Text classification of news artical using NLP

In [1]:
```
import pandas as pd
import numpy as np
import re
import nltk
```

In [2]:
```
from nltk.corpus import stopwords
from sklearn.feature_extraction.text import CountVectorizer, TfidfTransformer
from sklearn.naive_bayes import MultinomialNB
from sklearn.metrics import classification_report, confusion_matrix
```

In [3]:
```
# Load the data
df = pd.read_csv('BBC News.csv', encoding='utf-8')
```

In [6]:
```
# Preprocess the text data
corpus = []
for i in range(len(df)):
    article = re.sub('[^a-zA-Z]', ' ', df['Text'][i])
    article = article.lower()
    article = article.split()
    article = [word for word in article if word not in set(stopwords.words('english'))]
    article = ' '.join(article)
    corpus.append(article)
```

In [8]:
```
# Convert the text data to numerical features
cv = CountVectorizer()
X = cv.fit_transform(corpus)
tfidf = TfidfTransformer()
X = tfidf.fit_transform(X)
```

In [9]:
```
# Split the data into training and testing sets
y = df['Category']
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)
```

In [10]:
```
# Train the Naive Bayes model
nb = MultinomialNB()
nb.fit(X_train, y_train)
```

Out[10]:
```
MultinomialNB()
```

In [11]:
```
# Make predictions on the test set
y_pred = nb.predict(X_test)
```

In [12]:
```
# Evaluate the model's performance
print(confusion_matrix(y_test, y_pred))
print(classification_report(y_test, y_pred))
```

```
[[104   0   2   0   2]
 [  1  76   1   1   0]
 [  2   0  83   1   0]
 [  0   0   0 101   0]
 [  1   0   3   1  68]]
               precision    recall  f1-score   support

     business       0.96      0.96      0.96       108
entertainment       1.00      0.96      0.98        79
     politics       0.93      0.97      0.95        86
        sport       0.97      1.00      0.99       101
         tech       0.97      0.93      0.95        73

     accuracy                           0.97       447
    macro avg       0.97      0.96      0.97       447
 weighted avg       0.97      0.97      0.97       447
```

The text classification model has achieved an overall accuracy of 97%, indicating that it is very effective in predicting the category of a given news article.