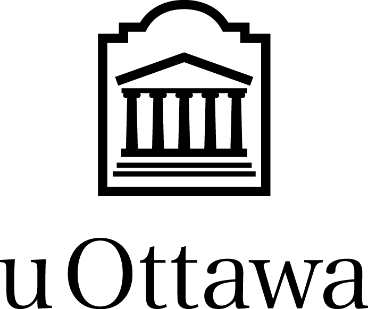
**University of Ottawa**

Faculty of Engineering

School of Electrical Engineering  
and Computer Scienc

School of Electrical Engineering  
and Computer Science



2021 Summer DTI5125 Data Science Applications

Assignment Three

Classification Assignment (Group) Report

**Group Members:**

Salma Mousa, Tokka Hassan, Reyad Melies, Shahenda Youssef

**Group ID.:** DSA\_202101\_ 10

**Under Supervision:** Dr. Arya Rahgozar

**Submission Date:** 20/06/2021

Table of Contents

**TABLE OF CONTENTS1**

**INTRODUCTION4**

REPORT BODY**4**

Data Preparation4

Books Choice4

Clean Data5

Partition Data5

Label Data6

Feature Extraction7

Bag of Words7

TFIDF7

Data Visualization8

Clustering8

K-means8

Expectation Maximization 15

Hierarchical clustering17

Champion Model19

Choosing Champion Model19

Error analysis19

#### 

# LIST OF FIGURES

[Figure 1 – Removing non-alphabetic characters code 5](#_Toc75087249)

[Figure 2 - Text partitioning code 5](#_Toc75087250)

[Figure 3 - Dataframe after text partitioning 6](#_Toc75087251)

[Figure 4 - Dataframe after labeling partitions 6](#_Toc75087252)

[Figure 5 – bag of words code 7](#_Toc75087253)

[Figure 6 – dataset obtained using BoW method 7](#_Toc75087254)

[Figure 7 - Dataset after TFIDF vectorization 7](#_Toc75087255)

[Figure 8 - Top Frequent words 8](#_Toc75087256)

[Figure 9 - Wordcloud 8](#_Toc75087257)

[Figure 10 - Shuffling data code 9](#_Toc75087258)

[Figure 11 – Data points distribution before and after clustering 9](#_Toc75087259)

[Figure 12 - metric scores of k-means with TFIDF dataset 10](#_Toc75087260)

[Figure 13 - Data points distribution before and after clustering with k-means model 10](#_Toc75087261)

[Figure 14 - metric scores of k-means model with TFIDF dataset and PCA 10](#_Toc75087262)

[Figure 15 - Data points distribution using k-means with TSNE 11](#_Toc75087263)

[Figure 16 - metric scores using the k-means model with TFIDF data and TSNE 11](#_Toc75087264)

[Figure 17 - Data points distribution using k-means with BoW dataset 12](#_Toc75087265)

[Figure 18 - metric scores of k-means model with BoW dataset 12](#_Toc75087266)

[Figure 19 - Data distribution using k-means model with BoW dataset and PCA 13](#_Toc75087267)

[Figure 20 - metric scores of k-means model with BoW dataset and PCA 13](#_Toc75087268)

[Figure 21 - Data points distribution using k-means with BoW dataset and TSNE 14](#_Toc75087269)

[Figure 22 - metric scores of k-means with BoW dataset and TSNE 14](#_Toc75087270)

[Figure 23 - original data distribution 15](#_Toc75087271)

[Figure 24 - data distribution after using EM with TFIDF dataset 15](#_Toc75087272)

[Figure 25 - EM with TFIDF scores 15](#_Toc75087273)

[Figure 26 - cross tabulation of EM with TFIDF 16](#_Toc75087274)

[Figure 27 - data distribution using EM with BoW 16](#_Toc75087275)

[Figure 28 - EM with BoW dataset scores 16](#_Toc75087276)

[Figure 29 - Cross tabulation plot of EM with Bow Dataset 17](#_Toc75087277)

[Figure 30 - Dendrogram for heirachial clustering 17](#_Toc75087278)

[Figure 31 - Data plot and score for HC with TFIDF dataset 18](#_Toc75087279)

[Figure 32 - Data plot and score for HC with BoW dataset 18](#_Toc75087280)

