# COEN 240 Final Project Report

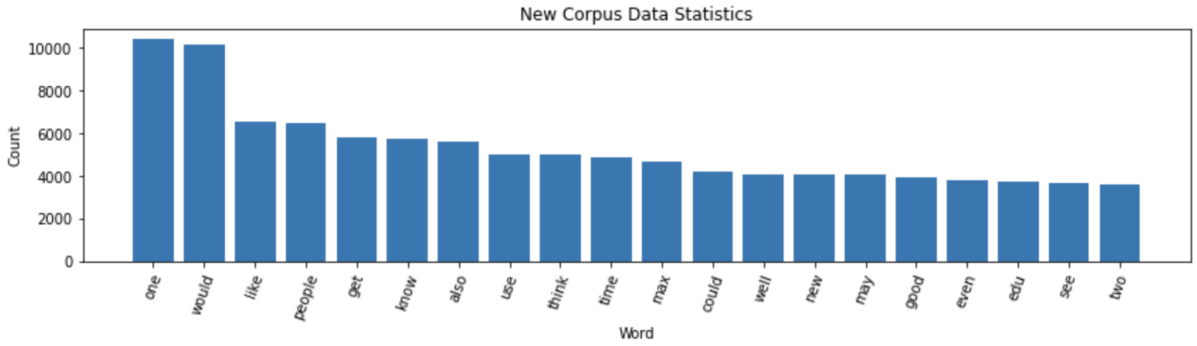
## About the Project:

The project takes the 20Newsgroup dataset, which contains 20,000 newsgroup documents partitioned between 20 newsgroups. We preprocess it and imagine the term- recurrence dispersion. Then we vectorize these archives utilizing BoW, TF-IDF, LDA, Word2Vec, and Doc2Vec models. Once the vector representation of the archives is obtained, we perform clustering using the K-Means Model. We mainly use the gensim module for training models and use sklearn module to perform clustering and supervised models.

### 1.Preprocessing

In the preprocessing steps we first converted capital letters to lowercase letters, then removed stopwords,and removed digits as well, guaranteeing word length is more than two. We then further utilized `spacy`, `gensim.parsing.preprocessing` libraries to further remove stopwords, new line characters and single quotes. We likewise eliminate headers, footers and statements from the records. To check our preprocessed information we utilize the bar chart to visualize the term frequency distribution in our documents. This helps us to check if we require any further preprocessing before we train the model.

| # of docs | # of words | # of unique words | min words | max words | mean words | std of words |
|---|---|---|---|---|---|---|
| 18846 | 1761662 | 95013 | 0 | 9300 | 138.735 | 392.667 |



## 2. Building Vocabulary and producing the BoW and TF-IDF

We use gensim and sklearn libraries to generate Vocab_v1 and Vocab_v2, and produce BoW and TF-IDF models with the same module.

