IMPORTING ALL THE NECESSARY LIBRARIES

```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
import re
import warnings
warnings.simplefilter("ignore")
```

DATASET

```
df=pd.read_csv("/content/Language Detection.csv")
df
```

	Text	Language
0	Nature, in the broadest sense, is the natural...	English
1	"Nature" can refer to the phenomena of the phy...	English
2	The study of nature is a large, if not the onl...	English
3	Although humans are part of nature, human acti...	English
4	[1] The word nature is borrowed from the Old F...	English
...	...	...
10332	ನಿಮ್ಮ ತಪ್ಪು ಏನು ಬಂದಿದೆಯೆಂದರೆ ಆ ದಿನದಿಂದ ನಿಮಗೆ ಒ...	Kannada
10333	ನಾರ್ಸಿಸಾ ತಾನು ಮೊದಲಿಗೆ ಹೆಣಗಾಡುತ್ತಿದ್ದ ಮಾರ್ಗಗಳನ್...	Kannada
10334	ಹೇಗೆ ' ನಾರ್ಸಿಸಿಸಮ್ ಈಗ ಮರಿಯನ್ ಅವರಿಗೆ ಸಂಭವಿಸಿದ ಎ...	Kannada
10335	ಅವಳು ಈಗ ಹೆಚ್ಚು ಚಿನ್ನದ ಬ್ರೆಡ್ ಬಯಸುವುದಿಲ್ಲ ಎಂದು ...	Kannada
10336	ಟೆರ್ರಿ ನೀವು ನಿಜವಾಗಿಯೂ ಆ ದೇವದೂತನಂತೆ ಸ್ವಲ್ಪ ಕಾಣು...	Kannada

10337 rows × 2 columns

DISPLAYING THE VALUE COUNT OF THE LANGUAGES MENTIONED IN THE DATASET

```
df["Language"].value_counts()
```

```
English       1385
French        1014
Spanish        819
Portugeese     739
Italian        698
Russian        692
Sweedish       676
Malayalam      594
Dutch          546
Arabic         536
Turkish        474
German         470
Tamil          469
Danish         428
Kannada        369
Greek          365
Hindi           63
Name: Language, dtype: int64
```

SPLITTING TARGET AND INDEPENDENT VALUES

```
X=df["Text"]
X
```

```
0         Nature, in the broadest sense, is the natural...
1        "Nature" can refer to the phenomena of the phy...
2        The study of nature is a large, if not the onl...
3        Although humans are part of nature, human acti...
4        [1] The word nature is borrowed from the Old F...
                              ...
10332    ನಿಮ್ಮ ತಪ್ಪು ಏನು ಬಂದಿದೆಯೆಂದರೆ ಆ ದಿನದಿಂದ ನಿಮಗೆ ಒ...
10333    ನಾರ್ಸಿಸಾ ತಾನು ಮೊದಲಿಗೆ ಹೆಣಗಾಡುತ್ತಿದ್ದ ಮಾರ್ಗಗಳನ್...
10334    ಹೇಗೆ ' ನಾರ್ಸಿಸಿಸಮ್ ಈಗ ಮರಿಯನ್ ಅವರಿಗೆ ಸಂಭವಿಸಿದ ಎ...
10335    ಅವಳು ಈಗ ಹೆಚ್ಚು ಚಿನ್ನದ ಬ್ರೆಡ್ ಬಯಸುವುದಿಲ್ಲ ಎಂದು ...
10336    ಟೆರ್ರಿ ನೀವು ನಿಜವಾಗಿಯೂ ಆ ದೇವದೂತನಂತೆ ಸ್ವಲ್ಪ ಕಾಣು...
Name: Text, Length: 10337, dtype: object
```

```
Y= df["Language"]
Y
```

```
0        English
1        English
2        English
3        English
4        English
          ...
10332    Kannada
10333    Kannada
10334    Kannada
10335    Kannada
10336    Kannada
Name: Language, Length: 10337, dtype: object
```