[Figure 33 – silhouette and Kappa scores vs models plot 19](#_Toc75087281)

[Figure 34 - cross tabulation of the champion model 19](#_Toc75087282)

[Figure 35 - coherence scores for LDA model 20](#_Toc75087283)

[Figure 36 - Top words of importance for topic one 21](#_Toc75087284)

[Figure 37 - Top words of importance for topic two 21](#_Toc75087285)

[Figure 38 - Top frequent words in all documents 23](#_Toc75087286)

# 

# INTRODUCTION

In some real life application, data is not always labeled and the process can be both time and financially exhausting, and hence using clustering techniques can be very important in many business cases. Since clustering is an unsupervised learning technique, we might have to accept clusters that don't really represent our data well, so it's very important to investigate how the different clustering algorithms work, and study the results they output, so that we can make sure we are tuning our models correctly and according to our application.

For our project, we are using clustering algorithms on different books and checking what errors cause the model to give us inaccurate results, this can give very important insights for projects that contain large number of document analysis.

## REPORT BODY

## Data Preparation

## Books Choice

For this project, we choose 5 books that are semantically different, written by different authors and have different genres.

The books share something in common that we believe will cause some issues with the clustering algorithms, they all revolve in different ways about biblical analogies and writings.

The books are:

1. Bible:   
   Genre: religious text
2. Father brown by Chesterton:   
   Genre: Mystery and fiction  
   Points of interest: The book is about the adventures of a priest solving crimes
3. Macbeth by Shakespeare:   
   Genre: tragedy   
   Points of interest: The book has many biblical parallels with the bible and the old testament stories
4. Paradise Lost by John Milton:  
   Genre: Epic poetry, Christian mythology  
   Points if interest: The poems all revolve around biblical stories
5. Moby dick by Herman Melville:

Genre: Novel, Adventure fiction

Points of interest: the main character is inspired from the bible and there are biblical analogies.

## Clean Data

The first step in data preparation is data cleaning from any unwanted data that can affect the model results.

We removed all non-alphabetic characters, stop words and then we used the stemming function to reduce the word to its stem.

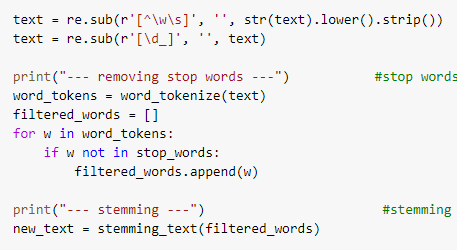


Figure 1 – Removing non-alphabetic characters code

Now the data consists of only words in lower case separated by white spaces and reduced to its stem.

## Partition Data

After cleaning the data, it is ready for partitioning. We started by splitting the text to words, then we made a list of partitions each of 150 words. Finally, we took random 200 partitions and repeated the steps for five books of the same genre.

All partitions and their book labels are added to a pandas dataframe.

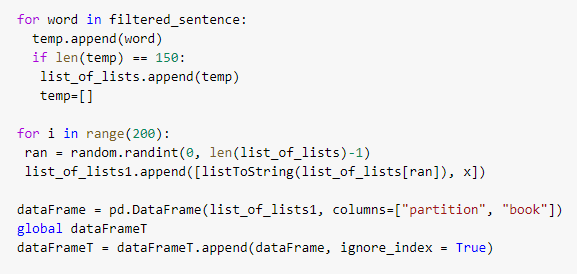


Figure 2 - Text partitioning code

And the resulted dataframe is as follows.

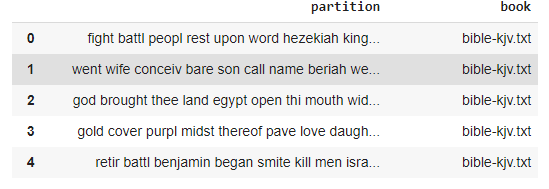


Figure 3 - Dataframe after text partitioning

## Label Data

Now the dataframe is ready with the partitions and their book names, we can start labeling the data. This is simply done by separating the authors name from the book label, then adding it to the dataframe as a new column.

