Data Mining Coursework 2

|  |  |  |
| --- | --- | --- |
|  | Daniel Lee University of Southampton - Electronics and Computer Science Southampton UK dl2g14@ecs.soton.ac.uk |  |

ABSTRACT

This report demonstrates how two clustering algorithms, K-Means Clustering and Hierarchical Document Clustering, using Ward Clustering, can be applied to antiquity text documents to group them according to how similar their contents are. Moreover, multidimensional scaling was also used to visualize the similarities of the documents, in the form of *cosine distances*, in a lower dimension (e.g. 2-D). The results show that these algorithms are coherent to each other, but convey the information in different ways.

1 DATA PRE-PROCESSING

1.1 Data Extraction

The first step that was carried out to apply these algorithms to the data was the pre-processing. The data was programmatically extracted from the root directory using the OS module in Python. The pages for each book were then sequentially concatenated and stored into document variables named after their folder names. These variables were used to build a single 2-D matrix variable, with the rows indicating the document names and the columns, their respective raw concatenated text.

1.2 Special Characters

After the data was organized and stored as described above, a loop was used to go over all the text contained in the 2-D matrix. For each of the texts, all the special characters were replaced with spaces using the re.subfunction with [^A-Z^a-z]+, as the input argument. This was done to avoid any parsing issues in the tokenization of the text.

1.3 Stop Words

It is very common for texts to have a high relative frequency of stop words. These types of words were assumed to not convey any essential meaning, and therefore, were filtered out from the texts to avoid any distortion in the analysis.

1.4 Stemming and Tokenizing

To reduce the computational overhead, as well as some semantic redundancy, the words were stemmed and grouped accordingly. These words were then tokenized. Lastly, to further reduce overhead, the words with less than three characters were removed with the assumption that they would not have a notable impact.

2 K-MEANS CLUSTERING

2.1 Vectorization with tf-idf

In order to be able to apply K-Means Clustering to a dataset, it is necessary to transform it in the form of vectors. To do this, a matrix containing the tokenized words and their counts in each of the documents was made. This was then used to compute the tf-idf score for each of the words and for each of the documents. The effect of applying tf-idf to the words’ counts, is that it amplifies the weights to those words that appear in less documents, and vice versa, in order to mitigate the relevance of general terms. Moreover, to filter out more of the less important words, the words with a document frequency of less than 10% and higher than 70%, were removed. Due to the reason that the number of tokenized words was excessive (> 4.5 million), the number of words, or dimensions, in the vector form, was limited to 200,000. These set of constraints and parameters were applied through the TfidVectorizer class from the sklearn library, which was used to produce the tf-idf matrix.

2.2 Clustering

With the data in vector form, it was now possible to apply the K-Means Clustering algorithm. This algorithm is used to split the data into an arbitrary number of groups, or clusters, by recursively shifting the means of the clusters, to the new mean of their respective data points. In this case, a number of 5 clusters was used to classify the documents with the use of the KMeans function.

3 HIERARCHICAL DOCUMENT CLUSTERING

3.1 Ward Clustering

The core concept of the hierarchical document clustering algorithm, is that all the data points (e.g. documents) are initialized as clusters of their own and in every iteration, these clusters are grouped in pairs, based on the minimization of an objective function (e.g. MSE), to form a higher a level cluster, until one cluster remains. In the Ward Clustering algorithm, the pair clustering is performed based on the criteria of minimizing the variance computed between each of these data points. This is known as the Ward’s minimum variance criterion [1].

3.2 Implementation

This algorithm was implemented by using the cosine similarity matrix. This matrix contains similarity scores between each of the documents, based on the cosine distances of the data points. This was obtained by using the cosine\_similarity function to the tf-idf matrix, previously generated.

4 MULTIDIMENSIONAL SCALING

The purpose of using multidimensional scaling is to graphically illustrate the similarities between objects of a dataset of N-dimensions in a lower dimension (e.g. 2-D). This can be achieved by using a matrix of the cosine distances between each data object, as the one generated in Section 3. This matrix was used as an input to the *MDS* function.