## 3.Training LDA model with vocab_v1 and Visualizing

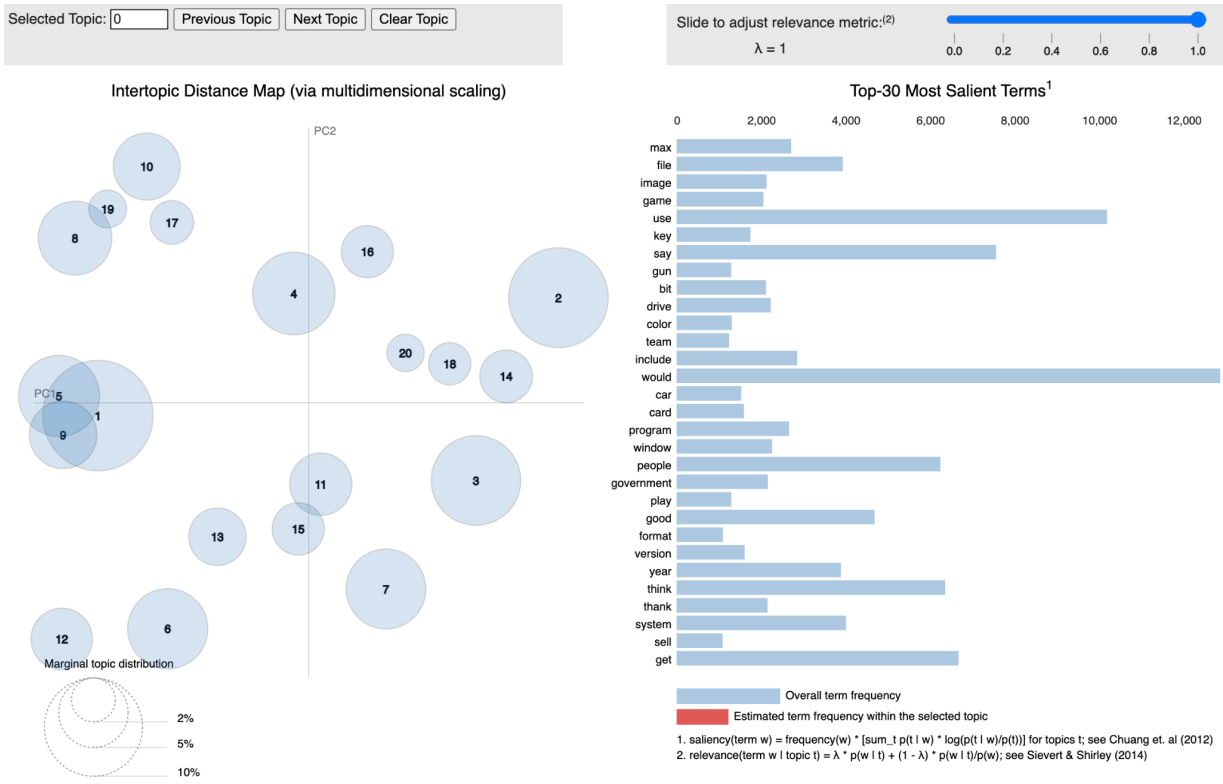
We use gensim to learn the LDA model, and we visualize the topic distribution by using word cloud and  plYDAvis.





