Task Description:

Leveraging the potency of machine learning, particularly Random Forest classifiers, is catalyzing a paradigm shift in healthcare, notably in heart disease prediction. These sophisticated models harness vast health data to predict heart disease risk with remarkable accuracy, facilitating tailored risk assessments and optimizing healthcare resource allocation. Similarly, they forecast the decade-long risk of heart disease by analyzing multifaceted variables including demographics, lifestyle habits, and medical history. Given the pervasive impact of stroke and heart disease on global morbidity and mortality rates, early identification of risk factors is imperative for proactive healthcare management. By championing data transparency and guiding individuals towards healthier lifestyles, these models aspire to mitigate complications associated with stroke and heart disease and bolster preventive healthcare strategies. In essence, they epitomize the profound potential of machine learning in advancing public health outcomes and enhancing societal well-being.

Methods used:

The application's features are extracted from the Random Forest classifier code titled "RForest-classifier". Datasets obtained from Kaggle are imported as CSV files, initiating an exploratory analysis phase. Utilizing visualizations such as, histograms, box plots, and pie chart, the underlying data patterns are thoroughly examined. Preprocessing steps include handling missing values, managing outliers through winsorization, and encoding target variables for model compatibility. After initialization, the Random Forest classifier is trained using selected features and labels. Model performance is assessed using metrics such as the confusion matrix, accuracy score, and classification report. Users can input new instances for prediction, highlighting the model's practical application. Through iterative deployment, continuous predictions are made until the user chooses to conclude, ensuring a seamless and interactive experience.

Hyperparameter Tuning:

The table illustrates the impact of hyperparameter tuning on the accuracy of a model. Firstly, varying the test size reveals fluctuations in accuracy percentages, with the highest accuracy of 88.04% achieved at a test size of 10%. Secondly, adjusting the number of estimators in the model indicates consistent accuracy percentages across different estimator values, with the highest accuracy also recorded at 89.13% for 20 estimators. These findings underscore the importance of fine-tuning hyperparameters to optimize model performance. Choosing five estimators strikes a balance between model complexity and performance.

Test Size Variation **Estimator Variation**

Test Size	Accuracy (%)	
0.10	**88.04**	
0.15	86.23	
0.20	83.15	
0.25	84.78	
0.30	86.23	n_estimators = 200

Estimators	Accuracy	
10	88.04	
20	**89.13**	
30	88.04	
100	88.04	
200	88.04	Test size = 0.10

Improvements:

1. **Pairplot**: A pairplot has been included to visualize the pairwise relationships between numerical features in the dataset, with each plot colored by the target variable, heart disease. This comprehensive visualization provides insights into potential patterns and correlations between different features, helping in understanding their relationships with each other and with the target variable.
2. **Line Plot**: A line plot has been added to illustrate the distribution of heart disease occurrences across different age groups. By plotting age against heart disease status, this visualization helps in understanding how heart disease prevalence varies with age, providing valuable insights into potential age-related patterns or trends.
3. **Pie Chart**: A pie chart has been included to visualize the distribution of different categories within the RestingECG feature. Each category is represented by a slice of the pie, with colors distinguishing between categories. This visualization offers a clear overview of the proportion of each category within the RestingECG feature, facilitating easy interpretation of the distribution pattern.
4. **Semi-Autonomous:** The iterative deployment of the model has been implemented using a semi-autonomous while loop. This loop allows for continuous predictions until the user decides to conclude, enhancing the user experience by providing a more interactive and user-friendly interface.

