13. Natural Language Processing (NLP) - Text Processing

Aim: To implement text processing using NLP library

Algorithm:

(1) Import the necessary libraries

(2) Import nltk ~~lib~~, string and regular expression libraries.

(3) We need to convert the all the texts into lowercase

(4) Next, we have to remove the numbers for the easy processing of textual data.

(5) Now, we need to integer numbers into words by importing inflect library

(6) The next step is to remove punctuations and multiespaces

(7) We, then, remove the frequently occuring stopwords. NLTK library has a set of stopwords and we can use these words to remove stopwords from our text and return a list of word tokens

(8) Stemming is performed to get the root form of words.

(9) Last step is to perform lemmatization. This step also converts word into its root but also ensures that word belongs to the language.

SRM INSTITUTE OF SCIENCE & TECHNOLOGY

14. Implementation of NLP programs

Aim: To discriminate between ham/spam messages automatically using UCI datasets.

Algorithm:

(1) Import important libraries such as pandas, numpy, nltk, seaborn, etc.

(2) Load the dataset and perform relabelling of the columns.

(3) sort the created dataframe

PROGRAM:

```
# Load libraries
from sklearn.ensemble import AdaBoostClassifier
from sklearn import datasets
# Import train_test_split function
from sklearn.model_selection import train_test_split
#Import scikit-learn metrics module for accuracy calculation
from sklearn import metrics
# Load data
iris = datasets.load_iris()
X = iris.data
y = iris.target
# Split dataset into training set and test set
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3) # 70% trainin

# Create adaboost classifer object
abc = AdaBoostClassifier(n_estimators=50,

learning_rate=1)
# Train Adaboost Classifer
model = abc.fit(X_train, y_train)
#Predict the response for test dataset

y_pred = model.predict(X_test)
# Model Accuracy, how often is the classifier correct?
print("Accuracy:",metrics.accuracy_score(y_test, y_pred))
```