Language Detection!!!

In [1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
```

In [2]:
```
data = pd.read_csv(r"C:\Users\ASUS\Downloads\archive (7)\language.csv")
```

In [3]:
```
data.head()
```

Out[3]:

	Text	language
0	klement gottwaldi surnukeha palsameeriti ning ...	Estonian
1	sebes joseph pereira thomas på eng the jesuit...	Swedish
2	ถนนเจริญกรุง อักษรโรมัน thanon charoen krung เ...	Thai
3	விசாகப்பட்டினம் தமிழ்ச்சங்கத்தை இந்துப் பத்திர...	Tamil
4	de spons behoort tot het geslacht haliclona en...	Dutch

In [4]:
```
data.tail()
```

Out[4]:

	Text	language
21995	hors du terrain les années et sont des année...	French
21996	ใน พศ หลักจากที่เสด็จประพาสแหลมมลายู ชวา อินเ...	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

In [5]:
```
data.sample(10)
```

Out[5]:

	Text	language
8467	nos últimos anos da década de ac preneste eme...	Portugese
20776	o maior astrólogo árabe foi albumazer seu livr...	Portugese
7317	le mars tag heuer annonce à l'occasion du ba...	French
5237	โนเกีย ลูเมีย อังกฤษ nokia lumia เป็นสมาร์ตโ...	Thai
1808	ต่อมาเมื่อสมเด็จพระเทพศิรินทราบรมราชินี มีพระป...	Thai
3324	ในปี โชกุนโทกุงะวะ อิเอะมิสึ มีคำสั่งให้สร้าง...	Thai
11852	年月日首張韓文同名單曲「ss」發行，月日ss開始後續曲〈never again〉宣傳活動。月...	Chinese
19072	cuchilla montes negros är en ås i colombia den...	Swedish
16760	koolikiusamine on koolivägivalla liik mis võib...	Estonian
4944	نیو یارک متحدہ امریکا کے شمال مشرق کی ایک ریاس...	Urdu

In [6]:
```
data.sample(25)
```

Out[6]:

	Text	language
16869	탄산 칼슘과 같은 제산제가 종종 즉각적인 치료에 사용되며 기반이 되는 병인에 따라 ...	Korean
8667	lake pollard är en sjö i australien den ligger...	Swedish
17319	レース実況や場内イベントは、松岡俊道と松岡の事務所のメンバーである高石順成、坂田博昭、吉本靖...	Japanese
21528	hipólito de roma † é geralmente considerado o...	Portugese
7694	دنمونې ځخه • cutting test دقطعه کولو امتحان- ...	Pushto
7602	ஆம் உலக சாரண ஜம்போறி என்பது இல் இடம்பெற்ற உலக...	Tamil
4536	இந்த முறையானது "பாரம்பரிய" முறையாக மிகத் துல்ல...	Tamil
17330	डिक्टाफोन एक अमेरिकी कंपनी थी जो श्रुतलेख मशीन...	Hindi
14500	manuri neograece μανούρι est nomen varietatis ...	Latin
2645	عظیم مفقیث شاعر اور ادیب ۔ پیدائشء	Urdu
7867	leaves of some holly species are used by some ...	English
11049	dan halutz bahasa ibrani דן חלוץ lahir di hago...	Indonesian
12755	albümün ilk klibi "vaziyetler" şarkısına emre ...	Turkish
20858	毛泽东对斯大林评价是：“斯大林不是在所有问题上，而是在一些问题上犯了错误。”“苏联过去把斯大...	Chinese
15047	le fc lourdes a également dominé le championna...	French
7387	ری... lochearn maryland انگریزی	Urdu
12512	総事業費億円規模で工事が進められてきたが、石灰岩の帯水層を貫くことから湧水がみられ、愛媛県側...	Japanese
5242	년 월 일 년 세계육상선수권대회가 개막하여 월 일 폐막하였다 이는 년 일본 도쿄 년...	Korean
6719	پ... کې چه کله حالات ډېر خراب شول نو زاهد خان ا	Pushto
11422	บาซิลิกาซานอันโตนิโอแห่งปาดัว อังกฤษ basilica ...	Thai
17649	...اروابښاد پوهاند ډاکټر نورکل حمزه خيل د زراعت يو	Pushto
5247	ในบริบทของกฎหมาย de jure แปลว่า "ตามกฎหมาย" แต...	Thai
12734	년 월에 민원식이 암살되자 김명준金明濬이 회장 대행으로 취임했다 그러나 대부분의 활...	Korean
7982	aastail – oli estonia ooperisolist koloratuurs...	Estonian
4504	बायैथलॉन भाग जर्मन बायैथलेट ईवी सेचेबैकर-स्टिल...	Hindi

In [7]:
```
data.shape
```

Out[7]: (22000, 2)

In [8]:
```
data.size
```

Out[8]: 44000

In [9]:
```
data.describe()
```

Out[9]:

	Text	language
count	22000	22000
unique	21859	22
top	haec commentatio automatice praeparata res ast...	Estonian
freq	48	1000

In [10]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 22000 entries, 0 to 21999
Data columns (total 2 columns):
 #   Column    Non-Null Count  Dtype 
---  ------    --------------  ----- 
 0   Text      22000 non-null  object
 1   language  22000 non-null  object
dtypes: object(2)
memory usage: 343.9+ KB
```

