```
import pandas as pd

# Create the dataset
data = {
    'Pregnancies': [6, 1, 8, 1, 0, 5, 3, 10, 2, 8],
    'Glucose': [148, 85, 183, 89, 137, 116, 78, 115, 197, 125],
    'BloodPressure': [72, 66, 64, 66, 40, 74, 50, 0, 70, 96],
    'SkinThickness': [35, 29, 0, 23, 35, 0, 32, 0, 45, 0],
    'Insulin': [0, 0, 0, 94, 168, 0, 88, 0, 543, 0],
    'BMI': [33.6, 26.6, 23.3, 28.1, 43.1, 25.6, 31.6, 0, 30.5, 0],
    'DiabetesPedigreeFunction': [0.627, 0.351, 0.672, 0.167, 2.288, 0.201, 0.248, 0.134, 0.158, 0.23
    'Age': [50, 31, 32, 21, 33, 30, 26, 29, 53, 54],
    'Outcome': [1, 0, 1, 0, 1, 0, 1, 0, 1, 1]
}

# Create a DataFrame
df = pd.DataFrame(data)

# Save it to CSV
df.to_csv("pima_indian_diabetes.csv", index=False)

print("Dataset saved as pima_indian_diabetes.csv")
```

```
Dataset saved as pima_indian_diabetes.csv
```