Python code:

```
"""

Created on Sun Feb 25, 2024

@author: Mohammed Abdulai

Random Forest Classifier Assignment - Heart Disease predictor

"""

import pandas as pd

import matplotlib.pyplot as plt

from sklearn.model_selection import train_test_split

from sklearn.ensemble import RandomForestClassifier
```

```
from sklearn.metrics import confusion_matrix, ConfusionMatrixDisplay, classification_report,
accuracy_score
from sklearn.preprocessing import LabelEncoder
import warnings
from scipy.stats import mstats
from sklearn.preprocessing import RobustScaler
import seaborn as sns
from plotly.subplots import make_subplots
warnings.filterwarnings('ignore')
label_encoder = LabelEncoder()
class RForest_model:
    def load_Data(self, filepath):
        df = pd.read_csv(filepath)
        self.df = df
        print(df.head())
        print(df.shape)
    def visualizeData_bef_modeling(self):
        print('\n Visualize Data Patterns Before Modeling')
        df = self.df
        plt.figure(figsize=(8, 6))
        sns.lineplot(x=df['Age'], y=df['HeartDisease'], data=df, linewidth=2)
        plt.title('Heart Disease for Ages', fontsize=16)
        plt.xlabel('Age')
        plt.ylabel('Heart Disease')
        plt.gca().set_facecolor('black')
        plt.show()
        #add a pairplot
        sns.pairplot(df, hue='HeartDisease')
        plt.show()
        resting_ecg_counts = df['RestingECG'].value_counts()
        print(resting_ecg_counts)
```

```
        colors =colors = ['skyblue', 'mediumturquoise', 'lightgreen']
        fig = plt.pie(x=resting_ecg_counts.values,
                      labels=resting_ecg_counts.index,
                      colors=colors,
                      autopct='%1.1f%%',
                      startangle=140)
        plt.title('Distribution of RestingECG')
        plt.axis('equal')  # Equal aspect ratio ensures that pie is drawn as a circle.
        plt.show()
    def data_preprocessing(self):
        df = self.df
          # Winsorization to handle outliers
        winsorized_df = df.copy()
        numerical_features = ['Age', 'RestingBP', 'Cholesterol', 'MaxHR', 'Oldpeak']
        for feature in numerical_features:
            winsorized_df[feature] = mstats.winsorize(df[feature], limits=[0.05, 0.05])
        # Robust scaling to handle outliers
        scaler = RobustScaler()
        winsorized_df[numerical_features] =
scaler.fit_transform(winsorized_df[numerical_features])
        df = winsorized_df
        print('\n DATA ENCODING')
        df['Sex_encoded'] = label_encoder.fit_transform(df['Sex'])
        df['ChestPainType_encoded'] = label_encoder.fit_transform(df['ChestPainType'])
        df['RestingECG_encoded'] = label_encoder.fit_transform(df['RestingECG'])
        df['ST_Slope_encoded'] = label_encoder.fit_transform(df['ST_Slope'])
        df['ExerciseAngina_encoded'] = label_encoder.fit_transform(df['ExerciseAngina'])

        features = df[['Age', 'Sex_encoded', 'ChestPainType_encoded', 'RestingBP', 'Cholesterol',
                       'FastingBS', 'RestingECG_encoded', 'MaxHR', 'ExerciseAngina_encoded',
'Oldpeak', 'ST_Slope_encoded']]
```

```
        label = df['HeartDisease']
        print(df)
        X_train, X_test, y_train, y_test = train_test_split(features, label, test_size=0.10,
random_state=0)
        print(X_train.shape, ': Training data size')