In [11]:
```
data.Text
```

Out[11]:
```
0        klement gottwaldi surnukeha palsameeriti ning ...
1        sebes joseph pereira thomas  på eng the jesuit...
2        ถนนเจริญกรุง อักษรโรมัน thanon charoen krung เ...
3        விசாகப்பட்டினம் தமிழ்ச்சங்கத்தை இந்துப் பத்திர...
4        de spons behoort tot het geslacht haliclona en...
                              ...
21995    hors du terrain les années  et  sont des année...
21996    ใน พศ  หลักจากที่เสด็จประพาสแหลมมลายู ชวา อินเ...
21997    con motivo de la celebración del septuagésimoq...
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...
21999     aprilie sonda spațială messenger a nasa și-a ...
Name: Text, Length: 22000, dtype: object
```

In [12]:
```
data.language
```

Out[12]:
```
0        Estonian
1         Swedish
2            Thai
3           Tamil
4           Dutch
           ...
21995      French
21996        Thai
21997     Spanish
21998     Chinese
21999    Romanian
Name: language, Length: 22000, dtype: object
```

In [13]:
```
data.isnull().sum()
```

Out[13]:
```
Text        0
language    0
dtype: int64
```

In [14]:
```
data.language.value_counts()
```

Out[14]:
```
language
Estonian      1000
Swedish       1000
English       1000
Russian       1000
Romanian      1000
Persian       1000
Pushto        1000
Spanish       1000
Hindi         1000
Korean        1000
Chinese       1000
French        1000
Portugese     1000
Indonesian    1000
Urdu          1000
Latin         1000
Turkish       1000
Japanese      1000
Dutch         1000
Tamil         1000
Thai          1000
Arabic        1000
Name: count, dtype: int64
```

In [15]:
```
import time
import nltk
import re
from nltk.stem.snowball import SnowballStemmer
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import MultinomialNB
from sklearn.metrics import accuracy_score
```

In [16]:
```
X = data.Text.values
```

In [17]:
```
Y = data.language.values
```

In [18]:
```
X.shape
```

Out[18]:
```
(22000,)
```

In [19]:
```
Y.shape
```

Out[19]:
```
(22000,)
```

In [20]:
```
vector = TfidfVectorizer()
vector.fit(X)
X = vector.transform(X)
```

In [21]:
```
X.shape
```

Out[21]:
```
(22000, 277720)
```

In [22]:
```
print(X)
```

```
  (0, 122429)   0.11632821567894927
  (0, 122098)   0.15245962403688545
  (0, 122097)   0.15245962403688545
  (0, 117124)   0.13392659423607992
  (0, 113245)   0.1389042716940385
  (0, 112024)   0.15245962403688545
  (0, 106285)   0.08285492222494331
  (0, 104967)   0.42661618752454356
  (0, 80288)    0.15245962403688545
  (0, 80287)    0.15245962403688545
  (0, 80056)    0.1464612850687559
  (0, 79323)    0.15245962403688545
  (0, 77619)    0.08182087878336176
  (0, 76696)    0.1389042716940385
  (0, 75304)    0.16625026948941637
  (0, 75247)    0.2290289414877052
  (0, 67654)    0.15245962403688545
  (0, 67653)    0.15245962403688545
  (0, 63450)    0.2433938789015453
  (0, 63122)    0.13392659423607992
  (0, 60954)    0.1464612850687559
  (0, 59244)    0.15245962403688545
  (0, 57772)    0.1389042716940385
  (0, 55264)    0.26785318847215983
  (0, 53103)    0.1322820868685367
  :     :
  (21999, 104844)       0.16248852574304734
  (21999, 103845)       0.18186228813180896
  (21999, 102254)       0.18987120980426156
  (21999, 101742)       0.3576348748525535
  (21999, 101537)       0.19555363016241606
  (21999, 97734)        0.07526526548636828
  (21999, 95539)        0.18546358214789696
  (21999, 88346)        0.20356255183486865
  (21999, 84356)        0.17385336645935637
  (21999, 81608)        0.10343937006427364
  (21999, 74014)        0.13510584168183312
  (21999, 70726)        0.18987120980426156
  (21999, 69551)        0.18987120980426156
  (21999, 69301)        0.3911072603248321
  (21999, 66036)        0.10270771489197011
  (21999, 43690)        0.20356255183486865
  (21999, 40786)        0.15215310275629662
  (21999, 38077)        0.1309466755562267
  (21999, 28194)        0.09160270401248888
  (21999, 25042)        0.19212934088982778
  (21999, 20053)        0.20356255183486865
  (21999, 17371)        0.20944489288925866
  (21999, 6037) 0.16016202442874922
  (21999, 6023) 0.14759970003791315
  (21999, 4888) 0.1290413507799354
```

