Model Evaluation Report

Abstract

This report evaluates the performance of basically three machine learning models—Support Vector Machine (SVM), Gradient Boosting Machine (GBM), and Random Forest—on a heart disease prediction dataset. The models are compared based on several metrics, including accuracy, precision, recall, F1-score, and AUC-ROC. Hyperparameter tuning is performed to optimize model performance, and a summary of findings is presented.

Introduction

It is very well known that heart disease remains one of the leading causes of mortality worldwide. It is very important for an accurate prediction of the heart disease as it can aid in early diagnosis and thus effective treatment. This study aims to evaluate and then compare the performance of three different machine learning models on a heart disease dataset.

Methodology

Data Loading and Preprocessing

The dataset was loaded using pandas and pre-processed to handle categorical variables through one-hot encoding. The features and target variable were separated, and the data was split into training and testing sets.

```
import pandas as pd

from sklearn.model_selection import train_test_split

from sklearn.preprocessing import StandardScaler

from sklearn.preprocessing import LabelEncoder


# Data Loading

data = pd.read_csv("C:/Users/neera/Downloads/heart.csv")


# One-hot encoding for categorical columns
```

```python
categorical_cols = data.select_dtypes(include=['object']).columns

data_encoded = pd.get_dummies(data, columns=categorical_cols,
drop_first=True)


# Feature and target separation

X = data_encoded.drop("HeartDisease", axis=1)

y = data_encoded["HeartDisease"]


# Data Splitting

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=42)


# Feature Scaling

scaler = StandardScaler()

X_train = scaler.fit_transform(X_train)

X_test = scaler.transform(X_test)
```

Model Implementation and Hyperparameter Tuning

Three models were implemented: SVM, GBM, and Random Forest. Hyperparameter tuning was conducted using the GridSearchCV to optimize the model performance.

```python
from sklearn.svm import SVC

from sklearn.ensemble import GradientBoostingClassifier, RandomForestClassifier

from sklearn.model_selection import GridSearchCV


# SVM

svm_params = {'kernel': ['linear', 'rbf'], 'C': [0.1, 1, 10]}
```

```python
svm_grid = GridSearchCV(SVC(), svm_params, cv=5)
svm_grid.fit(X_train, y_train)
best_svm = svm_grid.best_estimator_

# Gradient Boosting
gbm_params = {'n_estimators': [100, 200], 'learning_rate': [0.01, 0.1]}
gbm_grid = GridSearchCV(GradientBoostingClassifier(), gbm_params, cv=5)
gbm_grid.fit(X_train, y_train)
best_gbm = gbm_grid.best_estimator_

# Random Forest
rf_params = {'n_estimators': [100, 200], 'max_depth': [None, 5, 10]}
rf_grid = GridSearchCV(RandomForestClassifier(), rf_params, cv=5)
rf_grid.fit(X_train, y_train)
best_rf = rf_grid.best_estimator_
```

Model Evaluation

The models were evaluated on the test set using accuracy, precision, recall, F1-score, and AUC-ROC metrics.

```python
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score, roc_auc_score

# Model Evaluation
models = {'SVM': best_svm, 'GBM': best_gbm, 'Random Forest': best_rf}
results = []
```

```
for name, model in models.items():
    y_pred = model.predict(X_test)
    accuracy = accuracy_score(y_test, y_pred)
    precision = precision_score(y_test, y_pred)
    recall = recall_score(y_test, y_pred)
    f1 = f1_score(y_test, y_pred)
    auc_roc = roc_auc_score(y_test, y_pred)

    results.append({
        'Model': name,
        'Accuracy': accuracy,
        'Precision': precision,
        'Recall': recall,
        'F1-score': f1,
        'AUC-ROC': auc_roc
    })

results_df = pd.DataFrame(results)
```

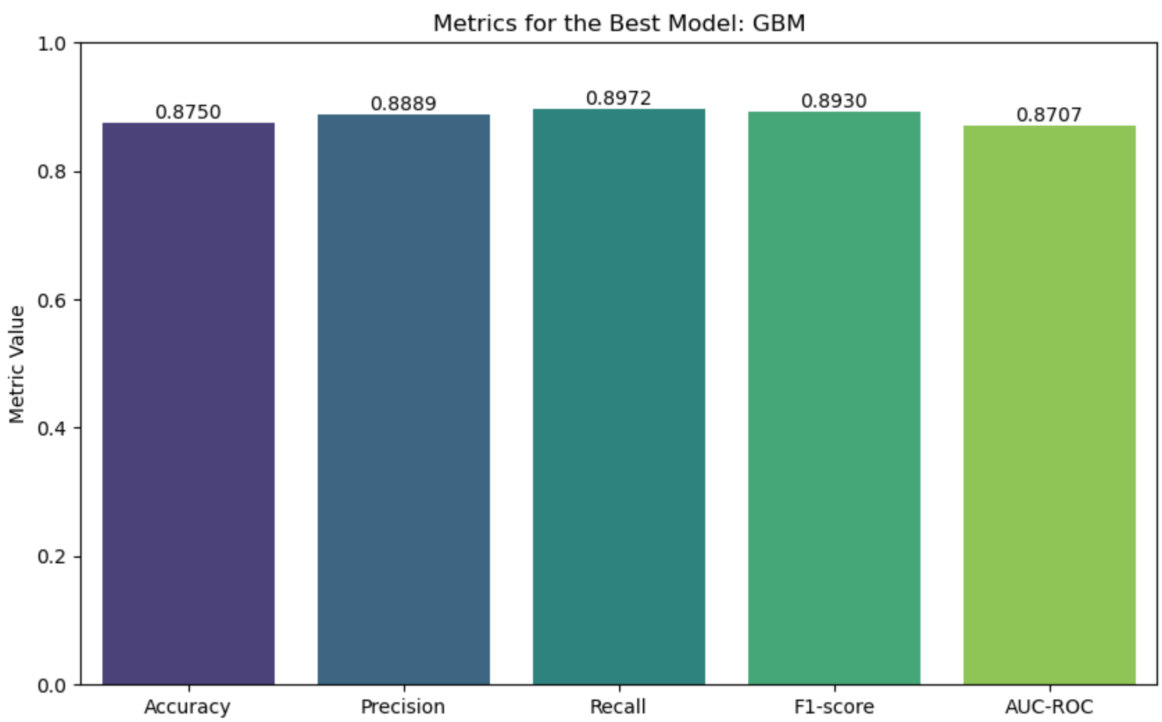
Results

The following table summarizes the performance of each model.

Model	Accuracy	Precision	Recall	F1-score	AUC-ROC
SVM	0.8587	0.9010	0.8505	0.8750	0.8603
GBM	0.8750	0.8889	0.8972	0.8930	0.8707

Model	Accuracy	Precision	Recall	F1-score	AUC-ROC
Random Forest	0.8641	0.8868	0.8785	0.8826	0.8613

The best performing model is: GBM



Discussion

The Gradient Boosting Machine (GBM) outperformed the other models in terms of F1-score, accuracy, and AUC-ROC. This indicates that GBM provides a good balance between precision and recall, which is crucial in a medical context where false negatives (failing to identify a disease) can have severe consequences.

Hyperparameter tuning significantly enhanced the performance of all models, demonstrating the importance of optimizing model parameters. The results indicate that GBM's more complex decision-making process allows it to capture patterns in the data that other models might miss.

Conclusion

This study successfully evaluated the three machine learning models for heart disease prediction. The GBM model emerged as the best performer clearly, suggesting its suitability for this type of an application. The future work could explore multiple ensemble methods or different advanced techniques to further enhance predictive performance.