        print(X_test.shape, ': Test data size')
        self.df = df
        self.X_train = X_train
        self.X_test = X_test
        self.y_train = y_train
        self.y_test = y_test
    def train_RF_predict_test(self):
        X_train = self.X_train
        y_train = self.y_train
        X_test = self.X_test
        RFclass = RandomForestClassifier(n_estimators=20, bootstrap=True, criterion='entropy',
random_state=0, n_jobs=-1)
        RFclass.fit(X_train, y_train)
        self.pred_y = RFclass.predict(X_test)
        self.RFclass = RFclass
    def evaluate_RFmodel(self):
        print('\n RANDOM FOREST PERFORMANCE EVALUATIONS')
        RFclass = self.RFclass
        cm = confusion_matrix(self.y_test, self.pred_y, labels=RFclass.classes_)
        disp = ConfusionMatrixDisplay(confusion_matrix=cm, display_labels=RFclass.classes_)
        disp.plot()
        plt.show()

        print("Accuracy Score = ", accuracy_score(self.y_test, self.pred_y))
        print(classification_report(self.y_test, self.pred_y))
    def predict_new_instance(self, new_instance):
```

```
        Age, Sex, ChestPainType, RestingBP, Cholesterol, FastingBS, RestingECG, MaxHR,
ExerciseAngina, Oldpeak, ST_Slope = new_instance
        Sex_encoded = label_encoder.fit_transform([Sex])[0]
        ChestPainType_encoded = label_encoder.fit_transform(['ChestPainType'])[0]
        RestingECG_encoded = label_encoder.fit_transform(['RestingECG'])[0]
        ExerciseAngina_encoded = label_encoder.fit_transform(['ExerciseAngina'])[0]
        ST_Slope_encoded = label_encoder.fit_transform(['ST_Slope'])[0]
        encoded_instance = [[Age, Sex_encoded, ChestPainType_encoded, RestingBP, Cholesterol,
                             FastingBS, RestingECG_encoded, MaxHR, ExerciseAngina_encoded,
Oldpeak, ST_Slope_encoded]]
        pred_instance_encoded = self.RFclass.predict(encoded_instance)
        pred_instance_type = 'Yes' if pred_instance_encoded[0] == 1 else 'No'
        print('\n Predicted instance type: = ', pred_instance_type)
# -------------------------class driver-------------------------------------------------------
handle = RForest_model()
filepath = 'heart.csv'
handle.load_Data(filepath)
handle.visualizeData_bef_modeling()
handle.data_preprocessing()
handle.train_RF_predict_test()
handle.evaluate_RFmodel()
while True:
    print("New instance description to be predicted: ")
    Age = int(input("Enter age: "))
    Sex = input("Sex (Male or Female): ")
    ChestPainType = int(input("Chest Pain Type (1, 2, 3, or 4): "))
    RestingBP = int(input("Resting Blood Pressure: "))
    Cholesterol = int(input("Cholesterol: "))
    FastingBS = int(input("Fasting Blood Sugar (0 or 1): "))
    RestingECG = int(input("Resting ECG (0, 1, or 2): "))
    MaxHR = int(input("Max Heart Rate: "))
```

```
    ExerciseAngina = int(input("Exercise Induced Angina (0 or 1): "))
    Oldpeak = float(input("Oldpeak: "))
    ST_Slope = int(input("ST Slope (0, 1, or 2): "))
    new_instance = [Age, Sex, ChestPainType, RestingBP, Cholesterol, FastingBS, RestingECG,
MaxHR, ExerciseAngina, Oldpeak, ST_Slope]
    print('\n Structured new instance description: ', new_instance)
    handle.predict_new_instance(new_instance)
    user_input = input("\nDo you want to predict another instance? (yes/no): ").lower()
    if user_input != 'yes':
        break
```