In [23]:
```
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.2, stratify = Y,random_state = 2)
```

In [24]:
```
print(X_train)
```

```
  (0, 194266)   0.17396412117237442
  (0, 190509)   0.15360935793211114
  (0, 186377)   0.1441449320049061
  (0, 179829)   0.19094088458465805
  (0, 179091)   0.33546012636973144
  (0, 178349)   0.19094088458465805
  (0, 177544)   0.3174115931599312
  (0, 173251)   0.1705861213443948
  (0, 171073)   0.15360935793211114
  (0, 170996)   0.13244891541159626
  (0, 168963)   0.12588001028706056
  (0, 168944)   0.3818817691693161
  (0, 167964)   0.19094088458465805
  (0, 166099)   0.15241360731910408
  (0, 166005)   0.16112169541718976
  (0, 165997)   0.1441449320049061
  (0, 162317)   0.19094088458465805
  (0, 161676)   0.1780984588294734
  (0, 161285)   0.17396412117237442
  (0, 160938)   0.13543684390682043
  (0, 160670)   0.15241360731910408
  (0, 159707)   0.1512903722740091
  (0, 154967)   0.12259441815163577
  (0, 154951)   0.16773006318486572
  (0, 152604)   0.08366621665604251
  :     :
  (17599, 26366)        0.059679632909940904
  (17599, 26365)        0.10661103553625474
  (17599, 26084)        0.059679632909940904
  (17599, 25042)        0.12045066340203094
  (17599, 24657)        0.1423762991316896
  (17599, 22944)        0.05915523259570576
  (17599, 22925)        0.06086173534704326
  (17599, 21307)        0.06153616600256736
  (17599, 20913)        0.05737688997840998
  (17599, 20580)        0.05737688997840998
  (17599, 18845)        0.06024602839798863
  (17599, 18321)        0.11475377995681996
  (17599, 18286)        0.06311516681756726
  (17599, 17371)        0.1125481831792744
  (17599, 14762)        0.05737688997840998
  (17599, 12804)        0.15862958363808308
  (17599, 11365)        0.06086173534704326
  (17599, 11292)        0.06311516681756726
  (17599, 10064)        0.06153616600256736
  (17599, 8949) 0.03861873959534258
  (17599, 8381) 0.06543781279239445
  (17599, 2687) 0.07292481586700715
  (17599, 2543) 0.047278377718548925
  (17599, 2493) 0.07292481586700715
  (17599, 2085) 0.052671407255985916
```

In [25]:
```
Y_train
```

Out[25]:
```
array(['Urdu', 'Spanish', 'English', ..., 'Hindi', 'Hindi', 'French'],
      dtype=object)
```

In [26]:
```
print(Y_train)
```

```
['Urdu' 'Spanish' 'English' ... 'Hindi' 'Hindi' 'French']
```

In [27]:
```
X_test
```

Out[27]:
```
<4400x277720 sparse matrix of type '<class 'numpy.float64'>'
        with 181787 stored elements in Compressed Sparse Row format>
```

