Text clustering model for categorizing books

Project Write-up by: Himani Kaushik

Abstract

A local library wants to implement a categorizing system based on the similarity of the genres and the authors. This text clustering model will enable the librarian to assign the books into groups and subgroups based on their labels. This unsupervised learning model will apply various transformation and clustering techniques to choose the best algorithm to assign the correct labels to the books. Five books, written by different authors and from different genres, were extracted from the Project Gutenberg website. The data was preprocessed and partitioned into 200 partitions and each partition consisted of 150 words. Different text processing and dimension reducing techniques were used to transform the data. Unsupervised clustering algorithms were used on all data transformations to find out the best algorithm.

Design

The purpose of the model is to use text clustering techniques to assign books into similar groups or clusters. The model uses the data obtained from the Project Gutenberg website. The data was preprocessed using NLTK library by removing stop words and lemmatizing every word to its origin. BOW, TD-IDF and LDA techniques were used to transform the data. K-means and Hierarchical clustering algorithms were used on all three data transformations, respectively. V-measure score and Silhouette score were used for evaluating the performance of clustering algorithms against the true authors. LDA-transformed data was used to compare different silhouette scores to figure out the optimum number of clusters using K-means algorithm.

Data

The data was obtained from the Project Gutenberg website - https://www.gutenberg.org. The following books were extracted from this website using python library – requests:

1. Chaldea
2. A Book About Lawyers
3. EBook of Darwinism
4. The Vicomte de Bragelonne
5. A Popular History of Astronomy During the Nineteenth Century

All the books are by different authors and from different genres. The data was preprocessed using NLTK. The stop words and garbage characters were removed, all the words were converted to lower case and lemmatization was performed to return every word to its origin. The books were randomly partitioned into 200 partitions and each partition consisted of 150 words. The books were labelled as [a,b,c,d,e].

Algorithms

- Data preprocessing:
    - removed stop words, garbage characters, all words converted to lower case, lemmatization performed

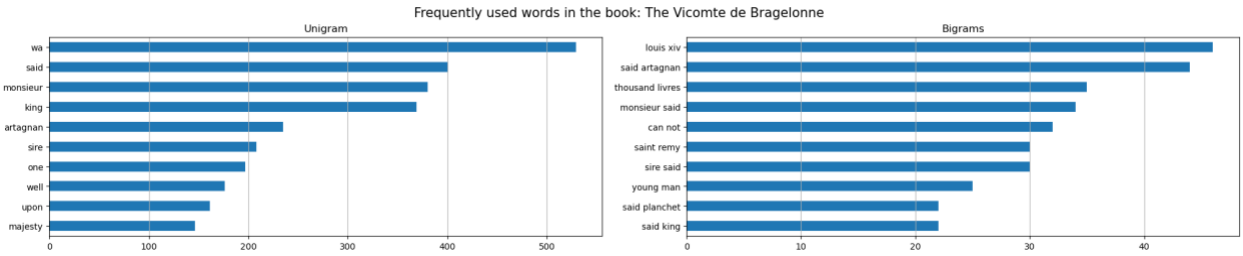
    - Unigrams/bigrams and Wordcloud used to find out the frequently used words in all 5 books
- Feature Engineering:
    - Bag of Words (BOW)
    - TF-IDF
    - LDA
- Clustering Algorithms:
    - K-means
    - Hierarchical clustering
- Evaluation scores:
    - V-measure
    - Silhouette score

Tools

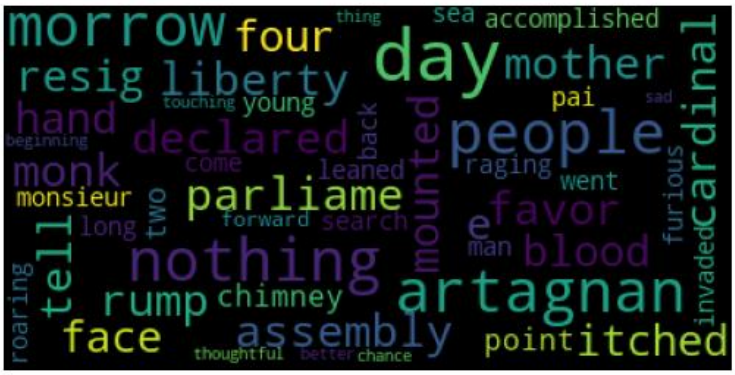
- Requests: For extracting books using their urls.
- Pandas and Numpy: For cleaning data and preprocessing.
- NLTK, gensim, scikit-learn: For processing text, topic modeling, clustering and evaluation
- Matplotlib, Seaborn, Wordcloud, pyLDAvis, scipy : For visualizing the data.

Communication

Unigram and bigram techniques were used to show frequently used words in all the five books. One example is shown below:



Wordcloud was used to show 50 frequently used words in all five books. One example is:



Feature engineering was performed using following techniques:

1. BOW transformation

```
BOW
```

	aaron	abandon	abandoned	abandoning	abandonment	abated	abb	abbe	abbey	abbott	...	zodiacal	zonal	zone	zool	zoologique	zoologist
0	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
995	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0
996	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0
997	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0
998	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0
999	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0

1000 rows × 16048 columns

2. TF-IDF transformation

```
TFIDF_Vector
```

	aaron	abandon	abandoned	abandoning	abandonment	abated	abb	abbe	abbey	abbott	...	zodiacal	zonal	zone	zool	zoologique	zoologist	zoology	zur	zwischen	zygomatic
0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
1	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
2	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
3	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
4	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
995	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
996	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
997	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
998	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
999	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

1000 rows × 16048 columns

3. LDA transformation

```
LDA.head()
```

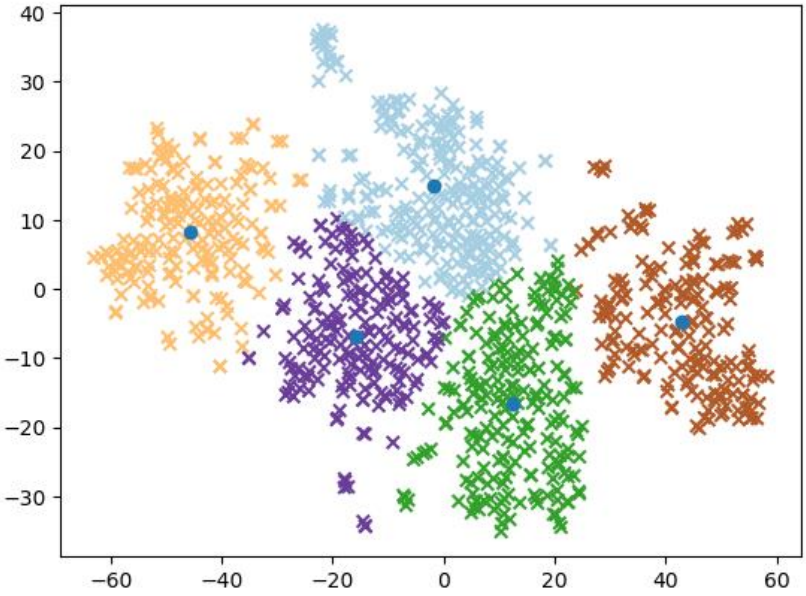
	1	2	3	4	5	res
0	0.121710	6.297770	10.518360	41.755322	91.040962	5
1	128.704269	6.219258	0.131042	15.295912	0.270618	1
2	84.970520	0.204380	6.978431	27.358265	30.191488	1
3	3.745756	0.206495	0.131131	146.006546	0.268842	4
4	81.228470	1.564073	0.131100	51.285492	15.966647	1

```
PredictedWords
```

