MONASH University

# FIT3152 Assignment 3

Semester 1 - 2024

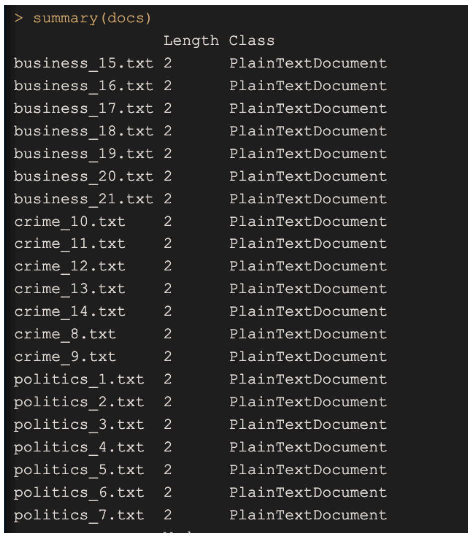
Ke Er Ang | 32581343

**Task 1**
First and foremost, I have identified my primary interest area and selected topics that would form the basis of my analysis. I have collected a set of documents, specifically a set of news stories for the analysis. The news stories collected were obtained from 9news, an established news agency recognised for its extensive coverage of a wide range of events happening around the world. The documents that I have chosen focus on 3 distinct topics, namely business, politics and crime. A total of 21 stories are picked, 7 stories from one topical area to ensure a diverse and equal representation of the chosen topics. Business articles mainly talked about the trend of market, corporate news and economic policies; Crime articles report on criminal cases ;Political articles discuss government decisions or any other political events.

**Task 2**
The first step of creating the Corpus is to compile the necessary documents. I created a Google Drive folder to hold all the news stories I have chosen. Each story was copied from the news article online and pasted in a separate Google Docs. To make the clustering and network graph visualisations process more efficient, a systematic naming convention is adopted for the documents. Each document was named based on its topic and a distinct sequential identifier. For business-related documents, they are labelled as ‘business_x’ ; for crime-related documents, they are labelled as ‘crime_x’; for politics-related documents, they are labelled as ‘politics_x’. Once the documents are properly organised in the folder, I exported all of them as plain text files (.txt) by using the “Download as” function.These files were then stored in a specifically named directory, Asgn3, located within the FIT3152 folder on my Desktop.

Next, I set the working directory in R to the location of these files by calling the code setwd("/Users/chloeang/Desktop/FIT3152"). Then creates a path that points to a directory named "Asgn3" within the current working directory by calling the code ‘cname = file.path(".", "Asgn3")’. Last, to create a text corpus object, I called the code ‘docs= Corpus(DirSource(cname))’. To verify the content, I ran ‘summary(docs)’ to inspect the contents and ensured all documents were successfully included in the correct format within the corpus. The picture below shows a snippet of the Corpus object (see Figure 2.1)

Figure 2.1 Summary of the Corpus object


```
> summary(docs)
                Length Class
business_15.txt 2      PlainTextDocument
business_16.txt 2      PlainTextDocument
business_17.txt 2      PlainTextDocument
business_18.txt 2      PlainTextDocument
business_19.txt 2      PlainTextDocument
business_20.txt 2      PlainTextDocument
business_21.txt 2      PlainTextDocument
crime_10.txt    2      PlainTextDocument
crime_11.txt    2      PlainTextDocument
crime_12.txt    2      PlainTextDocument
crime_13.txt    2      PlainTextDocument
crime_14.txt    2      PlainTextDocument
crime_8.txt     2      PlainTextDocument
crime_9.txt     2      PlainTextDocument
politics_1.txt  2      PlainTextDocument
politics_2.txt  2      PlainTextDocument
politics_3.txt  2      PlainTextDocument
politics_4.txt  2      PlainTextDocument
politics_5.txt  2      PlainTextDocument
politics_6.txt  2      PlainTextDocument
politics_7.txt  2      PlainTextDocument
```

**Task 3**

In order to improve the quality and usefulness of the data for future analysis, I did some processing for the documents. First, the corpus was tokenized, which involved removing digits and punctuation. Moreover, all text is converted to lowercase to prevent duplicate tokens that are merely different in their capitalisation. This normalisation technique is not only critical for minimising complexity in text data, but also ensures the text is in a consistent and standard format. Besides, the common stopwords from the English language, such as “a”, “and”, “but” which have little to no value in identifying document topics were also removed. Furthermore, certain text transformations were executed to remove non-printable letters and various dashes (- and —), which are frequently artefacts of text formatting and do not add to semantic analysis. These characters were substituted with spaces to prevent unintentional word concatenation.

On top of that, some targeted elimination also performed on the corpus to remove those frequently used but low-value words including “also”, “will”, “may”, “since”, “just” , “can”, “like”, as these words do not improve the identification of documents. Some words that are common conversational phrases and not directly related to the any of the selected topics are also removed which are “say”, “said”, “around”, “back”, “told”, “year”,“show”, “see”, “take”, “know”, “remain”, “get”, “make”, “made”. Some words that are more neutral and could fit in any 3 categories are also removed, such as “one”, “two”, “three”, “per”, “cent”,“become”, “increase”. By doing these steps, data noise is reduced and attention is drawn on the more significant terms that are more likely to distinguish between texts

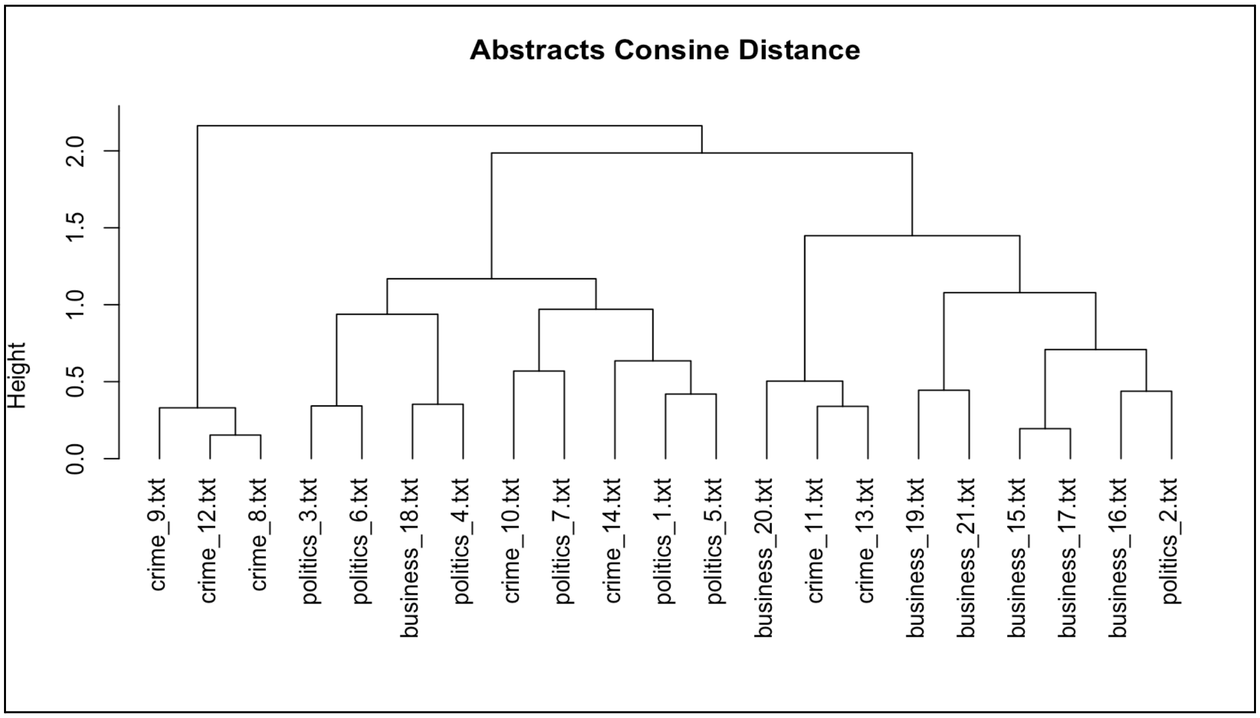
Additionally, the variation of the form of a verb and pluralisation of a word will also affect the text analysis, to solve these issues, stemming was performed to simplify the words to their root form and group all variations of a word together. Doing this step will help to improve the accuracy of the analysis as it consolidates word count and reduces redundancy.

