

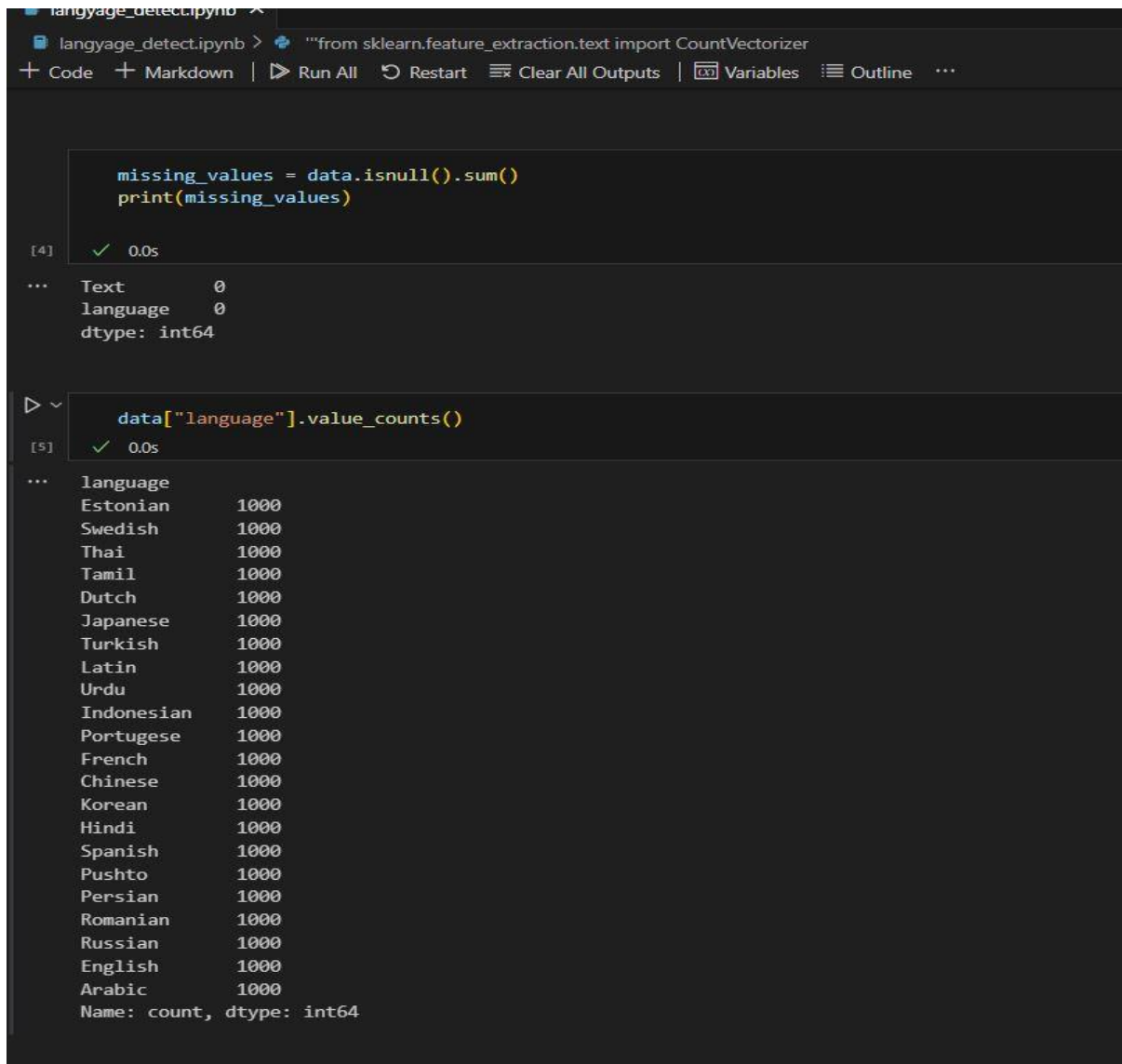
LANGUAGE DETECTION MODEL USING PYTHON + ML

By-Devansh Sharma

Git-Hub- <https://github.com/devanshh7>

Project Summary:

This project focuses on developing a language detection model capable of identifying 22 different languages using a dataset of 20,000 rows. The goal of the model is to accurately predict the language of a given text by leveraging machine learning techniques. The project uses various libraries and approaches for data manipulation, text processing, and model building. Over all this model has achieved 95.33% accuracy which train and testing.



```
language_detect.ipynb > from sklearn.feature_extraction.text import CountVectorizer
```