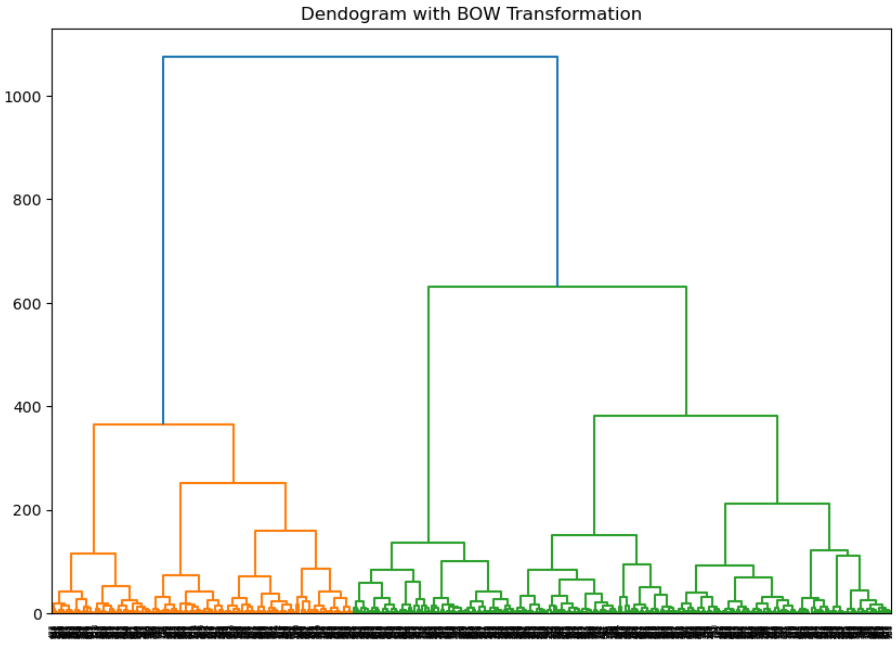
```
(array([[1.2170985e-01, 6.2977695e+00, 1.0518360e+01, 4.1755322e+01,
         9.1040962e+01],
        [1.2870427e+02, 6.2192578e+00, 1.3104180e-01, 1.5295912e+01,
         2.7061805e-01],
        [8.4970520e+01, 2.0437954e-01, 6.9784307e+00, 2.7358265e+01,
         3.0191488e+01],
        ...,
        [3.1825294e+00, 9.5775023e+00, 1.3119207e-01, 3.2911915e-01,
         1.3765221e+02],
        [1.2156795e-01, 3.0078484e+01, 1.3103145e-01, 1.2030704e+02,
         2.6176342e-01],
        [3.7553685e+00, 1.6282141e+01, 2.9358027e+01, 9.1212921e+01,
         1.0117374e+01]], dtype=float32),
 None)
```

Clustering algorithms on three transformations:

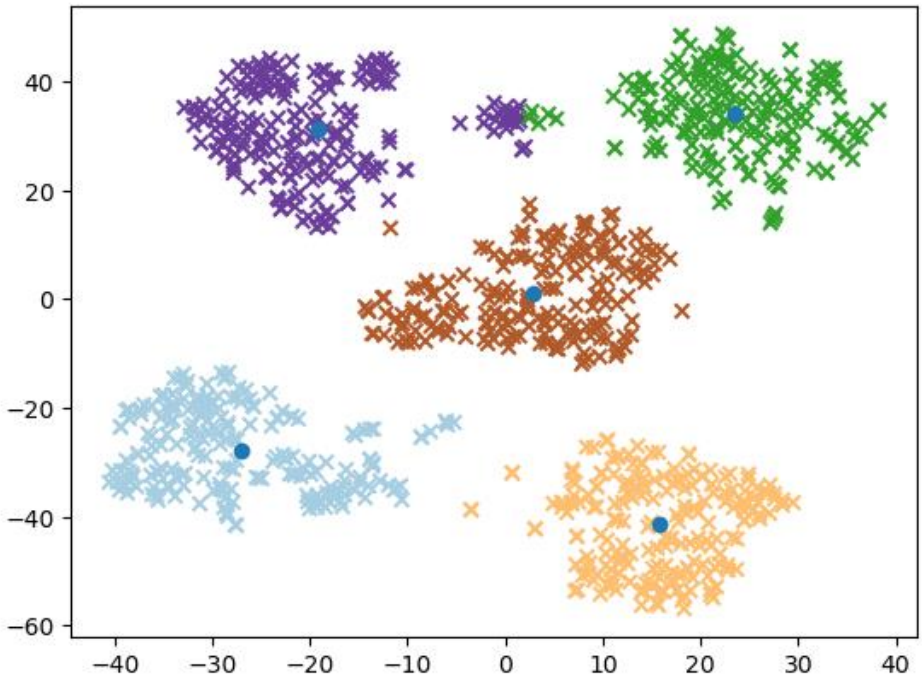
1. **BOW and K-Means algorithm:**



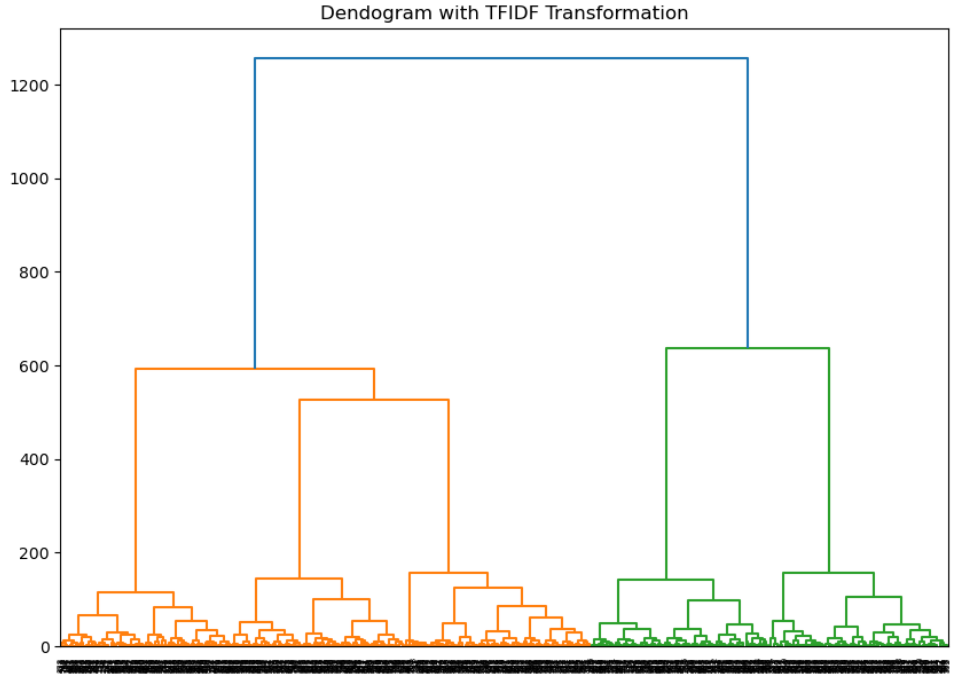
2. **BOW and Hierarchical clustering algorithm**



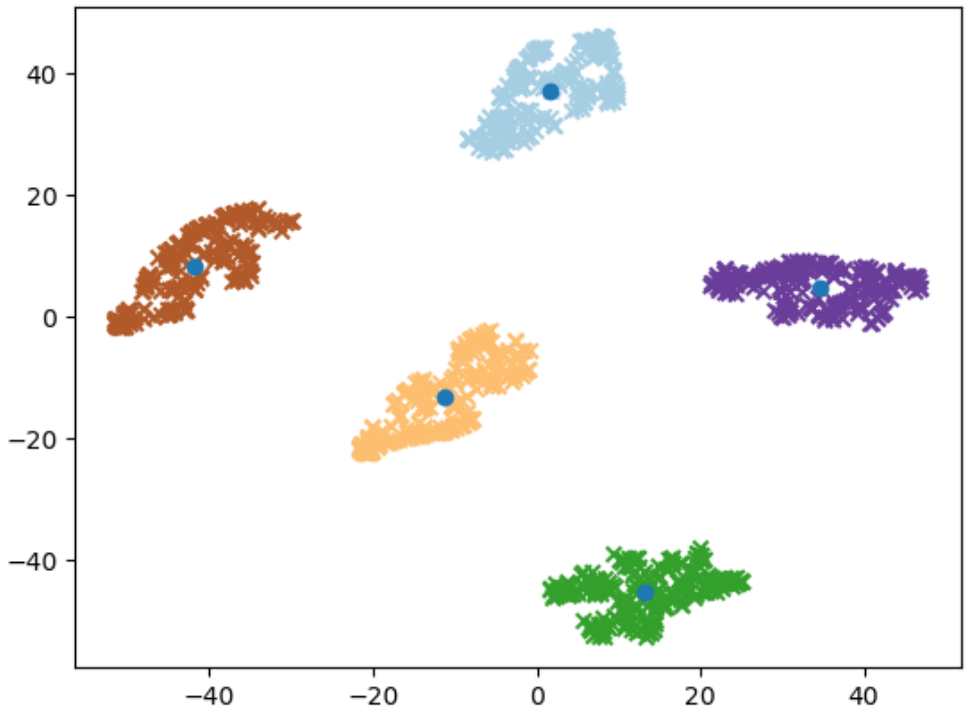
3. TF-IDF and K-Means algorithm



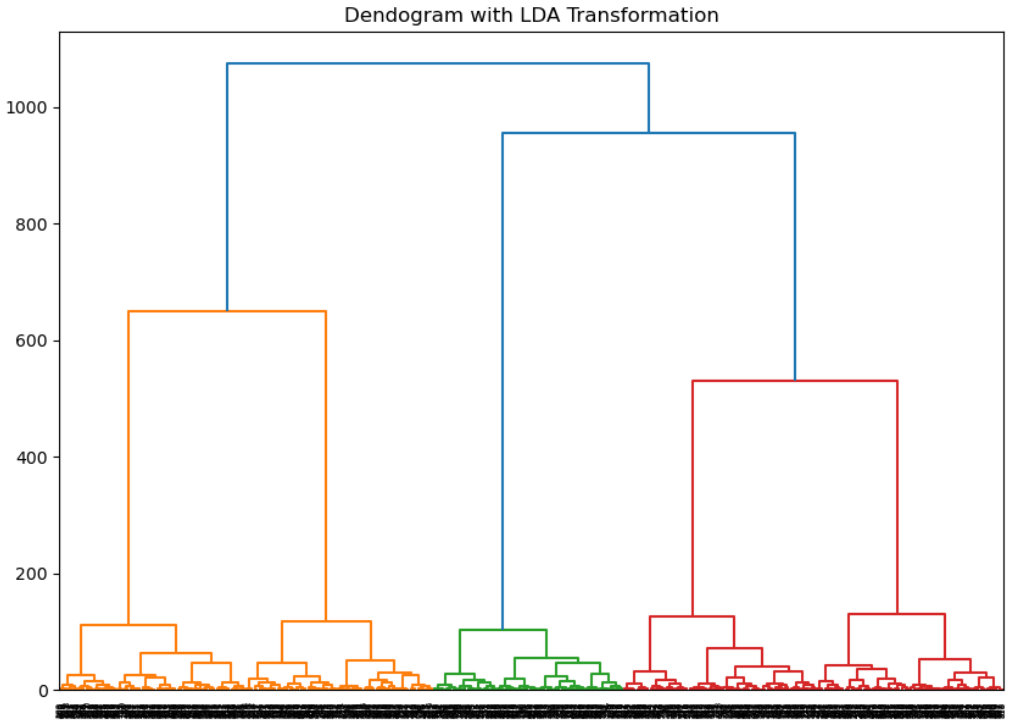
4. TF-IDF and Hierarchical clustering algorithm