Finally, after completing the text modifications, the Document-Term Matrix was created. The DTM began with a large number of tokens, and after a thorough process of trial and error of removing sparse terms, words that appeared at least 25% are retained. This approach successfully filters out phrases that provide little value to the analysis and the final DTM was pruned to only focus on the most related 27 tokens.

**Task 4**

In the next step of the analysis, hierarchical clustering has been applied to the Corpus using cosine distance to better understand the classification of documents based on the topics of business, crime and politics. Figure 4.1 shows the resulting cluster dendrogram.

Figure 4.1 Cluster Dendrogram



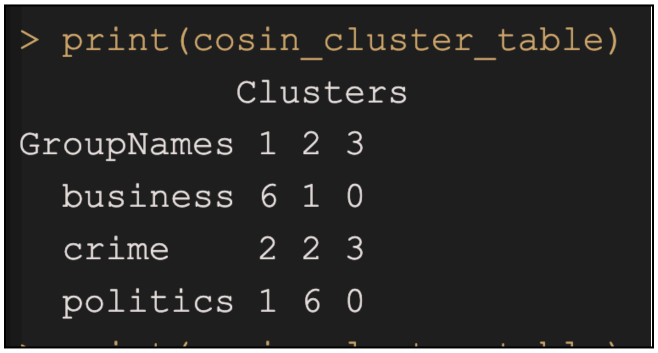
The dendrogram classified the documents into 3 main clusters.

Cluster 1 is the smallest branch, containing only 3 documents, but they are all crime-related documents, including crime_9.txt, crime_12.txt and crime_8.txt. In this branch, crime_12.txt and crime_8 are grouped together, indicating a higher similarity between these two documents compared to crime_9.txt.

Cluster 2 comprises politics-related and some other-related documents. Among all, there are 6 political-related documents, 2 crime-related documents and solely 1 business-related document, totaling 9 documents. These clusterings show that these domains have some interconnections between each other in the real-world. These documents might include discussions about rules, regulations, and government. For instance, the decisions of politicians might bring some impacts on the legal system, on the other hand, lawsuits might also lead to political discussion. These might explain the rationale behind the clustering of these documents.

Cluster 3 has the same amount of documents as Cluster 2, also containing 9 documents. This cluster is primarily made up of business-related documents, with 6 of them. Additionally , the other 3 documents are 2 crime-related documents and 1 political-related document. This suggests they might have shared themes, for example, the crime-related document might discuss some white-collar crime, economic crime such as money laundering, bribery, fraud. Also, the political document might talk about something related to the business or economic policy which causes all of them to be put under the business branch. Although the business-related document (business_20.txt) is in the same branch as the other 2 crime-related documents, it is still somewhat isolated from them. This indicates that although 3 of them might have some topical overlap but still have some distinct differences.

To evaluate the effectiveness of the clustering, a vector of topic labels has been created according to the order of the Corpus. Next, the dendrogram is divided into 3 clusters by calling the code 'group = cutree(fit, k=3)', 'fit' represents the hierarchical clustering model and 'k=3' specifies splitting the dendrogram into 3 clusters . A table is created to cross tabulate the true topic labels against the cluster numbers assigned by the clustering algorithm.



```
> print(cosin_cluster_table)
          Clusters
GroupNames 1 2 3
  business 6 1 0
  crime    2 2 3
  politics 1 6 0
```

Each document is assigned to the cluster having the largest number of members related to its specific true topic. It is presumed that the cluster containing the most documents for a specific true topic is the one that correctly clustered those documents. In this case, business is assigned to Cluster 1, crime to Cluster 3, and politics to Cluster 2.

The 6 documents in both Cluster 1 and Cluster 2, as well as the 3 documents in Cluster 3, are all assumed to be true positives. The accuracy is then calculated by summing the number of correct assignments and dividing by the total number of documents, specifically: (6+6+3)/ 21 = 0.714. This suggests that the method classified 71.4% of documents accurately which highlights the effectiveness of the clustering algorithm while also outlining there is room for improvement.

In particular, 4 out of 7 crime-related documents are classified wrongly, indicating the algorithm has some difficulty in differentiating crime-related documents from other documents in the corpus. Crime-related documents may share a lot of terminology with other topics and the DTM used may not be effective enough to capture unique aspects of crime-related documents.

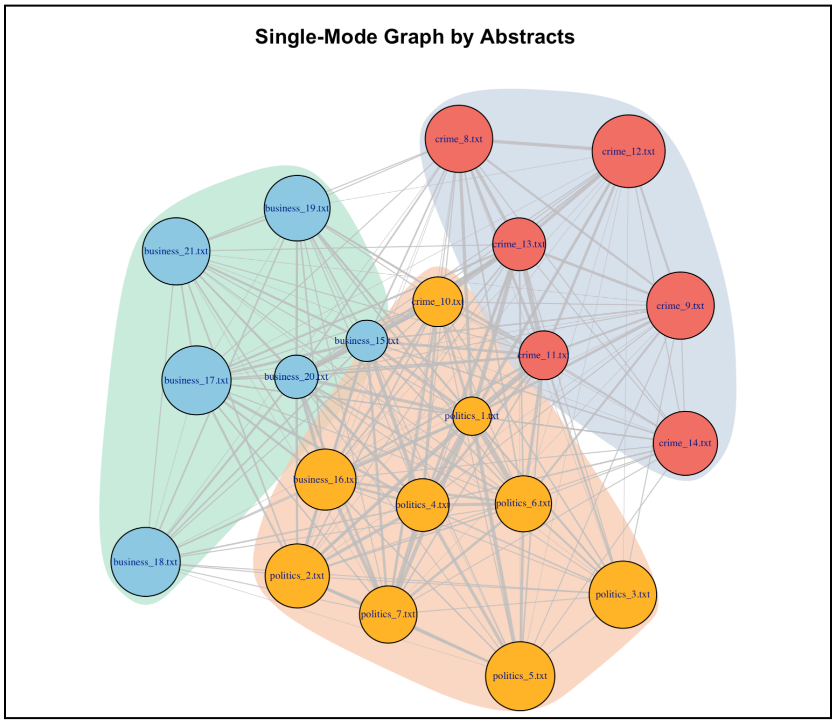
However, only 1 out of 7 business and politics-related documents is classified wrongly, suggesting the algorithm is able to identify documents from these two topics accurately.