In [28]:
```
print(X_test)
```

```
  (0, 118099)   0.12960711440046246
  (0, 117883)   0.11808360374682939
  (0, 117624)   0.10269308550475609
  (0, 115867)   0.0843663794270248
  (0, 114496)   0.1208899151078527
  (0, 104205)   0.12960711440046246
  (0, 104044)   0.12960711440046246
  (0, 103985)   0.07991524532191834
  (0, 101535)   0.08815781243016407
  (0, 100667)   0.18870264968831174
  (0, 100591)   0.09945055954164557
  (0, 100272)   0.12450787970297277
  (0, 100271)   0.25921422880092493
  (0, 100224)   0.12960711440046246
  (0, 97674)    0.12960711440046246
  (0, 93383)    0.12960711440046246
  (0, 89843)    0.08704414186449945
  (0, 89743)    0.11385204135240715
  (0, 89736)    0.09734964390713786
  (0, 89735)    0.09835628182514347
  (0, 89515)    0.07691692625893634
  (0, 86802)    0.11808360374682939
  (0, 85792)    0.09784289380058653
  (0, 85685)    0.0843663794270248
  (0, 82848)    0.12960711440046246
  :     :
  (4398, 36932) 0.2082454152570716
  (4398, 30613) 0.13732694474796733
  (4398, 25042) 0.055004482491003616
  (4398, 15760) 0.1833970507475676
  (4398, 15654) 0.26018878434908493
  (4398, 14855) 0.23309377976657558
  (4398, 4767)  0.23309377976657558
  (4398, 557)   0.20173863477330542
  (4398, 464)   0.21236915622364025
  (4399, 124311)        0.20484590771847613
  (4399, 121243)        0.26067227414430166
  (4399, 121188)        0.27134813745641645
  (4399, 111792)        0.27134813745641645
  (4399, 95821) 0.27134813745641645
  (4399, 90326) 0.2317459019552019
  (4399, 89064) 0.25309762857943147
  (4399, 80382) 0.26067227414430166
  (4399, 76629) 0.27134813745641645
  (4399, 74985) 0.14013018052061196
  (4399, 65532) 0.21500036946481585
  (4399, 58484) 0.27134813745641645
  (4399, 58122) 0.26067227414430166
  (4399, 50769) 0.2348471197024465
  (4399, 48473) 0.27134813745641645
  (4399, 7321)  0.27134813745641645
```

In [29]:
```
Y_test
```

Out[29]:
```
array(['Latin', 'Korean', 'Arabic', ..., 'Latin', 'Romanian', 'Estonian'],
      dtype=object)
```

In [30]:
```
print(Y_test)
```

```
['Latin' 'Korean' 'Arabic' ... 'Latin' 'Romanian' 'Estonian']
```

In [31]:
```
from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()
```

In [32]:
```
Y_train = le.fit_transform(Y_train)
Y_train
```

Out[32]:
```
array([21, 16,  3, ...,  6,  6,  5])
```

In [33]:
```
Y_test = le.fit_transform(Y_test)
Y_test
```

Out[33]:
```
array([10,  9,  0, ..., 10, 14,  4])
```

In [34]:
```
model = MultinomialNB()
model.fit(X_train, Y_train)
```

Out[34]:
```
▾ MultinomialNB
MultinomialNB()
```

In [35]:
```
model.predict(X_test)
```

Out[35]:
```
array([10,  9,  0, ..., 10, 14,  4])
```

In [36]:
```
X_train_prediction = model.predict(X_train)
training_data_accuracy = accuracy_score(X_train_prediction, Y_train)
```

In [37]:
```
training_data_accuracy
```

Out[37]:
```
0.9839772727272728
```

In [38]:
```
X_test_prediction = model.predict(X_test)
testing_data_accuracy = accuracy_score(X_test_prediction, Y_test)
```

In [39]:
```
testing_data_accuracy
```

Out[39]:
```
0.9525
```

In [40]:
```
data.Text[0]
```

Out[40]:
```
'klement gottwaldi surnukeha palsameeriti ning paigutati mausoleumi surnukeha oli aga liiga hilja ja oskamatult
palsameeritud ning hakkas ilmutama lagunemise tundemärke  aastal viidi ta surnukeha mausoleumist ära ja kremeer
iti zlíni linn kandis aastatel – nime gottwaldov ukrainas harkivi oblastis kandis zmiivi linn aastatel – nime g
otvald'
```

In [41]:
```
data.language[0]
```

Out[41]:
```
'Estonian'
```

In [42]:
```
testing = data.Text[0]
testing = [testing]
testing = vector.transform(testing)
prediction =  model.predict(testing)
print(prediction)
prediction = le.inverse_transform(prediction)
prediction
```

```
[4]
```

Out[42]:
```
array(['Estonian'], dtype=object)
```

In [43]:
```
user = input("Enter a text:")
user = [user]
user = vector.transform(user)
prediction =  model.predict(user)
print(prediction)
prediction = le.inverse_transform(prediction)
prediction
```

```
[3]
```

Out[43]:
```
array(['English'], dtype=object)
```

In [48]:
```
user = input("Enter a text:")
user = [user]
user = vector.transform(user)
prediction =  model.predict(user)
prediction = le.inverse_transform(prediction)
prediction
```

Out[48]:
```
array(['Thai'], dtype=object)
```

In [44]:
```
import pickle
pickle.dump(vector,open('vectorizer.pkl','wb'))
pickle.dump(model,open('model.pkl','wb'))
pickle.dump(le,open('labelencoder.pkl','wb'))
```

In [ ]: