TEXT MINING AND SENTIMENT ANALYSIS OF MULTI-GENRE CORPUS

Name: Chai Shou Zheng
Last Modified: 1:32pm 14/6/2025

Q1) Document Collection and Genre Labelling

I collected a total of 21 machine-readable text documents, each containing between 300 and 1300 words. The documents were sourced from reputable websites and saved in plain .txt format for compatibility with text mining tools in R

The three genres I selected are:

1. **Smartphone Reviews**
   Articles reviewing newly released smartphones, focusing on specifications, performance, camera quality, and battery life.
2. **Sports Articles**
   Reports and analyses of recent sports events.
3. **Car Reviews**
   Informative reviews discussing various aspects of new car models, including performance, safety features, fuel efficiency, design, and technology.

Q2) Corpus Creation Process

To create the corpus, I followed the folder-based method demonstrated in the lectures. The goal was to compile a well-organized, machine-readable text collection consisting of documents from three clearly defined genres: **Smartphone Reviews**, **Sports Articles**, and **Car Reviews**

1. **Document Collection**
   I manually sourced a total of 21 documents (7 per genre) by visiting reputable websites that publish high-quality content related to each genre. Each document was selected to ensure it contained average word from 300 – 1300 and featured **genre-specific vocabulary** to facilitate clustering and text analysis. This range provides enough content for meaningful text analysis while avoiding overly short or excessively long documents that could reduce clustering accuracy or introduce noise.

2. **Formatting and Saving**

   Since the source content was web-based, I copied the main article text (excluding ads, headers, and comments) from each page. I then pasted each document into Microsoft Word and saved each file in .txt format with UTF-8 encoding, which is machine-readable and suitable for text mining.

   To ensure document traceability, each file was named using a consistent and descriptive identifier:

   - Smartphone_review1.txt, Smartphone_review2.txt ... for Smartphone Reviews
   - Sports_article1.txt, Sports_article2.txt ... for Sports Articles
   - Car_review1.txt, Car_review2.txt, ... for Car Reviews

3. **Organizing into Corpus Folders**

   I created a main directory titled **corpus/**, and within it, I added three subfolders corresponding to each genre:

   corpus/

   ├——— Car_Reviews/

   ├——— Smartphone_Reviews/

   └——— Sports_Articles/

   Each .txt file was placed into the appropriate subfolder based on its genre.

4. **Quality Control**

   After saving each document, I manually reviewed each .txt file to ensure:

   - There were no HTML tags, image captions, or unrelated navigation content.
   - The content reflected its genre accurately and used genre-relevant vocabulary (e.g., "battery" for smartphone reviews, "points" for sports "horsepower" or "safety features" for car reviews).

Q3) Text Cleaning and Preprocessing

Before constructing the Document-Term Matrix (DTM), several text preprocessing steps were applied to clean and standardize the documents. These transformations aimed to reduce noise, normalize language, and retain only informative tokens that contribute meaningfully to analysis and clustering.

1. **Lowercasing**
   - All text was converted to lowercase to ensure consistent token representation (e.g., "Phone" and "phone" are treated the same)

2. **Number and Punctuation Removal**
   - Numeric characters and punctuation were removed to eliminate formatting artefacts and irrelevant symbols.

3. **Stopword Removal**:
   - Common English stopwords (e.g., "the", "is", "and") were removed.
   - In addition, we defined a custom list of extra stopwords (e.g., "use", "get", "make", "arent", "dont", "wont") that are common across all genres but do not contribute meaningfully to clustering.

4. **Whitespace Stripping**:
   - Extra whitespaces were removed to clean up the text format.

5. **Stemming**
    - Stemming was applied using stemDocument, reducing words to their base form (e.g., "running" → "run"), which helps consolidate semantically similar terms and reduce dimensionality.

Document-Term Matrix (DTM) Construction

```
<<DocumentTermMatrix (documents: 21, terms: 1398)>>
Non-/sparse entries: 3032/26326
Sparsity           : 90%
Maximal term length: 20
Weighting          : term frequency (tf)
```

The resulting DTM had:

- 21 documents (rows)
- 3,032 non-zero entries and 26,326 zero (sparse) entries, resulting in a sparsity of 90%
- The maximum term length is 20 characters

Handling Sparsity

A high sparsity level indicates that most terms appear in only a few documents, which can affect clustering performance by introducing noise and computational overhead.

Action Taken:

To address this, we first removed standard English stopwords using the stopwords("english") list. However, we noticed that certain additional words—such as *"use"*, *"get"*, *"make"*, and *"need"*—still appeared frequently across all genres but did not meaningfully contribute to clustering. These extra stopwords were therefore also removed to enhance the distinctiveness of genre-specific vocabulary.

After stopword removal, we applied the removeSparseTerms() function. After trial and error with different threshold values, sparsity threshold of 0.65 was found to be a suitable balance—it reduced the number of terms significantly (from 1398 to 23), while still preserving enough informative vocabulary to support meaningful clustering and genre differentiation.

Result:

```
> # Step 4: Remove sparse terms
> dtm_trimmed <- removeSparseTerms(dtm, 0.65)
> dtm_trimmed
<<DocumentTermMatrix (documents: 21, terms: 23)>>
Non-/sparse entries: 205/278
Sparsity           : 58%
Maximal term length: 8
Weighting          : term frequency (tf)
```

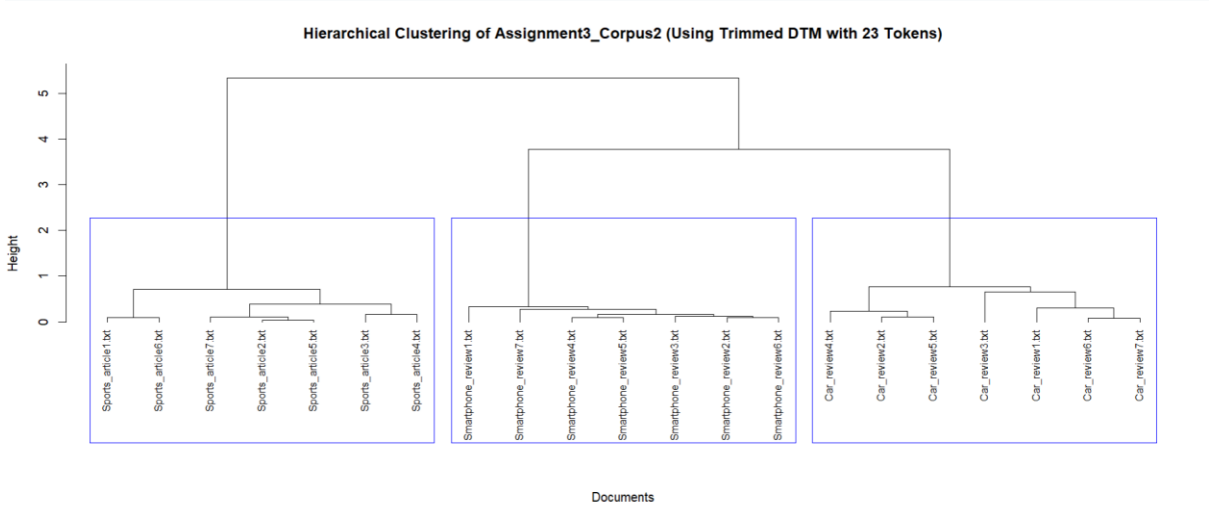
- The DTM was reduced to 23 terms/tokens, significantly lowering dimensionality.
- The new sparsity dropped to 58%, making the matrix denser and more suitable for clustering.

```
> colnames(as.matrix(dtm_trimmed)) # Shows all the terms (columns)
 [1] "batteri"  "charger"  "front"    "perform"
 [5] "point"    "power"    "price"    "rear"
 [9] "specif"   "speed"    "top"      "volum"
[13] "weight"   "aluminum" "display"  "first"
[17] "spec"     "camera"   "glass"    "back"
[21] "final"    "now"      "seri"
```

- These terms are **frequent and representative** across the corpus, enhancing the ability to group documents based on shared content, improving both clustering performance and interpretability.
- They were used to construct the **trimmed Document-Term Matrix**, which forms the basis for calculating cosine distance and performing **hierarchical clustering**.

Q4) Hierarchical Clustering and Dendrogram

To analyze the similarity between documents, hierarchical clustering was performed using the cosine distance between each document. As shown in the dendrogram, the documents were grouped into three distinct clusters, which correspond closely to the original genres: Car Reviews, Smartphone Reviews, and Sports Articles.



To evaluate the accuracy of the clustering, each document was assigned a genre label (Car_Reviews, Smartphone_Reviews, Sports_Articles) based on its original folder. These labels were then compared to the clustering result using a confusion matrix:

	Cluster 1	**Cluster 2**	**Cluster 3**
Car_Reviews	7	0	0
Smartphone_Reviews	0	7	0
Sports_Articles	0	0	7

Each genre was perfectly grouped into its own cluster. Accuracy was calculated by summing the number of correctly classified documents (maximum per row in the confusion matrix) and dividing by the total number of documents:

$$\text{Accuracy} = \frac{7+7+7}{21} = 1.00 = 100\%$$

This result indicates that the trimmed DTM was highly effective at distinguishing between genres (Car_Reviews, Smartphone_Reviews, Sports_Articles) using shared vocabulary. The clustering process successfully captured the genre-specific language patterns across all documents.

Qualitative Description of Clustering Quality

The clustering quality was excellent, as evidenced by the dendrogram and the perfect separation of documents into their respective genres. Each cluster grouped documents that shared distinctive vocabulary patterns consistent with their genre.

```
> # Get top 10 terms for each genre
> top_n_terms(genre_term_freq, "Car_Reviews")
    top   volum   speed   front perform    rear
     21      16      15      13      12      12
  power   price  weight  specif
     11       9       9       7
```

For example, documents in the Car_Reviews cluster frequently included terms like “top” (21), “volum” (16), “speed” (15), “front” (13), and “rear” (12)—all of which are common in automotive discussions and car review formats. These terms reflect specifications and driving experience commonly highlighted in car reviews.

```
> top_n_terms(genre_term_freq, "Smartphone_Reviews")
  camera    glass    front  display      now     back
      29       19       14       13       12        9
    spec     rear  batteri aluminum
       9        8        7        6
```

In contrast, the Smartphone_Reviews cluster was dominated by terms such as “camera” (29), “glass” (19), “display” (13), and “battery” (7), reflecting core features reviewed in mobile technology content.

```
> top_n_terms(genre_term_freq, "Sports_Articles")
  point   final    seri   first perform     now
     38      27      27      16       6       3
   back  specif   front   volum
      2       2       1       1
```

The Sports_Articles cluster stood out through its use of terms like “point” (38), “final” (27), “seri” (27), and “first” (16), which clearly reflect competition and match-related reporting typical in sports journalism.

These genre-specific word patterns enabled the clustering algorithm to separate the documents with high precision. The low intra-cluster distances and high inter-cluster separation observed in the dendrogram reinforce the strong cohesion within clusters and clear boundaries between them. This qualitative evidence aligns perfectly with the 100% clustering accuracy, confirming that the grouping is both numerically and semantically meaningful.

To further validate the clustering results, we created an augmented Document-Term Matrix (in appendix) by appending genre labels and cluster assignments to the trimmed DTM. This table was sorted by cluster ID to analyze the token distributions across groups. The result, saved as Augmented_DTM.csv, confirms strong separation between genres. No documents were misclassified, and token patterns such as the frequent use of "camera" in Smartphone_Reviews or "torque" in Car_Reviews remained cluster-specific. This analysis supports the qualitative and quantitative accuracy of the clustering.

