Data Mining: Understanding Data

Student ID: 29845998
MSc Artificial Intelligence

***Abstract*—Our task was to investigate 24 Antiquity related documents, which had been scanned into HTML format, with the aim of exploring/discovering how they relate to one another. We did this through a series of pre-processing steps and initial investigation of the data using K-Means. We then refine our data with LSA and use continue to explore the data with K-means, multidimensional scaling and hierarchical clustering.**

I. Pre-processing

Our first objective was to decide how to represent the 24 books currently in HTML format. Each book is split into individual HTML files (one for each page), which are all stored in the same folder. We converted each book to a text file using the BeautifulSoup library in Python. Now the books are consolidated in one place, examining the data and applying TF-IDF is much easier. We chose TF-IDF as it offers a more efficient way to investigate these large documents and allows us to use powerful tools, such as those provided by the Sklearn library.

To investigate the data further we used Sklearns' TfidfVectorizer (with stop words removed) on each document to create our sparse Matrix, X, and used the get_feature_names() method to see the features (the words). Immediately we can see that a lot of words have been wrongly transcribed and therefore unnecessary increase the number of columns in the matrix. For instance, words with leading numbers, numbers in the middle of words, and words with nonsensical characters (e.g. contain leading underscores). Clearly we have to remove the nonsensical characters but we also decided to remove all numbers, which can easily be done with a simple script. However, this does potentially risk losing other information such as dates. Now, words with leading numbers can now be considered properly. Of course, this does not help the words which have had the numbers in them removed, but there does not seem to be an easy way around this other ran running a auto spell correction on the texts. This has its own complications however, as we would have to filter for English and Greek (possibly other) words due to the contents of the documents.

II. Initial Investigation

First, we looked at the highest and lowest scoring TF-IDF words from TfidfVectorizer (with stop words removed). The top scoring words make sense in the context of the books being about classical history, they include "herod", "jews", "great", "king", "men". The bottom scoring words on the other hand are just wrongly transcribed words, mainly two words joined together, like "multiplythe" and "multitudeand". To reduce these words (which do not help the model) we can add a minimum document frequency cut off, df-min. Following on from this we tried K-means on the sparse matrix $(24x377441)$ to see if any further complications would be found. We could of used multidimensional scaling but as there is likely to be a lot of noise we believed it would not give us useful results. We chose our value of K by creating an elbow plot'. However the plot had quite a smooth curve so this might mean that k-means is not a good way of investigating the data, or the data is too noisy for K-means to work properly. There did seem to be a slight 'elbow' at cluster = 9. Curiously, one cluster only contained one book ,book 5. On inspection of its top scoring TF-IDF words, it became clear that the text has been incorrectly transcribed from the original. Words include: 'fenate', 'firft', 'fome', 'flate' and so on. Clearly, S's have been replaced with F's. In fact, the first real word 'commons' comes in at 14th place. Investigating further, we used Sklearn's cosine_similarity and found that, as expected, book 5 is not like any other book, as all its similarity scores are below 0.35. We expected to see book 5 ('The History of Rome') to be grouped in with other books on the topic. To correct this we decided to manually change the top 15 words to what they are most likely to be, we decide this by briefly looking at the first few instances in which the words appear in the document and their context. For instance, 'fome' is meant to be 'some' but not Rome, as the uppercase must have meant it got copied correctly. This will hopefully allow more relevant words to be weighted more.

Looking at the top scoring words for other documents we see a similar issue, only now the words are two separate words joined together. As we could not think of a good way of dealing with this we just hope that other documents have similar words. Although, we have began investigating the documents, this has allowed us to catch some issues early before getting onto the main investigation, this could be class as pre-processing.

III. Investigation

Now we know what to avoid we have set a min_df score of 0.1 on TfidfVectorizer after experimenting with a few values to see at which point the lowest scoring words mostly make sense. The features have been reduced considerably, from $\approx 370,000$ to $\approx 43,000$. Book 5 is much better now, for example the top 3 words are: "commons", "tribunes" and "city". Also, words which were sometimes misspelt, such as 'senate' (previously "fenate"), now appear in the top 15.

We used Sklearn's TruncatedSVD to reduce the dimensions further, as the term-document matrix is still noisy. The Sklearn documentation recommends using $n = 100$ for the number of components for LSA, but by summing over the variance ratios for each document we found $n = 15$ to be sufficient. Then

used LSA.fit_transform(X) to get a new X which can be used on K-means. Based on the graph below we found 8 to be a good cluster size, but it's still not a definite 'elbow' which probably means that K-means isn't a good way of grouping the data. This could be due to how similar the documents are to each other in terms of topic.

