Report of Text Analysis by Tokenization with R

By: Sarviin Hari

Data Collection

For the data collection process, I have decided to get 3 articles each for the categories of my choice which are Advertising, Food, AI, Climate Change, Cryptocurrency. Each of the 15 articles collected consists of at least 100 words.

Data Conversion

For the conversion of documents into a text format, since my data are from web-based articles, they are already text-based so I decided to copy the content of the document (from below the title until the end of the document) and paste into a text file with the labelling "CategoryName + Document Number (01 / 02 / 03)". Next, I create a folder called "CorpusAbstarcts" and within the folder, I created a folder called "txt" where I will place all the 15 text files of 5 categories. Next, I create a file path to the "txt" folder in "CorpusAbstarcts" and create a Corpus with the Directory Source, "DirSource" to be the file path.

Advertisement01.txt	Advertisement02.txt	Advertisement03.txt
2437	1003	1650
AI01.txt	AI02.txt	AI03.txt
5262	2010	3568
ClimateChange01.txt	ClimateChange02.txt	ClimateChange03.txt
3748	1123	2015
Food01.txt	Food02.txt	Food03.txt
1422	4968	1179
Savings01.txt	Savings02.txt	Savings03.txt
1018	1534	2330

Next, I calculated the number of words in each of the documents in the corpus. From this data, we can see that the document with the highest number of words is AI01.txt with 5262 words, while the document with the lowest number of words is Advertisement02.txt with 1003 words. The AI01.txt document has a high number of words because it covers multiple aspects of artificial intelligence (AI), including its types, examples, applications, governance, and history. This comprehensive study makes the document long as it includes a wide variety of topics related to AI. In contrast, the Advertisement02.txt document has fewer words because its main aim is to explain the importance of advertising. This topic has a focused scope of are, resulting in a smaller document size.

Data Transformation

For text transformation, I have decided to use a few approaches. The first approach is text transformation by substitution. This is because some words in the articles related to "Artificial Intelligence" have abbreviated words such as "AI", "ML" and "DL" which might not be useful when analysis as they do not provide any meaningful information and has a higher chance of being excluded analysis. Thus, by transforming the abbreviations to their original meaningful form, we can ensure that meaningful information can be extracted based on the repeated occurrence of the words throughout all documents.

The next approach that I used is tokenization. With tokenization I removed the Numbers, punctuations and convert all the words to lower case letters. This ensures that the data with no contributions to the analysis such as numbers and punctuations are omitted to prevent any noise being introduced to focus on the main contents of the documents and the text messages are standardized by setting to lower case, making them easier to analyse and compare.

Next, approach that I used is filtering. Since all documents are originated in English language, first I decided to remove all the stopwords that provide no meaningful information to the context of the analysis such as "and", "the", "is" etc. Since the occurrence of these words in frequent are highly likely removing these words prevents unwanted noise introduced by non-important words. Next I will strip all the additional whitespaces within the documents as well as performing stemming on the words in the corpus documents. A stemming is a process of conversion of words to their root words (i.e. eating -> eat) so that similar words that are of different tenses or forms can be normalized to a common representation to identify the similarity of the occurrences of these words within the corpus.

Next, I decided to view the details of the words by converting them to a document term matrix (dtm) and view the last 50 tokens of the highest frequency within the corpus. Upon viewing the documents, I identified a few forms of hyphens and apostrophes are still present within the documents, so I decided to replace these strings to a space. Next, from the last 50 words, I identified 19 words that has a very high frequency but provide no meaningful insight to the document analysis, for example 'can', 'use', 'also', 'like', 'new', 'may' and "'s". These words are repeated multiple times within the documents in the corpus but provide no meaningful information thus these documents

were replaced with a space. Next, I create a document term matrix from the processed data and identified that some of the tokens does not exhibit relationship to the documents.

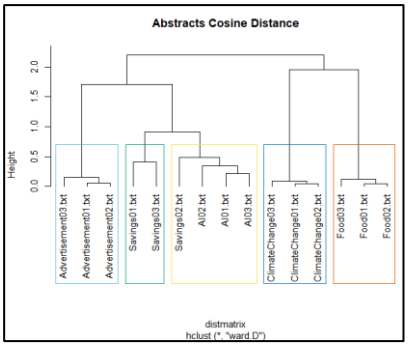
Next, I attempt to remove the sparse terms from the document by using a trial and error method and finally setting the sparsity threshold to 0.25 (to remove terms that are appearing <75% within the document). This is important in reducing the dimension of the document term metrics to get an analysis with less noisy data and more accurate data. Based in this I reviewed the tokens within the document term matrix and identified some of the tokens to be appearing within all documents multiple times without any distinct patterns to distinguish between different topics of documents as well as tokens that do not provide any meaningful information such as “amount”, “first”, “even”, “still” and many more. So, I decided to review these words against the documents collected by reviewing the type of use and frequency of use and removed the tokens that does not provide any meaningful information. This step is repeated for the sparsity threshold of 0.3, 0.35, 0.4 and 0.45. Once the threshold of 0.45 is reached, after removing the tokens that do not seem to provide any meaningful information, I observed that I was left with 25 tokens that might provide meaningful information for the document term matrix classification of topics. Since the threshold has met the criterion of reasonable number of tokens required for analysis (> 20 tokens with meaningful information), I decided to use the DTM with 25 tokens for the subsequent processing.

```
> dtms
<<DocumentTermMatrix (documents: 15, terms: 25)>>
Non-/sparse entries: 250/125
Sparsity           : 33%
Maximal term length: 7
Weighting          : term frequency (tf)
```

Based on the summary of the document term matrix, we can see that the resulting document term matrix have a sparsity of 33% which is still reasonable to derive a decisive conclusion about the document’s similarity based on the tokens and the longest-term length is 7. This indicates that there is some shared vocabulary of tokens, but also a significant amount of unique content within the tokens selection resulting in some of the tokens not appearing in some documents.

Text Analysis with Hierarchical Clustering

To create the hierarchical clustering model, I attempt to find the cosine matrix between each documents by using the code “distmatrix = proxy::dist(dtms, method = "cosine")”, where I used the dist function from the proxy package on the document term matrix (with 25 terms/tokens) using the method “cosine”. A cosine similarity works by identifying the cosine of the angle between vectors of documents in a multi-dimensional space. The values for the distance matrix based on cosine similarity ranges between 0 to 1 where 0 refers to highly identical document and 1 refers to highly non-identical documents.



Based on the distance matrix, I plotted a hierarchical clustering model with the method = “ward.D”. The diagram shows the hierarchical clustering model for all the documents. From this plot, qualitatively, we see that, for the first, fourth and fifth cluster the documents for Advertisements (light blue), Climate Change (dark blue) and Food (Orange) respectively are within the same cluster showing that the processed data of tokens is able to perfectly classify them to a cluster of their own. This is because documents that are within the same clusters are more similar to each other in comparison to those of different clusters suggesting the similarity in the topic chosen. As for the third cluster (Yellow), although we can see that all 3 documents of AI are within the same cluster, but there is an outlier, where Savings02 is clustered as the same as AI. This suggests that the document Savings02 seem to exhibit more similarity with AI than the cluster of its own category. As for the second category (Green), we can see the remaining 2 documents of “Savings” are clustered within the same cluster indicating the correct similarity between both documents.

Quantitatively, we can observe the height of the dendrogram which indicates the cosine distance between clusters and documents. The smaller the heights, the higher the similarity is between the clusters. From the dendrogram, we can see that the documents cluster of ClimateChange, Food and Advertisement have a relatively bugger height in comparison to each other and remaining clusters suggesting that they are able to be distinguished well by the clustering based on their cosine similarity. The two clusters, Savings and AI, have a smaller height compared to other clusters. This suggests a higher degree of similarity between the topics covered in these documents. Savings02.txt

Report of Text Analysis by Tokenization with R

By: Sarviin Hari

might be clustered under AI because the content of this document aligns more closely with AI topics than Savings topics based on the features used for clustering. This could be due to Savings02.txt containing tokens or concepts that are common to both Savings and AI domains.

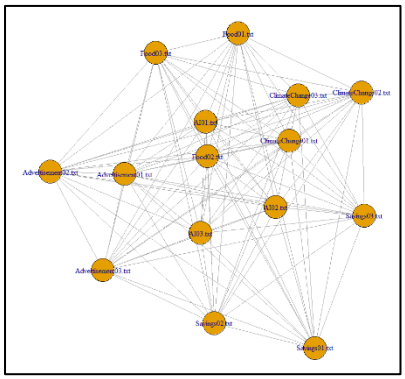
```
     1 2 3 4 5
adv  3 0 0 0 0
ai   0 3 0 0 0
clim 0 0 3 0 0
food 0 0 0 3 0
sav  0 1 0 0 2
> print((TA[1,1] + TA[2,2] + TA[3,3]+TA[4,4]+TA[5,5])/15*100)
[1] 93.33333
```

Next, I created a table of groups within the cluster vs the clustering of the 5 categories of documents to create a confusion matrix of actual vs expected output. Based on the hierarchical clustering and the confusion matrix, we can identify that the accuracy of the model is 93.33% suggesting that 14/15 documents have been correctly classified to the cluster of their own. An 93.33% accuracy suggests a good clustering based on the tokens of the document term matrix to derive the categories of documents.

Thus, based on the clustering of the documents based on the document term matrix created from the processed data with 25 tokens, the hierarchical clustering on the distance matrix (using cosine similarity) created with the method “ward.D” can classify all the documents by their topics with 93.33% accuracy. The 100% accuracy could not be achieved which is potentially due to some of the tokens (even after processing of data) inducing noise within the model due to the unimportant information induced within the model clustering. This is potentially why the document Savings02.txt was clustered within the cluster of AI’s rather than Savings.

Text Analysis with Single Mode Network (Document)

To create a single-mode network between documents based on number of shared items, first I converted the document term matrix (sparse matrix format) to R matrix. Then I create a matrix of document vs documents by multiplying the document term matrix (with documents as the row) with its transpose (with documents as the column) and setting the diagonals to 0 (since same documents must not be considered between each other). Next, I plot the undirected weighted adjacency matrix graph for the documents with the weighted parameter set to “True”.



From this plot, we can see that each of the nodes are represented by the document names, while each edge represents the connection between the documents (all documents pair except self-edges). We can also see that all the nodes are having the same colour and size, indicating that uniformness in a centrality measure in the base plot. From the graph as well, we can see the degree, which refers to the number of edges from one node to other nodes. This metric indicates the number of documents each of the document have a direct connection based on the documents shared terms. From the graph, we can see that all the documents have 14 edges “to” or “from” the document, suggesting that every document share term with each other document since the edges from nodes are maximised to all the other nodes within the graph. This implies that all the documents within the document term matrix are interconnected between each other with a high overlap between the terms as they have at least one shared term between them. This also suggest that out of the 25 tokens/words of the document term matrix, at least one of the words are commonly shared for all the documents indicating the tokens are spread among all the documents in the corpus.

The documents related to **Advertisements** (Advertisement01.txt, Advertisement02.txt, Advertisement03.txt) are closely grouped together. This suggests that these documents share a significant number of common terms, indicating a strong similarity in terms of the type of the documents. Similarly, the **Climate Change** documents (ClimateChange01.txt, ClimateChange02.txt, ClimateChange03.txt) are also closely grouped together. This tight clustering reflects a high degree of commonality in terms used within these documents. The **Food and AI documents** seem to show a more varied pattern. While Food01.txt and Food03.txt are close to each other as well as AI02.txt and AI03.txt being close to each other respectively, Food02.txt and AI01.txt seem to be closer to each other rather than being close to their own cluster of documents. This suggests that Food02.txt might share more terms with AI01.txt than with the other Food documents and vice versa. The **Savings** documents (Savings01.txt, Savings02.txt, Savings03.txt) are somewhat isolated from other clusters and seem to be slightly further away from each other,

Report of Text Analysis by Tokenization with R

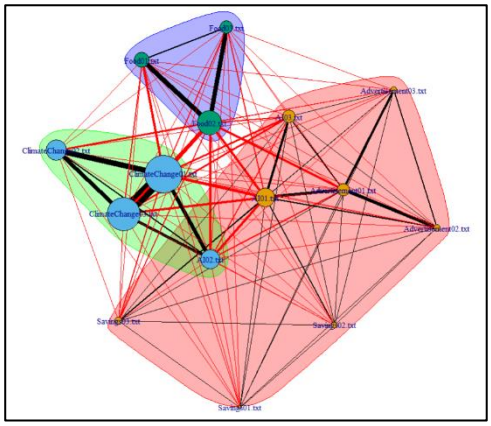
By: Sarviin Hari

indicating they share fewer common terms with documents from other categories as well as potentially their own categories.

	degree	Closeness	eig
Advertisement01.txt	14	0.0005020	0.3420
Advertisement02.txt	14	0.0008177	0.1889
Advertisement03.txt	14	0.0007599	0.1857
AI01.txt	14	0.0003906	0.5080
AI02.txt	14	0.0004733	0.5183
AI03.txt	14	0.0004902	0.3487
ClimateChange01.txt	14	0.0004638	1.0000
ClimateChange02.txt	14	0.0008271	0.5587
ClimateChange03.txt	14	0.0005417	0.8878
Food01.txt	14	0.0007375	0.4060
Food02.txt	14	0.0004808	0.6508
Food03.txt	14	0.0006631	0.3475
Savings01.txt	14	0.0009355	0.1181
Savings02.txt	14	0.0007599	0.1752
Savings03.txt	14	0.0006878	0.1999

Based on the plot, I also populated the degree, closeness and eigenvector centrality. As mentioned before the degree for all the documents is 14 indicating that there are similar words between all the documents with each other. A closeness centrality identifies how close is a node to other nodes within the single-flow network. The higher the closeness value, the higher the effectiveness of the documents to reach other documents efficiently. Based on the results, we can see that “Savings01.txt” has the highest closeness score (0.0009355), suggesting that this document is centrally located in the network in comparison to other documents. Furthermore, this also indicates that the node for “Savings01.tx” has a higher influence on the flow through the network in comparison to other nodes as it acts as a common connector between all other documents. Next, we can look at the Eigenvector centrality metric, which measures the influence of a node within the graph based on its influence with the neighbouring nodes. Based on the eigenvector metric, we can see that “ClimateChange01.txt” has the highest eigenvector score suggesting the higher strength of connection of this document with all the other documents in the graph. This implies that "ClimateChange01.txt" plays a central role in the document network, likely containing key tokens that is relevant across multiple documents.

To create an improved model from the base single-mode network between documents based on number of shared items, to obtain more valuable information, I decided to include three metrics which are the eigenvector of the vertices (determined by the size of the vertex based on the eigenvector score), weight of the edges (determined by the width size of the edges based on the weight assigned) and the communities or subgroups within the graph (determined by the “fast greedy” community detection algorithm which is a method based on modularity by building communities by adding vertices to community one by one based on the modularity score assigned). Based on these results, the below improved graph can be created.



Based on the graph, firstly, looking at the vertices, we can see that “ClimateChange01.txt” has the largest vertex size and “Savings01.txt” has the smallest vertex as the size of the vertices are determined by the eigenvector score, which is based on the node’s connection influenced by the high-scoring nodes. Based on this, we can also clearly see that the nodes that have a bigger size of vertex (eigenvector score), have the thicker size of edge to all the other nodes that has a higher eigenvector within the graph. This insight tells us that Eigenvector centrality is influenced by the weights of the edges connected to a node and the strength of its neighbouring nodes. A higher strength connection with a neighbouring node that has a high Eigenvector score suggests that the node will also have a higher Eigenvector score. Even so, a node with the highest edge weights will not necessarily have the highest Eigenvector centrality unless those weights lead to highly influential nodes. Although we can't see this diagrammatically due to the graph being a complete connected graph where all nodes are connected to each other, we can clearly see the influence of edge weights on the Eigenvector score as determined by the size of the vertex. So, distinguishing this metrics is important in the analysis of the network analysis graph.

The first cluster in purple consists of the documents from the “**Food**” category, where we can see the documents Food01.txt, Food02.txt and Food03.txt are placed in the same cluster. Within this cluster we can also view the strength of connection of the weights of the edges where we can see these 3 documents have a larger weight of edges within themselves in comparison to other documents. This suggests that the documents of the Food category were able to be clustered well based on the commonality of the tokens for the DTM created based upon the frequency term matrix which shows the number of times terms from one document appears in another. Thus, the stronger connection signifies that these documents share the almost the same number of tokens within them in comparison to other documents.

Report of Text Analysis by Tokenization with R

By: Sarviin Hari

As for the second cluster in green colour, we can see that the documents are from the "**Climate Change**" category, where the documents ClimateChange01.txt, ClimateChange02.txt, and ClimateChange03.txt are present with an outlier being the document **AI02.txt** (*from the base graph, it was predicted that Climate Change is a cluster of its own and AI02.txt and AI03.txt is a cluster of its own, which is not the case in this scenario with fast greedy clustering*). Although we can see that based on the strong connection of edges the Climate Change category documents were able to be clustered well together into one single cluster, the document AI02.txt being present in this cluster suggests that it has more common tokens with these documents especially ClimateChange03.txt and ClimateChange02.txt compared to other documents. This is potentially due to AI02.txt might be an AI document discussing the application of AI in climate change research which would explain the shared vocabulary with the Climate Change documents. Thus, upon revisiting back the documents of AI's, I realized that the documents AI01.txt and AI03.txt discusses on the intricate details on what is an AI from different perspectives while AI02.txt explains on the impact of AI on the world with a section for Climate Change. This suggests that the commonality of the topics might not only be the influence in the document clustering, but the content and contexts of these documents also pay an important role.

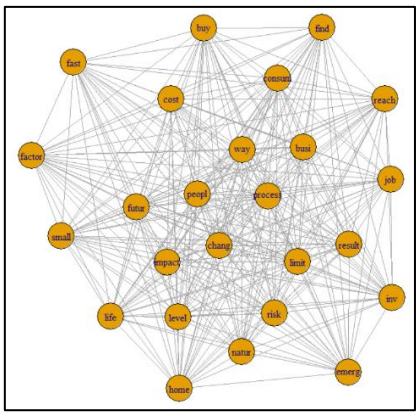
As for the third cluster we can see the documents for AI01.txt, AI03.txt and the documents for the Advertisement and Savings to be clustered among the same red group although they all are distinctively different in their categories. Within this cluster the three **Savings** documents appear to be further away and show weak connections, both with each other and with other documents. This indicates that the Savings documents are not central points for information within the network. The idea is further supported by the very low strength or size of weights of the edges connecting these documents to others, suggesting a weak relationship with the rest of the documents. Furthermore, the Savings document being further away is potentially due to the limited vocabulary focused on savings-specific terms in the Document Term Matrix leading to lesser variation of tokens within the documents (*as predicted from the base model, since there is lesser variation among Savings, they are clustered to the closets cluster which is the red cluster*). This is an effect of filtering of tokens, where the tokens that explains the Savings document better was not within the selected range of sparse terms. This leads to the topic of savings not being central to the overall focus of network. Thus, if other documents do not reference savings terms specifically, the Savings documents naturally exhibits weaker connections within the network. As for the **Advertisements** documents, although they seem to exhibit strong connection with each other, their connectivity with other documents within the red cluster is mildly strong which is why these documents appear to be in the red cluster rather than the cluster of their own. Similarly, the **two AI** documents show strong connections with each other and strong-mild relationship with documents within red cluster.

Text Analysis with Single Mode Network (Token)

To create a single-mode network between tokens based on number of shared items, first I converted the document term matrix (sparse matrix format) to R matrix. Then I create a matrix of token vs tokens by multiplying the transpose of the document term matrix (with rows as tokens) with the document term matrix itself (with columns as tokens) to get a matrix of tokens vs tokens and setting the diagonals to 0 (since same documents should not be considered between each other). Next, I plotted the undirected weighted adjacency matrix graph for the documents with the weighted parameter set to "True".

Report of Text Analysis by Tokenization with R

By: Sarviin Hari



From this plot, we can see that each of the nodes are represented by the tokens, while each edge represents the connection between the tokens (to all token pair except self-edges). From the graph as well, we can see the degree, which refers to the number of edges from one node to other nodes to be 24 edges "to" or "from" the tokens, suggesting that every token share document with each other documents since the edges from nodes are maximised to all the other nodes within the graph. This implies that all the tokens within the document term matrix are interconnected between each other with a high overlap between the documents as they have at least one shared document in between for each token. This indicates a significant overlap between the documents in terms of vocabulary.

From the tokens plot, we can see that the tokens, busi, consum, reach, and find are seemingly close together indicating they can be grouped together to form a cluster where these tokens are related to the topic of business which is most related to the category of **Advertisement** documents. Next, I can also see the tokens impact, chang, level, natur, and risk which are seemingly close together forming a cluster indicating a change occurring related to nature with some impact at a certain level which is most related to the climate, thus this cluster will most likely be within the category of **Climate Change** document. As for the tokens cost, way, peopl, futur and process which are seemingly near each other seem to indicate about the future and some cost (might be referring to price or computational cost) as well as the process and way of working which is most related to the category of **AI** but weakly since although these words do explain AI, but they are not the dominant terms like "intelligence" or "software" that clearly identifies AI related topics. As for the documents emerg, result, limit, and inv which are close together seems to indicate something related to an investment or emergency related details with some result or outcome which seems to relate more to the category of **Savings** as they seem to be explaining about money related information. As for the category document of **Food**, the tokens do not seem to be close to each other and scattered within the network that it cannot be identified without improving the network model.

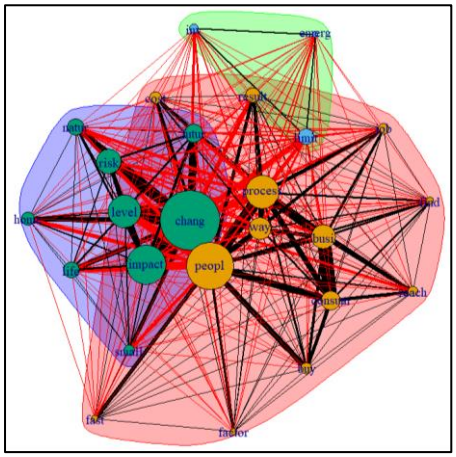
	degree	closeness	eig
busi	24	0.0006743	0.4128
buy	24	0.0010799	0.1924
chang	24	0.0005233	1.0000
consum	24	0.0009116	0.3273
factor	24	0.0013699	0.1196
fast	24	0.0014409	0.1295
find	24	0.0010638	0.1316
home	24	0.0009718	0.2304
inv	24	0.0011099	0.1502
job	24	0.0012870	0.1876
peopl	24	0.0003772	0.7942
process	24	0.0004706	0.5663
reach	24	0.0011148	0.1703
way	24	0.0007148	0.3791
emerg	24	0.0012255	0.1109
impact	24	0.0008123	0.6639
natur	24	0.0010060	0.2826
cost	24	0.0009355	0.2075
result	24	0.0010121	0.2458
small	24	0.0011507	0.1888
futur	24	0.0011148	0.3124
level	24	0.0008150	0.5497
life	24	0.0009124	0.2756
limit	24	0.0009285	0.2694
risk	24	0.0008993	0.4214

Based on the plot, I also populated the degree, closeness and eigenvector centrality. As mentioned before the degree for all the tokens is 24 indicating that all the nodes for tokens are completely connected with an edge. A closeness centrality identifies how close is a node to other nodes within the single-flow network. Based on the results, we can see that the token "fast" has the highest closeness score (0.0014409), suggesting that this token is located centrally in the graph and can reach other nodes effectively and also acts as an influential common connector between the node to other nodes. Next, we can look at the Eigenvector centrality metric, which measures the influence of a node within the graph based on its influence with the neighbour nodes. Based on the eigenvector metric, we can see that "chang" has the highest eigenvector score suggesting the higher strength of connection of this token with all the other tokens in the graph.

To create the improved model from the base single-mode network between tokens based on number of shared items, to obtain more valuable information, I used the same technique as in Question 5, where I used three metrics which are the eigenvector of the vertices (determined by the size of the vertex based on the eigenvector score), weight of the edges (determined by the width size of the edges based on the weight assigned) and the communities or subgroups within the graph (determined by the "fast greedy" community detection algorithm which is a method based on modularity by building communities by adding vertices to community one by one based on the modularity score assigned). Based on these results, the below improved graph can be created.

Report of Text Analysis by Tokenization with R

By: Sarviin Hari



Based on the base graph and improved graph, we predicted to have at least 4 distinct clusters which is not the case here as only 3 clusters were able to be identified with fast greedy approach suggesting that the interpretation from the base model might not have considered certain factors. Firstly, looking at the vertices, we can see that “chang” has the largest vertex size and “emerg” has the smallest vertex as the size of the vertices are determined by the eigenvector score, which is based on the node’s connection influenced by the high-scoring nodes. Based on this, we can also clearly see that the nodes that have a bigger size of vertex (eigenvector score), have the thicker size of edge to all the other nodes within the graph. The observation on the size of weights from “chang” and “emerg” clearly shows a relationship between the eigenvector score and the weight of edges as we can see that the thickness of the edges from “chang” to all nodes are the thickest in total in comparison to all other nodes while for “emerg”, the thickness of the edges are the thinnest in total in comparison to all other . This suggest that “chang” being the central token meaning it is highly connected and influential in the network. This could be because "chang" represents a common or pivotal concept that links many documents or topics together. Its high eigenvector score means it is well-connected to other high-scoring nodes, reinforcing its importance. The thickness of the edges from "chang" to all other nodes signifies strong connections or high weights, suggesting frequent or significant interactions with other nodes for all documents. As for 'emerg', the small eigenvector score, as indicated by the small size of the node, suggests that this token is more unique among the 15 documents. This uniqueness implies that 'emerg' appears less frequently and is less interconnected with other high-scoring tokens. This suggests that the token “emerg” plays a specialized role in the set of documents where it highlights distinct topics in dataset.

We can see that for the red cluster peopl, way, process, busi, consum, reach, buy, factor, fast, find, job, result, and cost are within the same cluster where they are more general and relate to various aspects of business or consumer behavior, such as finding jobs, processes, results, and costs. When we investigate it the tokens peopl, busi, consum, reach, buy and process are strongly related to each other (especially busi and consum) as they seem to be connected by thick edges among each other. And from an interpretation view, we can identify that these words are commonly used to identify a business or company marketing details. This suggests that although this cluster is within a large red cluster, but it seem to signify the **Advertisement** documents. Within the cluster we can also see the remaining tokens which are way, factor, fast, find, job, result and cost which seems to be far from the cluster within and have lesser strength of relationship within themselves suggesting that these values might be an outlier for advertisement cluster or they might have been allocated to the wrong cluster due to the not-well clustering within the tokens.

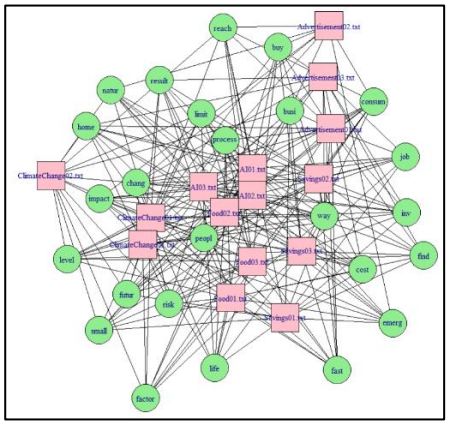
The green cluster consists of the tokens, emerg, limit and inv are within the same cluster. Although these tokens do not have a very thick size of edges between them, but these tokens seem to have the thickest among themselves compared to their edges towards any other tokens within the network suggesting that these tokens might refer to some meaningful output. From the words within this cluster, emerg may mean emergence or emergency, limit refers to the maximum and inv mean inverse or investment. Based on the meaning of these words in terms of investment, emergency and limit, these token cluster might refer to the documents on savings where these words are reflecting on ways of savings and when to save. Thus, these words have a strong correlation to the documents for **Savings**.

Next, we have the purple cluster where the words impact, chang, level, risk, futur, natur, reduc, home, life, and small are within the same cluster. Among this cluster the words impact, futur, chang, level, risk, and natur seems to have a very strong connection with each other in terms of their weighted edges between them. These words might be very related to nature and the risks to nature thus signifying (especially the word chang referring to the changes that are currently occurring accompanying with the words related to the world life natur) **Climate Change** topic.

Within these tokens, although I could identify the patterns based on the cluster of tokens for the documents Advertisement, Climate Change and Savings, I could not identify a clear pattern for the tokens relating to the **Food** and **AI**. This is potentially due to the documents related to Food and AI might have a broader or less specific focus, leading to a more widespread set of tokens. Although our quality of clustering for the tokens and documents are

high, potentially due to the dispersed set of tokens for these documents within different document clusters we might not be able to see a clear cluster for the tokens referring to these documents.

Text Analysis with Bipartite Network



To create a bipartite network of my corpus with document ID as one type of node and tokens as the other type of nodes, first I convert my document term matrix to a data frame and added a new column, "ABS", where the values are of the rownames of the dataframe. Next I will create a new data frame, where I will loop through each rows representing the documents and each columns representing the tokens to create a long data frame with the first column being the weight, the second column being the document and the third column being the tokens, where each row consists of the unique pair of the weight, document and the token. Next, I will remove the rows with a weight of 0, mainly to remove the pair of document and token that do not have any relationship (as weight = 0). Next, I will reorder the data frame where the first column is the document, followed by the token and weight. With the complete data processed, I will create a graph instance with "graph.data.frame" with the processed data as input and the directed edges is set to "False". Next, I will set the mapping of the graph instance to a bipartite graph set the type of the nodes to each of the respective labels of document or token with a logical vector. Next, the colour and the shape of the vertices are set based upon their types and the edges are set to black colour. With this tidy of data and graph instance, we can plot the graph for the bipartite network.

Based on the graph, we can see that the vertices for the tokens are identified by green node vertices with round shape while the documents are represented by the square node vertices with pink colour. From the graph the documents for the AI category and Food02.txt are closely clustered with the closest token being peopl and process at the centre which means these documents and tokens have a strong relationship with many documents or tokens as they seem to have the most edges to other documents or tokens. We can also see that the tokens like factor, and reach seem to be further away from other documents and tokens suggesting that they have less degree of relationship with other documents and tokens in the graph.

Advertisment01, 02, 03	busi, reach, buy, consum
AI01, 02, 03	process, limit, result
ClimateChange01, 02, 03	natur, home, impact, chang, level, futur, small
Food01, 02, 03	peopl, risk, life, factor
Savings01, 02, 03	way, cost, find, emerg, fast, inv, job

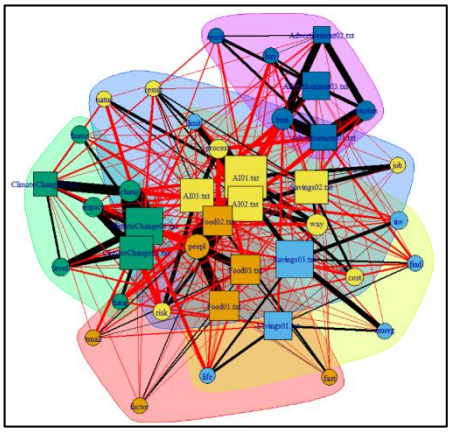
In terms of groups within the network by viewing the graph, I can identify the groups for the documents and their respective tokens as in the table. For the **Advertisement**, the words identified from the graph seems to be explaining about the business principle with the consumers as well as the market in terms of the reach of the advertised items to the consumers that consume or buy these products. Thus, these tokens are actually providing important information on the category of Advertisement. As for the **AI**, we can see the tokens seem to explain about a limit within the AI and the internal processes and their corresponding results. Although there seem to be interconnection for the documents of AI with these tokens, there does not seem to be a strongly explaining the topic of AI as these words might also be common within other documents. As for **Climate Change** the tokens are explaining about the nature, the changes that is occurring and the corresponding impact for future. Together these words themselves seem to explain the topic of Climate Change very well. There also seem to be some tokens that do not seem to be providing very meaningful information such as small, home and level suggesting these might be unimportant in terms of Climate Change analysis. As for **Food** although the tokens peopl, life and risk can be assumed to explain about how people may be affected with life-threatening issues due to unproper food, these tokens do not seem to be explaining the category of Food very well as the words seem to be common within other documents as well. As for **Savings** the terms seem to be explaining about investments and cost of usage as well as emergency situations and way (how to do or use) which are all seem to be explaining the topic Savings very well. There are also some tokens like find, fast and job which might also be common with other documents while might not also provide meaningful insight to Savings.

Report of Text Analysis by Tokenization with R

By: Sarviin Hari

	degree	closeness	eig
Advertisement01.txt	14	0.010309	0.18970
Advertisement02.txt	9	0.010309	0.08597
Advertisement03.txt	15	0.011765	0.08549
AI01.txt	23	0.011236	0.30305
AI02.txt	19	0.010000	0.27960
AI03.txt	18	0.011111	0.17477
ClimateChange01.txt	20	0.010309	0.91573
ClimateChange02.txt	13	0.010101	0.32301
ClimateChange03.txt	18	0.012195	0.66647
Food01.txt	16	0.011905	0.22110
Food02.txt	16	0.008929	0.53073
Food03.txt	16	0.010753	0.18128
Savings01.txt	15	0.011236	0.05581
Savings02.txt	18	0.011494	0.08447
Savings03.txt	20	0.012195	0.10073
busi	12	0.010638	0.17760
buy	9	0.010204	0.08085
chang	12	0.011628	1.00000
consum	11	0.011111	0.14464
factor	9	0.011628	0.05432
fast	9	0.012195	0.05911
find	9	0.009174	0.04863
home	10	0.011628	0.10835
inv	10	0.010989	0.06260
job	9	0.012048	0.07324
peopl	14	0.008772	0.68170
process	11	0.007752	0.27774
reach	9	0.009901	0.06891
way	12	0.010101	0.16367
emerg	9	0.011765	0.04288
impact	11	0.011236	0.36913
natur	9	0.010638	0.12999
cost	10	0.011494	0.08796
result	9	0.010101	0.10941
small	9	0.011494	0.08980
futur	9	0.009901	0.14437
level	10	0.010204	0.28772
life	9	0.010638	0.13184
limit	9	0.008547	0.11749
risk	10	0.010000	0.20580

Based on the plot, I also populated the degree, closeness and eigenvector centrality. From this table we can see that among the documents, AI01.txt has the highest degree with 23 edges to tokens while among the documents busi, chang, and way have the highest degree with 12 edges to doc. This suggests that AI01.txt has the highest number of edges to all the tokens, suggesting that 23/25 of the tokens can be seen within the document AI01.txt a sit covers a wide range of terms. As for the tokens of busi, chang, and way, at least one repetition of these tokens can be seen from 12/15 documents within the cluster. This shows the commonality of the tokens in the documents and their relationship within the document. Based on the closeness score, among the documents, ClimateChange03.txt and Savings03.txt has the highest closeness score while among the tokens, fast has the highest closeness score of 0.12195. The high closeness suggests that the documents and tokens are centrally located and can quickly interact with many tokens or documents respectively as it covers a broad number of nodes efficiently within the network. Based on the eigenvector score, among the documents, ClimateChange01.txt has the highest value of 0.91573 while among the tokens, chang has the highest value of 0.04288. The high eigenvector suggests that the document is connected to other influential nodes/tokens in the network which implies that the document displays a high relevance while high eigenvector suggests that the tokens are frequent and vital in connecting important documents together thus being a key component of the bipartite network.



To create an improved model from the base bipartite network between the documents and the tokens based on number of shared items, to obtain more valuable information, I decided to include three metrics which are the degree of the vertices (determined by the size of the vertex based on the number of edges from the node), weight of the edges (determined by the width size of the edges based on the weight assigned) and the communities or subgroups within the graph (determined by the “fast greedy” community detection algorithm which is a method based on modularity by building communities by adding vertices to community one by one based on the modularity score assigned). Based on these results, the improved graph can be created.

From the base graph, we predict to have 3 strong clusters in Advertisement, Savings and Food, while weak clusters for AI and Food, but we seem to get 5 distinct clusters from the improved graph with weak cluster AI and Savings documents where one Savings document is clustered wrongly to the AI cluster suggesting that the prediction from the base model is partially correct. The first cluster is the purple cluster which comprises of 3 Advertisement documents, with the tokens buy, busi, reach and consum. This seems to align with our previous prediction of the categories based on the base model, suggesting that the Advertisement document and their corresponding tokens are well clusters with strong connection of edges among themselves.

As for the second cluster which is the red cluster which comprises of 3 Food documents with the tokens peopl, factor, small, and fast seem to explain about the factors related to food choosing and the small portions of food to maintain a good health as well as the term fast referring to probably fast food. Even so, these tokens don’t seem to explain the food category well enough as these tokens seem to be just a small sample of words within the documents of Food. Although these tokens seem to have a good relationship with the Food category in terms of clustering, stronger tokens associated with food will provide a stronger association.

As for the third cluster which is the green cluster which comprises of 3 ClimateChange documents with the tokens level, futur, impact, chang and home which seems to explain the Climate Change well especially in terms of futur, impact and chang which gives a high-level analysis on the Climate Change. Even so, we can see the token natur to not be within the green cluster instead being in the yellow cluster, suggesting that the token natur seems to have a higher correlation with the yellow cluster with the documents within than the green cluster.

Report of Text Analysis by Tokenization with R

By: Sarviin Hari

In terms of the fourth cluster, we can see it comprises of the documents of all 3 AI documents as well as Savings02.txt with the tokens being natur, result, risk, process, way, cost and job where all these documents and tokens seems to have strong association within each other in terms of the strength of their edges. The documents related to AI seems to have a strong relationship within the cluster with cost, process, risk and result which might be explaining about the computational cost of an AI model, the processes within the AI with the resulting outcome and the potential risks associated with AI. As for the document on Saving02.txt which seems to be well associated with the token's way, cost, job explains how to minimize the cost with multiple ways and to increase money with jobs. Although AI and Savings02.txt doesn't seem to have any similarity in terms of context, the nature of the text of the Savings02.txt document which has common words with AI tokens is the reason why Savings02.txt is clustered with AI. This suggests that these words are not the strongest words associating with AI, as they seem to overlap with the Savings02.txt document even when they do not have similarity in context suggesting the categories of AI is not explained well enough.

As for the fifth cluster, we can see that the Savings01.txt and Savings03.txt documents are clustered among the tokens which are life, emerg, find and inv. These terms seem to be explaining about investments and usage of the savings money in emergency situations and its usefulness in life situation which are all seem to be explaining the topic Savings very well suggesting the clustering for Savings documents has been done well enough.

Conclusion

Summary

To measure the importance of documents for the question 5 and 6, I have decided to use the eigenvector centrality which measures the influence of each document or tokens based on the number and quality of the connection between the nodes. The weight of the edges plays an important role in the eigenvector centrality measure where documents or tokens with strong connections to other influential documents or tokens (high eigenvector scores) respectively will themselves have higher eigenvector scores. Even so, the eigenvector centrality does not solely depend on the edge weights as for a higher eigenvector score, the document or token must have the strong connection with other influential documents or tokens respectively with higher eigenvector centrality score themselves.

Based on question 5, the document with the highest eigenvector score is ClimateChange01.txt with a score of 1.0 which represents a high influence within the network. This suggests that this document most likely have a strong connection to other influential documents within the network and is potentially the central node. This suggests that this document might contain the key details (in terms of the tokens) for all the documents within the corpus. While ClimateChange01.txt might contain important information, it probably doesn't encompass all the details for every document as they are of different topic areas, but the commonality of the tokens used in this document in comparison to different documents might explain the strong centrality relationship it has with other documents.

Based on question 6, the token with the highest eigenvector score is chang with a score of 1.0 which represents a prominent and influential within the network. This suggests that this token frequently appear in the important documents within the network which connects various part of the corpus. This also highlights that the token chang is a key term across multiple documents within the network due to its commonality within all the other documents in comparison to all other tokens. This is potentially due to the interchangeability of the token chang which refers to the English word change, which is an important reference in all documents as it can be used to refer to AI, Advertisement, ClimateChange (prominently), Savings and Food.

Based on question 7, the group that is the most important based on the strength of documents and tokens (based on eigenvector scores) as well as the improved bipartite network with the "fast greedy" community detection is the group of Climate Change with the tokens chang, impact, level, futur and home as it contains the most influential document and the most influential token within the same cluster.

This analysis reveals the significant influence of climate change related documents and terms across the corpus, as shown by the single-mode document and token networks, and the bipartite network.

Report of Text Analysis by Tokenization with R

By: Sarviin Hari

Relative effectiveness of clustering versus social network analysis

A clustering model works by grouping similar documents (based on cosine distance) together with the relativity of the tokens based on the characteristics (tokens) of the data points using techniques like hierarchical clustering. A social network analysis works by analysing the relationship and connections between the data points by identifying the clusters present between the network using community detection techniques and centrality measures. Both techniques analyse the patterns within the documents where clustering groups based on similarity while network analysis focuses on interaction between data points.

Based on the Social Network Analysis (SNA) to identify the important group, we can see that the hierarchical clustering model and the bipartite network model both have the same clustering for all the groups, where documents of Climate Change, Food and Advertisement are within their respective clusters while Savings01.txt and Savings03.txt are within their own cluster as well as the documents of AI with Savings02.txt are within another cluster. Although SNA is generated based upon the relationship between the documents and the tokens where two tokens are connected if they appear in the same document and two documents are connected if they contain the same tokens while Hierarchical Clustering is generated by the cosine similarity between the documents with the token based upon the document term matrix, they were able to produce a similar output for both models. This suggests that the tokens chosen to classify the documents appears to be well segregated resulting in similar connection for both models. Although only a small number of tokens, 25 tokens have been used, in terms of SNA, the tokens might not have a complex community structure on the documents and contains well-defined tokens for each documents cluster which represents the similar structure based on similarity in the hierarchical clustering model that uses a distance metric to split the network to same communities. Furthermore, each topic areas of the documents might potentially have a strong set of dominant tokens that is able to distinguish between the groups well enough that similar pattern can be seen for the SNA and Hierarchical Clustering.

Natural Language processing

The first NLP technique that can be used to better distinguish between the documents processed is by using the lemmatization technique. Lemmatization is the process of reducing the words to the root for, but unlike stemming, it considers the context of the words to transform them into its meaningful root form. For instance, the words in past, present and future tense such as runs, ran and running would be lemmatized to run, while the words better and best would be lemmatized to good as its root form instead of the chopped prefix or suffix form as in stemming. It also considers the state of the words whether it is a verb or a noun when lemmatizing the words to its root form. This ensures that the context of the words is maintained as it uses the correct base form to be lemmatized instead of just removing prefixes or suffixes to produce words or tokens with no meaning. Furthermore, lemmatization also ensures that better performance can be seen for the NLP tasks especially during the document classification. Thus, lemmatization will be able to enhance the overall performance and accuracy of the text analysis. The downside of this approach is the longer time consumption in comparison to the stemming approach. This NLP technique of lemmatization can be generated using the “lemmatize_words” function in the library textstem in R.

The NLP technique introduced in 2018 that can be used is a sentence-embedding technique that is primarily used with BERT (Bidirectional Encoder Representations from Transformers). BERT is an NLP model designed to generate a model using an encoder mechanism. Bert processes the text in bidirectional manner to obtain a deeper understanding on the context and better capture the context of the words rather than relying on the words before and after only unlike traditional clustering that relies on the tokens count only without the importance to the contextual information of the words. BERT works by using a self-attention transformer for data processing which is efficient in handling complex dependencies between documents. a set of tokens are given as an input to the transformer that embeds them into vectors by weighing up the importance between different words and process it to a neural network (GeeksforGeeks, 2024). A sequence of vectors that represents the contextual meaning of the input tokens are generated as an output of the model. This offers a high-level tokenization that handles words in and out of the vocabulary based on the structure of the words to manage and distinguish complex words for improved text processing. The contextualized embeddings generated also manages to capture meaning of words based on the context rather than the meaning of the words only which provides high-level interpretation of the words. In the

context of document clustering, BERT can be used to fit the documents based on their tokens into high dimensional vector space to determine their semantic meaning thus improving the clustering. The “reticulate” and “keras_bert” package In R provides tools for implementation BERT NLP technique to tokenize the texts and generate the sentence embedding for the model to classify the documents into different categories to further enhance the accuracy of the modelling of clustering (Abdullayev, 2019).

The next NLP technique that can be used is Named Entity Recognition (NER). NER is an NLP model analyses textual data to identify the specific entities that categorizes the terms to a predefined class such as Person, Location, Day and many more. It is also essential in connecting the structured and unstructured data by extracting meaningful information. The process of converting raw data into meaningful categories that makes the data more suitable for information analysis based on the contextual meaning, unlike traditional clustering that relies on the tokens count only. The process of NER involves tokenization of the text into segments of words or phrases. Next, NER identifies the potential entities that the tokens might belong to by identifying the details or token patterns (Capitalization for Names / Locations) and formats (dates) and assign them to predefined context-dependent classes such as Person, Location or Organization (Awan, 2023). Furthermore, NER system also analyses the contextual information to identify the entities that distinguish between the words based on their context (i.e. Apple could refer to brand or fruit entity depending on the context) that might otherwise appear similar based on word counts alone in traditional methods. This ensures the meaning of the tokens in clustering depends on the textual definition instead of general interpretation, which improves the accuracy. This is because clustering algorithms can utilize this contextual information to group documents that share similar entity contexts. Furthermore, NER also reduces the dimensionality of the token space by grouping them into entities, which reduces the complexity while allowing the model to recognize the patterns between documents quickly and efficiently to determine the relationship between documents better. The “spacyr” and “quanteda” package in R provides tools for the implementation of NER techniques to tokenize and group the texts based on their entities with methods like “spacyr::spacy_parse” and “quanteda::entity_extract” to get the entities of the tokens identified (Using Spacyr for Named Entity Recognition - Inconsistent Results, n.d.).

Report of Text Analysis by Tokenization with R

By: Sarviin Hari

Appendix

	busi	buy	chang	consum	factor	fast	find	home	inv	job	peopl	process	reach	way	emerg	impact	natur	cost	result	small	futur	level	life	limit	risk
Advertisement01.txt	21	5	1	28	1	1	4	1	1	1	4	7	5	3	0	0	0	0	0	0	0	0	0	0	0
Advertisement02.txt	14	1	1	12	0	0	0	0	0	0	0	2	2	0	1	2	1	0	0	0	0	0	0	0	0
Advertisement03.txt	11	4	1	7	0	1	1	1	1	1	2	0	6	1	0	0	0	1	1	1	0	0	0	0	0
AI01.txt	11	1	6	1	1	0	3	2	1	7	2	28	3	8	2	0	6	6	5	1	5	1	1	8	2
AI02.txt	5	0	9	2	0	0	1	0	3	8	6	4	1	6	1	5	2	1	3	0	5	2	0	1	6
AI03.txt	8	0	2	1	0	0	0	0	2	1	3	9	0	6	1	3	1	1	2	0	2	3	1	2	8
ClimateChange01.txt	0	1	49	0	1	1	0	5	1	0	12	2	2	4	0	16	6	3	4	5	4	13	7	3	4
ClimateChange02.txt	1	0	18	0	1	0	0	2	0	0	6	0	2	0	0	2	3	0	0	1	1	4	0	3	2
ClimateChange03.txt	1	0	36	0	1	1	0	1	2	1	6	2	0	1	1	16	0	0	1	0	6	9	1	2	11
Food01.txt	0	0	2	2	2	1	3	0	0	0	18	2	0	2	1	1	0	1	2	1	0	3	1	0	1
Food02.txt	1	6	0	5	2	4	0	3	0	0	51	9	0	3	0	2	2	2	2	2	0	3	3	0	0
Food03.txt	0	2	0	2	2	2	3	0	0	0	14	6	1	0	1	1	1	0	0	2	0	1	3	2	4
Savings01.txt	4	0	0	0	0	0	2	1	1	1	1	0	1	1	4	1	0	1	0	2	3	0	5	0	3
Savings02.txt	2	3	1	1	0	1	3	1	3	3	2	2	0	5	0	1	0	5	2	2	1	0	0	2	0
Savings03.txt	1	1	1	1	1	1	3	2	7	1	3	0	0	5	10	0	1	2	0	0	1	1	4	8	1

Report of Text Analysis by Tokenization with R

By: Sarviin Hari

References

3 reasons why you should start saving money today (2024, February 2024). Discover. Retrieved May 31, 2024, from https://www.discover.com/online-banking/banking-topics/3-reasons-to-save-more-money/

Abdullayev, T. (2019, September 30). *Posit AI Blog: BERT from R*. Posit AI Blog. Retrieved June 5, 2024, from https://blogs.rstudio.com/ai/posts/2019-09-30-bert-r/

Awan, A. A. (2023, September 13). *What is Named Entity Recognition (NER)? Methods, Use Cases, and Challenges*. Retrieved June 5, 2024, from https://www.datacamp.com/blog/what-is-named-entity-recognition-ner

BBC News. (2024, February 8). *What is climate change? A really simple guide*. Retrieved May 23, 2024, from https://www.bbc.com/news/science-environment-24021772

What is the importance of advertising in business? [Full 2024 guide]. (n.d.). TimesPro. Retrieved May 23, 2024, from https://timespro.com/blog/what-is-the-importance-of-advertising-know-the-top-10-benefits

Climate change impacts. (n.d.). National Oceanic and Atmospheric Administration. Retrieved May 23, 2024, from https://www.noaa.gov/education/resource-collections/climate/climate-change-impacts

Daugherty, G. (2024, April 8). *How to save money for your big financial goals*. Investopedia. Retrieved May 21, 2024, from https://www.investopedia.com/how-to-save-money-4589942

GeeksforGeeks. (2024, January 10). *Explanation of BERT Model NLP*. GeeksforGeeks. Retrieved June 5, 2024, from https://www.geeksforgeeks.org/explanation-of-bert-model-nlp/

Hausman, A. (2024, January 6). *5 types of advertising to attract customers and convert them*. MKT Maven. Retrieved May 23, 2024, from https://www.hausmanmarketingletter.com/5-types-of-advertising-to-attract-customers-and-how-to-use-them/

Hetter, K. (2024, February 22). *It's not just what you eat, according to a doctor. It's when and how*. CNN. Retrieved May 23, 2024, from https://edition.cnn.com/2024/02/22/health/diet-food-heart-health-wellness/index.html

Lindwall (2022, October 24). *What are the effects of climate change?* NRDC. Retrieved May 23, 2024, from https://www.nrdc.org/stories/what-are-effects-climate-change#weather

Lockert, B. (n.d.). *How to save money*. Fortune Recommends. Retrieved May 21, 2024, from https://fortune.com/recommends/banking/how-to-save-money/

Report of Text Analysis by Tokenization with R

By: Sarviin Hari

National Geographic Society (2024, January 4). *Food*. National Geographic. Retrieved May 23, 2024, from https://education.nationalgeographic.org/resource/food/

O'Connor, A. (2021, December 21). How food affects mental health. *The New York Times*. Retrieved May 23, 2024, from https://www.nytimes.com/2021/05/06/well/eat/mental-health-food.html

Using spacyr for named entity recognition - inconsistent results. (n.d.). Stack Overflow. Retrieved June 5, 2024, from https://stackoverflow.com/questions/73850140/using-spacyr-for-named-entity-recognition-inconsistent-results

Wee, R. (2023, February 3). *The importance of advertising and why you should advertise - Equinet Academy*. Equinet Academy. Retrieved May 23, 2024, from https://www.equinetacademy.com/the-importance-of-advertising/

Report of Text Analysis by Tokenization with R

By: Sarviin Hari

R code

```
# install.packages("tm") # requires R 3.3.1 or later
# install.packages("slam")
# install.packages("SnowballC")
# install.packages("stringr")
# install.packages(c("igraph", "igraphdata"))

library(igraph)
library(igraphdata)
library(stringr)
library(slam) # for matrices and arrays
library(tm)
library(SnowballC) # for stemming

## Q2
rm(list = ls())

# creates a file path to the txt folder in CorpusAbstarcts
cname = file.path(".", "CorpusAbstracts", "txt")
cname

# displays the 15 files within the folder
dir(cname)
# create a Corpus with the directory's source to be the txt file path
docs = Corpus(DirSource((cname)))
# summary of all documents
summary(docs)

# Print the lengths of each document
doc_lengths <- sapply(docs, function(doc) {
  length(unlist(strsplit(as.character(doc), "\\W+")))
})
doc_lengths
```

Report of Text Analysis by Tokenization with R

By: Sarviin Hari

Q3

```
# data cleaning by pattern of repeating words from short to long form
toAI <- content_transformer(function(x, pattern) gsub(pattern, "Artificial Intelligence", x))
docs <- tm_map(docs, toAI, 'AI')

toML <- content_transformer(function(x, pattern) gsub(pattern, "Machine Learning", x))
docs <- tm_map(docs, toAI, 'ML')

toDL <- content_transformer(function(x, pattern) gsub(pattern, "Deep Learning", x))
docs <- tm_map(docs, toAI, 'DL')



# Tokenization
docs <- tm_map(docs, removeNumbers)
docs <- tm_map(docs, removePunctuation)
docs <- tm_map(docs, content_transformer(tolower))



# filter words - remove stop words and white space
docs <- tm_map(docs, removeWords, stopwords("english"))
docs <- tm_map(docs, stripWhitespace)

# filter words - stemming
docs <- tm_map(docs, stemDocument, language = "english")

# remove the approstrophies
toSpace <- content_transformer(function(x, pattern) gsub(pattern, " ", x))

# convert to document term matrix
dtm0 <- DocumentTermMatrix(docs)
# get the top most values and top min values count
```

Report of Text Analysis by Tokenization with R

By: Sarviin Hari

```
freq0 <- colSums(as.matrix(dtm0))

length(freq0)

ord0 = order(freq0)

freq0[head(ord0, 30)]

freq0[tail(ord0, 50)]




# freqeuncy of frequencies

head(table(freq0), 10)

tail(table(freq0), 10)


# Remove hyphens

docs <- tm_map(docs, toSpace, "-")

docs <- tm_map(docs, toSpace, "—")

docs <- tm_map(docs, toSpace, "’")

docs <- tm_map(docs, toSpace, '”')

docs <- tm_map(docs, toSpace, '“')


# out of top 50 highest I remove the words that dont seem to have any importance in classification

docs <- tm_map(docs, toSpace, 'can')

docs <- tm_map(docs, toSpace, 'use')

docs <- tm_map(docs, toSpace, 'also')

docs <- tm_map(docs, toSpace, 'like')

docs <- tm_map(docs, toSpace, 'includ')

docs <- tm_map(docs, toSpace, 'data')

docs <- tm_map(docs, toSpace, 'make')

docs <- tm_map(docs, toSpace, 'one')

docs <- tm_map(docs, toSpace, 'will')

docs <- tm_map(docs, toSpace, 'exampl')

docs <- tm_map(docs, toSpace, 'need')

docs <- tm_map(docs, toSpace, 'new')

docs <- tm_map(docs, toSpace, 'may')
```

Report of Text Analysis by Tokenization with R

By: Sarviin Hari

```
docs <- tm_map(docs, toSpace, "’s")

docs <- tm_map(docs, toSpace, 'mani')

docs <- tm_map(docs, toSpace, 'get')

docs <- tm_map(docs, toSpace, 'generat')

docs <- tm_map(docs, toSpace, 'custom')

docs <- tm_map(docs, toSpace, 'much')


# convert to document term matrix

dtm <- DocumentTermMatrix(docs)

dim(dtm)

dtms <- removeSparseTerms(dtm, 0.25)

dim(dtms)

as.matrix(dtms)


# from the sparse matrix I remove the words that dont seem to have any importance in classification

docs <- tm_map(docs, toSpace, "amount")

docs <- tm_map(docs, toSpace, "around")

docs <- tm_map(docs, toSpace, "come")

docs <- tm_map(docs, toSpace, "even")

docs <- tm_map(docs, toSpace, "first")

docs <- tm_map(docs, toSpace, "help")

docs <- tm_map(docs, toSpace, "just")

docs <- tm_map(docs, toSpace, "mean")

docs <- tm_map(docs, toSpace, "import")

docs <- tm_map(docs, toSpace, "increas")

docs <- tm_map(docs, toSpace, "right")

docs <- tm_map(docs, toSpace, "still")

docs <- tm_map(docs, toSpace, "time")

docs <- tm_map(docs, toSpace, "well")

docs <- tm_map(docs, toSpace, "take")


# docs <- tm_map(docs, toSpace, 'build')

docs <- tm_map(docs, toSpace, 'certain')
```

Report of Text Analysis by Tokenization with R

By: Sarviin Hari

```
docs <- tm_map(docs, toSpace, 'work')
docs <- tm_map(docs, toSpace, 'year')



# convert to document term matrix
dtm <- DocumentTermMatrix(docs)
dim(dtm)
dtms <- removeSparseTerms(dtm, 0.3)
dim(dtms)
as.matrix(dtms)

# from the sparse matrix I remove the words that dont seem to have any importance in classification
docs <- tm_map(docs, toSpace, "across")
docs <- tm_map(docs, toSpace, "howev")
docs <- tm_map(docs, toSpace, "live")
docs <- tm_map(docs, toSpace, "look")
docs <- tm_map(docs, toSpace, "often")
docs <- tm_map(docs, toSpace, "provid")
docs <- tm_map(docs, toSpace, "thing")
docs <- tm_map(docs, toSpace, "year")
docs <- tm_map(docs, toSpace, "contribut")
docs <- tm_map(docs, toSpace, "consid")
docs <- tm_map(docs, toSpace, "found")
docs <- tm_map(docs, toSpace, "less")
docs <- tm_map(docs, toSpace, "major")
docs <- tm_map(docs, toSpace, "differ")
docs <- tm_map(docs, toSpace, "everi")  # everyone, every



# convert to document term matrix
dtm <- DocumentTermMatrix(docs)
dim(dtm)
dtms <- removeSparseTerms(dtm, 0.35)
```

Report of Text Analysis by Tokenization with R

By: Sarviin Hari

```
dim(dtms)

as.matrix(dtms)


# from the sparse matrix I remove the words that dont seem to have any importance in classification

docs <- tm_map(docs, toSpace, "abl")  # able

docs <- tm_map(docs, toSpace, "accord")  # according

docs <- tm_map(docs, toSpace, "avail")  # available

docs <- tm_map(docs, toSpace, "build") ############################

docs <- tm_map(docs, toSpace, "continu")  # continue

docs <- tm_map(docs, toSpace, "effect") # KIV

docs <- tm_map(docs, toSpace, "good")

docs <- tm_map(docs, toSpace, "keep")

docs <- tm_map(docs, toSpace, "larg")  # large

docs <- tm_map(docs, toSpace, "part")  # part of ...

docs <- tm_map(docs, toSpace, "place") ############################

docs <- tm_map(docs, toSpace, "put")

docs <- tm_map(docs, toSpace, "set")

docs <- tm_map(docs, toSpace, "start")

docs <- tm_map(docs, toSpace, "three")

docs <- tm_map(docs, toSpace, "within")

docs <- tm_map(docs, toSpace, "avoid")

docs <- tm_map(docs, toSpace, "type")

docs <- tm_map(docs, toSpace, "creat")

docs <- tm_map(docs, toSpace, "day")

docs <- tm_map(docs, toSpace, "might")

docs <- tm_map(docs, toSpace, "number")

docs <- tm_map(docs, toSpace, "see")

docs <- tm_map(docs, toSpace, "quick")



# convert to document term matrix

# from the sparse matrix there is very small amount of tokens so none are removed

dtm <- DocumentTermMatrix(docs)
```

Report of Text Analysis by Tokenization with R

By: Sarviin Hari

```
dim(dtm)

dtms <- removeSparseTerms(dtm, 0.4)

dim(dtms)

as.matrix(dtms)



# convert to document term matrix

dtm <- DocumentTermMatrix(docs)

dim(dtm)

dtms <- removeSparseTerms(dtm, 0.45)

dim(dtms)

as.matrix(dtms)


# from the sparse matrix I remove the words that dont seem to have any importance in classification

docs <- tm_map(docs, toSpace, "est")

docs <- tm_map(docs, toSpace, "high")

docs <- tm_map(docs, toSpace, "know")

docs <- tm_map(docs, toSpace, "long")

docs <- tm_map(docs, toSpace, "requir")

docs <- tm_map(docs, toSpace, "whether")

docs <- tm_map(docs, toSpace, "sinc")

docs <- tm_map(docs, toSpace, "adapt")

docs <- tm_map(docs, toSpace, "opportun")

docs <- tm_map(docs, toSpace, "addit")  # in addition

docs <- tm_map(docs, toSpace, "better")

docs <- tm_map(docs, toSpace, "now")

docs <- tm_map(docs, toSpace, "big")

docs <- tm_map(docs, toSpace, "though")

docs <- tm_map(docs, toSpace, "anoth")

docs <- tm_map(docs, toSpace, "decis")

docs <- tm_map(docs, toSpace, "follow")

docs <- tm_map(docs, toSpace, "imag")

docs <- tm_map(docs, toSpace, "posit")
```

Report of Text Analysis by Tokenization with R

By: Sarviin Hari

```
docs <- tm_map(docs, toSpace, "possibl")
docs <- tm_map(docs, toSpace, "reli")
docs <- tm_map(docs, toSpace, "sinc")
docs <- tm_map(docs, toSpace, "toward")
docs <- tm_map(docs, toSpace, "among")
docs <- tm_map(docs, toSpace, "sever")
docs <- tm_map(docs, toSpace, "world")
docs <- tm_map(docs, toSpace, "reduc")
docs <- tm_map(docs, toSpace, "light")
docs <- tm_map(docs, toSpace, "key")
docs <- tm_map(docs, toSpace, "becom")
docs <- tm_map(docs, toSpace, "base")
docs <- tm_map(docs, toSpace, "social")
docs <- tm_map(docs, toSpace, "focus")
docs <- tm_map(docs, toSpace, "public")

# convert to final document term matrix
dtm <- DocumentTermMatrix(docs)
dim(dtm)
dtms <- removeSparseTerms(dtm, 0.45)
dim(dtms)
inspect(dtms)
as.matrix(dtms)

## Q4

# Save the dtm matrix to a CSV file
dtms = as.matrix(dtms)
write.csv(dtms, "dtms.csv")

# Compute the cosine distance matrix for the document-term matrix and plot dendogram
distmatrix = proxy::dist(dtms, method = "cosine")
fit = hclust(distmatrix, method = "ward.D")
```

Report of Text Analysis by Tokenization with R

By: Sarviin Hari

```
plot(fit, hang = -1, main = "Abstracts Cosine Distance")

# Create vector of topic labels in same order as corpus
topics = c("adv", "adv", "adv", "ai", "ai", "ai", "clim", "clim", "clim", "food", "food", "food", "sav", "sav", "sav")

# Cut the dendogram to create the required number of clusters (k = 5) and plot
groups = cutree(fit, k = 5)
rect.hclust(fit, k=5, border=2:6)

# Create a table of topic labels vs cluster numbers
clust_table = table(GroupNames = topics, Clusters = groups)
print(clust_table)

# convert table to data frame and prints the accuarcy
TA = as.data.frame.matrix(table(GroupNames = topics, Clusters = groups))
TA

# Calculate the accuracy as the sum of the diagonal elements of the table divided by the total number of documents
accuracy = (TA[1,1] + TA[2,2] + TA[3,3] + TA[4,4] + TA[5,5]) / 15 * 100
print(accuracy)

## Q5

# Base Graph
# Convert the document-term matrix to a matrix
dtmsx = as.matrix(dtms)

# Compute the product of the matrix and its transpose to get the adjacency matrix and remove self-loops
ByAbsMatrix = dtmsx %*% t(dtmsx)
diag(ByAbsMatrix) = 0

# Create an undirected graph from the adjacency matrix with weights
set.seed(32885741)
```

```
ByAbs = graph_from_adjacency_matrix(ByAbsMatrix, mode = "undirected", weighted = TRUE)
plot(ByAbs)

# Calculate the degree, closeness and eigenvector of each vertex in dataframe
degree = degree(ByAbs)
closeness = format(closeness(ByAbs), digits = 4)
eig = format(as.table(evcent(ByAbs)$vector), digits = 4)
ksum = as.data.frame(cbind(degree, closeness, eig))
ksum


# Improved Graph
# Create an undirected graph from the adjacency matrix with weights (base graph)
set.seed(32885741)
ByAbs = graph_from_adjacency_matrix(ByAbsMatrix, mode = "undirected", weighted = TRUE)

# Set the size of each vertex  to its eigenvector centrality
V(ByAbs)$size <- eigen_centrality(ByAbs)$vector*20

# Set the width of each edge to its weight
E(ByAbs)$width <- E(ByAbs)$weight*0.01

# Perform community detection using the fast greedy algorithm and plot
cfb = cluster_fast_greedy(ByAbs)
plot(cfb, ByAbs,vertex.label=V(ByAbs)$role,main="Fast Greedy")


## Q6

# Base Graph
# Convert the document-term matrix to a matrix
dtmsa = as.matrix(dtms)
```

Report of Text Analysis by Tokenization with R

By: Sarviin Hari

```
# Compute the product of the matrix and its transpose to get the adjacency matrix and remove self-loops
ByTokenMatrix = t(dtmsa) %*% dtmsa
diag(ByTokenMatrix) = 0

# Create an undirected graph from the adjacency matrix with weights
set.seed(32885741)
ByToken = graph_from_adjacency_matrix(ByTokenMatrix, mode = "undirected", weighted = TRUE)
plot(ByToken)

# Calculate the degree, closeness and eigenvector of each vertex in dataframe
degree = degree(ByToken)
closeness = format(closeness(ByToken), digits = 4)
eig = format(as.table(evcent(ByToken)$vector), digits = 4)
ksum = as.data.frame(cbind(degree, closeness, eig))
ksum

# Improved Graph
set.seed(32885741)
ByToken = graph_from_adjacency_matrix(ByTokenMatrix, mode = "undirected", weighted = TRUE)

# Set the size of each vertex  to its eigenvector centrality
V(ByToken)$size <- eigen_centrality(ByToken)$vector*30

# Set the width of each edge to its weight
E(ByToken)$width <- E(ByToken)$weight*0.02


# plot the graph of vertices and edges updated again
plot(ByToken)

# Perform community detection using the fast greedy algorithm and plot
cfb = cluster_fast_greedy(ByToken)
plot(cfb, ByToken,vertex.label=V(ByToken)$role,main="Fast Greedy")
```

Report of Text Analysis by Tokenization with R

By: Sarviin Hari

```
## Q7

# Convert the document-term matrix to a data frame

dtmsa = as.data.frame(dtms) # clone dtms


# Add row names as a new column named 'ABS' (Abstract)

dtmsa$ABS = rownames(dtmsa) # add row names


# Loop through each row and column of the data frame and create a new row with the weight, abstract, and token information

dtmsb = data.frame()

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 ) } } # close loops


# Rename the columns of dtmsb

colnames(dtmsb) = c("weight", "abs", "token")


# Filter out rows with a weight of 0 and rearrange the columns

dtmsc = dtmsb[dtmsb$weight != 0,] # delete 0 weights

dtmsc = dtmsc[,c(2,3,1)]

dtmsc


# Create a Bipartite plot graph from the data frame with edge weights

set.seed(32885741)

g <- graph.data.frame(dtmsc, directed=FALSE)


# Check the bipartite mapping of the graph and set it as the type of the graph

bipartite.mapping(g)
```

```
V(g)$type <- bipartite_mapping(g)$type

# Set vertex color and shape based on type: lightgreen/pink or circle/square
V(g)$color <- ifelse(V(g)$type, "lightgreen", "pink")
V(g)$shape <- ifelse(V(g)$type, "circle", "square")

# Set the edge color to black
E(g)$color <- "black"

# plot the base graph
plot(g)

# Calculate the degree, closeness and eigenvector of each vertex in dataframe
degree = degree(g)
closeness = format(closeness(g), digits = 4)
eig = format(as.table(evcent(g)$vector), digits = 4)
ksum = as.data.frame(cbind(degree, closeness, betweenness, eig))
ksum


# Improved Graph
set.seed(32885741)
g <- graph.data.frame(dtmsc, directed=FALSE)
bipartite.mapping(g)
V(g)$type <- bipartite_mapping(g)$type
V(g)$color <- ifelse(V(g)$type, "lightgreen", "pink")
V(g)$shape <- ifelse(V(g)$type, "circle", "square")
E(g)$color <- "black"

# Set the vertex size to its degree
V(g)$size <- degree(g)

# Set the edge width to its weight
```

Report of Text Analysis by Tokenization with R

By: Sarviin Hari

```
E(g)$width <- E(g)$weight


# Compute the layout for a bipartite graph and swap the columns
LO = layout_as_bipartite(g)
LO = LO[,c(2,1)]


# Scale the bipartite layout by the scaling factor
scaling_factor <- 3
layout_bipartite <- layout_as_bipartite(g)
layout_bipartite_scaled <- layout_bipartite * scaling_factor


# Community detection using the fast greedy algorithm and plot the graph
cfb = cluster_fast_greedy(g)
plot(cfb, g,vertex.label=V(g)$role,main="Fast Greedy")
```