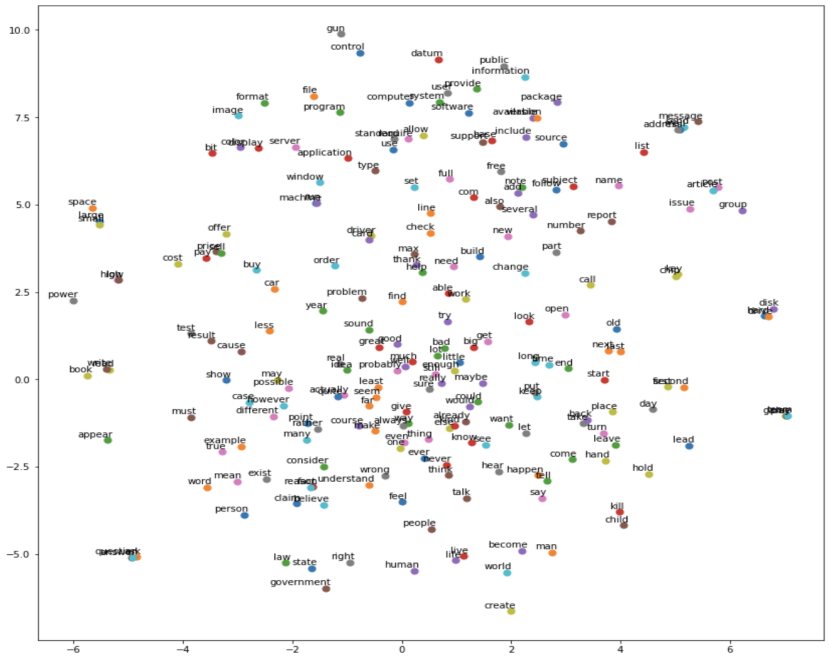




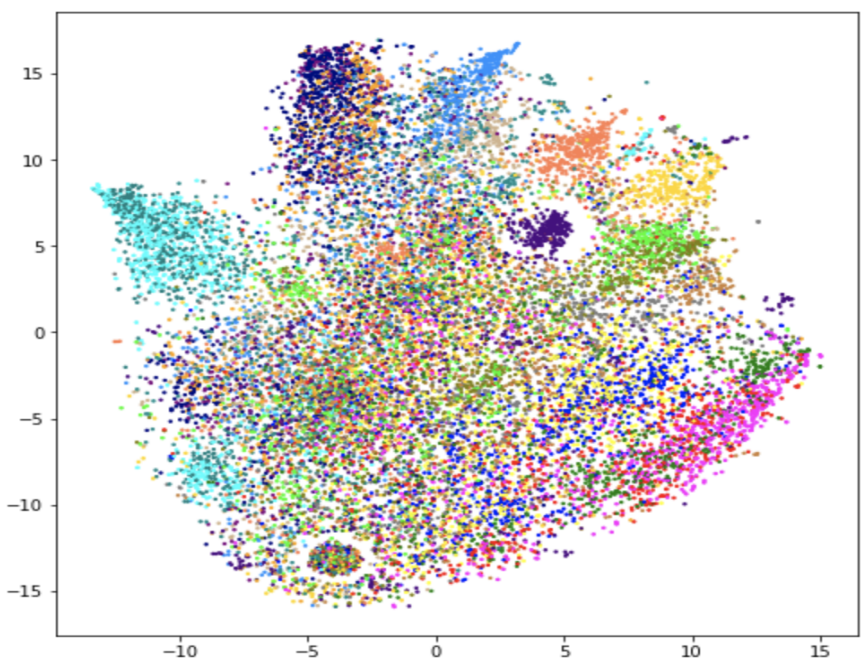


**4. We use gensim to learn the Word2Vec and Doc2Vec models, and visualize the results by t-SNE.**

Visualizing Word embedding space by TSNE:

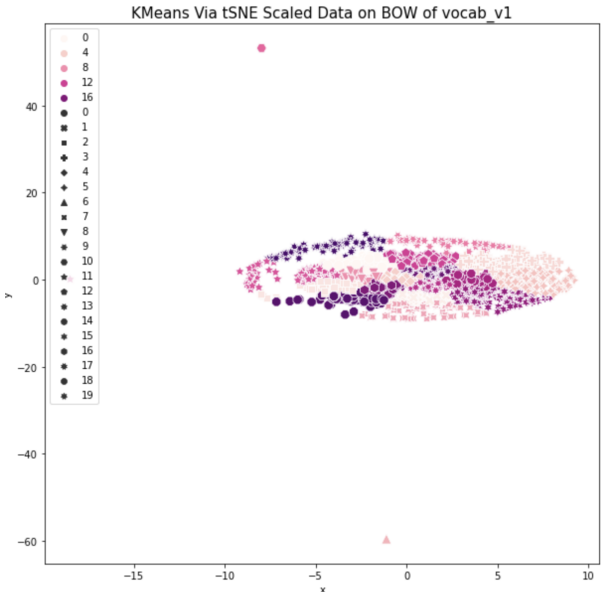


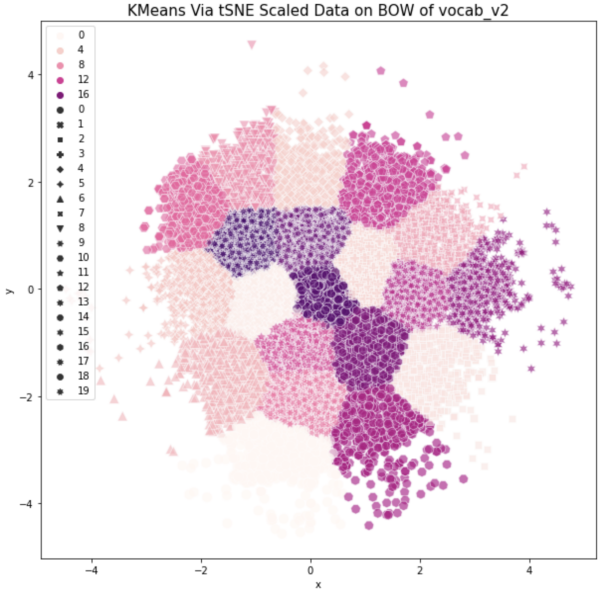
Visualizing Word embedding space by TSNE:



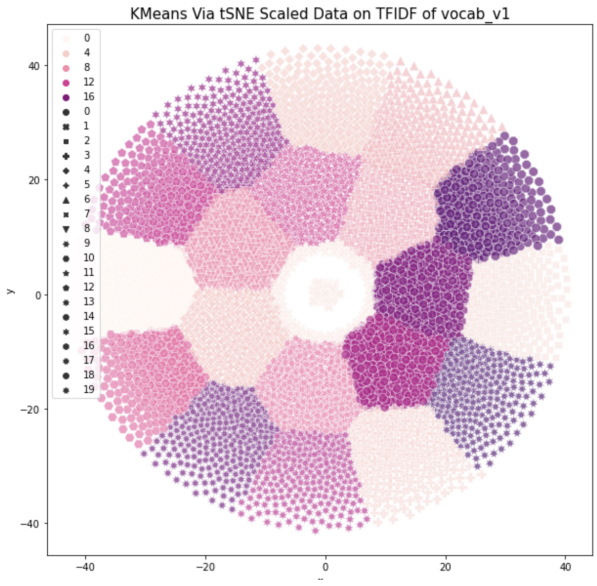
## 5. K-Means Clustering

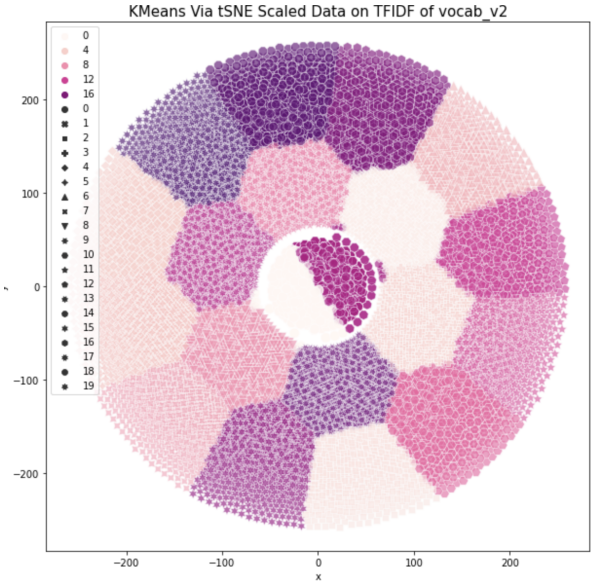
BoW:



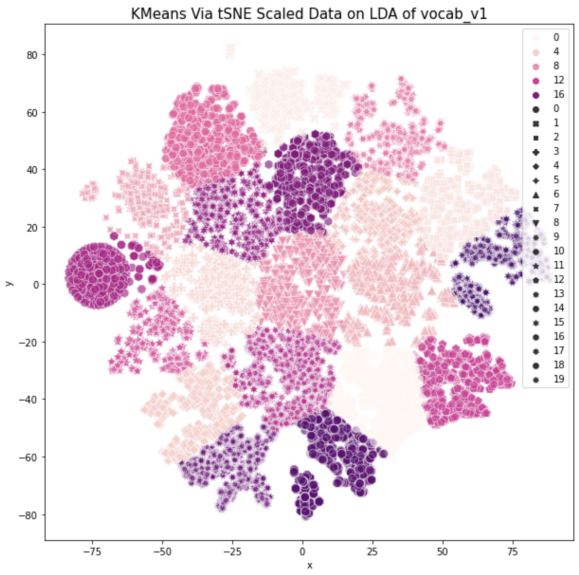



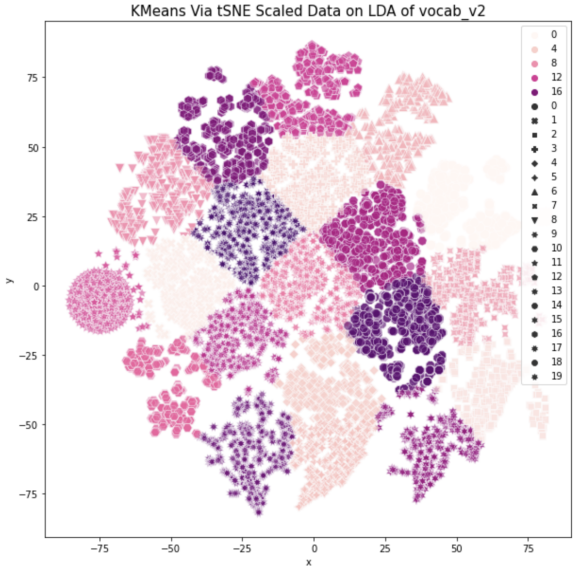

TF-IDF:



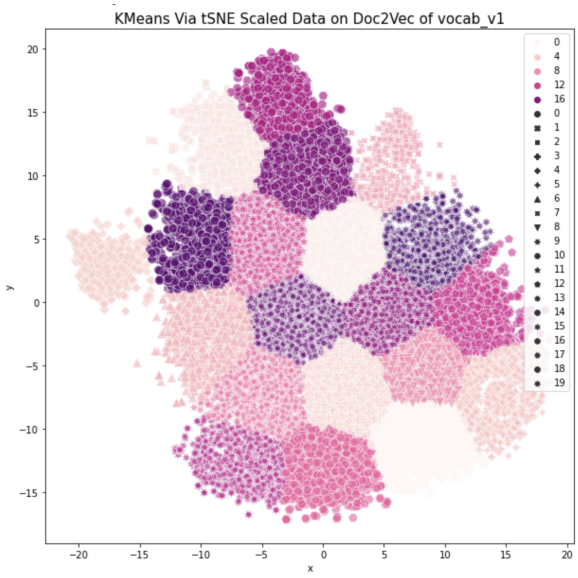



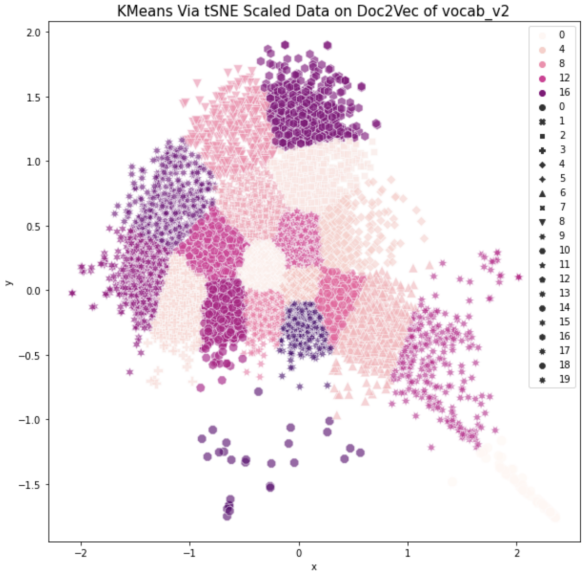
LDA:




Doc2Vec:



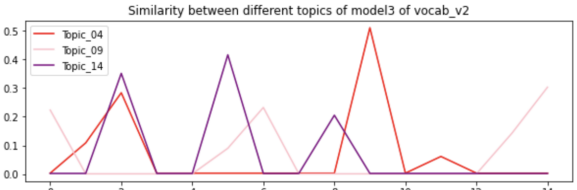

NMI Values:

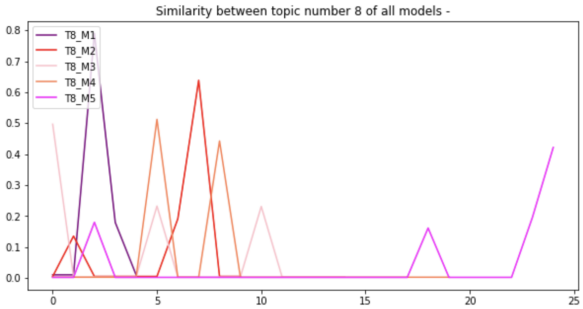
```
NMI values for all models on vocab_v1 -
NMI OF BOW MODEL: 0.011
NMI OF TFIDF MODEL: 0.274
NMI OF LDA MODEL: 0.282
NMI OF DOC2VEC MODEL: 0.254
```

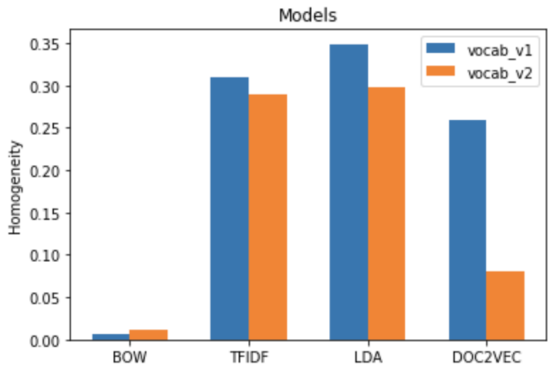
```
NMI values for all models on vocab_v2 -
NMI OF BOW MODEL: 0.018
NMI OF TFIDF MODEL: 0.236
NMI OF LDA MODEL: 0.291
NMI OF DOC2VEC MODEL: 0.044
```