EXPLORATORY DATA ANALYTICS

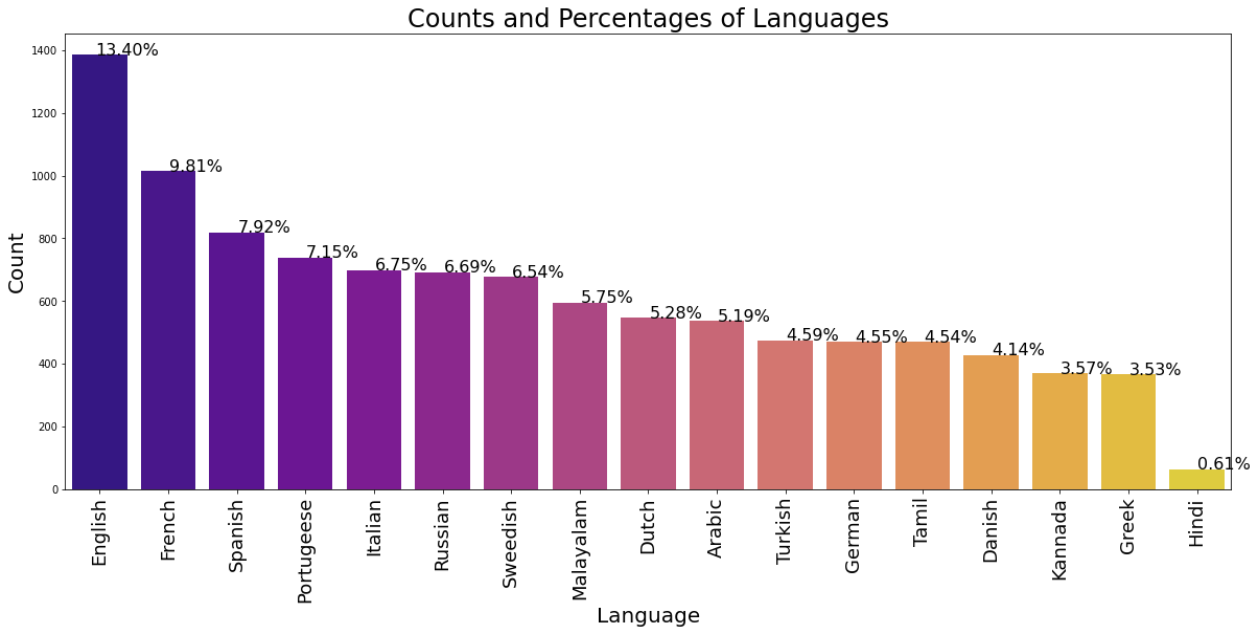
```
plt.figure(figsize=(20,8))

total=float(len(df['Language']))
ax=sns.countplot(x='Language',data= df,order= df['Language'].value_counts().index,palette='pl

for p in ax.patches:
    percentage='{:.2f}%'.format(100 * p.get_height()/total)
    x= p.get_x() + p.get_width()
    y= p.get_height()
    ax.annotate(percentage,(x,y),fontsize=16,ha='center')

plt.title('Counts and Percentages of Languages',fontsize=24)
plt.xlabel("Language",fontsize=20)
plt.ylabel("Count",fontsize=20)
```

```
plt.xticks(size=18,rotation=90)
plt.show()
```




```
language= df['Language'].value_counts().reset_index()
language
```