Results: The screenshots represent the code being executed successfully.

Image 1: The dataset has been loaded into a DataFrame, and the first five rows are presented using the head function. Additionally, the shape of the DataFrame is displayed using the shape function to provide an overview of its dimensions.

```
   Age Sex ChestPainType  RestingBP  Cholesterol  FastingBS RestingECG  MaxHR  \
0   40   M           ATA        140          289          0     Normal    172
1   49   F           NAP        160          180          0     Normal    156
2   37   M           ATA        130          283          0         ST     98
3   48   F           ASY        138          214          0     Normal    108
4   54   M           NAP        150          195          0     Normal    122

  ExerciseAngina  Oldpeak ST_Slope  HeartDisease
0              N      0.0       Up             0
1              N      1.0     Flat             1
2              N      0.0       Up             0
3              Y      1.5     Flat             1
4              N      0.0       Up             0
(918, 12)
```

Image 2: Data visualization is conducted prior to data preprocessing. The graph visually demonstrates that as age increases, there is a corresponding increase in the likelihood of heart disease occurrence. This insight is crucial for understanding the relationship between age and heart disease risk, indicating that older individuals may be more susceptible to developing heart-related conditions.

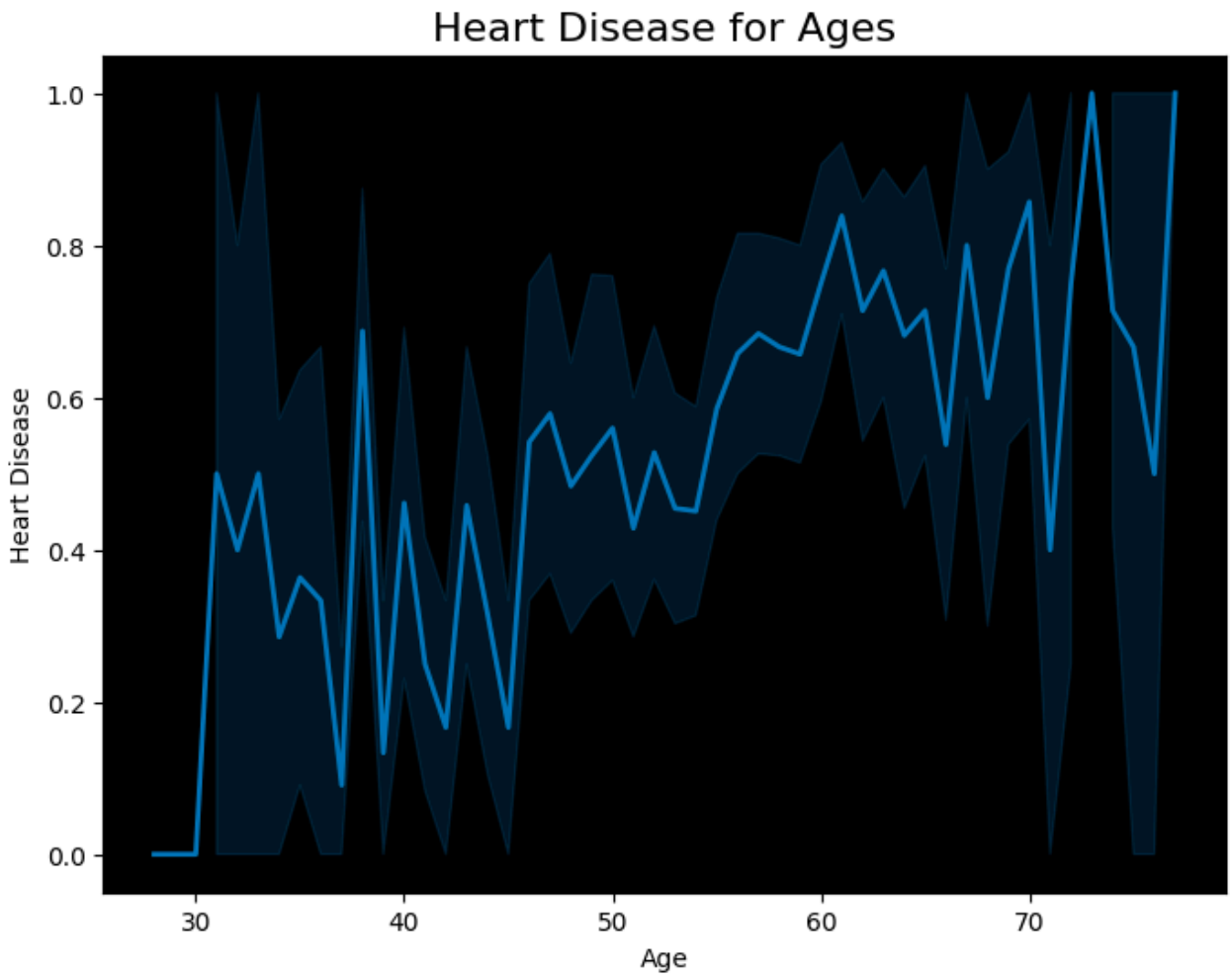


Image 3: The pair plot displays the pairwise relationships between different features in the dataset, with the hue parameter representing the presence or absence of heart disease.

Image 4: The pie chart visualizes the distribution of RestingECG categories in the dataset, including Normal (552), LVH (188), and ST (178). Each category is represented proportionally, with Normal having the highest count, followed by LVH and ST.