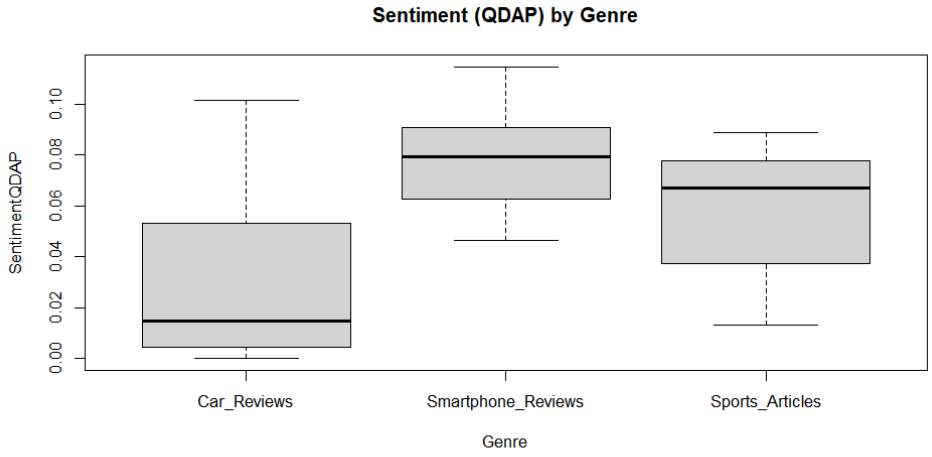
Q5) Sentiment Analysis

We used the SentimentAnalysis package in R, which applies several established sentiment dictionaries to assess and compare sentiment levels across genres such as Car Reviews, Smartphone Reviews, and Sports Articles. The results were analyzed visually using boxplots and statistically using t-tests to compare sentiment scores across genres.

Our primary focus was on two dictionaries:

1. QDAP: We choose QDAP as it is more general and design to compute general positivity and sentiment scores for quantitative analysis (SentimentQDAP)
2. GI (General Inquirer): Used for broader sentiment classification, allowing us to break down sentiment into components such as overall sentiment (NegativityGI).

Boxplot of SentimentQDAP



Genre	Median Sentiment	Range & Spread	Interpretation
Car_Reviews	Lowest (~0.01)	Narrow range, several	This suggests car reviews are mostly neutral or minimally

		documents near 0	positive, consistent with technical and objective writing.
Smartphone_Reviews	Highest (~0.08)	Tightly clustered around higher sentiment values	This genre is consistently more positive, likely due to promotional language used in tech reviews.
Sports_Articles	~0.07	Wider spread in values	Wider spread than Smartphone_Reviews, indicating greater variability in tone (possibly due to contrasting match outcomes or emotional storytelling).

Interpretation

- Smartphone_Reviews are the most positively toned genre according to QDAP, as tech reviews often highlight features and improvements.
- Sports_Articles are also positive but more varied. Sports articles swing between praise and critique.
- Car_Reviews are the least positive, with many documents scoring close to neutral (0). Car reviews tend to be more evaluative and balanced.

Performing T-test for SentimentQDAP

Mean SentimentQDAP Scores by Genre

Genre	**Mean SentimentQDAP Scores**	**Interpretation**
Car_Reviews	0.0331	Car Reviews have the lowest sentiment (0.0331), indicating a more neutral or objective tone, which is expected for technical product assessments.
Smartphone_Reviews	0.0783	Smartphone Reviews have the highest mean sentiment score (0.0783), suggesting they are generally written in a more positive tone, possibly due to promotional or marketing language.
Sports_Articles	0.0571	Sports Articles have a moderate score (0.0571), showing a mix of emotional tones, likely due to varying match outcomes and storytelling.

SentimentQDAP T-test Comparison Table

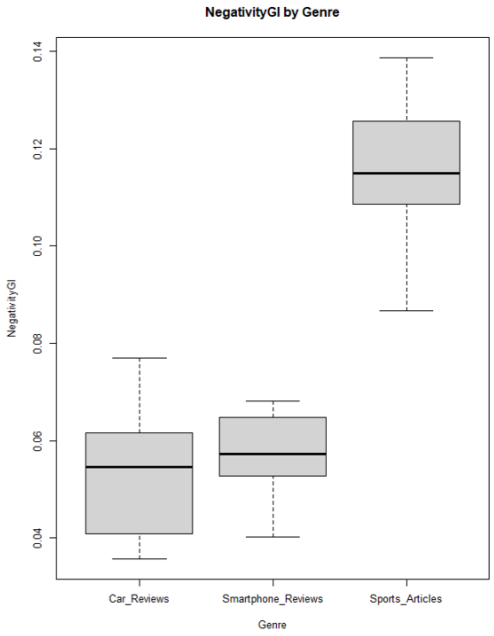
Comparison	**p-value**	**Interpretation**
Car Reviews vs Smartphone Reviews	0.9869	No evidence that Car Reviews are more positive

Car Reviews vs Sports Articles	0.8895	No evidence that Car Reviews are more positive
Smartphone Reviews vs Sports Articles	0.0784	Moderate evidence that Smartphone Reviews may be more positive (not significant)

Conclusion for SentimentQDAP

Overall, there is no statistically significant difference in SentimentQDAP scores between the genres, although Smartphone_Reviews appear to have the highest mean sentiment. This indicates that while Smartphone_Reviews tend to be more positive in tone—possibly due to promotional language—the observed differences are not strong enough to conclude a real difference statistically.

