Import Necessary Liberaries

In [ ]:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings('ignore')
```

Import Dataset

In [ ]:

```
dataset = pd.read_csv('news_data.csv')
dataset.head()
```

Out[ ]:

	topic	link	domain	published_date	title	lang
0	SCIENCE	https://www.eurekalert.org/pub_releases/2020-0...	eurekalert.org	2020-08-06 13:59:45	closer look solar fuel potenti	en
1	SCIENCE	https://www.pulse.ng/news/world/an-irresistibl...	pulse.ng	2020-08-12 15:14:19	irresist scent make locust swarm studi find	en
2	SCIENCE	https://www.express.co.uk/news/science/1322607...	express.co.uk	2020-08-13 21:01:00	artifici intellig warn ai know us better know	en
3	SCIENCE	https://www.ndtv.com/world-news/glaciers-could...	ndtv.com	2020-08-03 22:18:26	glacier could sculpt mar valley studi	en
4	SCIENCE	https://www.thesun.ie/tech/5742187/perseid-met...	thesun.ie	2020-08-12 19:54:36	perseid meteor shower 2020 time see huge brigh...	en

In [ ]:

```
# Shape of Dataset
print("Number of Rows :-> ", dataset.shape[0])
print("Number of Columns :-> ", dataset.shape[1])
```

```
Number of Rows :->  15774
Number of Columns :->  6
```

In [ ]:

```
dataset.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 15774 entries, 0 to 15773
Data columns (total 6 columns):
 #   Column          Non-Null Count  Dtype 
---  ------          --------------  ----- 
 0   topic           15774 non-null  object
 1   link            15774 non-null  object
 2   domain          15774 non-null  object
 3   published_date  15774 non-null  object
 4   title           15774 non-null  object
 5   lang            15774 non-null  object
dtypes: object(6)
memory usage: 739.5+ KB
```

Text Preprocessing

- Lower Case
- Tokenization
- Removing Special Characters
- Removing Stop words and punctuation
- Stemming

In [ ]:

```
import nltk
from nltk.corpus import stopwords
from string import punctuation
from nltk.stem import PorterStemmer

ps = PorterStemmer()
STOPWORDS = set(stopwords.words('english'))
```

In [ ]:

```
def tranform_text(text):
    text = text.lower()     # Converting to Lower case

    word_arr = nltk.word_tokenize(text)      # Tokenizing
    correct = []

    for word in word_arr:
        if (word.isalnum()) and (word not in STOPWORDS) and (word not in punctuation):      # Removal of special char, stop words, punc
            correct.append(ps.stem(word))                                                     # Stemming

    return " ".join(correct)
```

In [ ]:

```
dataset['title'] = dataset['title'].apply(tranform_text)
```

In [ ]:

```
df = dataset[["topic","title"]]
df.head()
```

Out[ ]:

	topic	title
0	SCIENCE	closer look solar fuel potenti
1	SCIENCE	irresist scent make locust swarm studi find
2	SCIENCE	artifici intellig warn ai know us better know

	topic	title
3	SCIENCE	glacier could sculpt mar valley studi
4	SCIENCE	perseid meteor shower 2020 time see huge brigh...

Text to Vectors

In [ ]:

```
# Using TF-IDF for Vectorizing
from sklearn.feature_extraction.text import TfidfVectorizer
tfidf = TfidfVectorizer(max_features=8000)
```

In [ ]:

```
text = tfidf.fit_transform(df['title']).toarray()
text
```

Out[ ]:

```
array([[0., 0., 0., ..., 0., 0., 0.],
       [0., 0., 0., ..., 0., 0., 0.],
       [0., 0., 0., ..., 0., 0., 0.],
       ...,
       [0., 0., 0., ..., 0., 0., 0.],
       [0., 0., 0., ..., 0., 0., 0.],
       [0., 0., 0., ..., 0., 0., 0.]])
```

In [ ]:

```
print("Shape of Transformed Text :-> ",text.shape)
```

```
Shape of Transformed Text :->  (15774, 8000)
```