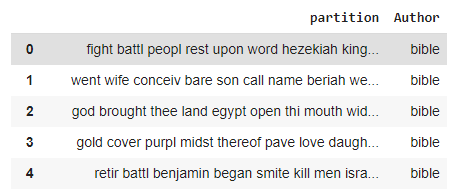


Figure 4 - Dataframe after labeling partitions

## Feature Extraction

* + 1. Bag of Words

For implementing the bag of words method, we used the count vectorizer approach, where the resultant represents a sparse matrix of words that are most frequent in the partitions, with a maximum number of 2500 words.

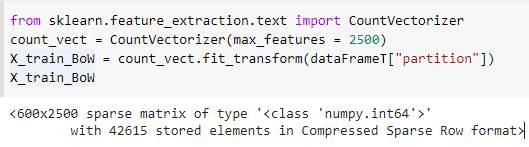


Figure 5 – bag of words code

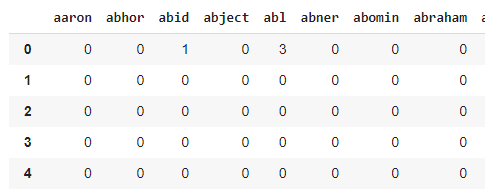


Figure 6 – dataset obtained using BoW method

* + 1. TFIDF

The second test we prepared is using the TFIDF vectorizer which transforms text to feature vectors that shows the importance of each word using the inverse document frequency calculation, this this data can be used as input to estimator. We set the parameter "max-features" to 2500.

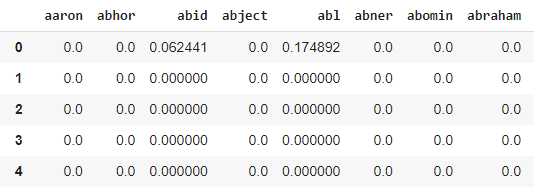


Figure - Dataset after TFIDF vectorization

* + 1. Data Visualization

We generated some plots that might give us an insight about the data presented.

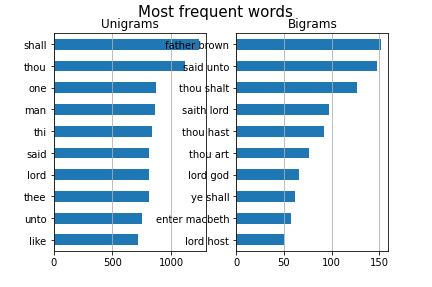


Figure 8 - Top Frequent words

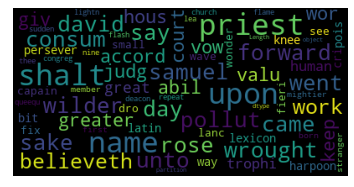


Figure 9 - Wordcloud

## Clustering

* + 1. K-means

After the feature extraction step, we can now start implementing our models, but first we shuffled the data to get the best results and to make sure the model does not train on a class more than the others do.

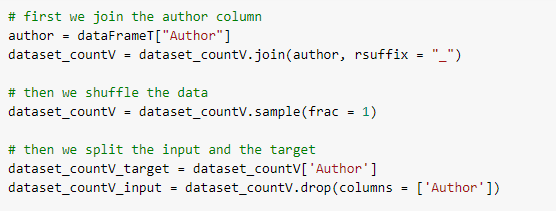


Figure 10 - Shuffling data code

Then we implemented the K-means model, we set k = 5 as we have 5 classes for 5 different authors.

First we tested with the TFIDF dataset, the first test was using the TFIDF dataset directly, we plotted the before and after clustered points.

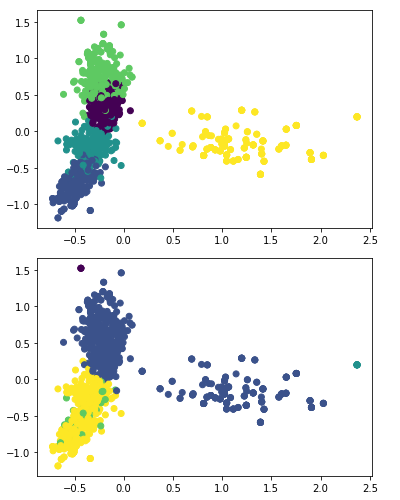


Figure 11 – Data points distribution before and after clustering

We can see that the clustering is not working very well and is not similar at all to original clustering of the data, and hence it gave a very bad score.



