import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

from sklearn.linear\_model import LogisticRegression

from sklearn.preprocessing import StandardScaler

from sklearn.model\_selection import train\_test\_split

# Sample dataset (replace with your dataset)

data = pd.DataFrame({

    'Age': np.random.randint(18, 80, size=100),

    'BMI': np.round(np.random.uniform(15, 40, size=100), 1),

    'BP Systolic': np.random.randint(90, 180, size=100),

    'BP Diastolic': np.random.randint(60, 120, size=100),

    'Cholesterol': np.random.randint(100, 300, size=100),

    'Disease': np.random.choice([0, 1], size=100)  # Binary target variable

})

# Splitting the dataset into features and target variable

X = data.drop('Disease', axis=1)

y = data['Disease']

# Standardize the features

scaler = StandardScaler()

X\_scaled = scaler.fit\_transform(X)

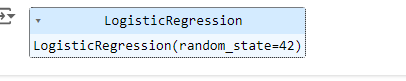
# Split data into training and testing sets

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X\_scaled, y, test\_size=0.3, random\_state=42)

# Fit Logistic Regression model

logreg = LogisticRegression(random\_state=42)

logreg.fit(X\_train, y\_train)



# Get coefficients

coefficients = logreg.coef\_[0]

features = X.columns

# Plot coefficients

plt.figure(figsize=(10, 6))

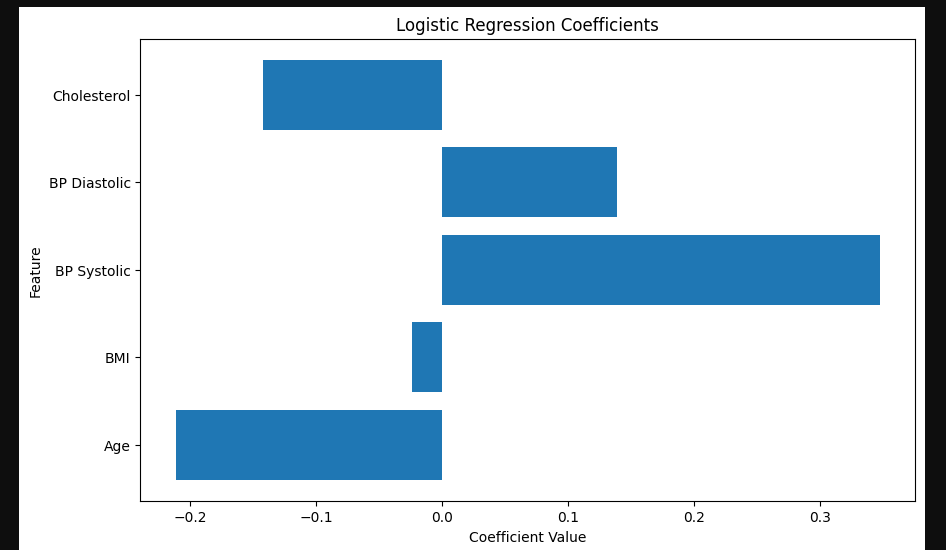
plt.barh(features, coefficients)

plt.xlabel('Coefficient Value')

plt.ylabel('Feature')

plt.title('Logistic Regression Coefficients')

plt.show()

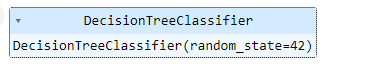


from sklearn.tree import DecisionTreeClassifier, plot\_tree

# Fit Decision Tree model

tree = DecisionTreeClassifier(random\_state=42)

tree.fit(X\_train, y\_train)