**Task 5**

A single-mode graph has been created to showcase the relationships between the documents according to the quantity of terms they shared. The first step to do this is to transform the DTM created into a binary matrix. Next, the binary matrix is multiplied by its transpose. With the binary transformation, the analysis is made simpler, as it concentrates on whether or not the document has a shared term rather than how frequently they do so. Following, the leading diagonal is set to zero to avoid self-loop and make the analysis to focus on meaningful relationships between documents.

Additionally, to further improve the visualisation of graphs to extract more insight, the edge width is mapped to the number of terms shared in the documents. The clusters within the graph are identified by using the Louvain algorithm, with the same node belonging to the same cluster having the same colour. The algorithm is picked for the analysis due to its efficiency and efficacy in identifying communities within an extensive network. It maximises modularity, which measures the density of connections inside communities compared to those between communities. The nodes within the cluster are more densely connected to one another than to the nodes outside of their cluster, creating a strong community structure. This, in turn, discloses the underlying structure of the network.

Figure 5.1 Single-mode network by abstracts



In figure 5.1, there are 3 key elements to be noted: the colour of the nodes, the size of the nodes, and the thickness of the edges between nodes.

This network effectively visualises the connections between documents based on shared terms. The graph clearly illustrates three distinct clusters, each represented by a different colour (yellow, red, blue). The documents that share more terms are grouped in the same cluster. The nodes in yellow are mostly politics-related documents; the blue nodes exclusively contain business-related documents while the red node consists of all crime-related documents. However, in the yellow cluster, the inclusion of 'business_16.txt' and 'crime_10.txt' suggests a minor misclassification.

Compared to the hierarchical clustering, this single-mode graph shows a significant improvement in document differentiation based on shared terms. In the hierarchical clustering , there are 6 documents misclassified, however in this network, there are only 2 documents misclassified, which shows a notable increase in accuracy in distinguishing across the various documents.

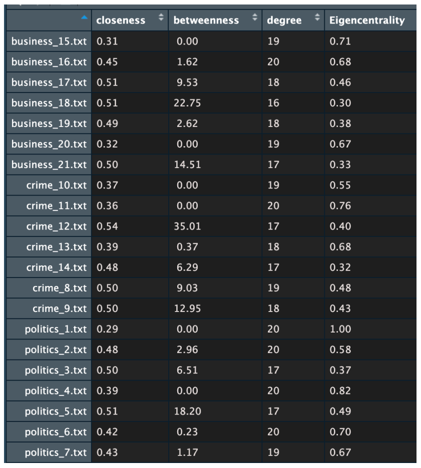
The size of the node shows the closeness centrality of the document. Larger size of the node indicates that the document has higher value of closeness centrality, meaning that the node is closer to all other nodes. The high closeness centrality makes them more important in the analysis as this shows they might contain key terms that are common across various documents. In contrast, smaller size nodes indicate they have lower closeness centrality, which means they are more specialised or concentrating on a topic that are not widely discussed in other documents.

The size of the edge shows the strength of connection between the documents, based on the number of shared terms. Ticker edge indicates more shared term between them, while thinner edge indicates less shared term between them. For example, the edge connecting 'politics_5.txt' and 'politics_7.txt' is relatively thicker, indicating a higher number of shared terms between these documents. On the other hand, the thinner edge, such as the one between 'business_18.txt' and 'business_17.txt' indicates these documents share fewer terms.

Figure 5.2 below shows the statistics of the centrality measures of the documents. Starting from closeness centrality, 'crime_12.txt' (0.54) is observed to have the highest value among all the documents,making it the most central document in the graph. Conversely, 'politics_1.txt' (0.29) has the lowest centrality closeness, suggesting it is more isolated from other nodes. These statistics are consistent with the size of nodes shown in the graph above, as the node of 'crime_12.txt' is one of the largest, reflecting  its higher centrality and the node of 'politics_1.txt' is the smallest, signifying its low centrality.

Furthermore, documents like 'business_18.txt' (22.75) , 'crime_12.txt' (35.01) and 'politics_5.txt' have high betweenness centrality, indicating they serve as crucial connectors within the network. Besides, degree centrality counts the number of connections a term has, high value of degree means the documents are directly connected to many other documents, such as 'business_16.txt' (20) , 'crime_11.txt', 'politics_1.txt' etc. Also, documents which have high eigenvector centrality suggesting they are linked to other highly influential documents, these documents including 'politics_1.txt' (1.00), 'politics_4.txt' (0.82), and 'crime_11.txt' (0.76).

Figure 5.2 Centrality Measure (betweenness, closeness, degree, eigenvector centrality)



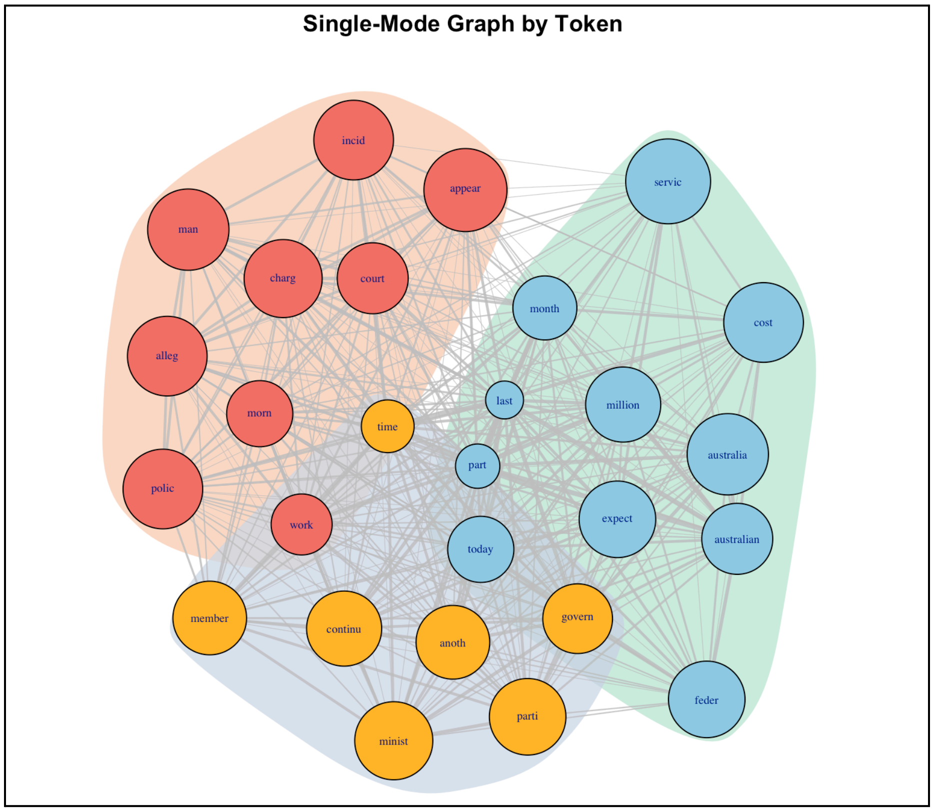
| | closeness | betweenness | degree | Eigencentrality |
|---|---|---|---|---|
| business_15.txt | 0.31 | 0.00 | 19 | 0.71 |
| business_16.txt | 0.45 | 1.62 | 20 | 0.68 |
| business_17.txt | 0.51 | 9.53 | 18 | 0.46 |
| business_18.txt | 0.51 | 22.75 | 16 | 0.30 |
| business_19.txt | 0.49 | 2.62 | 18 | 0.38 |
| business_20.txt | 0.32 | 0.00 | 19 | 0.67 |
| business_21.txt | 0.50 | 14.51 | 17 | 0.33 |
| crime_10.txt | 0.37 | 0.00 | 19 | 0.55 |
| crime_11.txt | 0.36 | 0.00 | 20 | 0.76 |
| crime_12.txt | 0.54 | 35.01 | 17 | 0.40 |
| crime_13.txt | 0.39 | 0.37 | 18 | 0.68 |
| crime_14.txt | 0.48 | 6.29 | 17 | 0.32 |
| crime_8.txt | 0.50 | 9.03 | 19 | 0.48 |
| crime_9.txt | 0.50 | 12.95 | 18 | 0.43 |
| politics_1.txt | 0.29 | 0.00 | 20 | 1.00 |
| politics_2.txt | 0.48 | 2.96 | 20 | 0.58 |
| politics_3.txt | 0.50 | 6.51 | 17 | 0.37 |
| politics_4.txt | 0.39 | 0.00 | 20 | 0.82 |
| politics_5.txt | 0.51 | 18.20 | 17 | 0.49 |
| politics_6.txt | 0.42 | 0.23 | 20 | 0.70 |
| politics_7.txt | 0.43 | 1.17 | 19 | 0.67 |