Figure - metric scores of k-means with TFIDF dataset

Then we plotted the data after performing PCA to reduce the dimensionality to 2, so we can plot the data and observe their distribution.

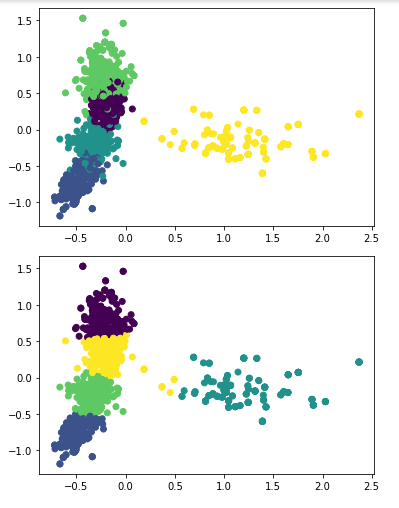


Figure - Data points distribution before and after clustering with k-means model

As we can see here the clusters are very similar to the original clusters.

This gave an accuracy of



Figure - metric scores of k-means model with TFIDF dataset and PCA

We also tested using TNSE instead of the PCA and obtained the following results.

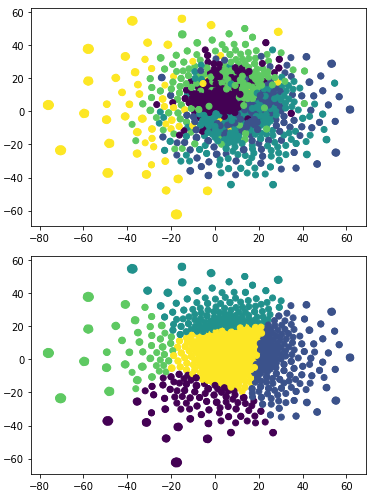


Figure - Data points distribution using k-means with TSNE

Since the points aren’t separable in the graph, it was expected that the accuracy acquired would be very low.



Figure - metric scores using the k-means model with TFIDF data and TSNE

Then we tested with the BoW dataset, first without using PCA.

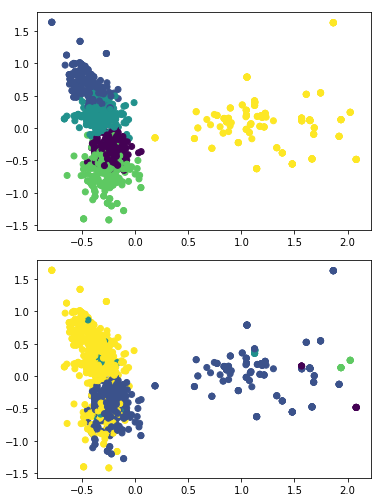


Figure - Data points distribution using k-means with BoW dataset



Figure - metric scores of k-means model with BoW dataset

Then using PCA before training the model.

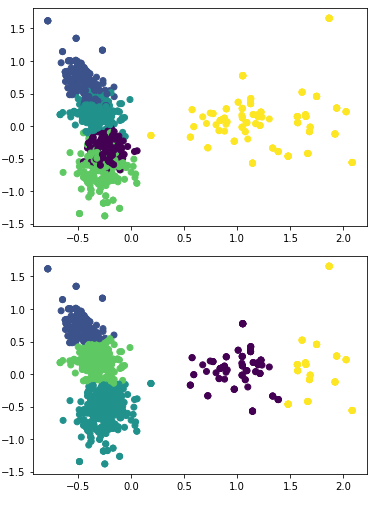


Figure - Data distribution using k-means model with BoW dataset and PCA



Figure - metric scores of k-means model with BoW dataset and PCA

Then using TSNE on the dataset before training the model.

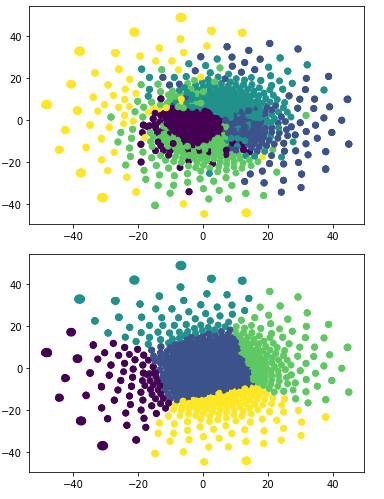


Figure - Data points distribution using k-means with BoW dataset and TSNE



Figure - metric scores of k-means with BoW dataset and TSNE

The accuracy is very low, as it's obvious from the plot that most points got misclassified.

* + 1. Expectation maximization

This is the plot of the original data.

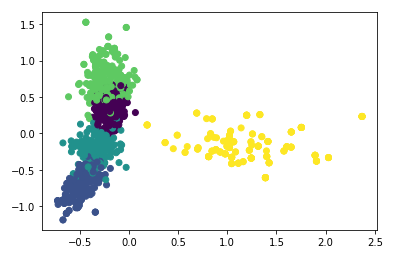


Figure - original data distribution

After training the model on the TFIDF dataset, we obtained these clusters.

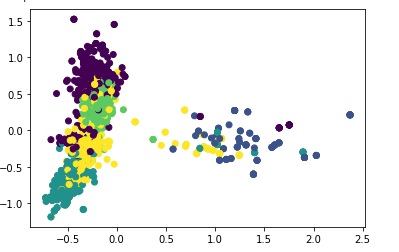


Figure - data distribution after using EM with TFIDF dataset

And this gave the following accuracy



Figure - EM with TFIDF scores

To gain more insight on the misclassification that happened we used the cross tabulation function to plot the different classes in each cluster.

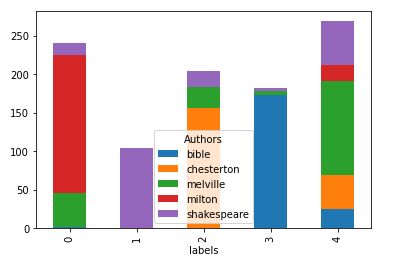


Figure - cross tabulation of EM with TFIDF

So here we can see that the data in cluster zero is not purely from a single class, while in class 1 it's all from the same class. The plot explains why the kappa accuracy was high for this model.

Then we tested with the BoW dataset and obtained these results.

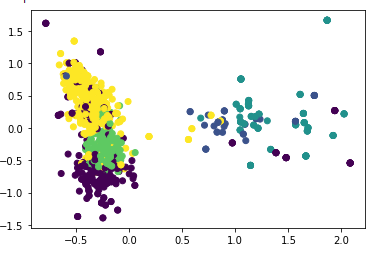


Figure - data distribution using EM with BoW



Figure - EM with BoW dataset scores

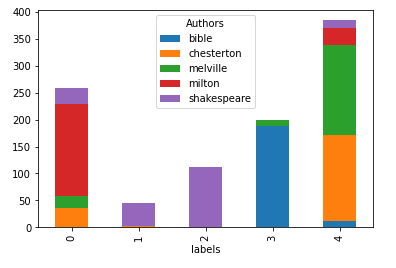


Figure - Cross tabulation plot of EM with Bow Dataset

## Hierarchical Clustering

First, we plotted the dendrogram, which shows that the best cluster number is 5 clusters.

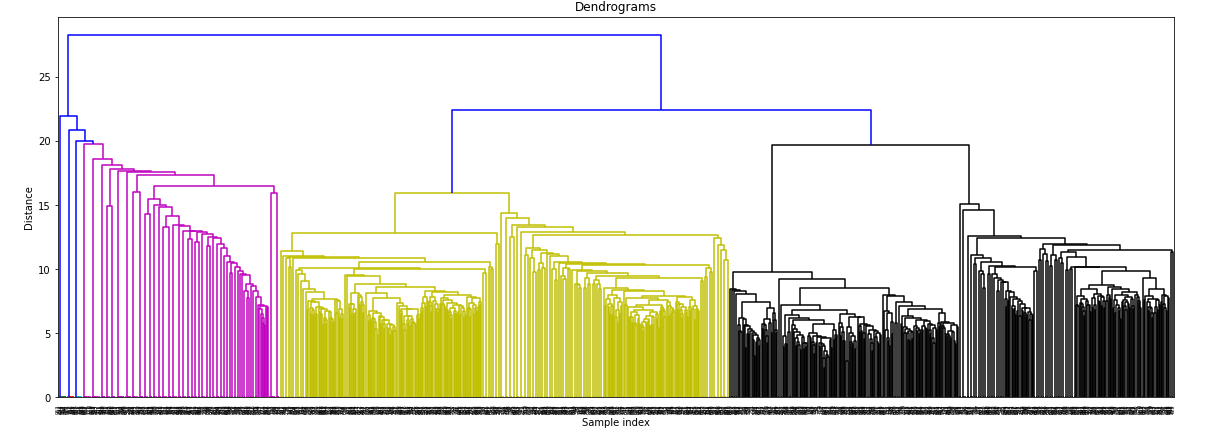


Figure - Dendrogram for heirachial clustering

Then we trained the model on the TFIDF dataset.

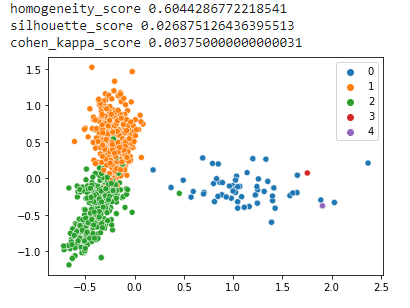


Figure - Data plot and score for HC with TFIDF dataset

And then we tested on the BoW dataset.

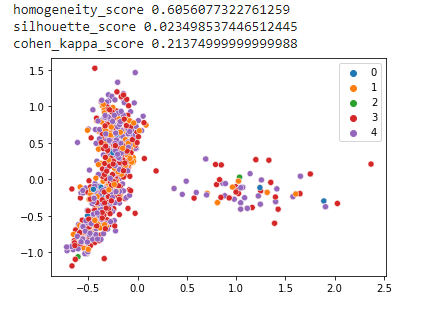


Figure - Data plot and score for HC with BoW dataset

* 1. Champion Model
     1. Choosing the champion model

In order to choose the champion model, we saved the silhouette and Kappa score from all the models we have tested and plotted the silhouette and kappa score on the same graph to be able to determine the point which maximizes both values.

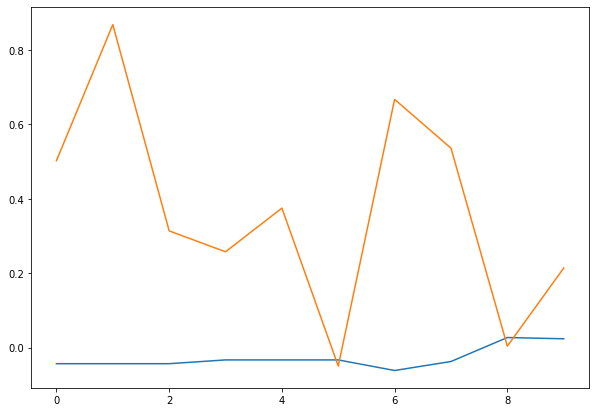


Figure 33 – silhouette and Kappa scores vs models plot

We can see that the model that has the best kappa value is the 2nd model which is the k-means with TFIDF after applying PCA, and since the silhouette values are very low and nearly non varying we will take the highest kappa score as the champion model.

We also plotted the cross tabulation plot of the winning model.

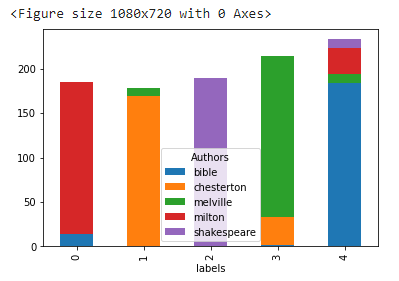


Figure - cross tabulation of the champion model

But even though this the champion model, it only achieves a kappa score of 0.87 and a very low score on the silhouette evaluation.

So we started investigating further by obtaining the wrongly classified data.

* + 1. Error Analysis
       1. LDA

We used the LDA (Latent Dirichlet Allocation), which is a topic modeling technique, to gain more insight about our data.

The LDA is trained over the whole documents and it starts extracting different topics from these data. We can specify the number of topics we want to extract but there is an optimal number of topics that gives the best results.

To obtain the optimal number of topics we used the coherence score as a measure to how these topics are good, so we tested the number of topics on a range from 2 to 50, plotting the coherence score of each.

