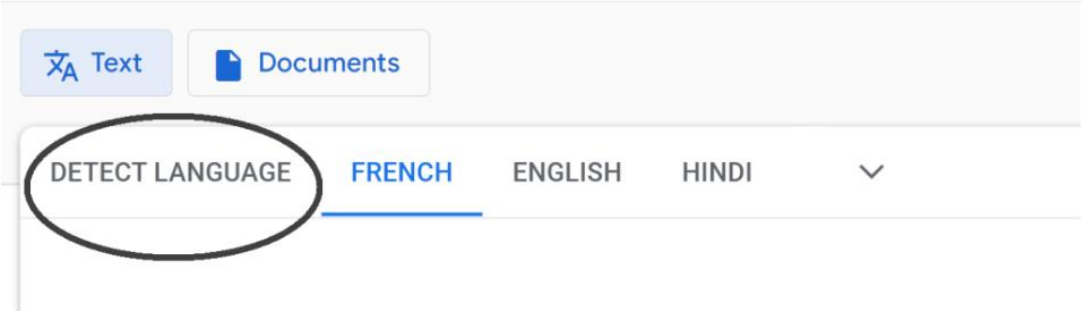
Language Detection with Machine Learning

Language detection is a natural language processing task where we need to identify the language of a text or document. Using machine learning for language identification was a difficult task a few years ago because there was not a lot of data on languages, but with the availability of data with ease, several powerful machine learning models are already available for language identification. So, if you want to learn how to train a machine learning model for language detection, then this article is for you. In this article, I will walk you through the task of language detection with machine learning using Python.

Language Detection

As a human, you can easily detect the languages you know. For example, I can easily identify Hindi and English, but being an Indian, it is also not possible for me to identify all Indian languages. This is where the language identification task can be used. Google Translate is one of the most popular language translators in the world which is used by so many people around the world. It also includes a machine learning model to detect languages that you can use if you don't know which language you want to translate.



Above image is an one of the example of language detection model.

The most important part of training a language detection model is data. The more data you have about every language, the more accurate your model will perform in real-time. The dataset that I am using is collected from Kaggle, so it will be an appropriate dataset for training a language detection model with machine learning. So in the section below, I will take you through how you can train a language detection model with machine learning using Python.