5 RESULTS

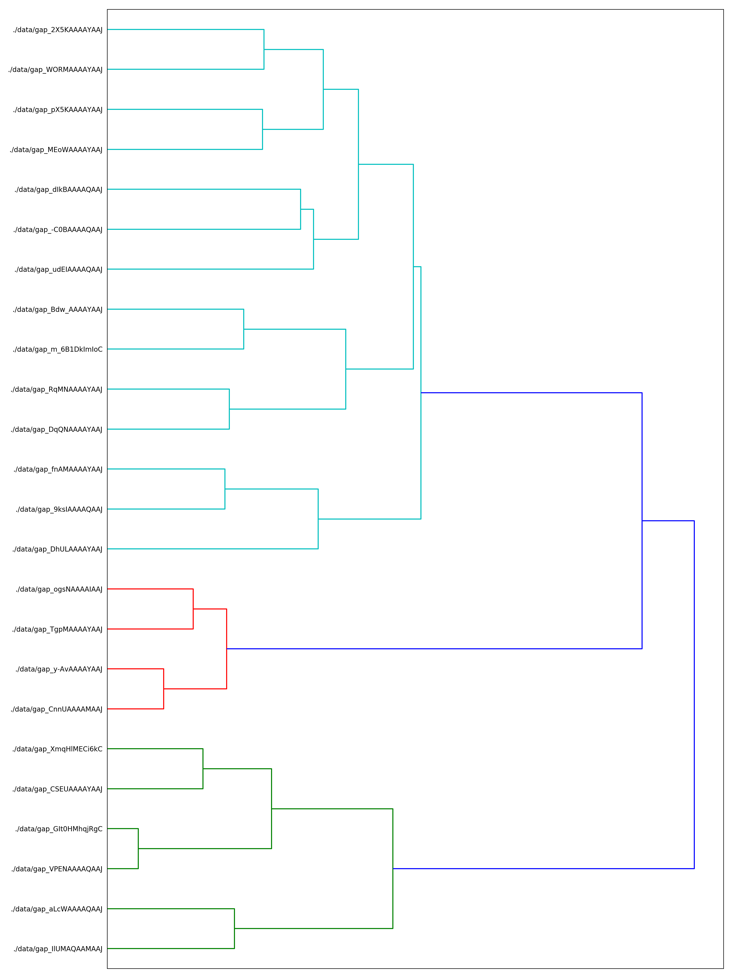
5.1 K-Means

The obtained results for the K-Means algorithm are shown below. The 6 words closest to the centroids of the clusters were also collected from the results to show a better insight of the clusters’ theme.

|  |  |  |
| --- | --- | --- |
| Cluster | Documents | Keywords |
| 1 | DhULAAAAYAAJ, 9ksIAAAAQAAJ, fnAMAAAAYAAJ, | Peloponnesians, monians, fays,  Lacedaemonians, hath, Thebans |
| 2 | MEoWAAAAYAAJ, pX5KAAAAYAAJ,  WORMAAAAYAAJ, 2X5KAAAAYAAJ | Nero, Tacitus, Germanicus, Otho, Vitellius, Galba |
| 3 | IlUMAQAAMAAJ, CSEUAAAAYAAJ,  VPENAAAAQAAJ, XmqHlMECi6kC,  GIt0HMhqjRgC, aLcWAAAAqAAJ | Justinian, lib, Christian, Constantinople, Belisarius, Constantine |
| 4 | TgpMAAAAYAAJ, CnnUAAAAMAAJ,  ogsNAAAIAAJ, y-AvAAAYAAJ | JEWS, herod, Josephus, Jerusalem, Hyrcanus, Judea |
| 5 | udEIAAAAQAAJ, DqQNAAAAYAAJ,  -C0BAAAAQAAJ, RqMNAAAAYAAJ,  m\_6B1DkImIoC, Bdw\_AAAAYAAJ,  dIkBAAAAQAAJ | Samnites, Lucius, Sulla, Carthaginian, Hannibal, Scipio |

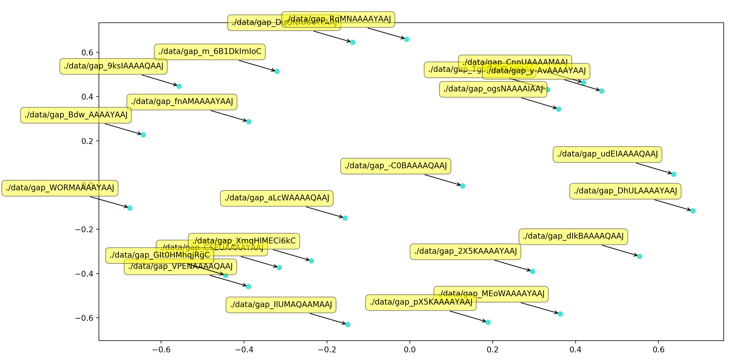
5.2 Hierarchical Document Clustering

To illustrate the results of the hierarchical document cluster, a dendrogram was plotted, as shown below. By comparing the clusters obtained by K-Means and the clusters of the dendogram, it can be observed that the K-Means clusters are also grouped in the same higher-level clusters of the dendogram. The advantage of using a dendogram however, is that allows us to see which of the documents in a cluster are closer together, by following down the roots. Moreover, by comparing these results with the multidimensional scaling results, in Section 5.3, we can observe that the multidimensional scaling algorithm groups the documents that are in the same K-Means clusters, spatially closer together, meaning that all of the algorithms are portraying the same information.



5.3 Multidimensional Scaling

The 2-D plot that was produced from applying multidimensional scaling using the cosine distance matrix, is shown below.



6 CONCLUSIONS

In summary, by looking at the results, we can see how all the algorithms are coherent with each other. However, each algorithm has a unique way of showing different insights of the data, such as, the themes of the data and the pairwise similarities.

ACKNOWLEDGMENTS

Main source: http://brandonrose.org/clustering

CODE

Github: