READ ME

Instructions to compile and run the code

- The code for the Multinomial Naïve Bayes Classifier is submitted in a jupyter notebook file in the source directory.
- To compile the code, open the file in jupyter notebook, then click on run to compile all the functions in the code.
- Then, you will be asked to specify the path of the train and test dataset. Here, enter the path to the 20news train and test dataset.
- After then, compile the remaining blocks, first all the file instances will be read into the memory sequentially, they will be vectorised and stop words and special characters containing tokens will be removed and the remainder shall reside nicely in a list. This is same for both train and test dataset and their class labels as well.
- After the vectors are generated, the running the next block shall train our classifier, the vocabulary, priors and conditional probabilities are computed at this time.
- The next step is to test the model. Running that block, for each test instance, prediction is obtained using the Argmax of class which had the max probability with it. And at last, the accuracy is displayed.
- Due to large number of instances, it may take a while, depending on the system for the Accuracy to be displayed.

Language

- Python, Jupyter Notebook to run and compile

Packages

- Numpy
- Random
- Math
- NLTK: Stopwords

Results and Analysis

This results are from a subset of train and test instances.

Run 1:

```
print("The 5 selected classes:\n", selected5Classes)
```

```
The 5 selected classes:
 ['soc.religion.christian', 'sci.electronics', 'comp.graphics', 'talk.religion.misc', 'comp.sys.ibm.pc.hardware']
```

```
print("Accuracy is: {:.3f}".format(accuracy*100))
```

```
Accuracy is: 82.196
```

Accuracy: 82.196

Classes:

1. soc.religion.christian
2. sci.electronics
3. comp.graphics
4. talk.religion.misc
5. comp.sys.ibm.pc.hardware

Run 2:

```
print("The 5 selected classes:\n", selected5Classes)
```

```
The 5 selected classes:
 ['talk.religion.misc', 'sci.med', 'comp.sys.mac.hardware', 'sci.crypt', 'rec.sport.baseball']
```

```
print("Accuracy is: {:.3f}".format(accuracy*100))
```

```
Accuracy is: 93.251
```

Accuracy: 93.251

Classes:

1. talk.religion.misc
2. sci.med
3. comp.sys.mac.hardware
4. sci.crypt
5. rec.sport.baseball

Run 3:

```
print("The 5 selected classes:\n", selected5Classes)
```

```
The 5 selected classes:
 ['comp.graphics', 'comp.os.ms-windows.misc', 'comp.sys.ibm.pc.hardware', 'comp.sys.mac.hardware', 'comp.windows.x']
```

```
print("Accuracy is: {:.3f}".format(accuracy*100))
```

```
Accuracy is: 74.353
```

Accuracy: 74.353

Classes:

1. comp.graphics
2. comp.os.ms-windows.misc
3. comp.sys.ibm.pc.hardware
4. comp.sys.mac.hardware
5. comp.windows.x

Run 4:

```
print("The 5 selected classes:\n", selected5Classes)
```

```
The 5 selected classes:
 ['comp.graphics', 'rec.motorcycles', 'sci.electronics', 'talk.politics.mideast', 'misc.forsale']
```

```
print("Accuracy is: {:.3f}".format(accuracy*100))
```

```
Accuracy is: 89.254
```

Accuracy: 89.254

Classes:

1. comp.graphics
2. rec.motorcycles
3. sci.electronics
4. talk.politics.mideast
5. misc.forsale

Observation:

When the classes chosen are from completely different category, the accuracy g oes up, and when the classes are related, see run 3, the accuracy falls down.

In the first case, it becomes easy for the classifier to differentiate between the c lasses, hence the higher accuracy, see run 2 and 4.

In the latter case, it is difficult for the classifier to differentiate as the files across classes are related.