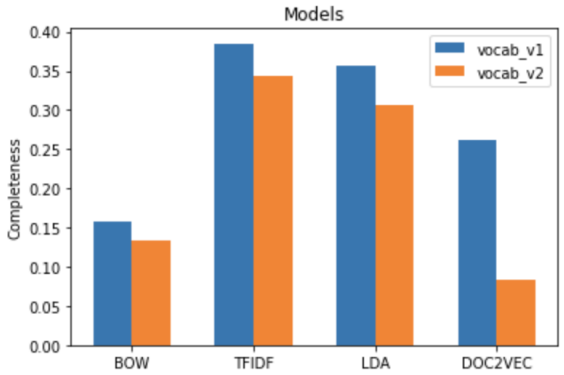
## 6. Experiment Analysis

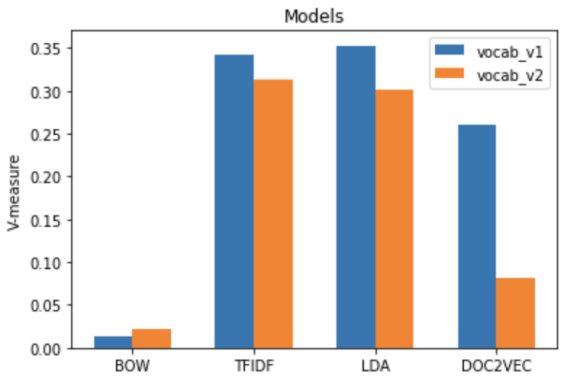


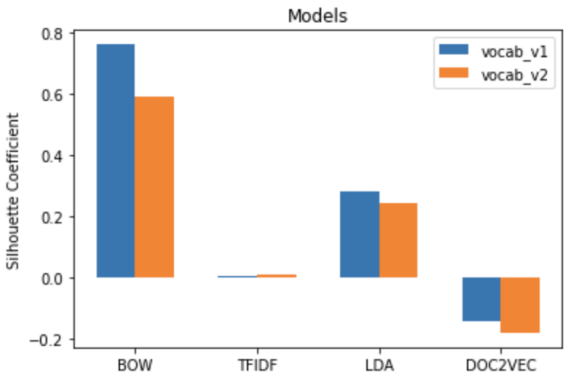


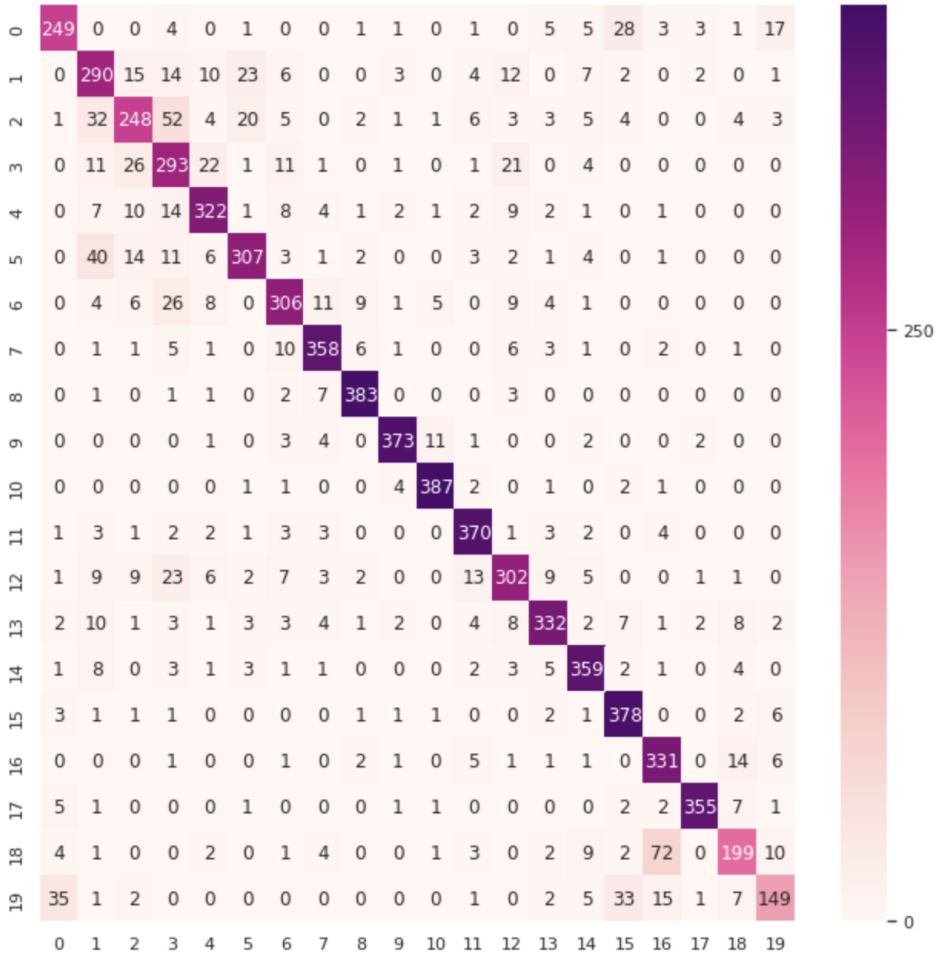



**7. Multinomial Naive Bayes Classifier**

The MultinomialNB is a part of a family of classifiers based on bayes' popular probability theorem known as to create simple powerful models, particularly in the areas of document classification. To select the Naive Bayes classifier (NBC) would be more desirable because of its high speed. MultinomialNB is preferred over other algorithms because for this size of problem, they have a parallel map-reduce implementation. The MultinomialNBC has excellent results for text data analysis. We utilized MultinomialNB to perform classification on the documents. We divided our data into two sets: training and testing. We first fit the model on training data and then tried to predict the response on test set and obtained the following results:

```
Predicted Class Labels: [ 7 11  0 ...  9  3 15]
0.8352363250132767
```




```
Classification Error of MultinomialNB: 0.16476367498672329
```

**8. Deep Neural Networks**

We are using DNN to  try to learn new doc and get an overall improvement.
We first trained the DNN model on the training set and predicted the result on test set and received following accuracy:

```
tf-idf with 35000 features
Epoch 1/4
