Source code

```
# STEP 1: Load Dataset
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.preprocessing import LabelEncoder, StandardScaler
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import classification_report,
confusion_matrix, accuracy_score

# Read the uploaded file (manually uploaded to /content/)
df = pd.read_csv('/content/patient_data.csv')

# STEP 2: Preview the Dataset
print("First 5 Records:")
print(df.head())

# Display summary
print("\nDataset Info:")
print(df.info())

# Drop 'name' column if it's not predictive
if 'name' in df.columns:
    names = df['name']
    df = df.drop(columns=['name'])  # Optional: store names
for later

# STEP 3: Handle Missing Values (if any)
df = df.dropna()

# STEP 4: Separate Features and Target
X = df.drop(columns=['disease'])
y = df['disease']

# Encode categorical variables
X = pd.get_dummies(X)

# Encode target

le = LabelEncoder()
y_encoded = le.fit_transform(y)
```

```
# STEP 5: Feature Scaling
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)

# STEP 6: Split Dataset
X_train, X_test, y_train, y_test = train_test_split(X_scaled, 
y_encoded, test_size=0.2, random_state=42)

# STEP 7: Train the Model
model = RandomForestClassifier(n_estimators=100, 
random_state=42)
model.fit(X_train, y_train)

# STEP 8: Evaluate the Model
y_pred = model.predict(X_test)

print("\nAccuracy:", accuracy_score(y_test, y_pred))
print("\nClassification Report:")
print(classification_report(y_test, y_pred, 
target_names=le.classes_))

# Confusion Matrix
conf_matrix = confusion_matrix(y_test, y_pred)
plt.figure(figsize=(8,6))
sns.heatmap(conf_matrix, annot=True, fmt='d', cmap='Blues', 
xticklabels=le.classes_, yticklabels=le.classes_)
plt.xlabel("Predicted")
plt.ylabel("Actual")
plt.title("Confusion Matrix")
plt.show()

# STEP 9: Predict New Patient Disease
print("\nEnter details of the new patient:")
new_patient = {}
for col in df.drop(columns=['disease']).columns:
    val = input(f"{col}: ")
    new_patient[col] = val if not val.replace('.', '', 
1).isdigit() else float(val)

# Convert to DataFrame and process
new_patient_df = pd.DataFrame([new_patient])
new_patient_df = pd.get_dummies(new_patient_df)

new_patient_df = new_patient_df.reindex(columns=X.columns, 
fill_value=0)
new_patient_scaled = scaler.transform(new_patient_df)

# Make prediction
```

```
pred = model.predict(new_patient_scaled)
predicted_disease = le.inverse_transform(pred)
print("\nPredicted Disease:", predicted_disease[0])
```