Detect Heart Disease using patient data

1. Objective

- The goal of this project is to build a machine learning model capable of predicting whether a patient has heart disease based on various medical features. The project involves the entire machine learning pipeline, including data exploration, preprocessing, model development, hyperparameter tuning, evaluation, and deployment.

2. Dataset

- The dataset used for this project contains medical information about patients, including features such as age, cholesterol, resting blood pressure, and other relevant medical data.
- The target variable is binary, where 1 indicates the presence of heart disease and 0 indicates its absence.

3. Steps Involved

Step 1: Exploratory Data Analysis (EDA)

- **Summary Statistics:** The dataset was analyzed to understand the basic statistics, including means, standard deviations, and counts for each feature.
- **Missing Values:** A check was performed for any missing data, with results indicating how many missing values were present in each feature.
- **Visualization:** Various visualizations were created to explore class distribution, numeric feature distributions, and feature correlations. These helped in identifying patterns and relationships between features.

Step 2: Data Preprocessing

- **Feature Selection:** The target variable was separated from the feature set. Categorical and numeric features were identified and handled separately.
- **Feature Scaling:** Numeric features were standardized using StandardScaler, and categorical features were encoded using OneHotEncoder.
- A preprocessing pipeline was set up using ColumnTransformer to apply the respective transformations to the appropriate features.

Step 3: Model Development

- **Train-Test Split:** The dataset was split into training and testing sets using an 80-20 split. The split was stratified to maintain the same distribution of the target variable in both sets.
- **Model Selection:** A RandomForestClassifier was selected as the initial model due to its effectiveness for classification problems and interpretability.

- **Hyperparameter Tuning:** A grid search with cross-validation (GridSearchCV) was used to optimize hyperparameters such as the number of estimators, max depth, and minimum samples required to split a node.

Step 4: Model Evaluation

- **Classification Report:** After training the model, a classification report was generated to assess precision, recall, f1-score, and support for each class.
- **Confusion Matrix:** A confusion matrix was plotted to visualize the performance of the model, showing the number of true positives, true negatives, false positives, and false negatives.
- **ROC-AUC Score:** The ROC-AUC score was calculated to assess the model's ability to distinguish between classes.
- **Feature Importance:** The importance of each feature in predicting the target was displayed through a bar plot of feature importances.

Step 5: Model Deployment and Testing

- **Model Saving:** The trained model was saved to a file using joblib, allowing for later use without retraining.
- **Model Loading and Testing:** The saved model was reloaded, and its performance was tested on the test set. The confusion matrix, ROC-AUC score, and ROC curve were visualized again to ensure the loaded model's results matched the initial model's performance.

4. Expected Outcome

- The final model should be able to accurately predict whether a patient has heart disease based on the given features.
- The model's performance will be evaluated using various metrics, including accuracy, precision, recall, F1-score, ROC-AUC, and feature importance, ensuring that it generalizes well to unseen data.

5. Skills and Technologies Used

- **Programming Languages:** Python
- **Libraries:** Pandas, NumPy, Matplotlib, Seaborn, scikit-learn
- **Machine Learning Techniques:** Data preprocessing, model training, hyperparameter tuning, model evaluation, model deployment
- **Tools:** Jupyter Notebooks for coding and visualization, joblib for model saving/loading

6. Future Improvements

- **Model Optimization:** Further experimentation with different machine learning models (e.g., XGBoost, SVM) to improve performance.
- **Feature Engineering:** Additional domain knowledge could be used to create new features or handle missing data more effectively.
- **Real-Time Prediction:** Integration with a real-time system where heart disease predictions can be made based on live patient data.