Boxplot of NegativityGI



Genre	Mean NegativityGI	Interpretation
Car_Reviews	0.0532	Car Reviews have the lowest negativity, aligning with their neutral and technical nature.
Smartphone_Reviews	0.0572	Smartphone Reviews also show low negativity, consistent with generally positive and promotional tone.
Sports_Articles	0.1155	Sports articles have the highest negativity score. This is likely because they include emotional words, especially when talking about wins, losses, or dramatic moments.

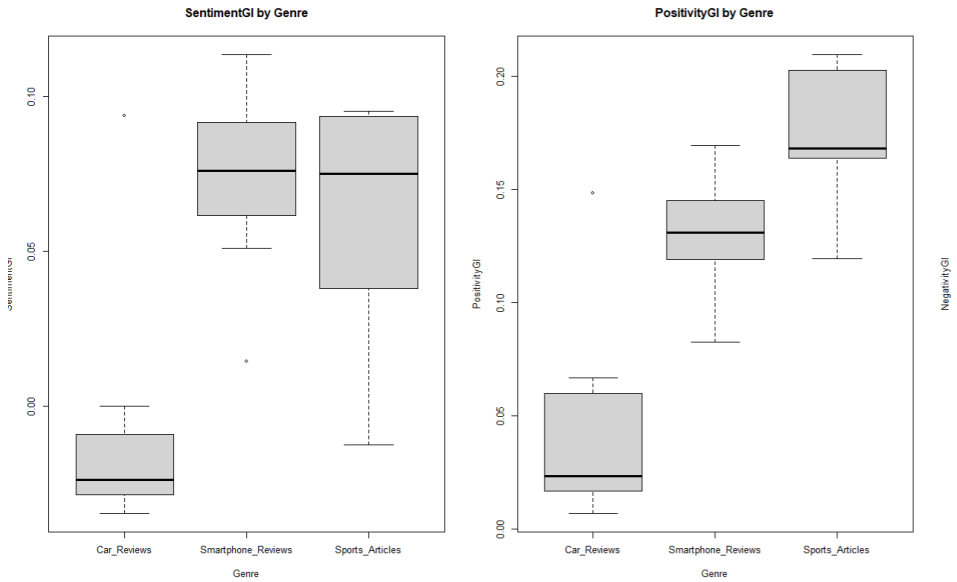
NegativityGI T-test Comparison Table

Comparison	p-value	Interpretation
Car Reviews vs Smartphone Reviews	0.718	No evidence that Car Reviews are more negative than Smartphone Reviews.
Car Reviews vs Sports Articles	1.000	No evidence that Car Reviews are more negative than Sports Articles.
Smartphone Reviews vs Sports Articles	1.000	No evidence that Smartphone Reviews are more negative than Sports Articles.

Conclusion for NegativityGI:

Overall, there is no statistically significant difference in NegativityGI scores between the genres. However, Sports Articles show the highest average negativity, which is likely due to emotional or dramatic language often used in reporting wins, losses, or intense moments in sports. In contrast, Car Reviews and Smartphone Reviews have lower negativity scores, reflecting their more neutral or promotional writing styles. Despite these differences in average scores, the t-tests show that these variations are not strong enough to be considered statistically significant.

Boxplot of SentimentGI and PositivityGI



SentimentGI

- Car_Reviews have the lowest sentiment values overall, centered just below 0, even dipping slightly into the negative range.
- Smartphone_Reviews show moderate sentiment, centered above 0.05.
- Sports_Articles show the highest overall sentiment, with a wide spread and a median close to 0.07–0.08.

Interpretation: Sports articles are generally more positive than the other two genres, and car reviews tend to be the least positive, sometimes even neutral or slightly negative.

PositivityGI

- Car_Reviews have the lowest positivity, with values clustering near 0.03–0.06.
- Smartphone_Reviews show moderate positivity, around 0.12.
- Sports_Articles show the highest positivity, with a tight spread around 0.17–0.21.

