Text Preprocessing using NLTK:

+ Code + Text

Aim: To preprocess the given text using NLTK

Description: NLTK is a leading platform for building Python programs to work with human language data. It provides easy-to-use interfaces to over 50 corpora and lexical resources such as WordNet, along with a suite of text processing libraries for classification, tokenization, stemming, tagging, parsing, and semantic reasoning, wrappers for industrial-strength NLP libraries, and an active discussion forum.

```
! pip install unidecode
```

```
import nltk
nltk.download('punkt')
```

```
[nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data]   Package punkt is already up-to-date!
True
```

```
!pip install -q -U --pre pycaret
```

```
import pandas as pd
import unidecode
import matplotlib.pyplot as plt
from collections import Counter
import re

from sklearn.model_selection import train_test_split

from pycaret.classification import *

from imblearn.over_sampling import SMOTE
```

```
from google.colab import drive
drive.mount('/content/drive')
```

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mount("/content/drive", force_remount=True).
```

```
import pandas as pd
path='/content/drive/MyDrive/NLP/cleaned.csv'
data = pd.read_csv(path)
```

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 23486 entries, 0 to 23485
Data columns (total 6 columns):
 #   Column                   Non-Null Count  Dtype 
---  ------                   --------------  ----- 
 0   Unnamed: 0               23486 non-null  int64 
 1   Title                    19676 non-null  object
 2   Review                   23486 non-null  object
 3   Rating                   23486 non-null  int64 
 4   Recommended IND          23486 non-null  int64 
 5   Positive Feedback Count  23486 non-null  int64 
dtypes: int64(4), object(2)
memory usage: 1.1+ MB
```

```
data.head()
```

	Unnamed: 0	Title	Review	Rating	Recommended IND	Positive Feedback Count
0	0	NaN	'absolutely wonderful silky sexy comfortable '	4	1	0
1	1	NaN	'love dress sooo pretty happened find store im...	5	1	4
2	2	Some major design flaws	' high hopes dress really wanted work initiall...	3	0	0
3	3	My favorite buy!	' love love love jumpsuit fun flirty fabulous ...	5	1	0

```
data.isna().any()
```

```
Unnamed: 0                False
Title                      True
Review                    False
Rating                    False
Recommended IND           False
Positive Feedback Count   False
dtype: bool
```

```
import re
import string
from nltk.tokenize import word_tokenize
```

```
import nltk
from nltk.corpus import stopwords
nltk.download('stopwords')
```

```
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
True
```

```
# Remove stop words
stoplist = stopwords.words('english')
stoplist = set(stoplist)
```

```
def preprocess_text(text):
    text=str(text)
    #formatted text
    text = text.replace('\\n', ' ').replace('\n', ' ').replace('\t',' ').replace('\\', ' ').replace('. com', '.com')
    # Remove URLs
    text = re.sub(r'http\S+', '', text)

    # Remove mentions and hashtags
    text = re.sub(r'@\w+|#\w+', '', text)

    # Remove punctuation and convert to lowercase
    text = text.translate(str.maketrans('', '', string.punctuation)).lower()
    # Remove extra whitespace
    text = re.sub('\s+', ' ', text).strip()
    # Removing all the occurrences of links that starts with https
    text = re.sub(r'http\S+', '', text)
    # Remove all the occurrences of text that ends with .com
    text = re.sub(r"\ [A-Za-z]*\.com", " ", text)
    text = re.sub(r'@\S+', '', text)
    text = re.sub(r'#\S+', '', text)
    text = unidecode.unidecode(text)
    text = text.lower()
    Pattern_alpha = re.compile(r"([A-Za-z])\1{1,}", re.DOTALL)
    # Limiting all the  repeatation to two characters.
    Formatted_text = Pattern_alpha.sub(r"\1\1", text)
    # Pattern matching for all the punctuations that can occur
    Pattern_Punct = re.compile(r'([.,/#!$%^&*?;:{}=_`~()+-])\1{1,}')
    # Limiting punctuations in previously formatted string to only one.
    Combined_Formatted = Pattern_Punct.sub(r'\1', Formatted_text)
    # The below statement is replacing repeatation of spaces that occur more than two times with that of one occurrence.
    Final_Formatted = re.sub(' {2,}',' ', Combined_Formatted)
    text = re.sub(r"[^a-zA-Z0-9:$-,%.?!]+", ' ',text)
    text = repr(text)
    # Text without stopwords
    No_StopWords = [word for word in word_tokenize(text) if word.lower() not in stoplist ]
    # Convert list of tokens_without_stopwords to String type.
    words_string = ' '.join(No_StopWords)
    return words_string
```

```
data['Review'] = data['Review'].apply(preprocess_text)
```

```
data=data.drop(['Title'],axis=1)
data.rename(columns={"Recommended IND":"label"},inplace=True)
```

```
data.to_csv('/content/drive/MyDrive/NLP/revpre.csv')
```

● ×