of 25% for this class compared to Random Forest's 43%. The SVM's ROCAUC score of 0.68 is also slightly better than the Random Forest's 0.63, showing a better ability to differentiate between the two classes overall (See Figure 2).

On the other hand, Random Forest performs exceptionally well for class 0 (No Heart Disease), achieving a precision of 85% and a recall of 99%, leading to an impressive F1score of 0.92 for this majority class. However, it struggles with the minority class (1), resulting in an F1-score of only 0.04 due to its inability to correctly identify most heart disease cases. In comparison, Weighted SVM achieves an F1-score of 0.35 for class 1, reflecting better handling of the imbalanced dataset **(see Table 1)**.

In terms of accuracy, Random Forest achieves a higher overall accuracy of 85% compared to SVM's 67%, but this metric is misleading due to the imbalanced dataset. Random Forest's high accuracy largely stems from its strong performance on the majority class, while Weighted SVM offers a more balanced approach to classifying both 0 and 1.

Confusion Matrix Comparison:

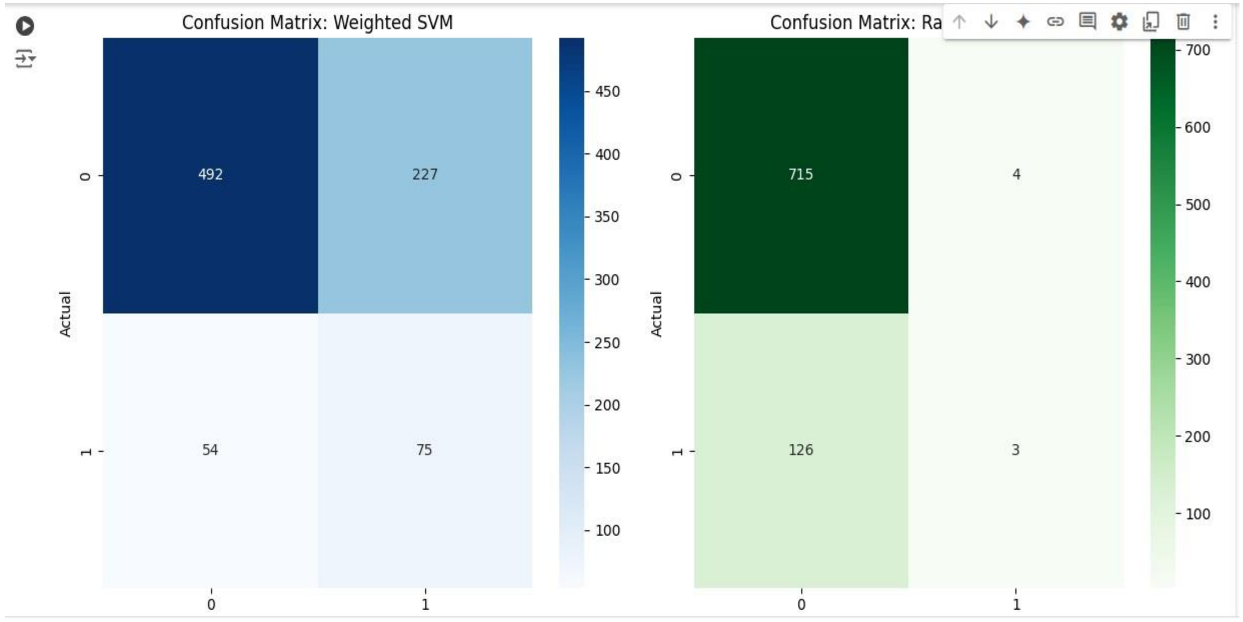


Figure 1

ROC - AUC Comparison:

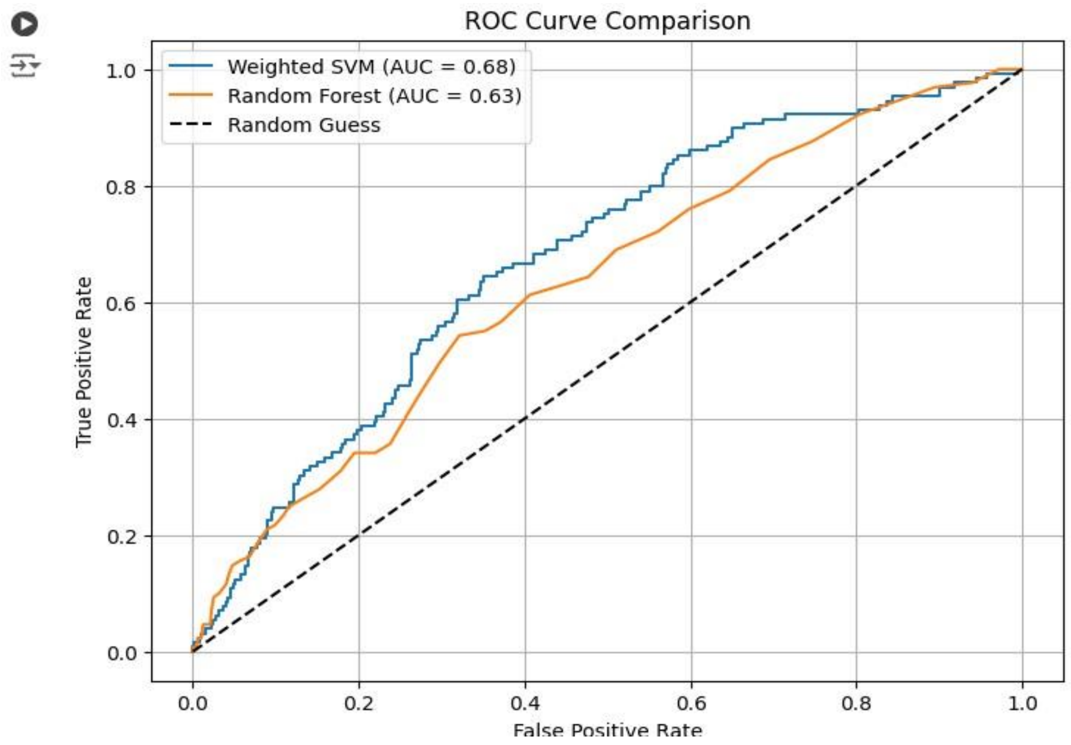


Figure 2

Key Insights:

In conclusion, Weighted SVM is better suited for identifying heart disease cases, as evidenced by its higher recall and F1-score for the minority class. However, Random Forest excels at classifying non-heart disease cases with high precision and recall but fails to generalize well for the minority class. The choice of the model ultimately depends on whether the focus is on minimizing false negatives for heart disease (favoring SVM) or achieving high overall accuracy and precision for non-heart disease cases (favoring Random Forest).

Model	Precision (Class 1)	Recall (Class 1)	F1-Score (Class 1)	ROCAUC
Weighted SVM	0.25	0.57	0.35	0.676
Random Forest	0.33	0.02	0.03	0.626

Interactive Application

An interactive Gradio interface was developed to make predictions based on user inputs:

Inputs: Age

Sex (Male/Female)

Cigarettes per Day

Total Cholesterol

Systolic Blood Pressure

Glucose Level

Model Type (SVM/Random Forest)

Output:

Predicted risk of heart disease along with probability.

Working:

Users input their health parameters into the interface. The application scales the input features and uses the chosen model (SVM or Random Forest) to make predictions. The output includes a risk classification (e.g., High Risk) and the probability score for interpretability.

Improved Error Handling:

If an invalid input is provided, the application returns a descriptive error message. The input scaling step ensures that features are in the expected range for accurate predictions.

Deployment:

The application enables real-time predictions and facilitates easy usage for non-technical users.

Conclusion

This project successfully implements machine learning models to predict heart disease risk, with a focus on handling class imbalance and optimizing performance metrics. The Weighted SVM model demonstrates better sensitivity, while the Random Forest model provides a robust and flexible alternative. The Gradio interface enhances usability by allowing interactive predictions.