89/89 - 29s - loss: 2.8241 - accuracy: 0.0963 - val_loss: 2.2666 - val_accuracy: 0.3139 - 29s/epoch - 327ms/step
Epoch 2/4
89/89 - 26s - loss: 1.4087 - accuracy: 0.5027 - val_loss: 0.9995 - val_accuracy: 0.6889 - 26s/epoch - 295ms/step
Epoch 3/4
89/89 - 26s - loss: 0.5640 - accuracy: 0.7933 - val_loss: 0.8698 - val_accuracy: 0.7516 - 26s/epoch - 294ms/step
Epoch 4/4
89/89 - 26s - loss: 0.2931 - accuracy: 0.8993 - val_loss: 0.9065 - val_accuracy: 0.7902 - 26s/epoch - 294ms/step
```

Accuracy Obtained:

```
0.7902283590015932
```

Also the NMI value obtained is greater than the KMeans NMI values, which represents a clear bump in the clustering performance.

```
NMI OF DNN MODEL: 0.715
```

**References:**

https://scikit-learn.org/stable/modules/generated/sklearn.cluster.KMeans.html
https://scikit-learn.org/stable/modules/generated/sklearn.cluster.SpectralClustering.html
https://radimrehurek.com/gensim/models/tfidfmodel.html
https://radimrehurek.com/gensim/corpora/dictionary.html
https://radimrehurek.com/gensim/models/ldamodel.html
https://radimrehurek.com/gensim/models/word2vec.html
https://radimrehurek.com/gensim/models/doc2vec.html
https://radimrehurek.com/gensim/matutils.html#gensim.matutils.corpus2dense
https://radimrehurek.com/gensim/parsing/preprocessing.html
https://radimrehurek.com/gensim/autoexamples/tutorials/runlda.html#sphx-glr-auto-examples-tutorials-run-lda-py
https://pyldavis.readthedocs.io/en/latest/readme.html