


```
from google.colab import drive
drive.mount('/content/drive')
```

 Mounted at /content/drive

```
!pip install pandas scikit-learn openpyxl
```

 Show hidden output

## Import Libraries

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.naive_bayes import MultinomialNB
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score
```

## Load the Dataset

```
# Load the dataset from the provided GitHub link
url = 'https://github.com/Sankaran36/MADM-NB/blob/main/Disease_Drugs_DS.xlsx?raw=true'
df = pd.read_excel(url)
```

## Preprocess the Data

```
# Combine symptoms into a single string for each row
df['Combined_Symptoms'] = df[['Symptom1', 'Symptom2', 'Symptom3']].apply(lambda x: ' '.join(x.dropna().astype(str)), axis=1)

# Split the dataset into training and test sets
X_train, X_test, y_train, y_test = train_test_split(df['Combined_Symptoms'], df['Disease'], test_size=0.2, random_state=42)

# Convert the text data to numerical data using CountVectorizer
vectorizer = CountVectorizer()
X_train_vec = vectorizer.fit_transform(X_train)
X_test_vec = vectorizer.transform(X_test)
```

 Show hidden output

## Train the Naive Bayes Model

```
# Initialize and train the MultinomialNB classifier
nb_classifier = MultinomialNB()
nb_classifier.fit(X_train_vec, y_train)
```

 Show hidden output

## Evaluate the Model

```
# Predict on the test set
y_pred = nb_classifier.predict(X_test_vec)

# Calculate evaluation metrics
accuracy = accuracy_score(y_test, y_pred)
precision = precision_score(y_test, y_pred, average='weighted')
recall = recall_score(y_test, y_pred, average='weighted')
f1 = f1_score(y_test, y_pred, average='weighted')

print(f'Accuracy: {accuracy}')
print(f'Precision: {precision}')
print(f'Recall: {recall}')
print(f'F1-score: {f1}')
```

## Predict a Disease Based on Given Symptoms

```
# Define the query symptoms
query = ["fever", "body ache", "chills"]

# Combine the query symptoms into a single string
query_str = ' '.join(query)

# Transform the query string using the same vectorizer
query_vec = vectorizer.transform([query_str])

# Predict the disease
predicted_disease = nb_classifier.predict(query_vec)

print(f'Predicted Disease: {predicted_disease[0]}')
```

 Show hidden output