Language Detection using Python

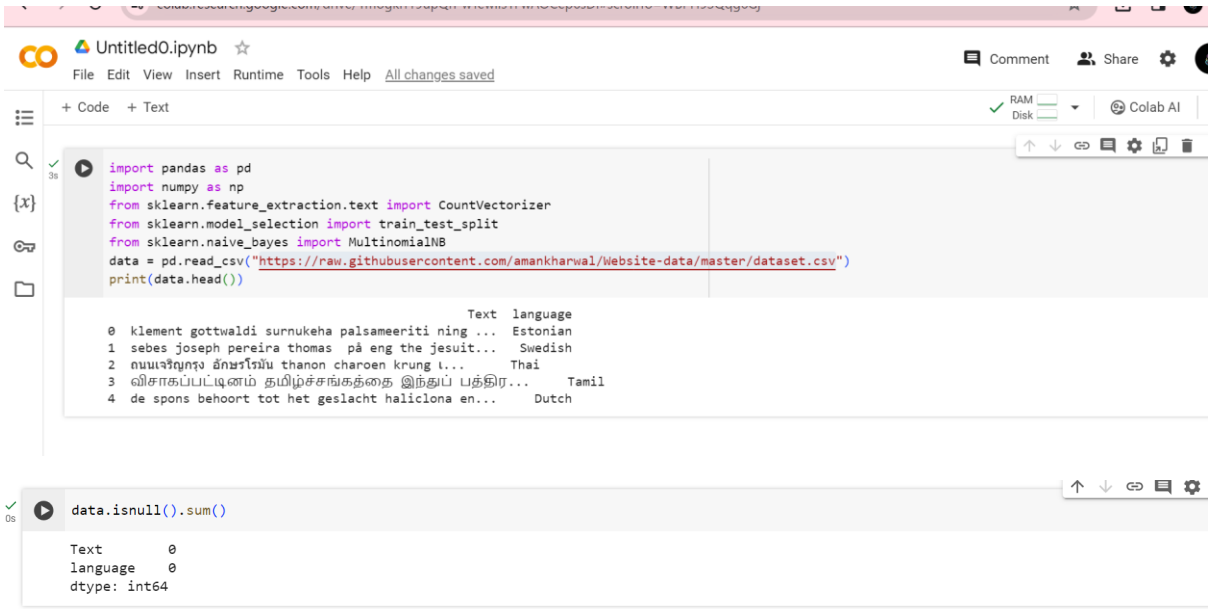
->go to google colab

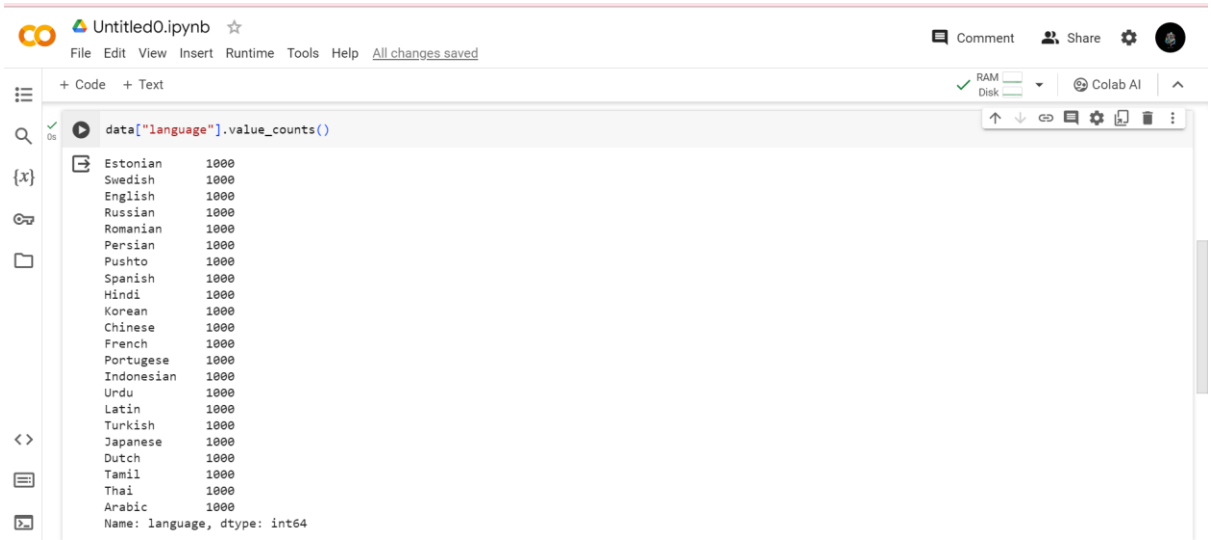
->at first download the dataset from kaggle

->upload to google colab

->https://github.com/amankharwal/Website-data/blob/master/dataset.csv

We have the required dataset there





This dataset contains 22 languages with 1000 sentences from each language. This is a very balanced dataset with no missing values, so we can say this dataset is completely ready to be used to train a machine learning model.

Language Detection Model

Now let's split the data into training and test sets:

```
[3] Latin        1000
    Turkish      1000
    Japanese     1000
    Dutch        1000
    Tamil        1000
    Thai         1000
    Arabic       1000
    Name: language, dtype: int64
```

```
x = np.array(data["Text"])
y = np.array(data["language"])

cv = CountVectorizer()
X = cv.fit_transform(x)
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.33,random_state=42)
```

As this is a problem of multiclass classification, so I will be using the Multinomial Naïve Bayes algorithm to train the language detection model as this algorithm always performs very well on the problems based on multiclass classification:

```
model = MultinomialNB()
model.fit(X_train,y_train)
model.score(X_test,y_test)
```

```
0.953168044077135
```

User input

```
user = input("Enter a Text: ")
data = cv.transform([user]).toarray()
output = model.predict(data)
print(output)
```

```
Enter a Text: तुम्हें देखकर मुझे खुशी होती है
['Hindi']
```

So as you can see that the model performs well. One thing to note here is that this model can only detect the languages mentioned in the dataset.

Conclusion:

Using machine learning for language identification was a difficult task a few years ago because there was not a lot of data on languages, but with the availability of data with ease, several powerful machine learning models are already available for language identification. I hope you liked this article on detecting languages with machine learning using Python. Feel free to ask your valuable questions in the comments section below.

Link is available at github:KAVYA_Sri05