## **Task 6**

To investigate how often a term is co-occur in the documents, a single-mode graph by token is constructed. The first step to create the graph is converting the DTM into a matrix form, then this frequency matrix is converted into a binary matrix, with 1 denoting the term's existence and 0 denoting its absence in the document. Subsequently, the co-occurrence matrix is created and the leading diagonal is set to 0 to prevent self-loop.

Similar to the previous graph, the graph has been improved to extract more information. First of all, this graph is also divided into clusters by using the Louvain algorithm, with the same nodes in the same cluster having the same colour. Also, the edge is mapped to the frequency of co-occurrence of the terms in the documents. A single-mode graph that visualises the relationship between tokens is shown below.

Figure 6.1 Single-mode network by token



In the graph shown above, there are 3 key elements that need to be noted: the node colour, the node size and the thickness of the edge between the nodes.

The graph displays 3 distinct clusters with different node colours for each cluster. These clusters correspond to groups of terms that appear frequently together in the documents. The yellow node cluster contains more term related to politics, such as "govern", "parti", "minist"; the red node cluster contains more crime related term, such as "incid", "polic"; while the blue node cluster contains more business-related terms, such as "million", "cost", "servic".

The centrality of closeness is shown by the size of the node, larger size means the node is more central and can spread information more efficiently which makes it more important. In contrast, a smaller size node means it is less central in the graph, hence less important in the analysis. For example, the term ‘last’ and ‘part’ have the smallest size of node among all, meaning they are less central and less frequently covered in the documents. The term ‘polic’, ‘minist’, ‘million’ have large node size, meaning that these terms may be widely used, and revealing its significance to the major topics covered in the document.

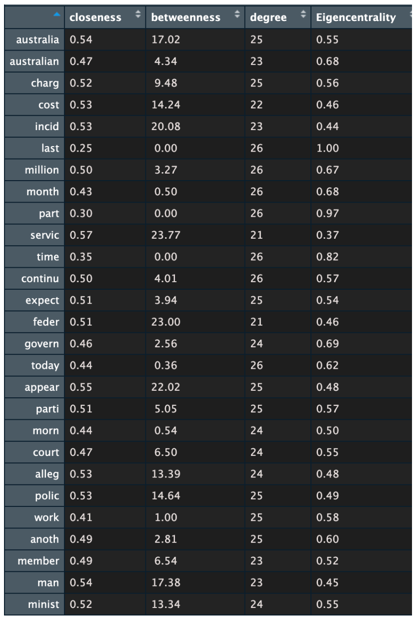
The thickness of the edge between the nodes reflects the frequency of the terms co-occur in the documents. Thinner edge indicates the terms are less likely to exist in the same document or less often to be discussed together. On the other hand, thicker terms mean the terms are closely related or frequently discussed together in the documents. For example, ‘court’ and ‘alleg’ have a thicker edge between them, suggesting they often appear together in the same document.

The graph below shows the value of centrality measures of the token, including betweenness, closeness, degree and eigenvector centrality (see Figure 6.2).

For closeness centrality, the token ‘part’ (0.30) has the lowest value which supports the smallest size of the node illustrated in Figure 6.1. Also, the ‘servic’ token with the highest value of 0.5652 has the largest size of node in the graph. Besides, terms such as ‘servic’(23.77), ‘appear’(22.02) and ‘feder’(23.00) have high betweenness centrality, meaning these terms act as crucial bridges in connecting other nodes in the network. Following that, terms with high degree of centrality like ‘last’ (26), ‘million’ (26) are significant nodes as this implies they are directly connected to a large number of other nodes. Additionally, terms which have high eigenvector centrality like ‘last’(1.00) and ‘part’(0.97), suggesting they are very influential in the network as they are connected to other highly influential phrases.

The low closeness centrality but high eigenvector centrality of ‘part’, suggesting that it might not play a significant role in linking the entire network but is important in certain discussion or topic in the corpus.

Figure 6.2 Centrality Measure (betweenness, closeness, degree, eigenvector centrality)

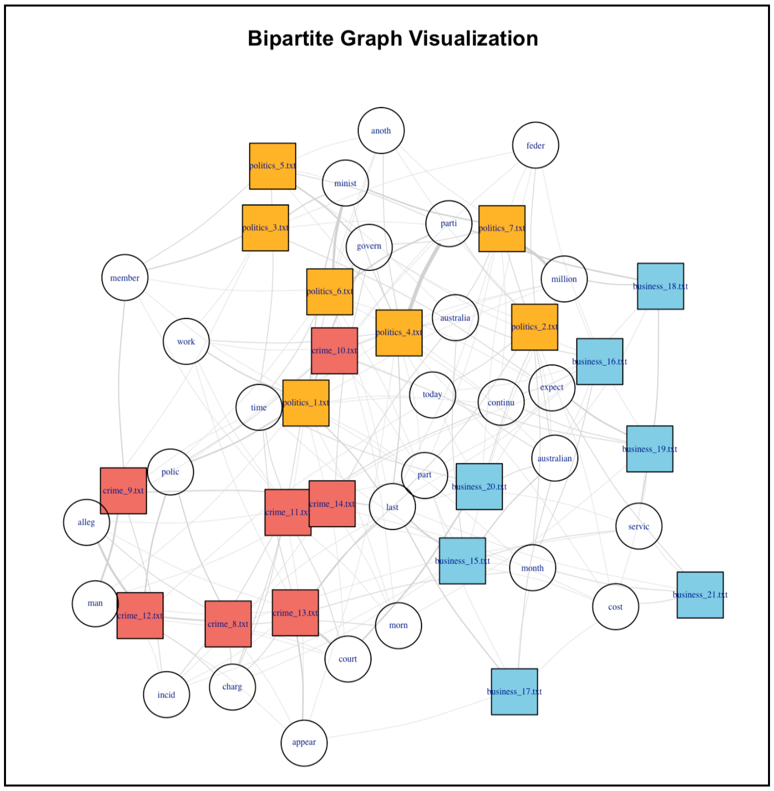


| | closeness | betweenness | degree | Eigencentrality |
|---|---|---|---|---|
| australia | 0.54 | 17.02 | 25 | 0.55 |
| australian | 0.47 | 4.34 | 23 | 0.68 |
| charg | 0.52 | 9.48 | 25 | 0.56 |
| cost | 0.53 | 14.24 | 22 | 0.46 |
| incid | 0.53 | 20.08 | 23 | 0.44 |
| last | 0.25 | 0.00 | 26 | 1.00 |
| million | 0.50 | 3.27 | 26 | 0.67 |
| month | 0.43 | 0.50 | 26 | 0.68 |
| part | 0.30 | 0.00 | 26 | 0.97 |
| servic | 0.57 | 23.77 | 21 | 0.37 |
| time | 0.35 | 0.00 | 26 | 0.82 |
| continu | 0.50 | 4.01 | 26 | 0.57 |
| expect | 0.51 | 3.94 | 25 | 0.54 |
| feder | 0.51 | 23.00 | 21 | 0.46 |
| govern | 0.46 | 2.56 | 24 | 0.69 |
| today | 0.44 | 0.36 | 26 | 0.62 |
| appear | 0.55 | 22.02 | 25 | 0.48 |
| parti | 0.51 | 5.05 | 25 | 0.57 |
| morn | 0.44 | 0.54 | 24 | 0.50 |
| court | 0.47 | 6.50 | 24 | 0.55 |
| alleg | 0.53 | 13.39 | 24 | 0.48 |
| polic | 0.53 | 14.64 | 25 | 0.49 |
| work | 0.41 | 1.00 | 25 | 0.58 |
| anoth | 0.49 | 2.81 | 25 | 0.60 |
| member | 0.49 | 6.54 | 23 | 0.52 |
| man | 0.54 | 17.38 | 23 | 0.45 |
| minist | 0.52 | 13.34 | 24 | 0.55 |

**Task 7**

The bipartite graph was constructed using the DTM to show how terms are connected to different documents. Square node represents documents, colour-coded by topic—blue for business, red for crime, and yellow for politics, and circular node represents the terms in the DTM. The edge connecting the node indicates the term present in the document.

Figure 7.1 Bipartite Graph



The graph above clearly reveals the clustering of documents, implying solid topic classification within the cluster. The blue square nodes which represents the business-related documents are tightly clustered, having multiple terms in common, like ‘cost’ and ‘servic’. Besides, the crime-related documents, which are square nodes in red, appear to be categorised together, and share common terms, such as ‘alleg’, ‘incid’, ‘man’. Similarly, political-related documents are also clustered together with some common terms, such as ‘parti’, ‘govern’.

An interesting observation is the proximity of ‘crime_10.txt’ to the political cluster, this document has always also been clustered in the political cluster in both dendrogram of hierarchy clustering and single-mode by abstract network, suggesting that this document might contain terminology that are related to political discourse. Moreover, some terms, like ‘feder’, ‘million’ , etc are linked to different topic categories, reflecting crossover topics, perhaps talking about federal concerns that have an impact on business and politics, or financial aspects influencing criminal and political aspects. These observations are important as they reflect how intricately various topics interconnect. Compared to both hierarchical clustering and single-node network, this bipartite graph does a better job in discriminating documentations as only one document is presumed to be classified wrongly.

Additionally, the size of the edge between the nodes acts as a quantitative indicator of the frequency of a term appearing in a document,  it reflects the strength of the term related to the document. Thicker edge means the terms are more frequently mentioned in the documents and thinner edge means the terms are less mentioned in the documents. In the graph above, it is obvious that the edge between the term ‘parti’ and the document ‘politics_4.txt’ appears to be relatively thicker compared to other edges, suggesting that ‘parti’ has been frequently used in that  particular document.

In this network, ‘part’(0.4947) and ‘crime_11.txt’(0.47) show high closeness centrality value. In the single-node by token network, ‘part’ has shown specific influence in the network by having high eigenvector centrality and in the single-node network, ‘crime_11.txt’ has also shown high eigenvector centrality. These nodes demonstrate high centrality values, showing their central roles in various networks underscore their significance in differentiating the documents inside the corpus.

These findings support the robustness of the network clustering approach. Unlike the single-mode networks, which primarily focus on document-to-document or term-to-term relationships based on abstract similarities or shared terms, this bipartite network offers more valuable insights into the complex relationship between terms and documents in the corpus as it shows the direct connection between them.

## **Task 8**

To further improve the text processing demonstrated, we could implement one of the text vectorisation scoring themes, namely the TF-IDF method to transform the words in a text document into an importance number through text vectorization. TF stands for Term Frequency and IDF stands for Inverse Document Frequency. So instead of using binary occurrences, this approach could be used to indicate a term's significance for a certain document inside the corpus, which could potentially enhance the topical focus of the clusters and in turn improve the accuracy of documents differentiation.

Secondly, to enhance the text analysis process, semantic term analysis can be performed to capture the semantic relationship between the terms. Semantic analysis entails comprehending and interpreting the documents in a way that accurately conveys the meaning of the terms in relation to their context. For example, “Apple is a fruit that is packed with vitamins”, the term “Apple” refers to a fruit, but in another context, it can refer to the leading technology company, “Apple Inc”. Hence, it is important to do semantic analysis and extract more information in order to correctly interpret the meaning of the terms based on their usage in a sentence.

One effective method that can perform semantic analysis is Word2Vec, which was created for word embedding training. This algorithm operates under a distributional hypothesis and is predicated on the idea that words that are close to one another in a text share semantic similarity. Word2Vec processes a text corpus by using cosine similarity to map words with similar semantic content to embedding vectors that are geometrically close. Cosine similarity is used to measure the semantic similarity between two terms. The two terms’ vectors are aligned in the same direction and suggested to have very similar meaning if the cosine similarity score is 1. On the other hand,the terms share no similarity if the cosine similarity is 0, indicating that they are independent of each other and do not share any semantic similarity.

**References**

Politics

1. https://www.9news.com.au/world/slovak-prime-minister-robert-fico-operation-remains-in-serious-condition/980afe2d-cc1d-4789-8e57-ee242eca49dd
2. https://www.9news.com.au/national/voters-worried-interest-rates-will-rise-again-post-budget-new-poll/6486091a-f8ce-44b2-84c0-8ba3262cff78
3. https://www.9news.com.au/national/darren-cheeseman-labor-mp-to-resign-over-inappropriate-behaviour-in-workplace/991fda3b-cfa5-407f-b339-a229b6739efb
4. https://www.9news.com.au/world/uk-politics-local-election-results-explained-ahead-of-general-election--rishi-sunak-keir-starmer/d8625a4f-6a23-4c35-8643-7163dadaa489
5. https://www.9news.com.au/national/geoff-brock-steps-down-from-sa-ministry-sparking-cabinet-shuffle/9d12dd55-85fe-411a-a616-0f8fcece1fd0
6. https://www.9news.com.au/world/solomon-islands-pro-china-leader-manasseh-sogavare-bows-out-of-prime-minister-contest/cbff1b58-7c3a-4de9-bf19-5d165a69fdc1
7. https://www.9news.com.au/national/federal-budget-2024-where-our-money-is-going-overseas/b1f6d8a2-3e9d-40aa-bb62-95da5df29951

Crime

1. https://www.9news.com.au/national/townsville-thai-restaurant-stabbing-joy-thai-two-injured/2b46735a-fd95-4807-b8f7-982062c4ce84
2. https://www.9news.com.au/national/port-melbourne-armed-robber-who-wore-underwear-as-disguise-on-the-run-in-melbourne/e950a999-9c87-45d1-bed5-6f933da887c0
3. https://www.9news.com.au/national/police-appeal-over-multimillion-dollar-mdma-seizure-on-ship-docked-in-wa/a9b407e1-e1f7-4150-bb77-f56247caa366
4. https://www.9news.com.au/national/glenroy-two-teenage-boys-arrested-over-tobacco-store-fires-in-melbournes-north/0fbe4faf-5dfc-4d7d-8105-d3731a6f70fa
5. https://www.9news.com.au/national/man-charged-after-allegedly-assaulting-four-women-on-train-to-adelaide/ff3657bf-d756-4d44-ae65-80f487deb68d
6. https://www.9news.com.au/national/man-to-finally-face-court-after-allegedly-murdering-ex-wife-in-car-crash/575e40e5-4e2c-47e5-9b91-c58e4f42244b
7. https://www.9news.com.au/national/widow-of-murdered-bikie-boss-nick-martin-sues-wa-government/dbf7b1f1-8b27-4644-9757-db034879b52d

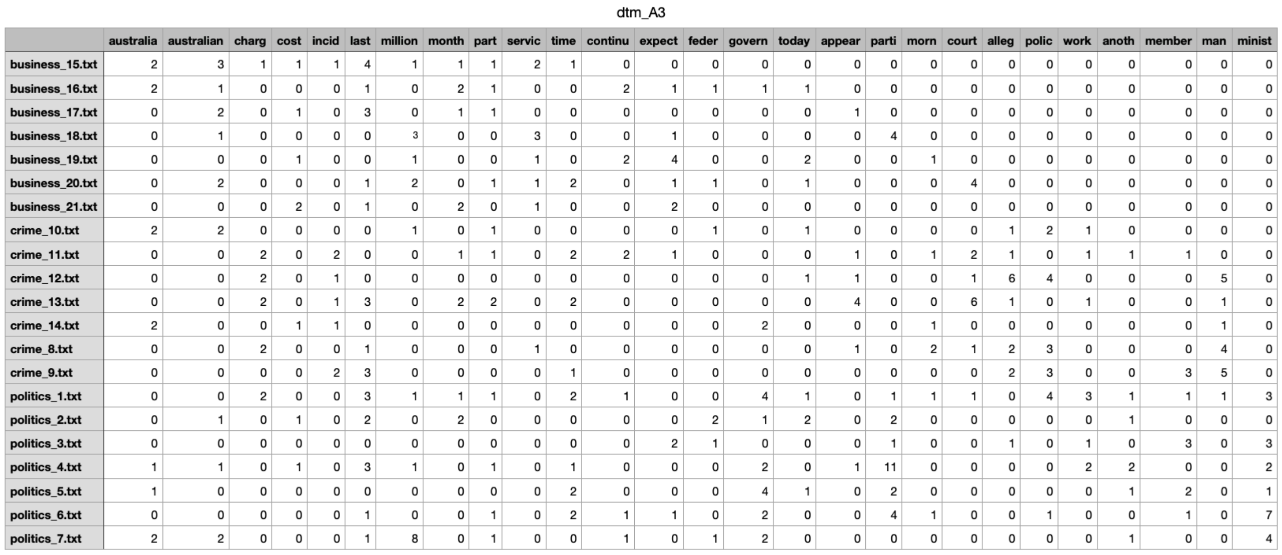
Business

1. https://www.9news.com.au/finance/optus-owner-singtel-talks-sell-australian-telco-canadian-private-equity-firm-brookfield/37df211a-ddc7-4669-ad87-98d6e50ae294
2. https://www.9news.com.au/national/australia-domestic-aviation-recovery-covid-pandemic/e112d3a8-3d6d-403c-bf65-4ba74562c935
3. https://www.9news.com.au/finance/business-collapses-australia-tax-office-chasing-debt/ce11a2f7-5566-4689-83b1-2eb1c48af1d4
4. https://www.9news.com.au/finance/macquarie-bank-10-million-dollar-fine-fraud-controls/49416d30-8fca-4c84-bb37-3f4f8f9a7af7
5. https://www.9news.com.au/finance/jb-hi-fi-half-year-results-profits-down-20-per-cent-finance-news-australia/4282b3ee-9a09-4b64-800c-ff45ce6b6297
6. https://www.9news.com.au/national/mazda-ordered-to-pay-millions-over-misleading-conduct/10000c32-b3a1-4faa-9c8c-38e36b14c768

7. https://www.9news.com.au/world/meta-profit-rise-online-advertising-up-cost-cutting/fe1d11fc-c30a-4aff-98bf-9926fac922aa

## **Appendix**

DTM



dtm_A3

| | australia | australian | charg | cost | incid | last | million | month | part | servic | time | continu | expect | feder | govern | today | appear | parti | morn | court | alleg | polic | work | anoth | member | man | minist |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| business_15.txt | 2 | 3 | 1 | 1 | 1 | 4 | 1 | 1 | 1 | 2 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| business_16.txt | 2 | 1 | 0 | 0 | 0 | 1 | 0 | 2 | 1 | 0 | 0 | 2 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| business_17.txt | 0 | 2 | 0 | 1 | 0 | 3 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| business_18.txt | 0 | 1 | 0 | 0 | 0 | 0 | 3 | 0 | 0 | 3 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 4 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| business_19.txt | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 2 | 4 | 0 | 0 | 2 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| business_20.txt | 0 | 2 | 0 | 0 | 0 | 1 | 2 | 0 | 1 | 1 | 2 | 0 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 4 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| business_21.txt | 0 | 0 | 0 | 2 | 0 | 1 | 0 | 2 | 0 | 1 | 0 | 0 | 2 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| crime_10.txt | 2 | 2 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 2 | 1 | 0 | 0 | 0 | 0 |
| crime_11.txt | 0 | 0 | 2 | 0 | 2 | 0 | 0 | 1 | 1 | 0 | 2 | 2 | 1 | 0 | 0 | 0 | 1 | 0 | 1 | 2 | 1 | 0 | 1 | 1 | 1 | 0 | 0 |
| crime_12.txt | 0 | 0 | 2 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 6 | 4 | 0 | 0 | 0 | 5 | 0 |
| crime_13.txt | 0 | 0 | 2 | 0 | 1 | 3 | 0 | 2 | 2 | 0 | 2 | 0 | 0 | 0 | 0 | 0 | 4 | 0 | 0 | 6 | 1 | 0 | 1 | 0 | 0 | 1 | 0 |
| crime_14.txt | 2 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 2 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| crime_8.txt | 0 | 0 | 2 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 2 | 1 | 2 | 3 | 0 | 0 | 0 | 4 | 0 |
| crime_9.txt | 0 | 0 | 0 | 0 | 2 | 3 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 2 | 3 | 0 | 0 | 3 | 5 | 0 |
| politics_1.txt | 0 | 0 | 2 | 0 | 0 | 3 | 1 | 1 | 1 | 0 | 2 | 1 | 0 | 0 | 4 | 1 | 0 | 1 | 1 | 1 | 0 | 4 | 3 | 1 | 1 | 1 | 3 |
| politics_2.txt | 0 | 1 | 0 | 1 | 0 | 2 | 0 | 2 | 0 | 0 | 0 | 0 | 0 | 2 | 1 | 2 | 0 | 2 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| politics_3.txt | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 2 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 1 | 0 | 3 | 0 | 3 |
| politics_4.txt | 1 | 1 | 0 | 1 | 0 | 3 | 1 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 2 | 0 | 1 | 11 | 0 | 0 | 0 | 0 | 2 | 2 | 0 | 0 | 2 |
| politics_5.txt | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 2 | 0 | 0 | 0 | 4 | 1 | 0 | 2 | 0 | 0 | 0 | 0 | 0 | 1 | 2 | 0 | 1 |
| politics_6.txt | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 2 | 1 | 1 | 0 | 2 | 0 | 0 | 4 | 1 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 7 |
| politics_7.txt | 2 | 2 | 0 | 0 | 0 | 1 | 8 | 0 | 1 | 0 | 0 | 1 | 0 | 1 | 2 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 4 |

```
'-Name: Ke Er Ang -
-Student ID: 32581343 -'

rm(list = ls())

library(NLP)
library(tm)
library(SnowballC)
library(igraph)

'---------------Question 1 & 2---------------'
# Set working directory
setwd("/Users/chloeang/Desktop/FIT3152")
# Point to the correct folder in the working directory
cname = file.path(".", "Asgn3")
# Create a Corpus object
docs = Corpus(DirSource(cname))
# Inspect the Corpus object
summary(docs)

writeLines(as.character(docs[[1]]))
writeLines(as.character(docs[[2]]))
writeLines(as.character(docs[[3]]))

'---------------Question 3---------------'
# Preprocess the corpus
# Tokenization
# Remove numbers
docs = tm_map(docs,removeNumbers)
# Remove punctuation
docs = tm_map(docs, removePunctuation)
# Convert text to lowercase
docs = tm_map(docs, content_transformer(tolower))
# Remove common stopwords
docs = tm_map(docs, removeWords, stopwords("english"))

# Create a transformer function to replace specific patterns
# Remove dash lines
toSpace = content_transformer(function(x, pattern) gsub(pattern, " ", x))
docs = tm_map(docs,toSpace, "—")
docs = tm_map(docs,toSpace, "–")
# Remove non-printable characters
removeNonPrintable = content_transformer(function(x) gsub("\\p{C}", "", x, perl = TRUE))
docs = tm_map(docs, removeNonPrintable)

# Remove those frequently used but low-value words
docs = tm_map(docs,toSpace, "also")
docs = tm_map(docs,toSpace, "may")
docs = tm_map(docs,toSpace, "since")
docs = tm_map(docs,toSpace, "will")
docs = tm_map(docs,toSpace, "just")
docs = tm_map(docs,toSpace, "can")
docs = tm_map(docs,toSpace, "like")

# Remove words that are common conversation but not directly related to the any of the selected
topics
docs = tm_map(docs,toSpace, "say")
docs = tm_map(docs,toSpace, "said")
docs = tm_map(docs,toSpace, "around")
docs = tm_map(docs,toSpace, "back")
docs = tm_map(docs,toSpace, "make")
docs = tm_map(docs,toSpace, "made")
docs = tm_map(docs,toSpace, "told")
docs = tm_map(docs,toSpace, "year")
docs = tm_map(docs,toSpace, "take")
docs = tm_map(docs,toSpace, "become")
docs = tm_map(docs,toSpace, "increase")
docs = tm_map(docs,toSpace, "show")
docs = tm_map(docs,toSpace, "see")
docs = tm_map(docs,toSpace, "know")
docs = tm_map(docs,toSpace, "remain")
docs = tm_map(docs,toSpace, "get")

# Remove too generic words that could fit in any topics
```

```r
docs = tm_map(docs,toSpace, "one")
docs = tm_map(docs,toSpace, "two")
docs = tm_map(docs,toSpace, "three")
docs = tm_map(docs,toSpace, "per")
docs = tm_map(docs,toSpace, "cent")

# Stemming documents
docs = tm_map(docs,stemDocument, language = "english")
# Remove white space
docs = tm_map(docs, stripWhitespace)

# Create a Document-Term Matrix
dtm = DocumentTermMatrix(docs)
inspect(dtm)

# Remove sparse terms
dtm_reduced = removeSparseTerms(dtm, sparse = 0.75)
inspect(dtm_reduced)

# Check and insepct word frequencies
freq = colSums(as.matrix(dtm_reduced))
length(freq)
sort(freq, decreasing = T)

ord = order(freq)
sort(ord)

# Inspect lowest and highest frequency
freq[head(ord)]
freq[tail(ord)]

# Check frequency of frequencies
head(table(freq), 10)
tail(table(freq), 10)

# Check dimension of dtm
dim(dtm_reduced)

# Identify key terms
findFreqTerms(dtm_reduced)

# Save dtm as csv
dtms = as.matrix(dtm_reduced)
write.csv(dtms, "dtm_A3.csv")

'---------------Question 4---------------'

# Calculate cosine distance
distMatrix = proxy:: dist(as.matrix(dtms), method="cosine")

# Perform hierarchical clustering
fit = hclust(distMatrix, method = "ward.D")

# Plot the hierarchical clustering
plot(fit, hang=-1, main = "Abstracts Consine Distance")

# Create topic vector
topics = c("business","business","business","business","business","business","business",
           "crime","crime","crime","crime","crime","crime","crime",
           "politics","politics","politics","politics","politics","politics","politics")

# Divide the dendrogram into 3 clusters
cosin_groups = cutree(fit, k = 3)

# make a table of topic labels vs cluster numbers
cosin_cluster_table = table(GroupNames = topics, Clusters = cosin_groups)
print(cosin_cluster_table)

# Calculate accuracy
accuracy = (6+3+6)/(6+1+2+2+3+1+6)
accuracy


'---------------Question 5---------------'
```

```
# Create networks by asbtract
dtmsx = as.matrix(dtms)
# transform into a binary matrix
dtmsx = as.matrix((dtmsx > 0)+0)
# multiply binary matrix by its transpose to find shared term
ByAbsMatrix = dtmsx%*%t(dtmsx)
# make leading diagonal zero to remove self-loops
diag(ByAbsMatrix)= 0
# create graph object
ByAbs = graph_from_adjacency_matrix(ByAbsMatrix, mode = "undirected", weighted = TRUE)
# plot the graph
plot(ByAbs,vertex.label.cex=0.6)

# calc average path length
a = average.path.length(ByAbs)
# calc graph density
g = graph.density(ByAbs)
# calc transitivity
t = transitivity(ByAbs)
# calc betweenness
b = format(betweenness(ByAbs),digits=2)
# calc closeness centrality
closeness.centrality.abs = closeness(ByAbs,normalized = TRUE)
c = format(closeness.centrality.abs,digits = 2)
# calc degreeness
d = degree(ByAbs)
# calc Eigencentrality
e = format(evcent(ByAbs)$vector, digits =2)

# combine centrality measures into a dataframe
Asum = as.data.frame(cbind(c,b,d,e))
# rename the data frame
colnames(Asum) = c("closeness","betweenness", "degree","Eigencentrality")


# Graph Improvement
# Edge Weight
E(ByAbs)$width = E(ByAbs)$weight / max(E(ByAbs)$weight) * 5

# Calculate communities using the Louvain method
communities = cluster_louvain(ByAbs)

# Define the custom color palette
custom_colors = c("#8ecae6", "#ffb703", "#f07167")
# Assign colour to node based on communities membership
node_colors = custom_colors[communities$membership]

# Plot the graph
plot(communities, ByAbs ,
     vertex.color = communities$membership,
     edge.width = E(ByAbs)$width,
     vertex.size = closeness.centrality.abs * 50,
     vertex.label.cex = 0.6,
     col = node_colors,
     mark.border = NA,
     mark.col = adjustcolor(brewer.pal(n = 3, name = "Pastel2"),0.7),
     edge.color = adjustcolor("gray", alpha.f = 0.8),
     main = "Single-Mode Graph by Abstracts")



'---------------Question 6---------------'
# Create networks by token
# Token matrix
dtmsx.token = as.matrix(dtms)
# transform into a binary matrix
dtmsx.token = as.matrix((dtmsx.token > 0)+0)
# multiply binary matrix by its transpose to find co-occurrence
ByTokenMatrix = t(dtmsx.token)%*% dtmsx.token
# make leading diagonal zero to remove self-loops
diag(ByTokenMatrix)= 0
# create graph object
ByToken = graph_from_adjacency_matrix(ByTokenMatrix, mode = "undirected", weighted = TRUE)
# plot the graph
```

```r
plot(ByToken, vertex.label.cex = 0.6)

# calculate average path length
a = average.path.length(ByToken)
# calculate graph density
g = graph.density(ByToken)
# calculate transitivity
t = transitivity(ByToken)

# calculate betweenness
b = format(betweenness(ByToken),digits=2)

# calculate closeness centrality
closeness.centrality.token = closeness(ByToken,normalized = TRUE)
c = format(closeness.centrality.token, digits = 2)

# calculate  degreeness
d = degree(ByToken)

# calculate Eigencentrality
e = format(evcent(ByToken)$vector, digits =2)

# combine all centrality measures as a data frame
Tsum = as.data.frame(cbind(c,b,d,e))
# rename the data frame
colnames(Tsum) = c("closeness","betweenness","degree","Eigencentrality")
# View(Tsum)

# Graph Improvement
# adjust the width of edges based on their weights
E(ByToken)$width = E(ByToken)$weight / max(E(ByToken)$weight) * 5

# Find communities using the Louvain method
communities.token = cluster_louvain(ByToken)

# Define the custom color palette
custom_colors = c("#8ecae6","#f07167" ,"#ffb703" )
# assign colour based on their community membership
node_colors = custom_colors[communities.token$membership]

# plot the graph
plot(communities.token, ByToken ,
     edge.width = E(ByToken)$width,
     vertex.size = closeness.centrality.token  * 50,
     vertex.label.cex = 0.6,
     col = node_colors,
     mark.border = NA,
     mark.col = adjustcolor(brewer.pal(n = 3, name = "Pastel2"),alpha =0.7),
     edge.color = adjustcolor("gray", alpha.f = 0.8),
     main = "Single-Mode Graph by Token")

'---------------Question 7---------------'
# Create bipartite network
# convert dtm into a data frame
dtmsa = as.data.frame(dtms)

# assign row names of dtm into a new column 'ABS' in dtmsa
dtmsa$ABS = rownames(dtm)
# create an empty data frame
dtmsb = data.frame()

# A loop that create a data frame, with each row representing a term-document combination.
# The colums include the document's identifier, the term's frequency in the document and the term
itself.
for(i in 1:nrow(dtmsa)){
  for (j in 1:(ncol(dtmsa)-1)){
    touse = cbind(dtmsa[i,j],dtmsa[i,ncol(dtmsa)],colnames(dtmsa[j]))
    dtmsb = rbind(dtmsb,touse)
  }
}
# rename columns, weight = frequency, abs = document, token = the term itself
colnames(dtmsb) = c("weight","abs","token")
# remove  entries where the term frequency is zero
dtmsc = dtmsb[dtmsb$weight != 0,]
```

```
# rearrange columns
dtmsc = dtmsc[,c(2,3,1)]

# create graph data frame
g =  graph.data.frame(d = dtmsc[,c("abs","token", "weight")], directed=FALSE)
# assign node types for bipartite structure
V(g)$type = bipartite_mapping(g)$type

# Colour the nodes
# Add a new column which categorize each row into a group based on the keywords found
dtmsc$group = ifelse(grepl("business", dtmsc $abs), "business",
                      ifelse(grepl("crime", dtmsc $abs), "crime",
                             ifelse(grepl("politics", dtmsc $abs), "politics", NA)))
# Create a unique pair of abstract and their corresponding group to ensure each abstract is
listed only once
abs_to_group = unique(dtmsc[, c("abs", "group")])
# Assign groups to the nodes
V(g)$group = abs_to_group$group[match(V(g)$name, abs_to_group$abs)]
# Set colors for different groups
group_colors = c("business" = "skyblue", "crime" = "#f07167", "politics" = "#ffb703")
# Assign colour to the nodes based on their group
V(g)$color = group_colors[V(g)$group]

# calculate closeness centrality
closeness.centrality.bipartite = closeness(g,normalized = TRUE)

# set node shape based on type
V(g)$shape = ifelse(V(g)$type, "circle", "square")
# set node size based on closeness centrality
V(g)$size = closeness.centrality.bipartite * 0.2
# set edges colours
E(g)$color = "lightgray"
# set edges width based on weight
E(g)$width = 0.3* as.numeric(E(g)$weight)

# plot the graph
plot(g,
     vertex.size=15,
     vertex.label.cex=0.5,
     edge.curved=0.1,
     main="Bipartite Graph Visualization")
```