CSE299.1 , Group 5, Al Imran(2122071642), MD. Mukzanul Alam Nishat(2212445042), Khondkar Sayif Ali(2111323642), Arman Hossain Nawmee(2221395042), MD Araf UI Haque Dhrubo(2021493042) 1

Weekly progress report 3:

This week, we extended our heart disease prediction project by implementing three additional machine learning models: Random Forest, Support Vector Machine (SVM), and Decision Tree. Our aim was to compare these models with our previous implementations (Linear Regression and Logistic Regression) and to evaluate improvements in classification performance and model interpretability.

Implementation of the Random Forest Model

For predicting binary heart disease outcomes, we implemented a Random Forest model following these key steps:

- Trained the model using Scikit-learn's RandomForestClassifier.
- Made predictions on the test set and evaluated performance using:
  - Accuracy Score (achieved **98%** accuracy)
  - Precision, Recall, and F1-score
- Plotted key visualizations, including a confusion matrix to assess misclassifications, a feature importance plot to identify key predictors, a decision tree visualization (limited to depth 3) for interpretability, and a ROC curve to assess the model's discrimination capability across various thresholds.

The Random Forest model's ensemble approach provided robust predictions with minimal overfitting, highlighting its strength in handling complex relationships in our dataset.

Implementation of the SVM Model

For classifying heart disease outcomes, we implemented a Support Vector Machine (SVM) model following these key steps:

- Trained the model using Scikit-learn's SVC class with an appropriate kernel (e.g., RBF) and regularization parameters.
- Made predictions on the test set and evaluated performance using:
  - Accuracy Score
  - Precision, Recall, and F1-score
- Plotted the ROC curve and computed the AUC score to dynamically determine the optimal decision threshold.

While the SVM achieved competitive classification performance, it required careful tuning of hyperparameters to balance the decision boundary and generalization.

Implementation of the Decision Tree Model

For clear, interpretable classification, we implemented a Decision Tree model following these key steps:

- Trained the model using Scikit-learn's **DecisionTreeClassifier**.
- Made predictions on the test set and evaluated the model using:
  - Accuracy Score(achieved **96%** accuracy)
  - Precision, Recall, and F1-score

- Visualized the decision tree to inspect feature splits and decision rules, which aided in understanding the model’s behavior.

The Decision Tree provided intuitive insights into the data but was more prone to overfitting compared to ensemble methods, reinforcing the need for advanced techniques like Random Forest.

Challenges and Proposed Solutions

Challenge: Data Imbalance and Overfitting

- Small dataset size (1,000 rows) and class imbalance may cause models to overfit.

Solution:

- Use cross-validation and hyperparameter tuning (e.g., GridSearchCV).
- Consider resampling techniques (like SMOTE) to balance the data.

Challenge: Hyperparameter Sensitivity in SVM

- SVM performance is very sensitive to the choice of kernel and regularization parameters.

Solution:

- Perform grid search to find the best parameters.
- Use ROC and Precision-Recall analysis to set an optimal threshold.