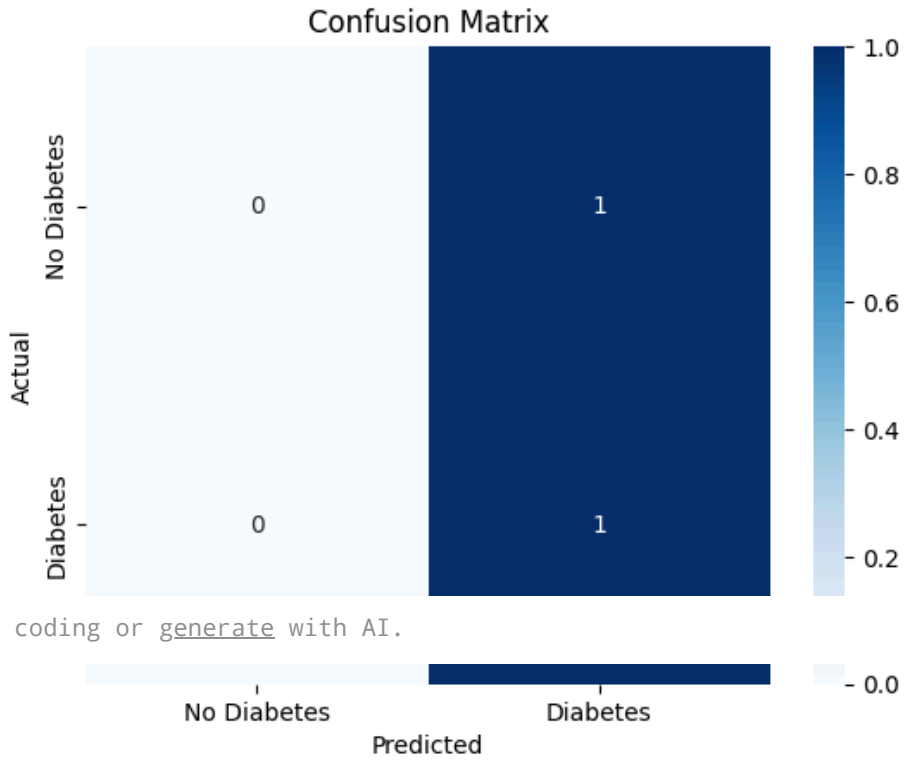
```
# Load the dataset
df = pd.read_csv('pima_indian_diabetes.csv')

# Print the first few rows of the dataset to check it
print(df.head())
```

```
   Pregnancies  Glucose  BloodPressure  SkinThickness  Insulin   BMI  \
0            6      148             72             35        0  33.6
1            1       85             66             29        0  26.6
2            8      183             64              0        0  23.3
3            1       89             66             23       94  28.1
4            0      137             40             35      168  43.1

   DiabetesPedigreeFunction  Age  Outcome
0                     0.627   50        1
1                     0.351   31        0
2                     0.672   32        1
3                     0.167   21        0
4                     2.288   33        1
```

```
# Import necessary libraries
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, confusion_matrix, classification_report
from sklearn.preprocessing import StandardScaler
import seaborn as sns
import matplotlib.pyplot as plt

# Load the dataset (from the previously saved CSV file)
df = pd.read_csv('pima_indian_diabetes.csv')
```

```
df = pd.read_csv('pima_indian_diabetes.csv')

# Display basic info about the dataset (optional)
print(df.info())

# Split the data into features (X) and target (y)
X = df.drop('Outcome', axis=1)  # Features (all columns except 'Outcome')
y = df['Outcome']  # Target variable (1 = Diabetic, 0 = Non-diabetic)

# Split the data into training and testing sets (80% train, 20% test)
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Standardize the features (important for models like Random Forest)
scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)

# Initialize the Random Forest Classifier model
rf_model = RandomForestClassifier(n_estimators=100, random_state=42)

# Train the model
rf_model.fit(X_train_scaled, y_train)

# Predict on the test set
y_pred = rf_model.predict(X_test_scaled)

# Evaluate the model's accuracy
accuracy = accuracy_score(y_test, y_pred)
print(f'Accuracy: {accuracy * 100:.2f}%')

# Confusion Matrix
conf_matrix = confusion_matrix(y_test, y_pred)
print(f'Confusion Matrix:\n{conf_matrix}')

# Classification Report (precision, recall, F1-score)
print(f'Classification Report:\n{classification_report(y_test, y_pred)}')

# Plot the confusion matrix
sns.heatmap(conf_matrix, annot=True, fmt="d", cmap="Blues", xticklabels=["No Diabetes", "Diabetes"],
plt.xlabel('Predicted')
plt.ylabel('Actual')
plt.title('Confusion Matrix')
plt.show()

# Feature Importances (to understand which features are important for prediction)
feature_importances = rf_model.feature_importances_
features = X.columns
feature_importance_df = pd.DataFrame({'Feature': features, 'Importance': feature_importances})
feature_importance_df = feature_importance_df.sort_values(by='Importance', ascending=False)

# Display feature importance
print(f'Feature Importance:\n{feature_importance_df}')
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10 entries, 0 to 9
Data columns (total 9 columns):
 #   Column                    Non-Null Count  Dtype  
---  ------                    --------------  -----  
 0   Pregnancies               10 non-null     int64  
 1   Glucose                   10 non-null     int64  
 2   BloodPressure             10 non-null     int64  
 3   SkinThickness             10 non-null     int64  
 4   Insulin                   10 non-null     int64  
 5   BMI                       10 non-null     float64
 6   DiabetesPedigreeFunction  10 non-null     float64
 7   Age                       10 non-null     int64  
 8   Outcome                   10 non-null     int64  
dtypes: float64(2), int64(7)
memory usage: 852.0 bytes
None
Accuracy: 50.00%
Confusion Matrix:
[[0 1]
 [0 1]]
Classification Report:
              precision    recall  f1-score   support

           0       0.00      0.00      0.00         1
           1       0.50      1.00      0.67         1

    accuracy                           0.50         2
   macro avg       0.25      0.50      0.33         2
weighted avg       0.25      0.50      0.33         2

/usr/local/lib/python3.11/dist-packages/sklearn/metrics/_classification.py:1565: UndefinedMetr
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
/usr/local/lib/python3.11/dist-packages/sklearn/metrics/_classification.py:1565: UndefinedMetr
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
/usr/local/lib/python3.11/dist-packages/sklearn/metrics/_classification.py:1565: UndefinedMetr
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
```



Start coding or generate with AI.

```
Feature Importance:
                    Feature  Importance
```

1	Glucose	0.229967
6	DiabetesPedigreeFunction	0.186719
5	BMI	0.155099
7	Age	0.147550
2	BloodPressure	0.128113
0	Pregnancies	0.081667
3	SkinThickness	0.036204
4	Insulin	0.034683