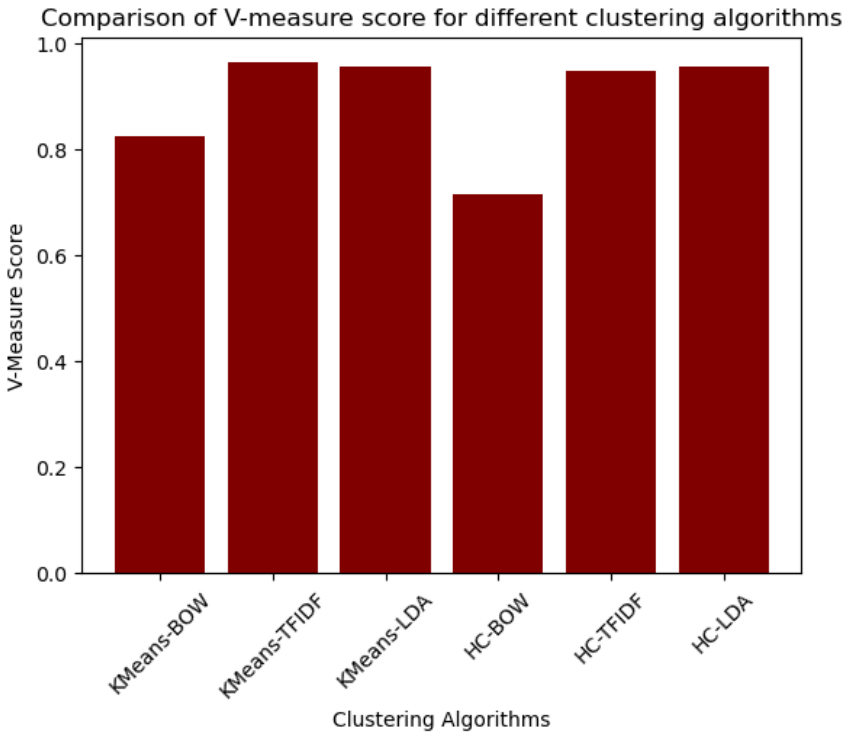
5. LDA and K-Means algorithm



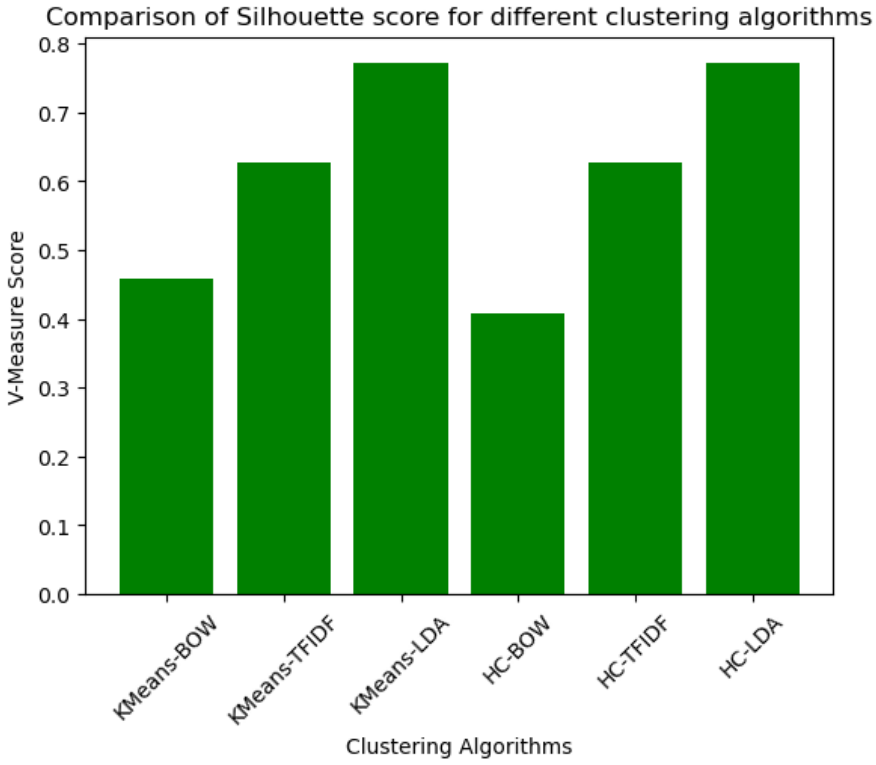
6. LDA and Hierarchical clustering algorithm



