Analysis of 20 Newsgroup Dataset

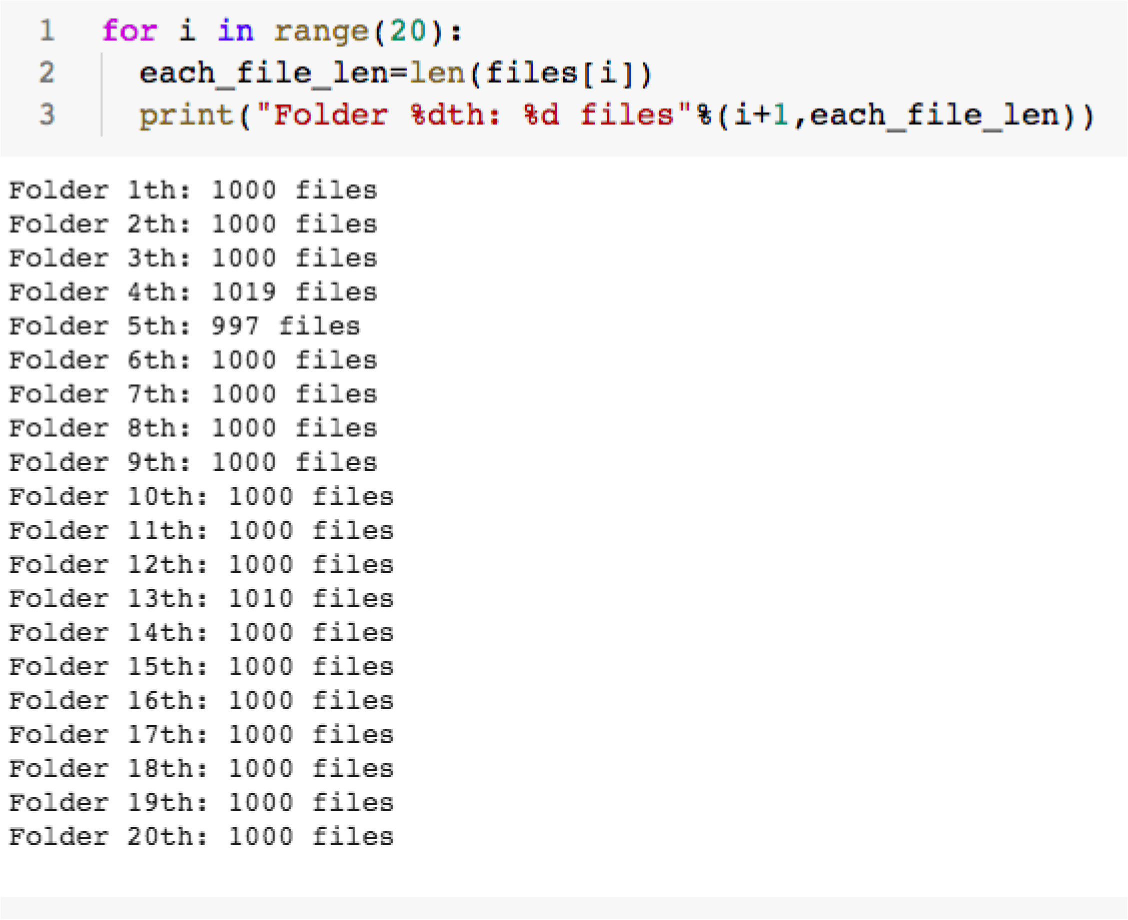
Gayatri Basude
19223031
IV Sem, MCA 2019-2022

Introduction

1. This data set consists of 20016 messages taken from 20 newsgroups.
2. Approx 1000 Usenet articles were taken from each of the following 20 newsgroups.
3. The articles are typical postings and thus have headers including subject lines, signature files, and quoted portions of other articles.
4. Each newsgroup is stored in a subdirectory, with each article stored as a separate file.
5. Followings are the 20 name of these newsgroups:

```
['alt.atheism',
 'comp.graphics',
 'comp.os.ms-windows.misc',
 'comp.sys.ibm.pc.hardware',
 'comp.sys.mac.hardware',
 'comp.windows.x',
 'misc.forsale',
 'rec.autos',
 'rec.motorcycles',
 'rec.sport.baseball',
 'rec.sport.hockey',
 'sci.crypt',
 'sci.electronics',
 'sci.med',
 'sci.space',
 'soc.religion.christian',
 'talk.politics.guns',
 'talk.politics.mideast',
 'talk.politics.misc',
 'talk.religion.misc']
```

