IMPORTING LIBRARIES

```
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.naive_bayes import MultinomialNB
from sklearn.metrics import accuracy_score, mean_squared_error
from sklearn.preprocessing import LabelEncoder

df=pd.read_csv('language.csv')
df.head()
```

```
                                                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
```

SOME BASIC EDA

```
df.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
```

```
df.isnull().sum()
```

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

```
df.nunique()
```

```
Text        21859
language       22
dtype: int64
```

```
df['language'].value_counts()
```

```
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
```

```
df.sample(10)
```

```
                                                    Text    language
14735  in januari van  veroverde zengi de steden sarj...       Dutch
21370  aktör  aralık  tarihinde joaquin phoenixin kar...     Turkish
13804  由于当年消息欠公开，之后的情况以及最终的遇难人数未有官方正式公布。现在的媒体报道为、、、 人...     Chinese
20124  ا على الأمريكية أويل أمين شركة حصلت سبتمبر في...      Arabic
14165  al-andalus atau semenanjung iberia spanyol dan...  Indonesian
9420   beberapa waktu setelah tahun  peneliti kitab s...  Indonesian
16391  burada görev yaparken anadoluda celali isyanl...     Turkish
13418  jason barnes en mark tullo maakten een goede r...       Dutch
14330  el municipio de homer se encuentra ubicado en ...     Spanish
10099  em  john consegue contrato com a fantasy recor...   Portugese
```

```
df[df['language']=='Hindi'].sample(5)
```

```
                                                    Text language
2450   छह टीमों को टूर्नामेंट में भाग लेंगे बांग्लादे...     Hindi
13641  राजकुमारी का मन लगाने के लिए सखी-सहोलियाँ थीं।...     Hindi
6252   मनुष्य का जीवनकाल अत्यन्त कम है और् ज्ञान का व...     Hindi
10376  घास शाकीय पौधों या शिंबी पादपों को काटने के बा...     Hindi
8165   अंग्रेज़ी में samely  कोई क्रियाविशेषण नहीं है।...     Hindi
```

Data Preprocessing

1. Count vectorizer----> BOW

```
x=np.array(df['Text'])
y=np.array(df['language'])
```

```
cv=CountVectorizer()

X=cv.fit_transform(x)

print(X[0])
```

```
  (0, 57772)    1
  (0, 43363)    1
  (0, 104967)   3
  (0, 80287)    1
  (0, 75304)    2
  (0, 80056)    1
  (0, 67653)    1
  (0, 77619)    1
  (0, 2193)     1
  (0, 63122)    1
  (0, 47020)    1
  (0, 53103)    2
  (0, 79323)    1
  (0, 80288)    1
  (0, 45293)    1
  (0, 49445)    1
  (0, 60954)    1
  (0, 112024)   1
  (0, 136) 1
  (0, 117124)   1
  (0, 106285)   1
  (0, 67654)    1
  (0, 122429)   1
  (0, 59244)    1
  (0, 122097)   1
  (0, 63450)    2
  (0, 55264)    2
  (0, 153) 2
  (0, 75247)    2
  (0, 43364)    1
  (0, 113245)   1
  (0, 45787)    1
  (0, 76696)    1
  (0, 122098)   1
  (0, 43365)    1
```

```
x[0]
```

```
'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 kremeeriti zlíni linn kandis aastatel – nime gottwaldov
ukrainas harkivi oblastis kandis zmiivi linn aastatel – nime gotvald'
```

```
print(y)
```

```
['Estonian' 'Swedish' 'Thai' ... 'Spanish' 'Chinese' 'Romanian']
```

Fiting the Count Vectorizer Model

```
cv=CountVectorizer()
X=cv.fit_transform(x)

X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.3,rando
m_state=42)

model1=MultinomialNB()

model1.fit(X_train,y_train)
model1.score(X_test,y_test)
```

```
0.9528787878787879
```

```
user=input("Enter a text")
data=cv.transform([user]).toarray()
output=model1.predict(data)
print(output)
```

```
Enter a text میری طرف دیکھو

['Urdu']
```

2. TF-IDF

```
tfidf=TfidfVectorizer()

X=tfidf.fit_transform(x)

print(X[0])
```

```
  (0, 43365)    0.15245962403688543
  (0, 122098)   0.15245962403688543
  (0, 76696)    0.13890427169403846
  (0, 45787)    0.15245962403688543
  (0, 113245)   0.13890427169403846
  (0, 43364)    0.15245962403688543
  (0, 75247)    0.22902894148770514
  (0, 153) 0.2155525828166711
  (0, 55264)    0.2678531884721598
  (0, 63450)    0.24339387890154524
  (0, 122097)   0.15245962403688543
  (0, 59244)    0.15245962403688543
  (0, 122429)   0.11632821567894924
  (0, 67654)    0.15245962403688543
  (0, 106285)   0.0828549222249433
  (0, 117124)   0.1339265942360799
  (0, 136) 0.08509082541842236
  (0, 112024)   0.15245962403688543
  (0, 60954)    0.14646128506875586
  (0, 49445)    0.15245962403688543
```

```
  (0, 45293)    0.12265170453053793
  (0, 80288)    0.15245962403688543
  (0, 79323)    0.15245962403688543
  (0, 53103)    0.13228208686853668
  (0, 47020)    0.14646128506875586
  (0, 63122)    0.1339265942360799
  (0, 2193)     0.10029161912723429
  (0, 77619)    0.08182087878336174
  (0, 67653)    0.15245962403688543
  (0, 80056)    0.14646128506875586
  (0, 75304)    0.16625026948941635
  (0, 80287)    0.15245962403688543
  (0, 104967)   0.42661618752454344
  (0, 43363)    0.15245962403688543
  (0, 57772)    0.13890427169403846
```

```
X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.3,rando
m_state=42)

model2=MultinomialNB()

model2.fit(X_train,y_train)
model2.score(X_test,y_test)
```

```
0.9571212121212122
```

```
user=input("Enter a text")
data=cv.transform([user]).toarray()
output=model2.predict(data)
print(output)
```

```
Enter a text O primeiro andar deste prédio está vazio

['Portugese']
```

n_grams

```
cv=CountVectorizer(ngram_range=(1,2))
X=cv.fit_transform(x)

X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.3,rando
m_state=42)

model3=MultinomialNB()

model3.fit(X_train,y_train)
model3.score(X_test,y_test)
```

```
0.953939393939394
```

Conclusion

Applying BOW technique we get an accuracy of 95.28%

Applying Tf-Idf technique we get an accuracy of 95.7%

Applying n_grams technique we get an accuracy of 95.3%

Thank you