# Experiment – 08

## Understand NLP Basics

1. **Aim:**
   To understand the basics of Natural Language Processing and implement a simple application.

2. **Objectives:**
   - Understand the concept of Natural Language Processing.
   - Implement a simple application using a programming language like Python.

3. **Brief Theory:**
   NLP, meaning Natural Language Processing, is a branch of artificial intelligence (AI) that focuses on the interaction between computers and humans using human language. Its primary objective is to empower computers to comprehend, interpret, and produce human language effectively.
   - Tokenization
     Tokenization breaks down text into smaller units, typically words or subwords. These smaller units are called tokens. Tokenization is the first step in most NLP tasks. It's essential because computers can't understand raw text; they need structured data.
   - Part of Speech Tagging
     Part-of-speech tagging labels each word in a sentence with its corresponding part of speech (e.g., noun, verb, adjective, etc.).
   - Stemming
     It involves cutting off prefixes or suffixes of words to derive their root form. While stemming is faster and simpler, it may only sometimes produce valid words.
   - Lemmatization
     It involves reducing words to their base or dictionary form (lemma), considering the word's context. Lemmatization usually requires a dictionary lookup to determine the lemma of a word, making it more accurate but slower than stemming.
   - Stop Word Removal
     Stop words are commonly used in a language without significant meaning and are often filtered out during text preprocessing. Examples of stop words include "the," "is," "and," "are," etc.

4. **Dataset Description:**
   The dataset was downloaded from Kaggle. The dataset used is Language Detection.csv. It's a small language detection dataset. This dataset consists of text details for 17 different languages. The languages are English, Hindi, Malayalam, Tamil, Kannada, French, Spanish, Portuguese, Italian, Russian, Swedish, Dutch, Arabic, Turkish, German, Danish, and Greek.

5. **<u>Implementation of application (Language Detection Model):</u>**

   **Python code:**

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

#Read the dataset file as data
data = pd.read_csv('/content/Language Detection.csv',encoding='latin-1')

#Read the headings in the dataset
data.head()

#Consider only two needed headings
df = data[['Text','Language']]
df.head()

#Get to know the number of words of each language
df['Language'].value_counts()

#Separate the dependent and independent variables
X = df['Text']
y = df['Language']

#Label encoding of the data
from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()
y = le.fit_transform(y)

#Text reprocessing
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)

#Text to Numerical form
from sklearn.feature_extraction.text import CountVectorizer
cv = CountVectorizer()
X = cv.fit_transform(data_list).toarray()

X.shape

#Training the model
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test = train_test_split(X,y,test_size=0.20)

#Model prediction
```

```
from sklearn.naive_bayes import MultinomialNB
model = MultinomialNB()
model.fit(x_train,y_train)

#Prediction
y_pred = model.predict(x_test)

#Model evaluation
from sklearn.metrics import accuracy_score,confusion_matrix, classification_report
ac = accuracy_score(y_test,y_pred)
cm = confusion_matrix(y_test,y_pred)
cr = classification_report(y_test,y_pred)

#Print the above metrics
print("Accuracy is :",ac)
print("Classification report is :",cr)
#Visualising the confusion matrix
plt.figure(figsize=(10,6))
sns.heatmap(cm,annot=True)
plt.show

#Testing the model
def predict(text):
  x = cv.transform([text]).toarray() #converting text to bag of words model (vector)
  lang = model.predict(x) #predicting the language
  lang = le.inverse_transform(lang) #finding the language corresponding the
predicted value
  print("The langauge is in",lang[0]) #printing the language
```

6. **Result:**

```
#English
predict("Niti is a good girl")

The langauge is in English

#French
predict("Je suis beau")

The langauge is in French

Accuracy is : 0.9850096711798839
```

7. **Conclusion:**

This experiment provided a foundational understanding of Natural Language Processing (NLP) by implementing a simple language detection model. Through the preprocessing steps of tokenization, part-of-speech tagging, stemming, lemmatization, and stop-word removal, the dataset was prepared for analysis. These techniques highlight the importance of structuring data in a way that makes text understandable for machine learning models. By applying a Naive Bayes classifier, the model achieved an impressive accuracy of 98.5%, demonstrating the effectiveness of this algorithm in language classification tasks. This project illustrates how NLP transforms unstructured human language into structured data, enabling computational processing. The successful prediction of languages in sample texts reflects the potential of NLP applications to bridge language barriers and enhance human-computer interaction.

8. **Link to the code uploaded on:** https://github.com/Niti0209/Language-Detection-using-NLP

9. **References:**

- https://www.simplilearn.com/tutorials/artificial-intelligence-tutorial/what-is-natural-language-processing-nlp
- https://www.youtube.com/watch?v=6I-Alfkr5K4&t=223s
- https://www.youtube.com/watch?v=I8-122otJl0
- https://www.kaggle.com/datasets/basilb2s/language-detection/data