Classifying newsgroup topics with Support Vector Machines

Code used –

Importing libraries

Libraries used: -

1. Sci-kit learn (machine learning library)
2. Natural Language Toolkit (natural language processing library) also known as NLTK

```
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.datasets import fetch_20newsgroups
from nltk.corpus import names
from nltk.stem import WordNetLemmatizer
```

Downloading required nltk library packages using the command - nltk.download ('package_name')

```
import nltk
nltk.download('names')
nltk.download('stopwords')
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
nltk.download('wordnet')
nltk.download('omw-1.4')
```

Lemmatization is the process of grouping together different inflected forms of words having the same root or lemma for better NLP analysis and operations. The lemmatization algorithm removes affixes from the inflected words to convert them into the base words (lemma form). For example, “running” and “runs” are converted to its lemma form “run”.

Wordnet is a popular lexical database of the English language that is used by NLTK internally. WordNetLemmatizer is an NLTK lemmatizer built using the Wordnet database.

```
all_names=set(names.words())
```

```
lemmatizer=WordNetLemmatizer()


def is_letter_only(word):
   return word.isalpha()


from nltk.corpus import stopwords
stop_words= stopwords.words('english')


def clean_text(docs):
   docs_cleaned=[]
   for doc in docs:
      doc=doc.lower()
      doc_cleaned=' '.join(lemmatizer.lemmatize(word) for word in doc.split()
                     if is_letter_only(word) and word not in all_names and word not in stop_words)
      docs_cleaned.append(doc_cleaned)
   return docs_cleaned


#Fetching dataset 20newsgroups


fetch_20newsgroups
```

```
cleaned_test = clean_text(data_test.data)
label_test = data_test.target
```

Binary classification is a supervised machine learning technique where the goal is to predict categorical class labels which are discrete and unordered such as Pass/Fail, Positive/Negative, etc.

```
# Binary Classification
categories= ['comp.graphics','sci.space']
data_train= fetch_20newsgroups(subset='train', categories= categories, random_state=42)
data_test= fetch_20newsgroups(subset='test', categories=categories, random_state=42)
cleaned_train= clean_text(data_train.data)
label_train= data_train.target
```

Tfidfvectorizer convert a collection of raw documents to a matrix of TF-IDF features. And it is equivalent to CountVectorizer followed by TfidfTransformer.

```
from collections import Counter
Counter(label_train)
tfidf_vectorizer= TfidfVectorizer(stop_words='english',max_features=None)
term_docs_train= tfidf_vectorizer.fit_transform(cleaned_train)
term_docs_test=tfidf_vectorizer.transform(cleaned_test)
```

Importing Support Vector Machine from sklearn library using the command - from sklearn.svm import SVC (using linear svc here)

```
from sklearn.svm import SVC
svm=SVC(kernel='linear', random_state=42)
svm.fit(term_docs_train, label_train)
accuracy= svm.score(term_docs_test, label_test)
print("The accuracy of binary classification is : {0:.1f}%".format(accuracy*100))
```

- ***Our model's accuracy of binary classification is 96.4%***

Multiclass classification is a popular problem in supervised machine learning.

Problem – Given a dataset of m training examples, each of which contains information in the form of various features and a label. Each label corresponds to a class, to which the training example belongs.

```
# Multiclass classification
categories=[
   'alt.atheism',
   'talk.religion.misc',
   'comp.graphics',
   'sci.space',
   'rec.sport.hockey'
]
```

Model for Multiclass classification

```
data_train= fetch_20newsgroups(subset='train', categories= categories, random_state=42)
data_test= fetch_20newsgroups(subset='test', categories=categories, random_state=42)
cleaned_train= clean_text(data_train.data)
label_train= data_train.target
cleaned_test= clean_text(data_test.data)
label_test= data_test.target
tfidf_vectorizer= TfidfVectorizer(stop_words='english',max_features=None)
term_docs_train= tfidf_vectorizer.fit_transform(cleaned_train)
term_docs_test=tfidf_vectorizer.transform(cleaned_test)

svm=SVC(kernel='linear', random_state=42)
svm.fit(term_docs_train, label_train)
accuracy= svm.score(term_docs_test, label_test)
print("The accuracy of 5-class classification is : {0:.1f}%".format(accuracy*100))
```

- ***Our model's accuracy of Multiclass classification is 88.6%***

```
from sklearn.metrics import classification_report
```

```
import matplotlib.pyplot as plt
prediction= svm.predict(term_docs_test)
report= classification_report(label_test, prediction)
print(report)


plt.plot(prediction)
plt.show()
```

- **Classification Report:**

Class	Precision	Recall	F1-Score	Support
0	*0.79*	*0.77*	*0.78*	*319*
1	*0.92*	*0.96*	*0.94*	*389*
2	*0.98*	*0.96*	*0.97*	*399*
3	*0.93*	*0.94*	*0.93*	*394*
4	*0.74*	*0.73*	*0.73*	*251*

Accuracy	*0.89*	*0.89*	*0.89*	*1752*
Macro avg	*0.87*	*0.87*	*0.87*	*1752*
Weighted avg	*0.89*	*0.89*	*0.89*	*1752*