```python
from google.colab import files
uploaded = files.upload()
```

```
<IPython.core.display.HTML object>
```

```
Saving test (1).csv to test (1).csv
```

```python
from google.colab import files
uploaded = files.upload()
```

```
<IPython.core.display.HTML object>
```

```
Saving train (1).csv to train (1).csv
```

```python
import pandas as pd
df = pd.read_csv("train (1).csv")
print("Shape:", df.shape)
df.head()
```

```
Shape: (15000, 40)
```

```
{"type":"dataframe","variable_name":"df"}
```

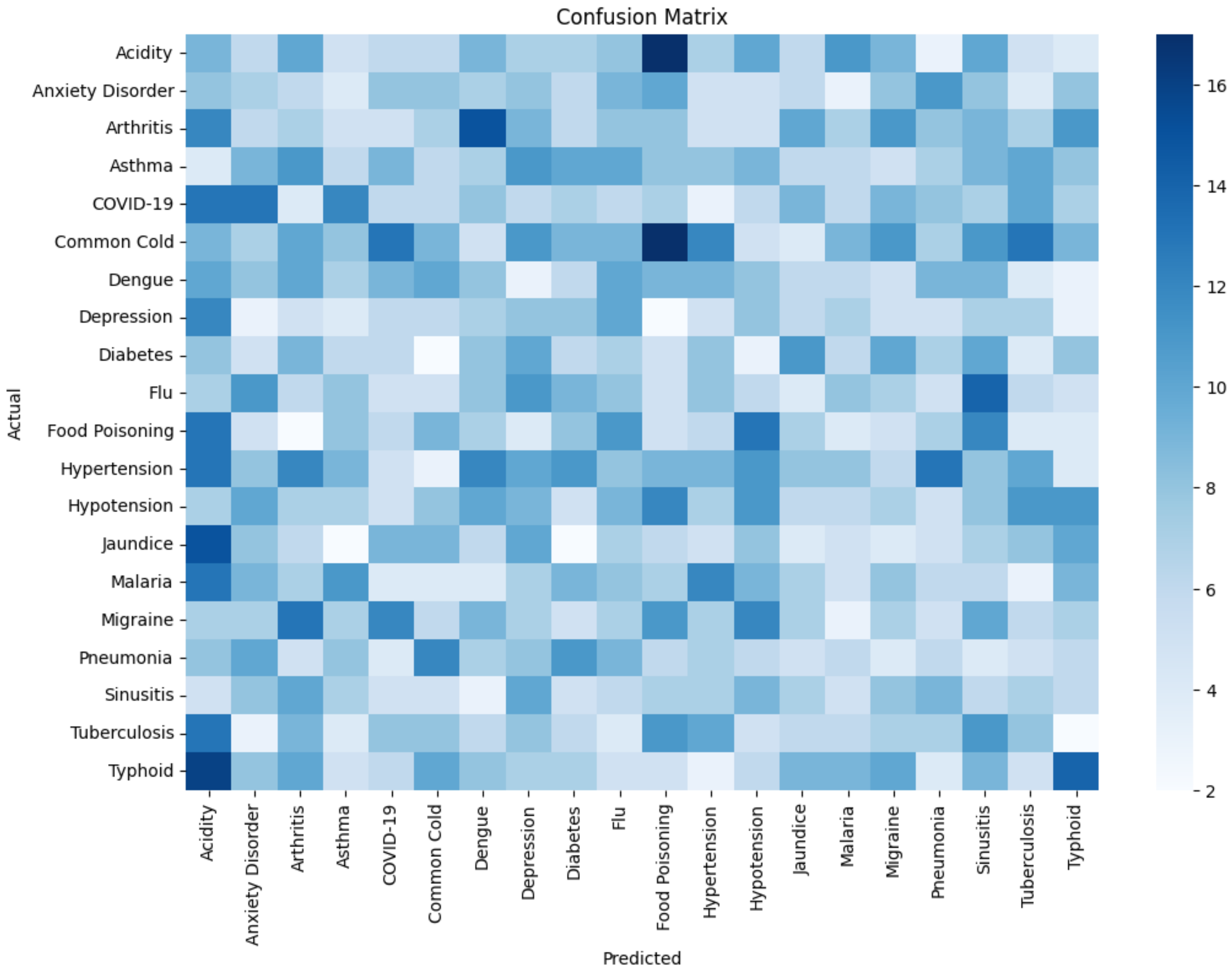
```python
import pandas as pd
df = pd.read_csv("train (1).csv")
print("Shape:", df.shape)
df.head()
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder

X = df.drop("prognosis", axis=1)
y = df["prognosis"]

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

X_train, X_test, y_train, y_test = train_test_split(X, y_encoded,
test_size=0.2, random_state=42)
```

```
Shape: (15000, 40)
```

```python
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import classification_report, confusion_matrix
import seaborn as sns
import matplotlib.pyplot as plt

model = RandomForestClassifier()
model.fit(X_train, y_train)

y_pred = model.predict(X_test)
print(classification_report(y_test, y_pred, target_names=le.classes_))
```

```
cm = confusion_matrix(y_test, y_pred)
plt.figure(figsize=(12, 8))
sns.heatmap(cm, annot=False, cmap="Blues", xticklabels=le.classes_,
yticklabels=le.classes_)
plt.title("Confusion Matrix")
plt.xlabel("Predicted")
plt.ylabel("Actual")
plt.show()
```

```
                  precision    recall  f1-score   support

         Acidity       0.04      0.06      0.05       155
Anxiety Disorder       0.05      0.05      0.05       139
       Arthritis       0.04      0.04      0.04       161
          Asthma       0.05      0.04      0.04       159
        COVID-19       0.04      0.04      0.04       153
     Common Cold       0.06      0.05      0.06       188
          Dengue       0.05      0.05      0.05       149
      Depression       0.05      0.06      0.06       124
        Diabetes       0.04      0.04      0.04       139
             Flu       0.05      0.05      0.05       146
  Food Poisoning       0.03      0.04      0.03       140
    Hypertension       0.06      0.05      0.06       177
     Hypotension       0.07      0.07      0.07       161
        Jaundice       0.03      0.03      0.03       136
         Malaria       0.04      0.03      0.04       148
        Migraine       0.05      0.05      0.05       155
       Pneumonia       0.04      0.04      0.04       137
       Sinusitis       0.03      0.04      0.04       135
    Tuberculosis       0.06      0.06      0.06       142
         Typhoid       0.10      0.09      0.09       156

        accuracy                           0.05      3000
       macro avg       0.05      0.05      0.05      3000
    weighted avg       0.05      0.05      0.05      3000
```



```
test = pd.read_csv("test (1).csv")
test_X = test.drop("prognosis", axis=1)
test_y = test["prognosis"]

test_preds = model.predict(test_X)
test_preds_labels = le.inverse_transform(test_preds)

test["Predicted"] = test_preds_labels
test.to_csv("predicted_results.csv", index=False)

from google.colab import files
files.download("predicted_results.csv")
```

```
<IPython.core.display.Javascript object>

<IPython.core.display.Javascript object>
```

```
import ipywidgets as widgets
from IPython.display import display

symptom_widgets = {symptom: widgets.Checkbox(value=False,
```

```python
description=symptom.replace("_", " ").title()) for symptom in
X.columns}

print("□ Select Symptoms:")
for sw in symptom_widgets.values():
    display(sw)

button = widgets.Button(description="Predict Disease",
button_style='success')
output = widgets.Output()

def on_button_click(b):
    symptom_input = [int(symptom_widgets[s].value) for s in X.columns]
    pred = model.predict([symptom_input])[0]
    disease = le.inverse_transform([pred])[0]
    precautions = {
        "Flu": ["Rest", "Fluids", "Paracetamol"],
        "Diabetes": ["Monitor sugar", "Insulin", "Diet"],
        "Asthma": ["Inhaler", "Avoid triggers", "Medication"],
        "COVID-19": ["Isolation", "Hydration", "Consult doctor"],
        "Malaria": ["Antimalarials", "Rest", "Drink water"],
        # Add more here
    }

    with output:
        output.clear_output()
        print(f"□ Predicted Disease: {disease}")
        print("□ Recommended Precautions:")
        for item in precautions.get(disease, ["No data available."]):
            print(f" - {item}")

button.on_click(on_button_click)
display(button, output)
```

```
□ Select Symptoms:
```

```
{"model_id":"7b87626d3d704568a35132dc68313c98","version_major":2,"vers
ion_minor":0}

{"model_id":"b322e09ec20a49288c8160604010bc21","version_major":2,"vers
ion_minor":0}

{"model_id":"4e2f9c347a31447ba69e871e9d23d934","version_major":2,"vers
ion_minor":0}

{"model_id":"50f5b9431ff14687b7619dc7ec0b5d70","version_major":2,"vers
ion_minor":0}

{"model_id":"52d65b303bde48a5b123ea6ce8c04685","version_major":2,"vers
ion_minor":0}
```

```
{"model_id":"64d79505fee04c45b4fd586e02c61696","version_major":2,"version_minor":0}

{"model_id":"807cef9845364dfda42450ac93b20922","version_major":2,"version_minor":0}

{"model_id":"45279562854646d18fe1095218d3cbde","version_major":2,"version_minor":0}

{"model_id":"15dc8a52d2954414b24fcf342755c670","version_major":2,"version_minor":0}

{"model_id":"f2c3bd9654ff4c2b8e49891d1bd296b8","version_major":2,"version_minor":0}

{"model_id":"79dcde3ec160469b9f51f2d915df1baa","version_major":2,"version_minor":0}

{"model_id":"9310e85d166e4ec79a5bd6fab8daaf40","version_major":2,"version_minor":0}

{"model_id":"f3a11db81ce34386bd3741adfe8594f7","version_major":2,"version_minor":0}

{"model_id":"81d8bcb289d94dc1b209a3150cec4a6e","version_major":2,"version_minor":0}

{"model_id":"e2658c7148b5495c9fc2fed435ca1005","version_major":2,"version_minor":0}

{"model_id":"d4f0e6505a974a90804aa9e62b73ba6b","version_major":2,"version_minor":0}

{"model_id":"b15d79d8fcf441249af938f9d2be4364","version_major":2,"version_minor":0}

{"model_id":"9d1321e745794afbb029cc14b5d046dd","version_major":2,"version_minor":0}

{"model_id":"09b2201140c748958e0d58570877510f","version_major":2,"version_minor":0}

{"model_id":"dc42974ab62648ac85c45913e0d375bc","version_major":2,"version_minor":0}

{"model_id":"4c845d7a827a43a4a947a2e9ee15cba1","version_major":2,"version_minor":0}

{"model_id":"4a03716eea3941f2824ae4f1d8479b92","version_major":2,"version_minor":0}

{"model_id":"32e5d8f4037e441ab41eb64abc1f220a","version_major":2,"version_minor":0}
```

```
{"model_id":"718510fa23b64f919a79020ef5e26ea9","version_major":2,"version_minor":0}

{"model_id":"b6f10e47ef814cacad304015b3240db5","version_major":2,"version_minor":0}

{"model_id":"f2a44d78657c4a9990dd6a325cfa7df8","version_major":2,"version_minor":0}

{"model_id":"1a9e20d4c80f4d7181fcc163ca1e7d0a","version_major":2,"version_minor":0}

{"model_id":"dddad290e21f451494cbc866b039856d","version_major":2,"version_minor":0}

{"model_id":"63bd0f4dcc1b42068643b8e0aa03bc3f","version_major":2,"version_minor":0}

{"model_id":"623af44da37a415797aa5331695a5c7f","version_major":2,"version_minor":0}

{"model_id":"e7133c6fcde447479d29bec7306b3911","version_major":2,"version_minor":0}

{"model_id":"b5ed7e200180421ca29e2d460afd3164","version_major":2,"version_minor":0}

{"model_id":"d4ecf05b54434e6daaeb438bcf66397f","version_major":2,"version_minor":0}

{"model_id":"6c1ea7399f2d43f2ad18c1fba700ae44","version_major":2,"version_minor":0}

{"model_id":"93025cd48fdf4b6494f4f72090ff7890","version_major":2,"version_minor":0}

{"model_id":"8907e1ceae184ec8953169b7083b1ec0","version_major":2,"version_minor":0}

{"model_id":"083fd339d41b41608490035005f7718b","version_major":2,"version_minor":0}

{"model_id":"fabc7e7e0f734641a3e5b08b3d741688","version_major":2,"version_minor":0}

{"model_id":"5c56ef257bc14918b0f5fd0ea8b37a4a","version_major":2,"version_minor":0}

{"model_id":"04f0d153b4d1474fb720cf226cecdc61","version_major":2,"version_minor":0}

{"model_id":"c3d5b647d23e4596aefbbe5e12c47881","version_major":2,"version_minor":0}
```

```
/usr/local/lib/python3.11/dist-packages/sklearn/utils/
validation.py:2739: UserWarning: X does not have valid feature names,
but RandomForestClassifier was fitted with feature names
  warnings.warn(
```