Rhea Sharma **1001679252**

CSE 6363: Machine Learning
Project 2: Report

About the dataset:

The dataset being used is the 20 Newsgroup Dataset, which is available for download for free at https://www.cs.cmu.edu/afs/cs/project/theo-20/www/data/news20.html. This dataset is one of the best-known datasets used for document classification and text clustering.

This dataset consists of 20,000 messages, collected from 20 different netnews newsgroups, partitioned (nearly) evenly across 20 different newsgroups. 1000 messages from each of the twenty newsgroups were chosen at random and partitioned by newsgroup name.

The 20 different newsgroups that the documents are partitioned into are:

1) alt.atheism
2) talk.politics.guns
3) talk.politics.mideast
4) talk.politics.misc
5) talk.religion.misc
6) soc.religion.christian
7) comp.sys.ibm.pc.hardware
8) comp.graphics
9) comp.os.ms-windows.misc
10) comp.sys.mac.hardware
11) comp.windows.x
12) rec.autos
13) rec.motorcycles
14) rec.sport.baseball
15) rec.sport.hockey
16) sci.crypt
17) sci.electronics
18) sci.space
19) sci.med
20) misc.forsale

Method:

The algorithm that is applied to this is the **Naïve Bayes Classifier**.

Naïve Bayes Classifier is a linear classifier based on the Bayes Probability Theorem. Mathematically, Bayes Theorem states that:

$$\boxed{P(A \mid B) = \frac{P(B \mid A)P(A)}{P(B)}}$$

where A and B are events and $P(B) \neq 0$.

Naive Bayes is a classification algorithm for binary and multiclass classification problems. Often, probabilities are used for prediction and that is why Naïve Bayes Classifier is used for document classification.

Implementation:

This model has been implemented using Python 3, which categorizes the data into one of the twenty available newsgroups. The dataset has been subjected to pre-processing where I divided the news group data into two equal parts for training and testing each, after which the model has been trained for classification.

- **Pre-processing:** The file: **preprocessing.py** is the file that is splitting the dataset into training and testing data. The steps that need to be followed to run this program are as follows:
    - First, the dataset – 20 Newsgroup should be downloaded and unzipped.
    - Once the dataset has been unzipped, this file can be run.
    - When this file is run, it creates a folder named: "**train_data**".
    - The dataset is then randomly split into two equal parts – one part for training and one part for testing.
    - The training data is moved to the train_data folder, while the testing data remains.

The file to be executed for training and testing is: **main.py**.

- **Training:**
    - I have first taken the word count for all the classes and then created a dictionary of all the words with its frequency.
    - Next, I have removed the less frequent words from the dictionary for those words that have a frequency count less than 3.
    - The classes have then been trained by calculating the probabilities of each word per class.
    - If there is a word in the dictionary, which is not found in any of the training classes, the default count is set to 2 for that word, which is the lowest as the counts of other words is at least 3.

- **Testing:**
    - After the training is complete, the other half of data is used for testing.
    - For each word of a class, I have calculated the logarithmic value of the probability for the word present in the class. The summation of these values is considered. This is done for all classes and the results are stored in a dictionary.
    - The class having the highest probability for a file is considered for the test output, which is further compared with the actual class. If they match, the accuracy increases, hence decreasing the overall error and increasing the accuracy of the model.
    - The correct instances are then compared with the total instances which gives us the accuracy for each class.

- The average of all the class accuracies is the final accuracy for the developed Naive Bayes classifier.

Note:

Python libraries used in this model are:

- os
- time
- math
- random

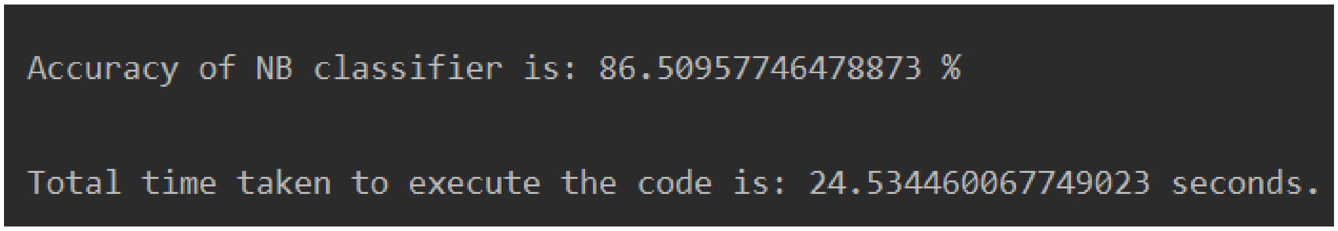
Results

The accuracy for each class results in the following:

Sl. No.	Class	Accuracy
1	alt.atheism	86.2%
2	talk.politics.guns	90.4%
3	talk.politics.mideast	91.8%
4	talk.politics.misc	73.0%
5	talk.religion.misc	47.199999999999996%
6	soc.religion.christian	98.59154929577466%
7	comp.sys.ibm.pc.hardware	84.39999999999999 %
8	comp.graphics	84.0%
9	comp.os.ms-windows.misc	80.4%
10	comp.sys.mac.hardware	90.0%
11	comp.windows.x	87.6%
12	rec.autos	89.4%
13	rec.motorcycles	93.2%
14	rec.sport.baseball	94.8%
15	rec.sport.hockey	97.2%
16	sci.crypt	94.6%
17	sci.electronics	86.0%
18	sci.space	92.0%
19	sci.med	91.60000000000001%
20	misc.forsale	77.8%

I have then taken the mean of the accuracies obtained for each class. This comes up to 86.5096%.

The **final accuracy** of the entire model is approximately: **86.5096%**.
The **time** taken to run this model is approximately: **24.535 seconds**.



```
Accuracy of NB classifier is: 86.50957746478873 %

Total time taken to execute the code is: 24.534460067749023 seconds.
```

References:

1) http://blog.yhat.com/posts/naive-bayes-in-python.html
2) http://machinelearningmastery.com/naive-bayes-classifier-scratch-python/
3) http://guidetodatamining.com/chapter7/
4) Stackoverflow
5) Wikipedia: https://en.wikipedia.org/wiki/Bayes%27_theorem
6) https://towardsdatascience.com/implementing-a-naive-bayes-classifier-for-text-categorization-in-five-steps-f9192cdd54c3
7) https://www.cs.cmu.edu/afs/cs/project/theo-20/www/data/news20.html