6. Dataset Link: https://archive.ics.uci.edu/ml/datasets/Twenty+Newsgroups



```python
for i in range(20):
    each_file_len=len(files[i])
    print("Folder %dth: %d files"%(i+1,each_file_len))
```

```
Folder 1th: 1000 files
Folder 2th: 1000 files
Folder 3th: 1000 files
Folder 4th: 1019 files
Folder 5th: 997 files
Folder 6th: 1000 files
Folder 7th: 1000 files
Folder 8th: 1000 files
Folder 9th: 1000 files
Folder 10th: 1000 files
Folder 11th: 1000 files
Folder 12th: 1000 files
Folder 13th: 1010 files
Folder 14th: 1000 files
Folder 15th: 1000 files
Folder 16th: 1000 files
Folder 17th: 1000 files
Folder 18th: 1000 files
Folder 19th: 1000 files
Folder 20th: 1000 files
```

Problem Statement

Here our task is to make a model which identifies or predicts the name of the NewsGroup based on its text and contents.

Strategies Used:

1. Dataset:
    - 1.1. 20016 files taken from 20 Newsgroup
    - 1.2. Output: Name of the Newsgroup (1 to 20)

1.3. Null Values: 0

2. Problem type:

2.1. Supervised

2.2. Categorical

2.3. Multi-class Classification

3. Splitted the dataset into two:

3.1. train(75%) - instances

3.2. test(25%)- instances

4. Conversion of strings (content of file) into vectors (numeric form) using the following process, as the model can process only numerical data.

4.1. Clearing:

4.1.1. Removal of punctuations, such as

```
'!()-[]{};:'"\,<>./?@#$%^&*_~'
```

4.1.2. Removal of white spaces

4.2. Tokenization: Splitting sequence of strings into words.

4.3. Removing Stopwords: Stopwords are commonly used words in any language.

4.4. Casing: Making the case of all words same (here lowering).

4.5. Stemming: The process of reducing a word to its word stem that affixes to suffixes.

4.6. Vectorization: Conversion of word into number

5. The process described above can be implemented using `CountVectorizer` or `TfidfVectorizer`

6. CountVectorizer: We **only count the number of times a word appears in the document** which results in biasing in favour of most frequent words. This ends up in ignoring rare words which could have helped us in processing our data more efficiently.

7. TfidfVectorizer: We consider **overall document weightage** of a word. It helps us in dealing with most frequent words. Using it we can penalize them. TfidfVectorizer weights the word counts by a measure of how often they appear in the documents.

7.1. Formula:

TF= No. of repeat of words in sentence / no. of words in sentence

IDF= log(no. of sentences/ no. of sentences containing words)

Tf-IDF= TF*IDF

8. Trained Using Multinomial Classification

8.1. MultinomialNB implements the naive Bayes algorithm for multinomial distributed data, and is one of the two classic naive Bayes variants used in text classification (where the data are typically represented as word vector counts, although tf-idf vectors are also known to work well in practice).

8.2. The distribution is parametrized by vectors for each class , where is the number of features (in text classification, the size of the vocabulary) and is the probability of features appearing in a sample belonging to class .

9. Evaluation

9.1. Accuracy

9.2. Confusion Matrix

9.3. Precision and Recall

9.4. F1 Score

Code

https://github.com/gayatribasude/MachineLearning/blob/master/20Newsgroup.ipynb

Result and Conclusion

In this way we have created a suitable model for prediction of the name of the NewsGroup based on its text and contents.

MultinomialNB with	CountVectorizer	TfidfVectorizer
Accuracy	0.88	0.87
F1 Score	0.88	0.87

Reference

1. https://scikit-learn.org/stable/modules/naive_bayes.html#multinomial-naive-bayes
2. https://www.youtube.com/user/krishnaik06
3. https://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.TfidfVectorizer.html

4. https://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.CountVectorizer.html