Artificial Intelligence LAB 4

Document Classification

By Shubham Chawla
239571

1. Scope

Representation of text documents as vectors of word frequencies.
Train a classifier based on Machine Learning for recognizing documents that belong to one of the selected categories on the basis of the content of the documents.

2. Data Description

There are 20 newsgroup data set each consisting of around roughly 700 files.

ClassName	Number of Files	Class Number
alt.atheism	480	0
comp.graphics	584	1
comp.os.ms-windows.misc	591	2
comp.sys.ibm.pc.hardware	590	3
comp.sys.mac.hardware	578	4
comp.windows.x	593	5
misc.forsale	585	6
rec.autos	594	7
rec.motorcycles	598	8
rec.sport.baseball	597	9
rec.sport.hockey	600	10
sci.crypt	595	11
sci.electronics	591	12
sci.med	594	13
sci.space	593	14
soc.religion.christian	599	15
talk.politics.guns	546	16
talk.politics.mideast	564	17
talk.politics.misc	465	18
talk.religion.misc	377	19

3. Category Selection

I selected 8 different categories having no similarities making the classification better.

```
"alt.atheism", "soc.religion.christian",
"comp.graphics", "sci.med","sci.space", "comp.os.ms-
windows.misc", "comp.sys.ibm.pc.hardware",
"comp.sys.mac.hardware"
```

4. Toolkit Selection

I have used Scikit-learn toolkit :

- CountVectorizer():
- TfidTransformer

In this project, Na ïve Bayes Classification model is built based on the 20 newsgroup dataset.
This model can be used to test any text classification. Any new document will be classified into any of these 20 categories. Model can be trained on any type of Dataset.

5. Feature Selection

- CountVectorizer()
  I am using a CountVectorizer component.
  It takes what we call is a Bag of words approach. Each message is separated into tokens and number of times each token occurs in a message is counted.
- Tf-id
  I am using tf-idf as the tokenization method.
  Term frequency is the weight representing how often a word occurs in a document. More it is higher is the tf value.
  Inverse frequency is another weight how common a word is occurring across documents.

Its basically related to stopwords. It is higher in a particular document but it decreases as we go through more than 1.
The pros of tf-idf includes that the inverse-document-frequency will alleviate the impact of some common meaningless words; the vector is normalized so the classification is based on cosine similarity and the length of the article will not affect topic classification. The cons are that it is still based on bag-of-words so the correlation of adjacent terms will not be taken into consideration.

6. Classifiers used.

- ### MultinomialNB for Na ïve Bayes

  To begin with, multinomial naive Bayes model may serve as a good baseline with the assumption that among features there is no correlation.

  1. Finding prior probabilities
  2. Conditional probability of the word occurring in the document.
  3. Finding the most probable category

- ### SGD Classifier for SVM (Support Vector Machines)

  SVM is known to be good at text classification based on bag-of-words because the text vector has high dimensional feature space, most features are relevant, and text vectors are sparse [1]. Since most text categorization problems are linearly separable, here a SVM model with linear kernel is implemented. Hinge loss and L-2 norm is applied.

7. Testing

Classifier	Accuracy(percent)	Time(sec)

Na ïve Bayes	87.38	5.27
SVM	95.4	6.73

8. Results

The biggest difference between the models you're building from a "features" point of view is that Naive Bayes treats them as independent, whereas SVM looks at the interactions between them to a certain degree, as long as you're using a non-linear kernel (Gaussian, rbf, poly etc.). So if you have interactions, and, given your problem, you most likely do, an SVM will be better at capturing those, hence better at the classification task you want.

SVM was more accurate but it takes more time.