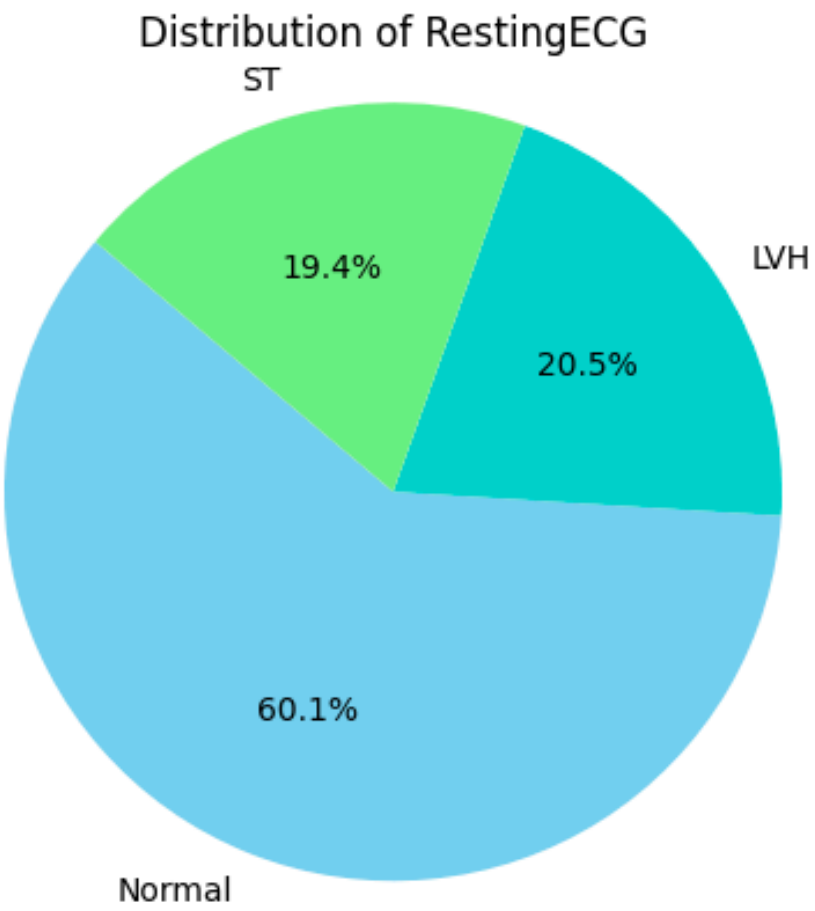
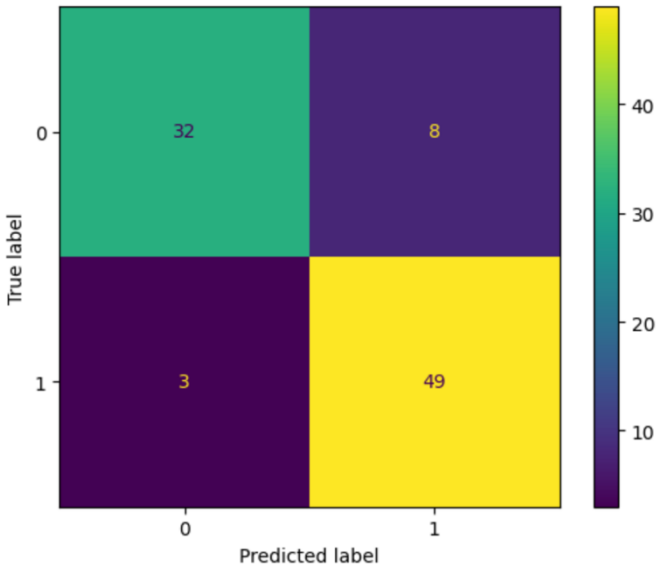