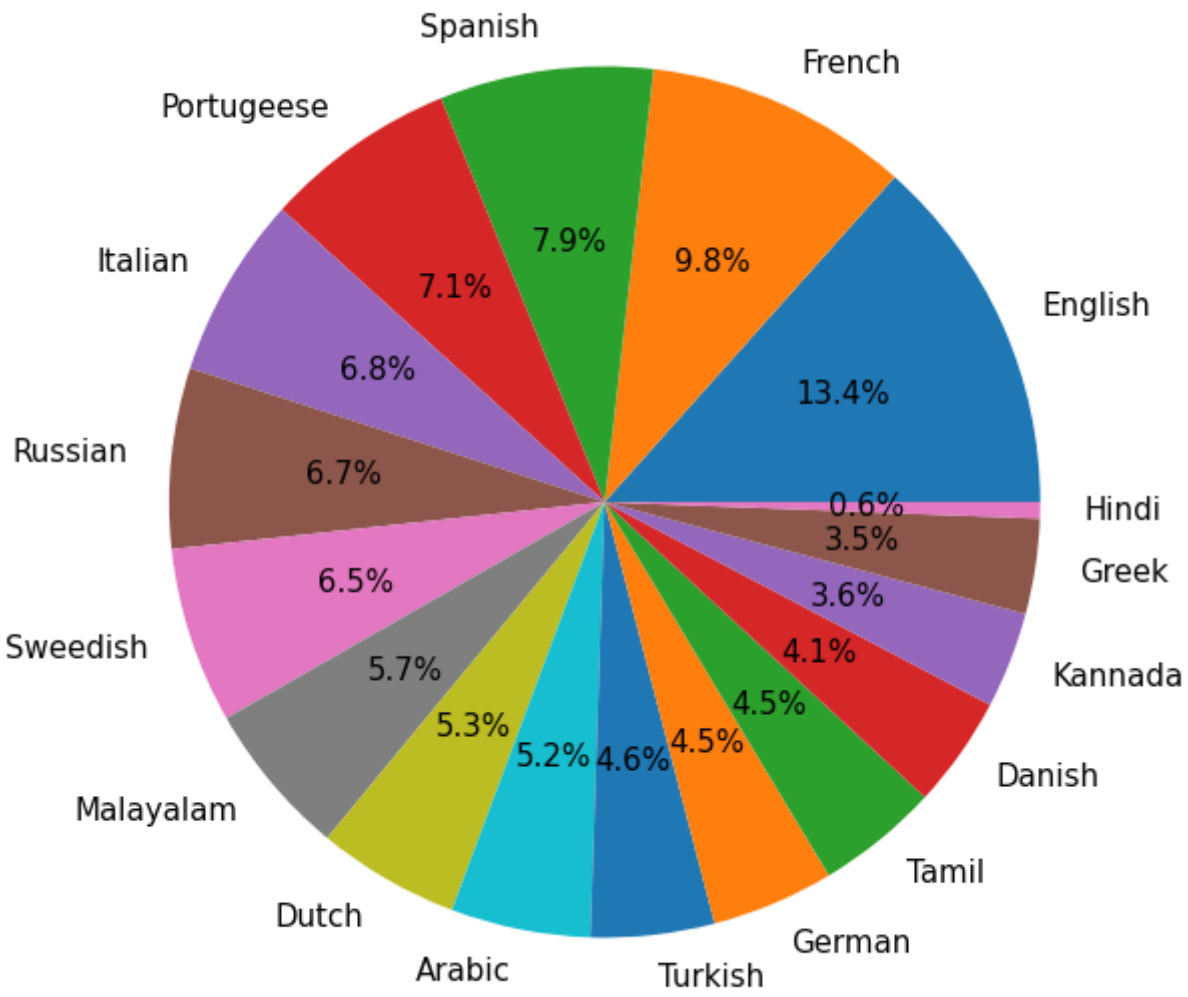
	index	Language
0	English	1385
1	French	1014
2	Spanish	819
3	Portugeese	739
4	Italian	698
5	Russian	692
6	Sweedish	676
7	Malayalam	594
8	Dutch	546
9	Arabic	536
10	Turkish	474
11	German	470
12	Tamil	469
13	Danish	428
14	Kannada	369
15	Greek	365
16	Hindi	63

```
plt.figure(figsize=(10,10))

labels=language['index']

plt.pie(language["Language"], labels= labels, autopct='%.1f%%', textprops={'fontsize': 15})
```

```
plt.show()
```



LABEL ENCODING

```
from sklearn.preprocessing import LabelEncoder
le=LabelEncoder()
Y=le.fit_transform(Y)
```

TEXT PREPROCESSING

Text processing involves creating a empty list and applying regular expression in order to remove the special characters in the text.

Appending the regularized text to the list

```
data_list=[]
for text in X:
    text = re.sub(r'[!@#$(),n"%^*?:;~`0-9]',' ',text)
    text = re.sub(r'[[]]',' ',text)
    text = text.lower()
    data_list.append(text)
```

USING COUNT VECTORIZER FUNCTION

```
from sklearn.feature_extraction.text import CountVectorizer
cv=CountVectorizer()
X=cv.fit_transform(data_list).toarray()
```

```
X.shape
```

```
(10337, 34937)
```

SPLITTING TRAINING AND TESTING DATA

```
from sklearn.model_selection import train_test_split
X_train,X_test,Y_train,Y_test=train_test_split(X,Y,test_size=0.2,random_state=2)
```

APPLYING MULTINOMIAL NAIVE BAYES ALGORITHM

```
from sklearn.naive_bayes import MultinomialNB
nb=MultinomialNB()
nb.fit(X_train,Y_train)
Y_pred=nb.predict(X_test)
```

PREDICTING ALL THE NECESSARY METRICS

```
from sklearn.metrics import accuracy_score,confusion_matrix,classification_report
ac=accuracy_score(Y_test,Y_pred)
print("ACCURACY SCORE",ac)
```

```
ACCURACY SCORE 0.9772727272727273
```

```
cr=classification_report(Y_test,Y_pred)
print(cr)
```