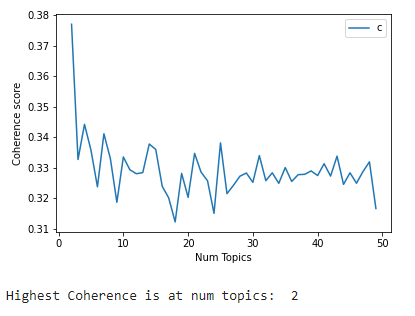


Figure - coherence scores for LDA model

We can see that the highest coherence score is obtained when we use only 2 number of topics. Since the coherence scores measures the degree of semantic similarity between high scoring words in the topic, then this indicates that the words are confined in only 2 topics.

To investigate these results further, we plotted the highest occurring words for each topic and we found some interesting results.

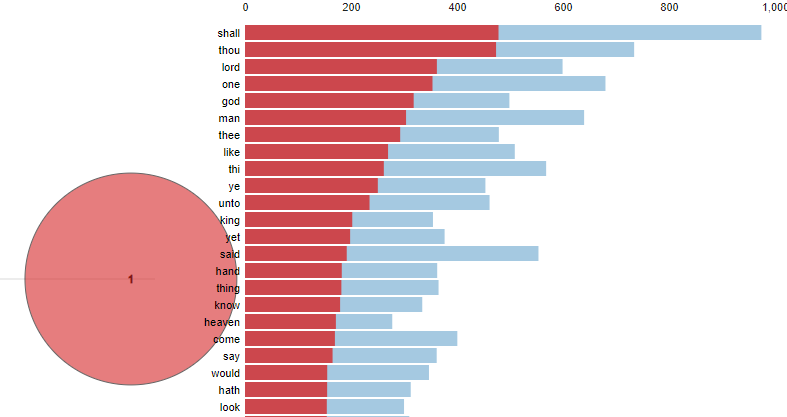


Figure - Top words of importance for topic one

This is a list of the top important words in the documents, The blue line shows the total number of the word in the whole document and the red part indicates how much the first topic takes from the number of occurrences of the word.

We can notice that for topic one, the percentage it takes from the frequency of the word is nearly half.

Now plotting the second topic.

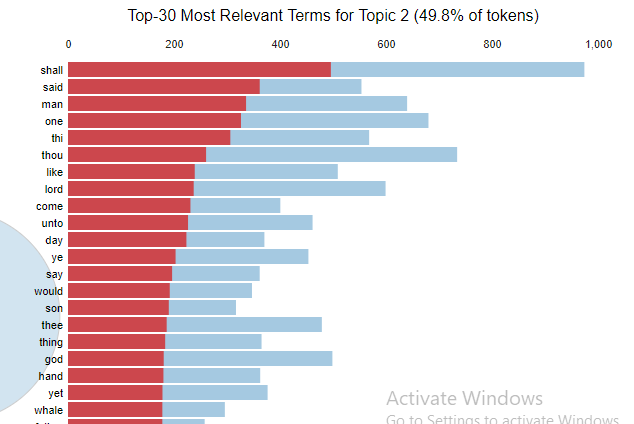
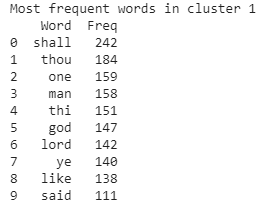
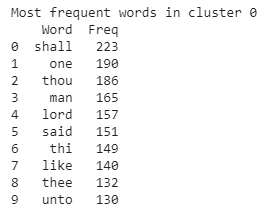


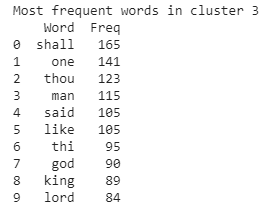
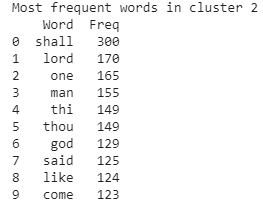
Figure - Top words of importance for topic two

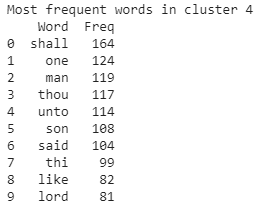
Here we can see nearly the same exact words as the first topic, and also the second topic takes about half of the frequency of these words.

So this can explain why the machine is unable to cluster the data in an efficient way, and this part explains why we obtained a very low silhouette score on all of the models. Since the silhouette score measures how similar an object is to its own cluster compared to other clusters, the low value indicates that the data points has more similarity with the other clusters than it should, or has equal similarity with most clusters, meaning it can't confirm that it belongs to the correct cluster.

To gain more insight on this we obtained the top 10 words for each cluster in the wrongly clustered data points.







We can see that the words "shall", "one", "thou", "lord", "said", etc are repeated over all clusters. And we can see that these words are the same we obtained from the top frequent words over all documents.

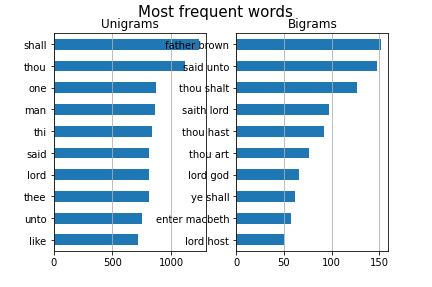


Figure - Top frequent words in all documents

And the words obtained from the topic modeling using LDA plot as well, and this takes us to the main idea of choosing the books, where they are very different in terms of genre, content and semantically, but given that they revolve around the bible in different ways like writing styles or analogies and quoting from it, this causes the data to become very similar.

Another test we made using doc2vec, we compared some random documents from the bible and moby dick, and between the bible and brown books.

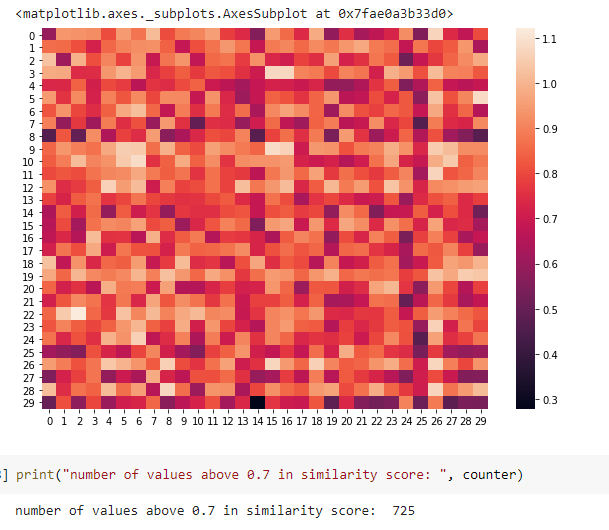
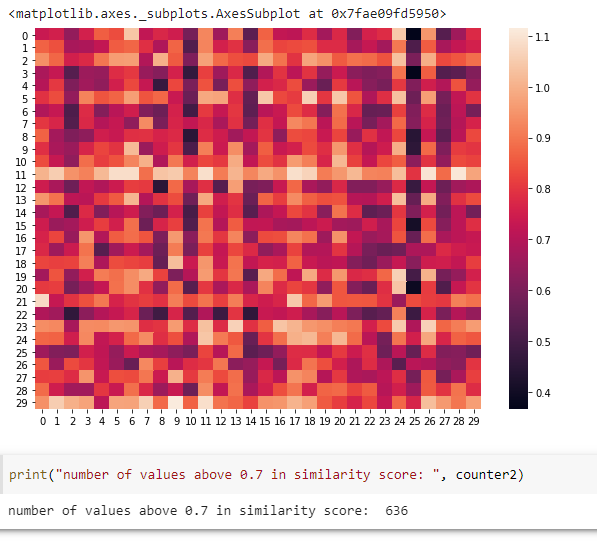


Figure - camparing documents bible and moby dick

And we printed the number of documents having a similarity more than 0.7, which were 725 out of 900 books



And as for the bible and brown books, the number of similar documents were 636 documents.

### REFERENCES

<https://stats.stackexchange.com/questions/375062/how-does-topic-coherence-score-in-lda-intuitively-makes-sense>

<https://machinelearningmastery.com/develop-word-embeddings-python-gensim/>

<https://cfss.uchicago.edu/notes/topic-modeling/>

<https://dylancastillo.co/nlp-snippets-cluster-documents-using-word2vec/>

<https://thinkinfi.com/gensim-doc2vec-python-implementation/>

# 