NOTES

import pandas as pd:

- Used to handle and manipulate data in tabular format (DataFrames), useful for storing text data and labels.

import numpy as np:

- Used for numerical operations and handling arrays, especially when preparing data for machine learning.

CountVectorizer:

- Converts text into a numeric format (word count vectors), making it usable for machine learning models.

train_test_split:

- Splits the data into training and testing sets to evaluate the model's performance on unseen data.

MultinomialNB:

- A Naive Bayes classifier that works well for text classification, predicts the language based on word frequencies.

The code **data.isnull().sum()** is used in **pandas** to check for missing values (null values) in a DataFrame.

The code **data['language'].value_counts()** counts the occurrences of each unique value in the 'language' column of the DataFrame data

- **x = np.array(data["Text"])**:
  Converts the "Text" column from the DataFrame data into a NumPy array and assigns it to x. This represents the input features (text data) used for training the model.
- **y = np.array(data["language"])**:
  Converts the "language" column from the DataFrame data into a NumPy array and assigns it to y. This represents the target labels (the actual language) for each text in x.

- **cv = CountVectorizer()**: This initializes a **CountVectorizer** object, which is used to convert a collection of text documents into a matrix of token counts (word frequency). It tokenizes the text and counts the occurrences of each word. It is commonly used to prepare text data for machine learning models.

- **X = cv.fit_transform(x)**: This line converts the text data (x) into a matrix of word counts
- **model = MultinomialNB()**: Initializes the Naive Bayes model.
- **model.fit(X_train, y_train)**: Trains the model with training data.
- **model.score(X_test, y_test)**: Tests the model's accuracy on test data.

1. **user = input("Enter a text: ")**:
    - Asks the user to type some text.
2. **data = cv.transform([user]).toarray()**:
    - Turns the user's text into numbers that the model can understand.
3. **output = model.predict(data)**:
    - The model guesses the language of the text.
4. **print(output)**:
    - Shows the predicted language.