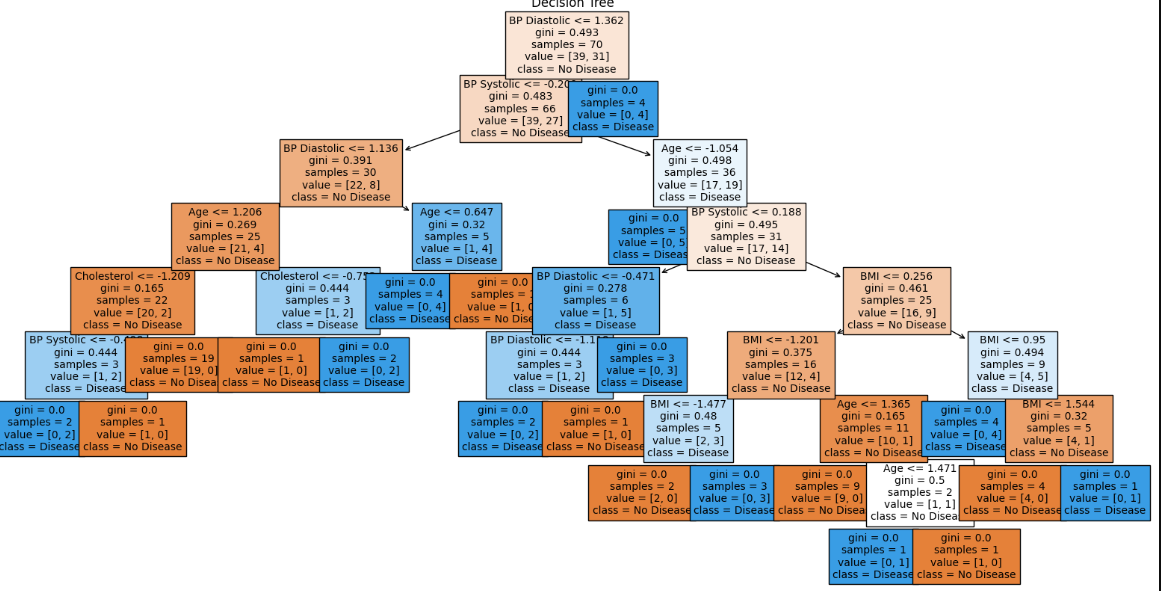
# Plot the Decision Tree

plt.figure(figsize=(20, 10))

plot\_tree(tree, feature\_names=X.columns, class\_names=['No Disease', 'Disease'], filled=True, fontsize=10)

plt.title('Decision Tree')

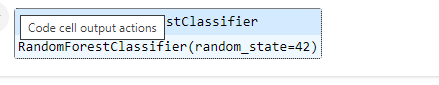
plt.show()



from sklearn.ensemble import RandomForestClassifier

forest = RandomForestClassifier(random\_state=42)

forest.fit(X\_train, y\_train)



# Get feature importances

importances = forest.feature\_importances\_

indices = np.argsort(importances)[::-1]

# Plot feature importances

plt.figure(figsize=(10, 6))

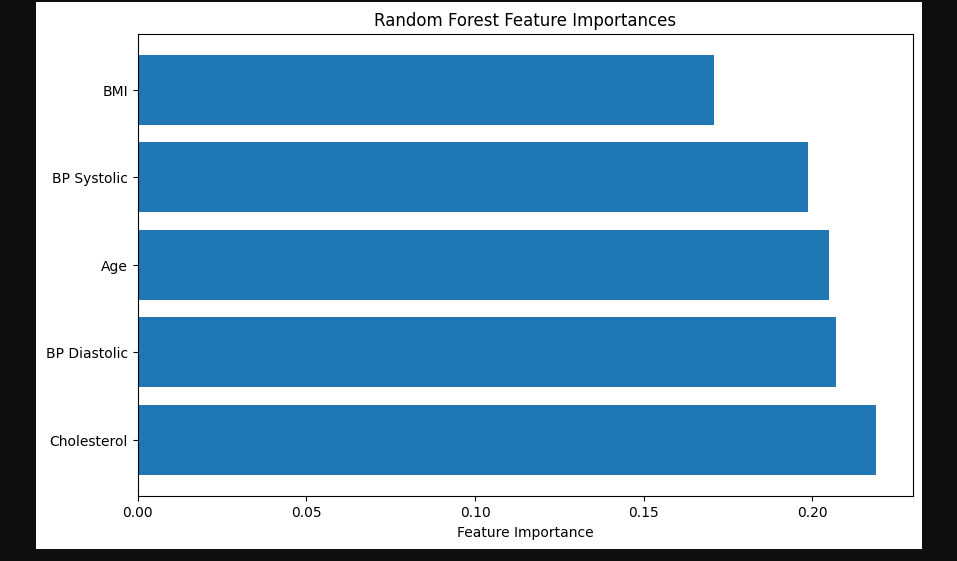
plt.title('Random Forest Feature Importances')

plt.barh(range(X.shape[1]), importances[indices], align='center')

plt.yticks(range(X.shape[1]), [features[i] for i in indices])

plt.xlabel('Feature Importance')

plt.show()

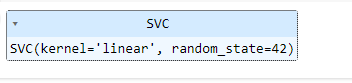


from sklearn.svm import SVC

# Fit Linear SVM model

svm = SVC(kernel='linear', random\_state=42)

svm.fit(X\_train, y\_train)



# Get coefficients

svm\_coefficients = svm.coef\_[0]

# Plot coefficients

plt.figure(figsize=(10, 6))

plt.barh(features, svm\_coefficients)

plt.xlabel('Coefficient Value')

plt.ylabel('Feature')

plt.title('Linear SVM Coefficients')

plt.show()