V-Measure for Evaluating Clustering Performance against the true authors

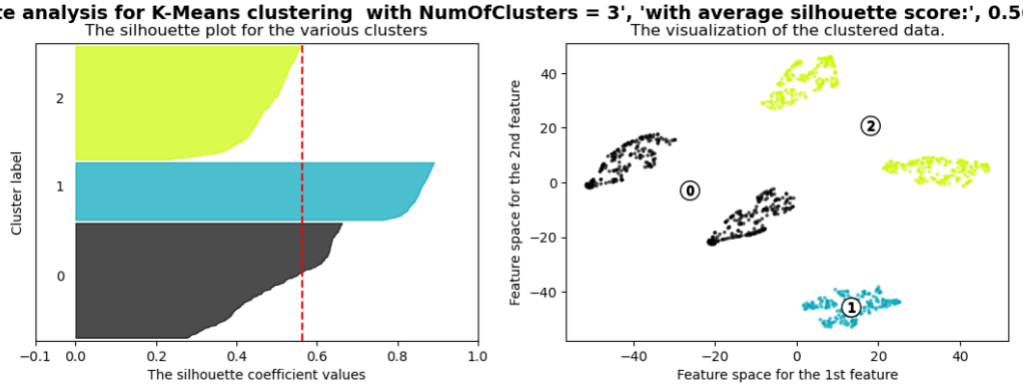


Silhouette score for Evaluating Clustering Performance against the true authors

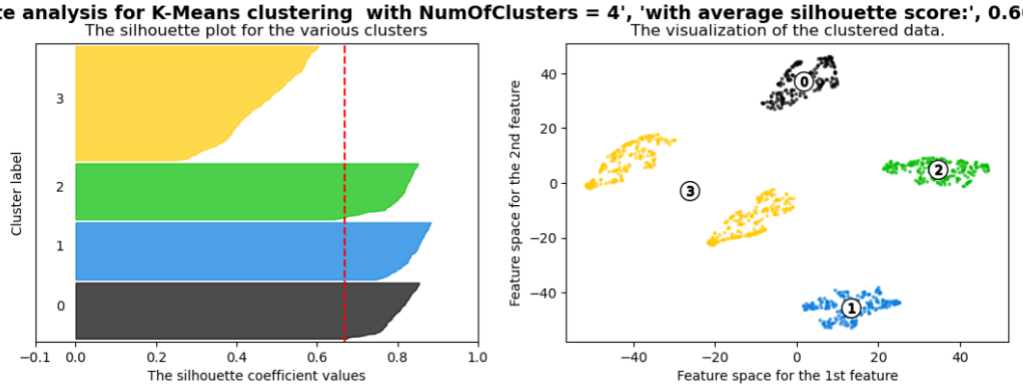


Comparison between Silhouette scores using different K numbers on LDA transformed data

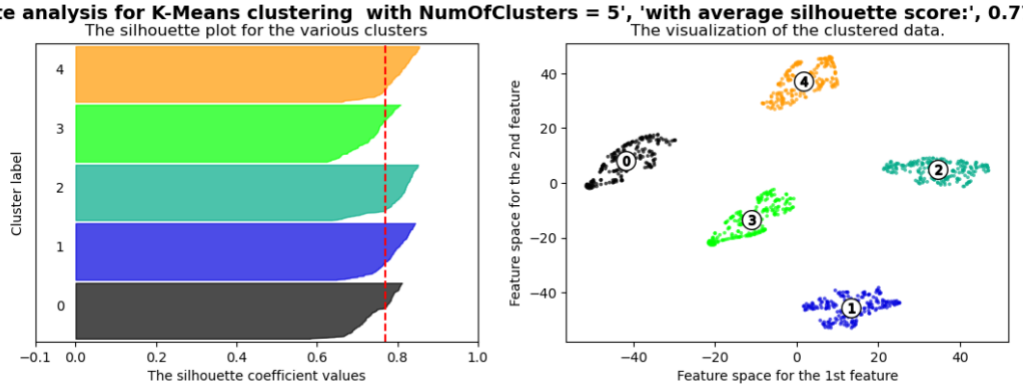
('Silhouette analysis for K-Means clustering with NumOfClusters = 3', 'with average silhouette score:', 0.56309605)