+ Code + Markdown | ▶ Run All ↺ Restart ≡ Clear All Outputs | [x] Variables ≡ Outline ...

```
missing_values = data.isnull().sum()
print(missing_values)
```

[4] ✓ 0.0s

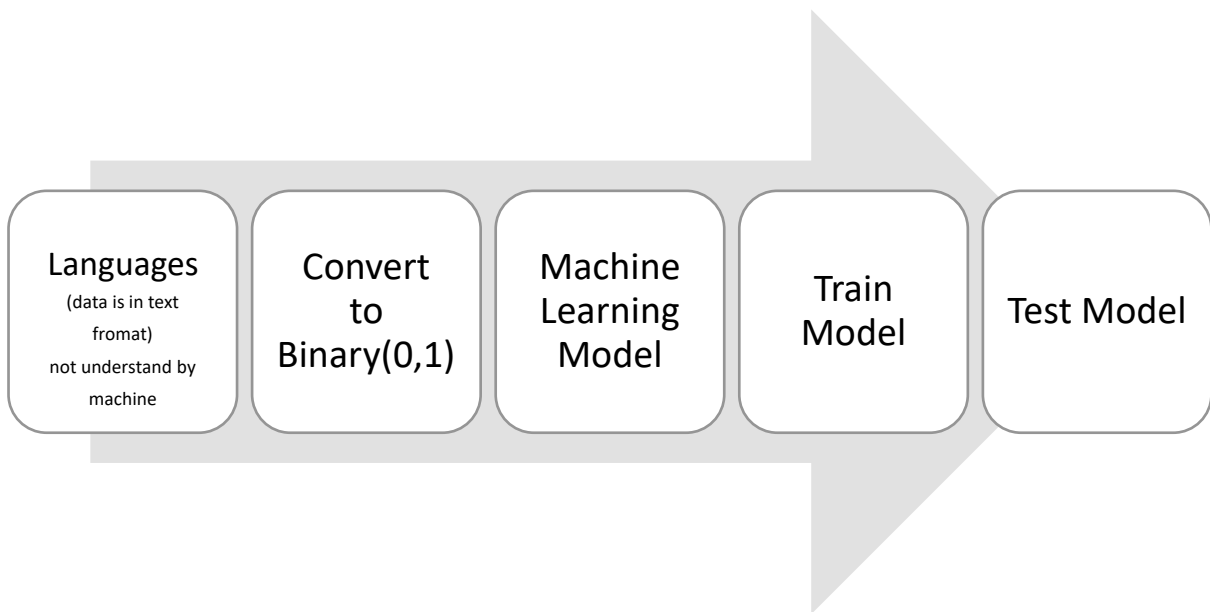
```
... Text      0
    language  0
    dtype: int64
```

▶ ▾ `data["language"].value_counts()`

[5] ✓ 0.0s

```
... language
Estonian      1000
Swedish       1000
Thai          1000
Tamil         1000
Dutch         1000
Japanese      1000
Turkish       1000
Latin         1000
Urdu          1000
Indonesian    1000
Portuguese    1000
French        1000
Chinese       1000
Korean        1000
Hindi         1000
Spanish       1000
Pushto        1000
Persian       1000
Romanian      1000
Russian       1000
English       1000
Arabic        1000
Name: count, dtype: int64
```

PROCESS :



The image illustrates the process of building a language detection model. Here's a brief explanation of each step:

1. **Languages (Text Data):** The data consists of text in different languages, which machines cannot understand directly.
2. **Convert to Binary (0,1):** The text is transformed into a format that a machine can interpret. This is done using techniques like one-hot encoding or Bag of Words, which convert text into numerical representations (binary).
3. **Machine Learning Model:** A machine learning model is then built to process the numerical data and learn patterns that differentiate the languages.
4. **Train Model:** The model is trained on a dataset to learn the features of different languages.
5. **Test Model:** After training, the model is tested on unseen data to evaluate its accuracy in detecting languages.

This sequence ensures that the model can learn from data and generalize well to new text inputs.

Libraries Used:

Pandas: This library was used to load, manage, and manipulate the dataset efficiently. Pandas helped in handling large amounts of data, allowing for quick data cleaning and pre-processing.