Image 5: This stage involves data preprocessing, which includes tasks such as cleaning the data and encoding textual columns into numerical values. Additionally, the sizes of the training and testing datasets are specified.

```
 DATA ENCODING
          Age Sex ChestPainType  RestingBP  Cholesterol  FastingBS RestingECG  \
0   -1.076923   M           ATA        0.5     0.704000          0     Normal
1   -0.384615   F           NAP        1.5    -0.458667          0     Normal
2   -1.307692   M           ATA        0.0     0.640000          0         ST
3   -0.461538   F           ASY        0.4    -0.096000          0     Normal
4    0.000000   M           NAP        1.0    -0.298667          0     Normal
..        ...  ..           ...        ...          ...        ...        ...
913 -0.692308   M            TA       -1.0     0.437333          0     Normal
914  1.076923   M           ASY        0.7    -0.320000          1     Normal
915  0.230769   M           ASY        0.0    -0.981333          0     Normal
916  0.230769   F           ATA        0.0     0.138667          0        LVH
917 -1.230769   M           NAP        0.4    -0.512000          0     Normal

        MaxHR ExerciseAngina   Oldpeak ST_Slope  HeartDisease  Sex_encoded  \
0    0.944444              N -0.400000       Up             0            1
1    0.500000              N  0.266667     Flat             1            0
2   -1.111111              N -0.400000       Up             0            1
3   -0.833333              Y  0.600000     Flat             1            0
4   -0.444444              N -0.400000       Up             0            1
..        ...            ...       ...      ...           ...          ...
913 -0.166667              N  0.400000     Flat             1            1
914  0.083333              N  1.600000     Flat             1            1
915 -0.638889              Y  0.400000     Flat             1            1
916  1.000000              N -0.400000     Flat             1            0
917  0.972222              N -0.400000       Up             0            1

     ChestPainType_encoded  RestingECG_encoded  ST_Slope_encoded  \
0                        1                   1                 2
1                        2                   1                 1
2                        1                   2                 2
3                        0                   1                 1
4                        2                   1                 2
..                     ...                 ...               ...
913                      3                   1                 1
914                      0                   1                 1
915                      0                   1                 1
916                      1                   0                 1
917                      2                   1                 2

     ExerciseAngina_encoded
0                         0
1                         0
2                         0
3                         1
4                         0
..                      ...
913                       0
914                       0
915                       1
916                       0
917                       0

[918 rows x 17 columns]
(826, 11) : Training data size
(92, 11) : Test data size
```

Image 6: The performance of the classifier is assessed, displaying metrics such as the confusion matrix, an accuracy score of 88.04%, as well as precision, recall, and f1 scores.

RANDOM FOREST PERFORMANCE EVALUATIONS



Accuracy Score = 0.8804347826086957

	precision	recall	f1-score	support
0	0.91	0.80	0.85	40
1	0.86	0.94	0.90	52
accuracy			0.88	92
macro avg	0.89	0.87	0.88	92
weighted avg	0.88	0.88	0.88	92

Image 7: The model is put into operation and tested with the initial prediction instance. Upon entering the data for the first patient, the prediction indicates that they are unlikely to have a heart disease.

```
New instance description to be predicted:
Enter age: 50
Sex (Male or Female): Male
Chest Pain Type (1, 2, 3, or 4): 3
Resting Blood Pressure: 150
Cholesterol: 280
Fasting Blood Sugar (0 or 1): 1
Resting ECG (0, 1, or 2): 2
Max Heart Rate: 170
Exercise Induced Angina (0 or 1): 1
Oldpeak: 1
ST Slope (0, 1, or 2): 2

 Structured new instance description:  [50, 'Male', 3, 150, 280, 1, 2, 170, 1, 1.0, 2]

 Predicted instance type: =  Yes

Do you want to predict another instance? (yes/no): yes
```

Image 8: Upon entering the data of the third patient, the prediction indicates that they are likely to have a heart disease. This prediction may be attributed to their age and their resting blood pressure.

```
New instance description to be predicted:
Enter age: 45
Sex (Male or Female): Female
Chest Pain Type (1, 2, 3, or 4): 3
Resting Blood Pressure: 150
Cholesterol: 290
Fasting Blood Sugar (0 or 1): 1
Resting ECG (0, 1, or 2): 2
Max Heart Rate: 171
Exercise Induced Angina (0 or 1): 1
Oldpeak: 1
ST Slope (0, 1, or 2): 2

 Structured new instance description:  [45, 'Female', 3, 150, 290, 1, 2, 171, 1, 1.0, 2]

 Predicted instance type: =  Yes

Do you want to predict another instance? (yes/no): yes
```