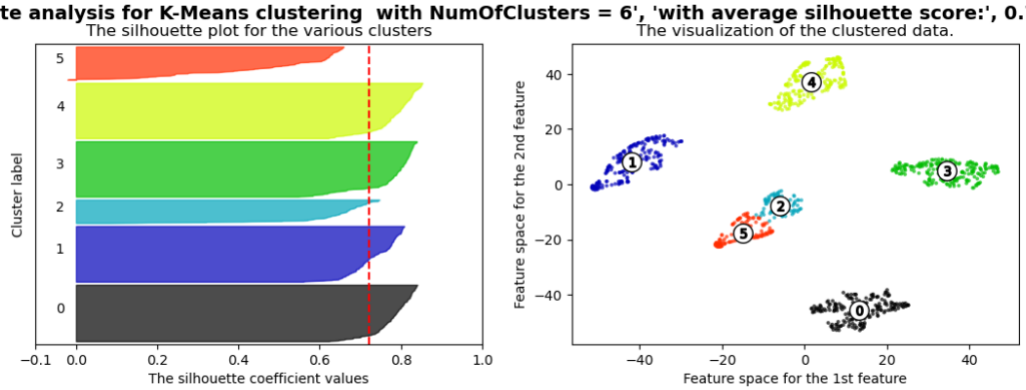
('Silhouette analysis for K-Means clustering with NumOfClusters = 4', 'with average silhouette score:', 0.66956854)



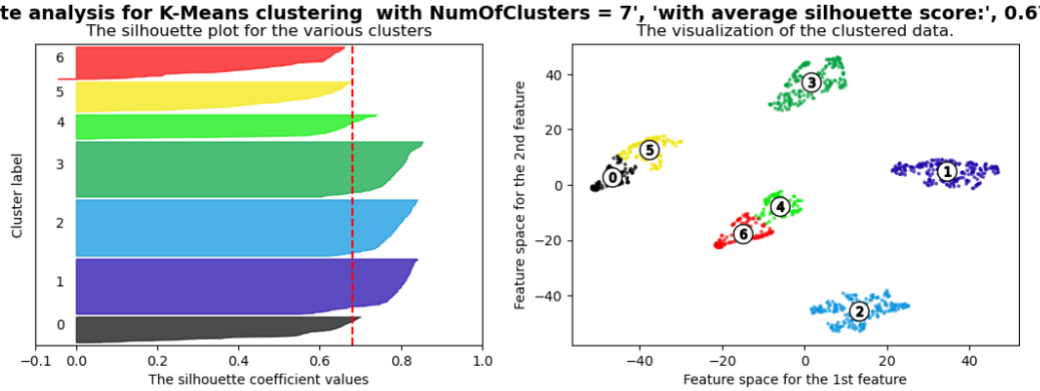
('Silhouette analysis for K-Means clustering with NumOfClusters = 5', 'with average silhouette score:', 0.77111685)



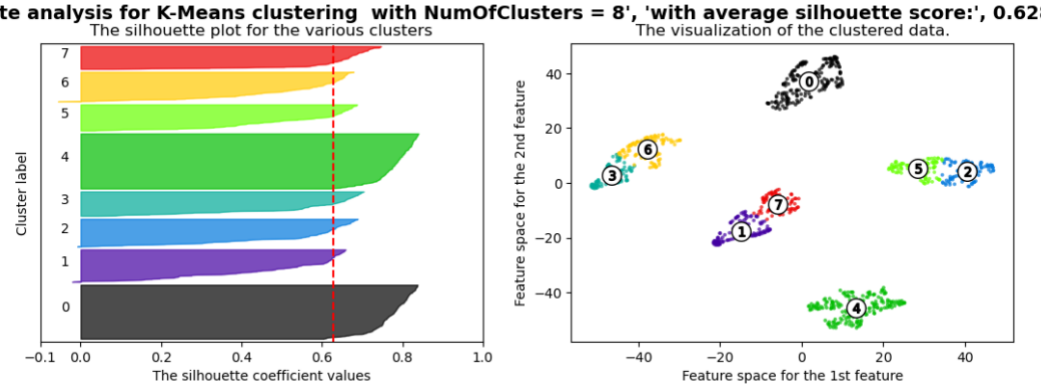
('Silhouette analysis for K-Means clustering with NumOfClusters = 6', 'with average silhouette score:', 0.7222523)



('Silhouette analysis for K-Means clustering with NumOfClusters = 7', 'with average silhouette score:', 0.6795865)



('Silhouette analysis for K-Means clustering with NumOfClusters = 8', 'with average silhouette score:', 0.62865674)



K = 5 was figured out as the optimum number of clusters as the average score is the highest and all the clusters have the same size and clear clusters.

Hence, K-means algorithm on LDA transformed data where number of clusters is 5 is the best algorithm for the text clustering project.