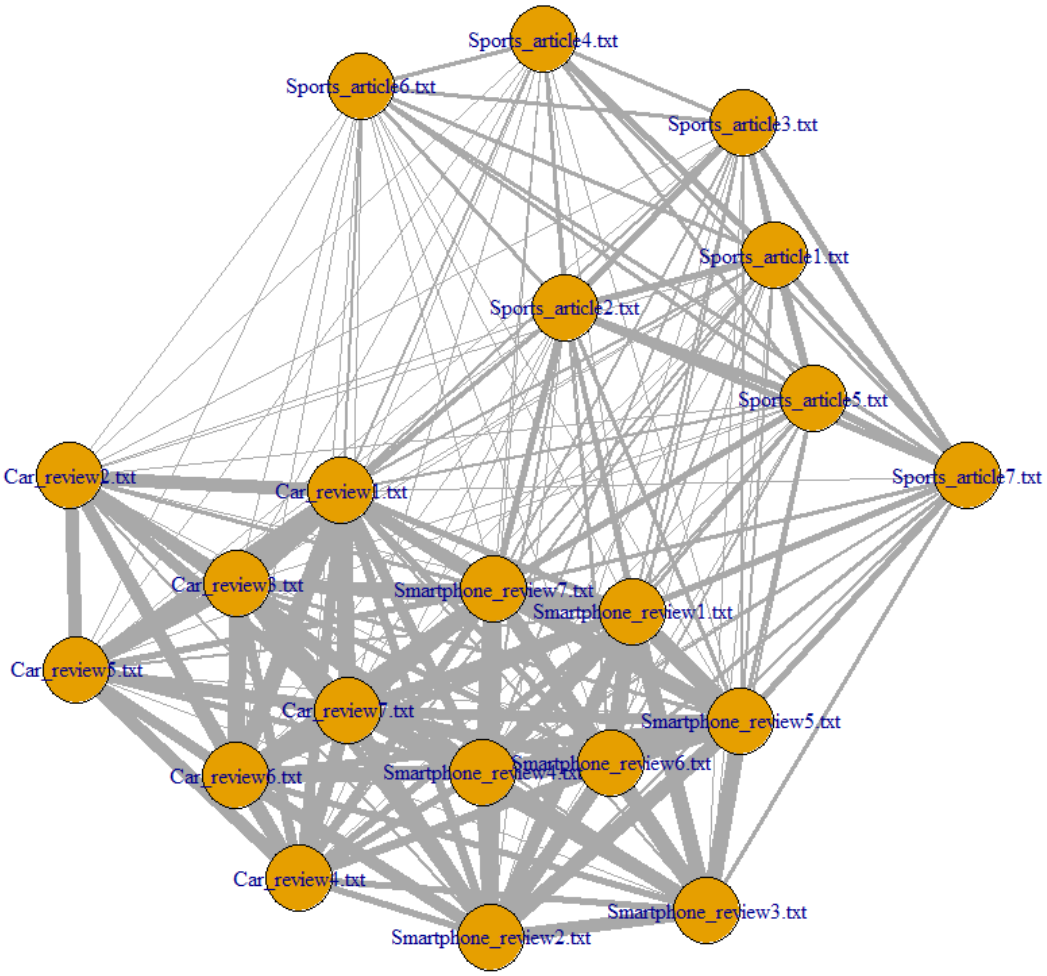
Interpretation: Sports articles use more positively connoted words. Car reviews are comparatively neutral or restrained in tone.

Overall Summary:

- Sports_Articles are the most emotionally expressive — they score highest in both positivity and negativity, leading to higher overall sentiment.
- Smartphone_Reviews are moderately positive with low negativity.
- Car_Reviews are generally neutral to slightly negative, with low positivity and negativity, indicating a more objective or technical tone.

Q6) Single-Mode Network (Documents)

Basic Network Graph (Documents)



Observation

The network shows **two main clusters**: one consisting of both Smartphone Reviews and Car Reviews, and the other consisting of Sports Articles.

The lower portion of the network shows a highly dense cluster made up primarily of Car Reviews and Smartphone Reviews. These documents form a highly interconnected cluster in the lower half of the network. These documents share a large number of terms with one another, as indicated by the dense web of thick edges. This suggests a strong lexical similarity between the two genres, likely due to shared technical and descriptive language commonly found in product reviews.

