# Import necessary libraries

import pandas as pd

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

from sklearn.preprocessing import StandardScaler

from sklearn.ensemble import RandomForestClassifier

from sklearn.metrics import accuracy\_score, classification\_report

# Load preprocessed data

data = pd.read\_csv('blood\_flow\_data.csv')

# Split data into training and testing sets

X\_train, X\_test, y\_train, y\_test = train\_test\_split(data.drop('disease\_status', axis=1), data['disease\_status'], test\_size=0.2, random\_state=42)

# Scale the data

scaler = StandardScaler()

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

X\_test = scaler.transform(X\_test)

# Train a random forest classifier

clf = RandomForestClassifier(n\_estimators=100, random\_state=42)

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

# Evaluate the model on the test set

y\_pred = clf.predict(X\_test)

print("Accuracy:", accuracy\_score(y\_test, y\_pred))

print(classification\_report(y\_test, y\_pred))

In this program, we load blood flow data from a CSV file and select six attributes (age, weight, height, glucose level, cholesterol level, and blood pressure) as input features. We split the data into training and testing sets and scale the data using the StandardScaler. We then train a random forest regression model to predict blood flow values based on the input features. We make predictions on the test set and evaluate the model using R2 score. You can modify this code to use different input features or machine learning algorithms as needed

# Import necessary libraries

import pandas as pd

import numpy as np

from sklearn.preprocessing import StandardScaler

from sklearn.ensemble import RandomForestRegressor

from sklearn.metrics import mean\_squared\_error, r2\_score

# Load data

data = pd.read\_csv('blood\_flow\_data.csv')

# Select relevant attributes

X = data[['age', 'blood\_pressure', 'heart\_rate', 'oxygen\_saturation']]

y = data['blood\_flow']

# Split data into training and testing sets

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

# Scale the data

scaler = StandardScaler()

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

X\_test = scaler.transform(X\_test)

# Train a random forest regressor

regressor = RandomForestRegressor(n\_estimators=100, random\_state=42)

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

# Make predictions on the test set

y\_pred = regressor.predict(X\_test)

# Evaluate the model using mean squared error and R2 score

mse = mean\_squared\_error(y\_test, y\_pred)

r2 = r2\_score(y\_test, y\_pred)

print('Mean Squared Error:', mse)

print('R2 Score:', r2)

In this program, we load blood flow data from a CSV file and select six attributes (age, weight, height, glucose level, cholesterol level, and blood pressure) as input features. We split the data into training and testing sets and scale the data using the StandardScaler. We then train a random forest regression model to predict blood flow values based on the input features. We make predictions on the test set and evaluate the model using R2 score. You can modify this code to use different input features or machine learning algorithms as needed

# Import necessary libraries

import pandas as pd

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

from sklearn.preprocessing import MinMaxScaler

from sklearn.linear\_model import LinearRegression

from sklearn.metrics import r2\_score

# Load data

data = pd.read\_csv('blood\_flow\_data.csv')

# Select relevant attributes

X = data[['age', 'weight', 'height', 'glucose\_level']]

y = data['blood\_flow']

# Split data into training and testing sets

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

# Scale the data

scaler = MinMaxScaler()

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

X\_test = scaler.transform(X\_test)

# Train a linear regression model

regressor = LinearRegression()

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

# Make predictions on the test set

y\_pred = regressor.predict(X\_test)

# Evaluate the model using R2 score

r2 = r2\_score(y\_test, y\_pred)

print('R2 Score:', r2)

In this program, we load blood flow data from a CSV file and select six attributes (age, weight, height, glucose level, cholesterol level, and blood pressure) as input features. We split the data into training and testing sets and scale the data using the StandardScaler. We then train a random forest regression model to predict blood flow values based on the input features. We make predictions on the test set and evaluate the model using R2 score. You can modify this code to use different input features or machine learning algorithms as needed

# Import necessary libraries

import pandas as pd

import numpy as np

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

from sklearn.preprocessing import StandardScaler

from sklearn.ensemble import RandomForestRegressor

from sklearn.metrics import r2\_score

# Load data

data = pd.read\_csv('blood\_flow\_data.csv')

# Select relevant attributes

X = data[['age', 'weight', 'height', 'glucose\_level', 'cholesterol\_level', 'blood\_pressure']]

y = data['blood\_flow']

# Split data into training and testing sets

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

# Scale the data

scaler = StandardScaler()

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

X\_test = scaler.transform(X\_test)

# Train a random forest regression model

regressor = RandomForestRegressor(n\_estimators=100, random\_state=42)

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

# Make predictions on the test set

y\_pred = regressor.predict(X\_test)

# Evaluate the model using R2 score

r2 = r2\_score(y\_test, y\_pred)

print('R2 Score:', r2)

In this program, we load blood flow data from a CSV file and select six attributes (age, weight, height, glucose level, cholesterol level, and blood pressure) as input features. We split the data into training and testing sets and scale the data using the StandardScaler. We then train a random forest regression model to predict blood flow values based on the input features. We make predictions on the test set and evaluate the model using R2 score. You can modify this code to use different input features or machine learning algorithms as needed