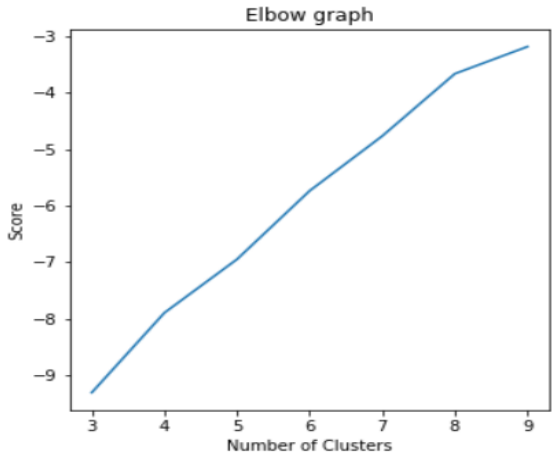

Fig. 1. K-Means for different cluster sizes against their corresponding scores

The 8 clusters consisted of 4 medium (4-6 documents), 2 small (2-3 documents) and two singular clusters which contained books 20 and 22. To see whats happening here we looked at the top words for each cluster. Book 20 (Natural History by Pliny the Elder) consists of words like 'gold', 'silver' and 'rings'. While Book 22 (Histories by Tacitus) has a lot of unique smaller words which is probably what is keeping it separate, such as 'cf' , 'et' , 'sc' and 'note'. These clusters and their top scoring words can be seen visually in Figure 2. This figure is created using Sklearn's MDS (Multidimensional Scaling). MDS allows us to get down to two dimensions so we can easily see the clusters, there is of course a loss of information from doing this. We used cosine distance because we care more about the similarity than the euclidean distance of the documents.

Most of the clusters groupings make sense, however, the documents around the dark blue cluster do not. Meaning that they must be grouped closer in higher dimensions. As there is not a clearly defined set of clusters we can see why the elbow curve was quite smooth. Of course, some information would be lost in the reduction but it is still a good indicator.

To get a better understanding of the relation between the groups we used scipy.cluster.hierarchy.linkage which is a agglomerative clustering algorithm. Specifically, we used the minimum (single linkage) clustering as it tends to give long thin clusters which will likely fit our data better, judging from Figure 2. The documents have been put into similar groups like before, as we are using the cosine distance again. Interestingly though, the dark blue cluster from Figure 2 (Book16 et al), and its 2-dimensional neighbours are grouped together very late in the process despite being close in 2D. This would suggest that the 2D representation is not good and is missing a lot of information.

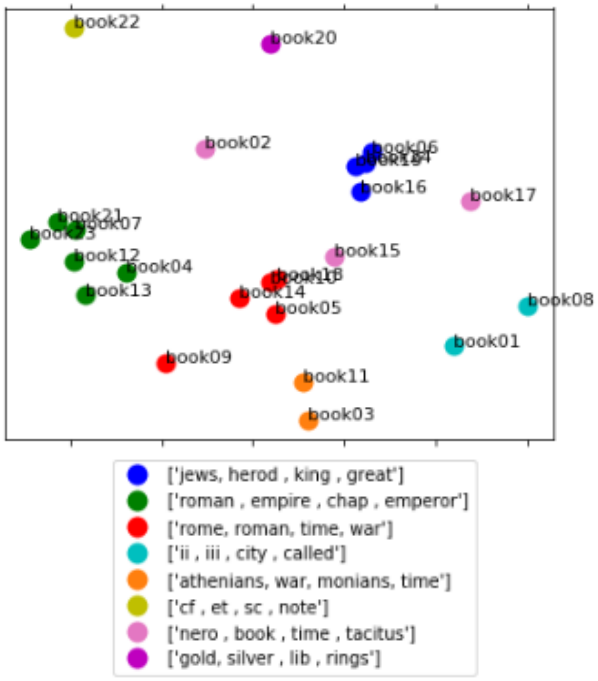

Fig. 2. The 8 clusters represented in 2D space after applying multidimensional scaling using cosine distance. Along with the top scoring words for each cluster and their colours.

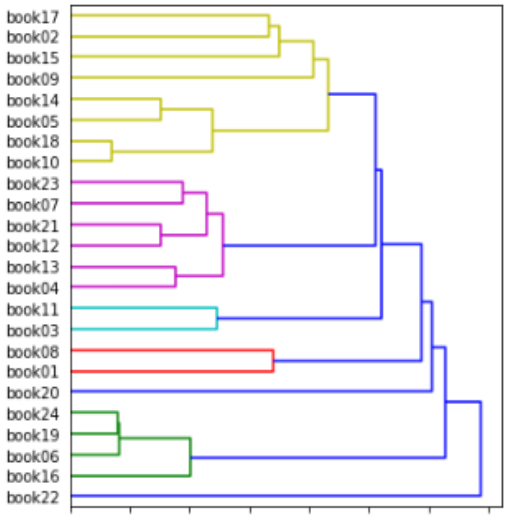

Fig. 3. Hierarchical Clustering with single linkage clustering using cosine distance

IV. Future Works

In the future we would like to use n-grams on the data to gain a better understanding of the structure of the documents. This is because TF-IDF is not good enough on its own as the documents are describing similar things but in different ways. For instance many documents frequently mention "Rome" but are discussing them in different contexts. Also, we would like to try and create a decision tree for the clusters to see at which words the data splits. This was attempted but could not get it working.