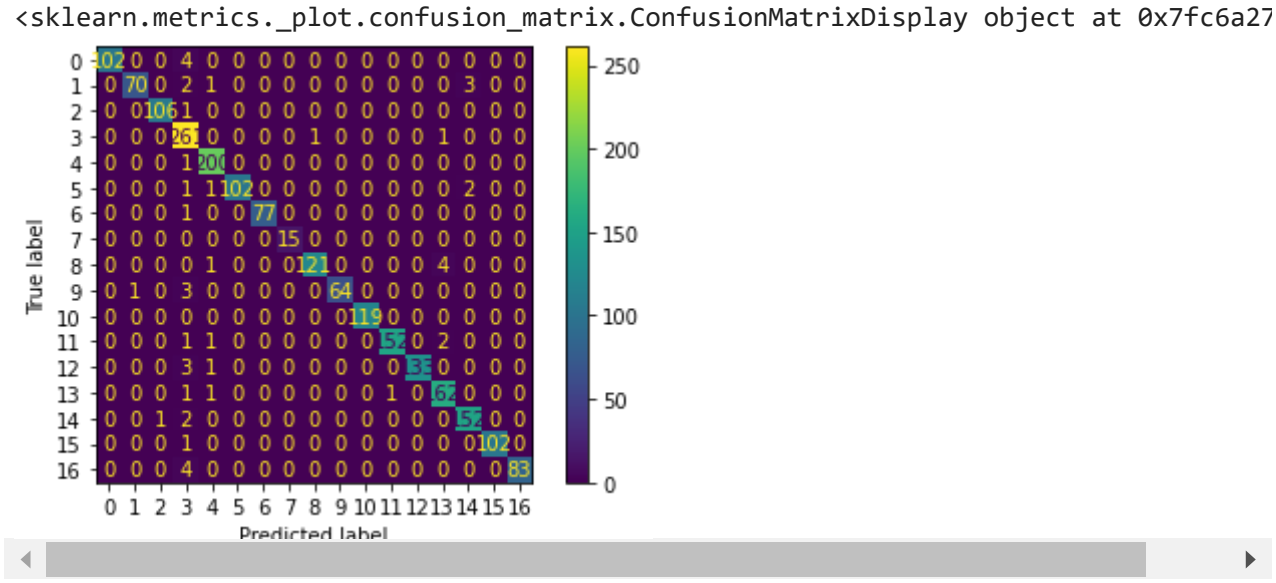
```
              precision    recall  f1-score   support

           0       1.00      0.96      0.98       106
           1       0.99      0.92      0.95        76
           2       0.99      0.99      0.99       107
           3       0.91      0.99      0.95       263
           4       0.97      1.00      0.98       201
           5       1.00      0.96      0.98       106
           6       1.00      0.99      0.99        78
           7       1.00      1.00      1.00        15
           8       0.99      0.96      0.98       126
           9       1.00      0.94      0.97        68
          10       1.00      1.00      1.00       119
          11       0.99      0.97      0.98       156
          12       1.00      0.97      0.99       137
          13       0.96      0.98      0.97       165
          14       0.97      0.98      0.97       155
          15       1.00      0.99      1.00       103
          16       1.00      0.95      0.98        87

    accuracy                           0.98      2068
   macro avg       0.99      0.97      0.98      2068
weighted avg       0.98      0.98      0.98      2068
```

```
from sklearn.metrics import ConfusionMatrixDisplay
print(ConfusionMatrixDisplay.from_predictions(Y_test,Y_pred))
```

```
<sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay object at 0x7fc6a27
```



RANDOM PREDICTION OF TEXT

```
def prediction(text):
    x= cv.transform([text]).toarray()
    lang= nb.predict(x)
    lang= le.inverse_transform(lang)
    print("Given sentence is in {} language.".format(lang[0]))
```

```
prediction("Your memory improves as you learn a language.In addition,since your brain will au
```

```
Given sentence is in English language.
```

```
prediction("ನೀವು ಭಾಷೆಯನ್ನು ಕಲಿತಂತೆ ನಿಮ್ಮ ಸ್ಮರಣೆಯು ಸುಧಾರಿಸುತ್ತದೆ.ಹೆಚ್ಚುವರಿಯಾಗಿ, ನಿಮ್ಮ ಮೆದುಳು ಸ್ವಯಂಚಾಲಿತವಾಗಿ ಅ
```

```
Given sentence is in Kannada language.
```

```
prediction("L'apprentissage d'une langue améliore la mémoire")
```

```
Given sentence is in French language.
```

✓ 0s completed at 4:45 PM