NumPy: NumPy provided support for numerical operations, enabling efficient handling of arrays and matrices, which are essential for data transformation and mathematical computations in machine learning.

```
language_detect.ipynb ×
language_detect.ipynb > "from sklearn.feature_extraction.text import CountVectorizer
+ Code + Markdown | ▶ Run All ⌂ Restart ☰ Clear All Outputs | 📄 Variables ☰ Outline ...

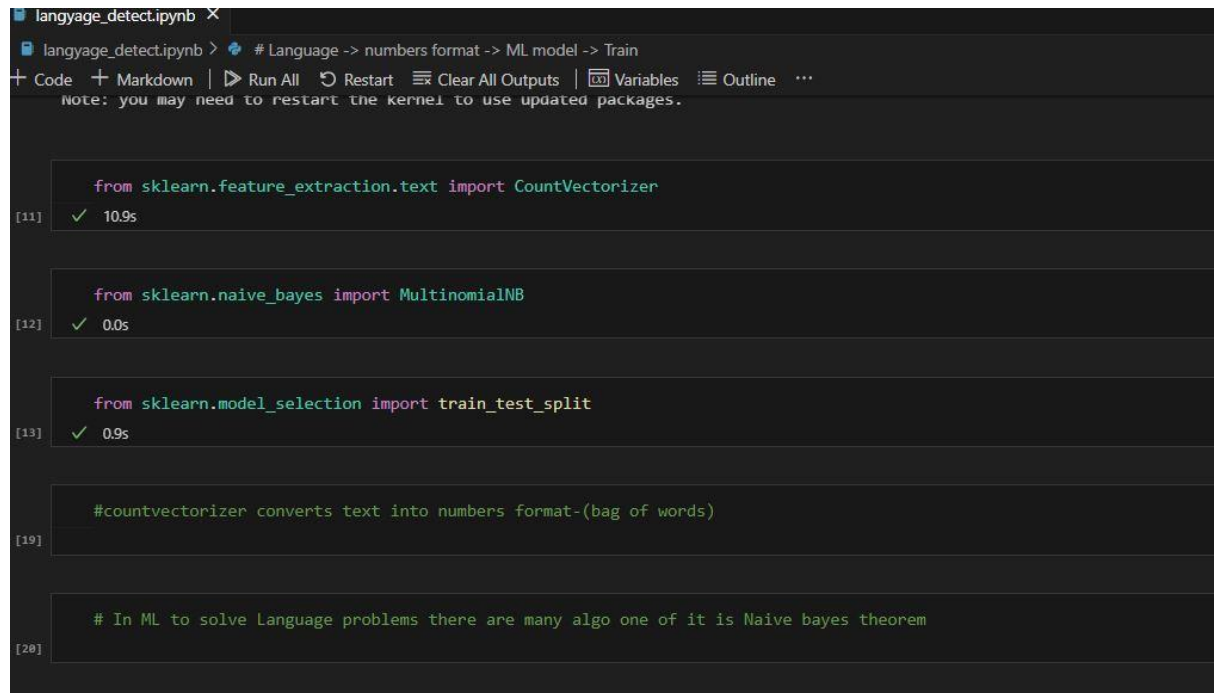
▶
import pandas as pd
import numpy as np
[1] ✓ 6.6s

data = pd.read_csv("language.csv")
[2] ✓ 1.4s

data
[3] ✓ 0.1s
...
      Text  language
0  klement gottwaldi surnukeha palsameeriti ning ...  Estonian
1  sebes joseph pereira thomas på eng the jesuit...  Swedish
2  ถนนเจริญกรุง อัครราชธาน์ thanon charoen krung L...  Thai
3  விசாகப்பட்டுனம் தமிழ்ச்சங்கத்தை இந்துப் பத்திர...  Tamil
4  de spons behoort tot het geslacht haliclona en...  Dutch
...  ...  ...
21995  hors du terrain les années et sont des année...  French
21996  ใน พศ หลังจากที่เราได้ประพาสแหลมมลายู ชาว ฮิน...  Thai
21997  con motivo de la celebración del septuagésimoq...  Spanish
21998  年月，當時還只有歲的她在美國出道，以mai-k名義推出首張英文《baby i like》，由...  Chinese
21999  aprilie sonda spațială messenger a nasa și-a ...  Romanian

22000 rows × 2 columns
```

⇒ **Scikit-learn (sklearn)**: A comprehensive library for machine learning, Scikit-learn was used for implementing various algorithms and utilities, including feature extraction, model building, and evaluation.



```
language_detect.ipynb X
language_detect.ipynb > # Language -> numbers format -> ML model -> Train
+ Code + Markdown | ▶ Run All ⌂ Restart ⌂ Clear All Outputs | 📄 Variables 📄 Outline ...
Note: you may need to restart the kernel to use updated packages.

[11] ✓ 10.9s
from sklearn.feature_extraction.text import CountVectorizer

[12] ✓ 0.0s
from sklearn.naive_bayes import MultinomialNB

[13] ✓ 0.9s
from sklearn.model_selection import train_test_split

[19] #countvectorizer converts text into numbers format-(bag of words)

[20] # In ML to solve Language problems there are many algo one of it is Naive bayes theorem
```

⇒ **Count Vectorizer** (from `sklearn.feature_extraction.text`): CountVectorizer is a tool for converting text data into a matrix of token counts. It transforms the dataset into numerical features by counting the frequency of each word in the text, which serves as input for the machine learning model.

Key Concepts:

Bag of Words (BoW) Approach: The BoW approach was used to represent the text in a way that counts the occurrence of unique words across the dataset. It ignores grammar and word order, focusing on the frequency of words in a sentence or document. This technique helped in creating a numerical representation of text data, enabling the model to distinguish between languages based on word usage.

MultinomialNB (from `sklearn.naive_bayes`): Multinomial Naive Bayes is a probabilistic algorithm commonly used for text classification. In this project, it was applied to predict the language of a given text by estimating the likelihood of word frequencies belonging to different languages.

```
language_detect.ipynb
language_detect.ipynb > # Language -> numbers format -> ML model -> Train
+ Code + Markdown | Run All | Restart | Clear All Outputs | Variables | Outline | ...

[14] ✓ 4.6s
vec = CountVectorizer()
a = vec.fit_transform(x)

[15] ✓ 0.1s
a_train, a_test, y_train, y_test = train_test_split(a, y, test_size= 0.33 , random_state= 42)

[16] ✓ 0.0s
a_train

... <Compressed Sparse Row sparse matrix of dtype 'int64'
    with 613529 stored elements and shape (14740, 277720)>

[25]
# a is input and y is output

[17] ✓ 0.1s
print(a_train)

... <Compressed Sparse Row sparse matrix of dtype 'int64'
    with 613529 stored elements and shape (14740, 277720)>
    Coords      Values
    (0, 197295)    2
    (0, 197708)    1
    (0, 197801)    1
    (0, 198388)    1
    (0, 197467)    1
    (0, 197865)    2
    (0, 197604)    1
    (0, 198428)    1
    (0, 198501)    1
    (0, 198556)    1
    (0, 197332)    1
    (0, 197485)    2
    (0, 198123)    1
    (0, 197892)    1
    (0, 197888)    1
```

train_test_split: This function from Scikit-learn was used to divide the dataset into training and testing sets. The training set is used to train the model, while the testing set evaluates its performance. Splitting the data helps in ensuring that the model can generalize well to unseen data, improving accuracy and reducing overfitting.

This combination of tools and methods resulted in an effective language detection model that can accurately classify text into one of 22 languages.

```
language_detect.ipynb X
language_detect.ipynb > # Language -> numbers format -> ML model -> Train
+ Code + Markdown | ▶ Run All ↺ Restart ≡ Clear All Outputs | [31] Variables ≡ Outline ...

print(y_test)
[18] ✓ 0.0s
... ['Japanese' 'Russian' 'Latin' ... 'Turkish' 'Arabic' 'English']

model = MultinomialNB()
[19] ✓ 0.0s

model.fit(a_train, y_train)
[20] ✓ 0.7s
...
  ▾ MultinomialNB ⓘ ?
  MultinomialNB()

model.score(a_test, y_test) # 95.3% accuracy
[21] ✓ 0.1s
... 0.953168044077135

user = input("Enter a text to check language:")
data = vec.transform([user]).toarray()
output= model.predict(data)
print(user,":",output)
[32] ✓ 4.8s
... this is : ['English']
```

Output Detecting Language

Accuracy is 95.33%.