Image 9: Upon entering the data for the fourth patient, the prediction indicates that the individual is unlikely to have heart disease. This conclusion may be attributed to factors such as the individual's age and resting blood pressure, which fall within ranges that are typically associated with lower risk of heart disease.

```
New instance description to be predicted:
Enter age: 70
Sex (Male or Female): Female
Chest Pain Type (1, 2, 3, or 4): 1
Resting Blood Pressure: 100
Cholesterol: 200
Fasting Blood Sugar (0 or 1): 0
Resting ECG (0, 1, or 2): 0
Max Heart Rate: 100
Exercise Induced Angina (0 or 1): 0
Oldpeak: 0
ST Slope (0, 1, or 2): 0

 Structured new instance description:  [70, 'Female', 1, 100, 200, 0, 0, 100, 0, 0.0, 0]

 Predicted instance type: =  No

Do you want to predict another instance? (yes/no): yes
```

Image 10: Upon entering the details for the new instance, the model predicts that the individual is unlikely to have heart disease. This conclusion is drawn considering factors

such as the individual's young age, low resting blood pressure, cholesterol levels, and max heart rate, all of which typically indicate a lower risk of heart disease.

```
New instance description to be predicted:
Enter age: 22
Sex (Male or Female): Female
Chest Pain Type (1, 2, 3, or 4): 1
Resting Blood Pressure: 90
Cholesterol: 150
Fasting Blood Sugar (0 or 1): 0
Resting ECG (0, 1, or 2): 0
Max Heart Rate: 85
Exercise Induced Angina (0 or 1): 0
Oldpeak: 0
ST Slope (0, 1, or 2): 0

 Structured new instance description:  [22, 'Female', 1, 90, 150, 0, 0, 85, 0, 0.0, 0]

 Predicted instance type: =  No

Do you want to predict another instance? (yes/no): no
```

Comparative Analysis: Decision Tree(DT) Heart Disease Predictor VS Random Forest(RF) Heart Disease Predictor

	RF Classifier Predictor		**DT Classifier Predictor**	
Accuracy	88.04%		82.60%	
Outcome Class	0	1	0	1
Precision	0.91	0.86	0.76	0.88
Recall	0.80	0.94	0.84	0.81
F1-score	0.85	0.90	0.80	0.85

- **Accuracy**: The Random Forest classifier achieves a higher accuracy of 88.04% compared to the Decision Tree classifier's accuracy of 82.60%. This indicates that the Random Forest model performs better overall in accurately predicting whether an individual has heart disease or not.
- **Outcome Class 0 Precision**: The precision for predicting class 0 (no heart disease) is higher for the Random Forest classifier (0.91) compared to the Decision Tree classifier (0.76). This suggests that the Random Forest model is more precise in correctly identifying individuals without heart disease.
- **Outcome Class 1 Precision**: The precision for predicting class 1 (heart disease) is slightly higher for the Decision Tree classifier (0.88) compared to the Random Forest classifier (0.86). However, the difference is relatively small.
- **Outcome Class 0 Recall**: The recall for class 0 is higher for the Decision Tree classifier (0.84) compared to the Random Forest classifier (0.80). This indicates that the Decision Tree model is better at capturing individuals without heart disease.
- **Outcome Class 1 Recall**: The recall for class 1 is higher for the Random Forest classifier (0.94) compared to the Decision classifier (0.81). This suggests that the Decision Tree model is better at identifying individuals with heart disease.

- **F1-score**: The F1-score, which is the harmonic mean of precision and recall, is higher for both classes in the Random Forest classifier compared to the Decision Tree classifier. This further confirms that the Random Forest model generally performs better overall.

Conclusion:

In conclusion, the comparative analysis between the Random Forest (RF) and Decision Tree (DT) classifiers for heart disease prediction underscores the superiority of the RF model. The RF model's ensemble approach, which aggregates multiple decision trees and averages their predictions, leads to improved generalization and robustness. By mitigating overfitting and reducing variance, the RF model demonstrates higher accuracy, precision, and F1-score compared to the DT model. Furthermore, the RF model's ability to handle high-dimensional datasets with correlated features effectively contributes to its superior performance in predicting heart disease. This superior performance is further emphasized by the Random Forest classifier's enhanced ability to generalize and make accurate predictions, positioning it as a robust and reliable model for heart disease prediction tasks. Overall, the RF classifier emerges as the preferred choice for heart disease prediction due to its enhanced predictive power, resilience against overfitting, and effectiveness in handling the complexities of heart disease classification.