In contrast, the Sports Articles (green nodes) form a separate and less dense cluster in the upper portion of the network. While they are still connected to the other documents, their edges are generally thinner and less frequent, indicating a more specialized vocabulary that is less

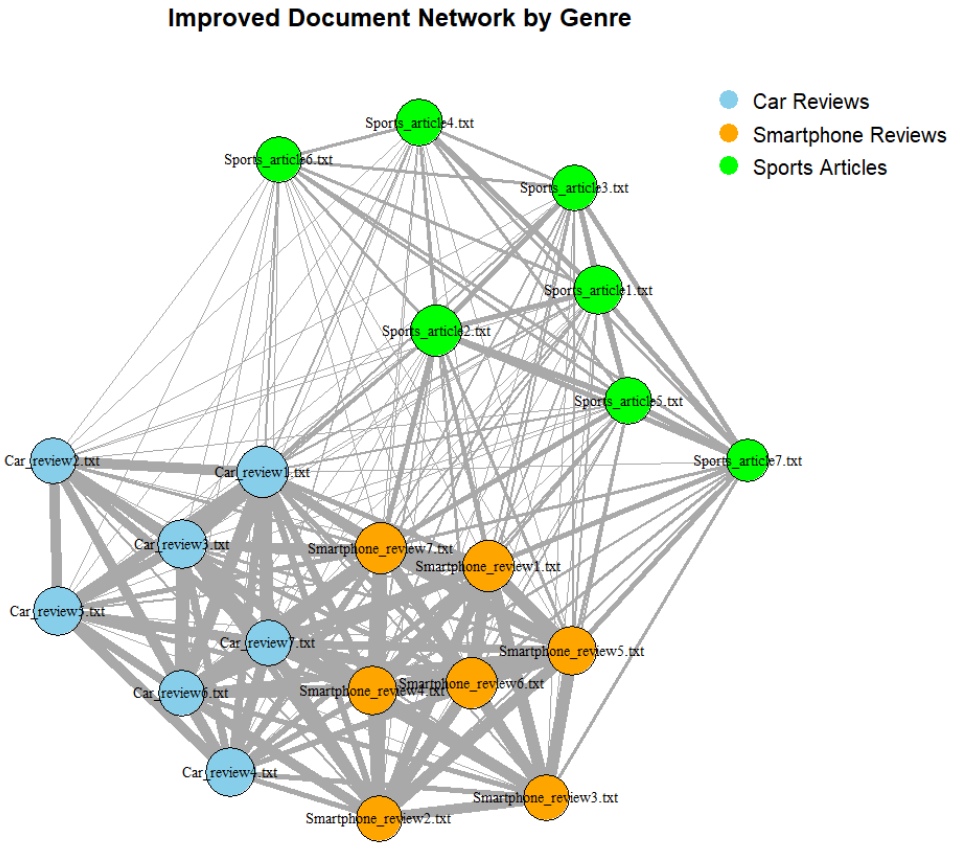
commonly shared with the review genres. This separation highlights the genre-specific language used in sports reporting.

Most Important (Central) Documents in the Network

From the observation, Smartphone_review1.txt, Smartphone_review7.txt, and Car_review1.txt appear to be the most important documents in the network. They are positioned at the center of the graph and are visually connected to many other nodes.

```
[1] "Top documents by Degree Centrality:"
> print(sorted_degree)
       Car_review1.txt Smartphone_review1.txt Smartphone_review6.txt Smartphone_review7.txt    Sports_article2.txt        Car_review3.txt        Car_review4.txt
                    20                     20                     20                     20                     20                     19                     19
       Car_review5.txt Smartphone_review4.txt Smartphone_review5.txt    Sports_article1.txt    Sports_article4.txt    Sports_article5.txt        Car_review2.txt
                    19                     19                     19                     19                     18                     18                     17
       Car_review6.txt        Car_review7.txt Smartphone_review2.txt Smartphone_review3.txt    Sports_article3.txt    Sports_article6.txt    Sports_article7.txt
                    17                     17                     17                     17                     17                     17                     15
```

We further supported this observation using Degree Centrality, where Car_review1.txt, Smartphone_review1.txt, Smartphone_review6.txt, Smartphone_review7.txt, and Sports_article2.txt were found to have the highest degree centrality. This confirms that these documents are the most connected in the network, reinforcing the visual observation of their central and influential positions.



Improved Network Visualization

Node Color by Genre:

- To differentiate documents by category, each node was colored based on its genre (Car Reviews, Smartphone Reviews, or Sports Articles). This clustering by color makes it easy to identify genre-based groupings within the network.

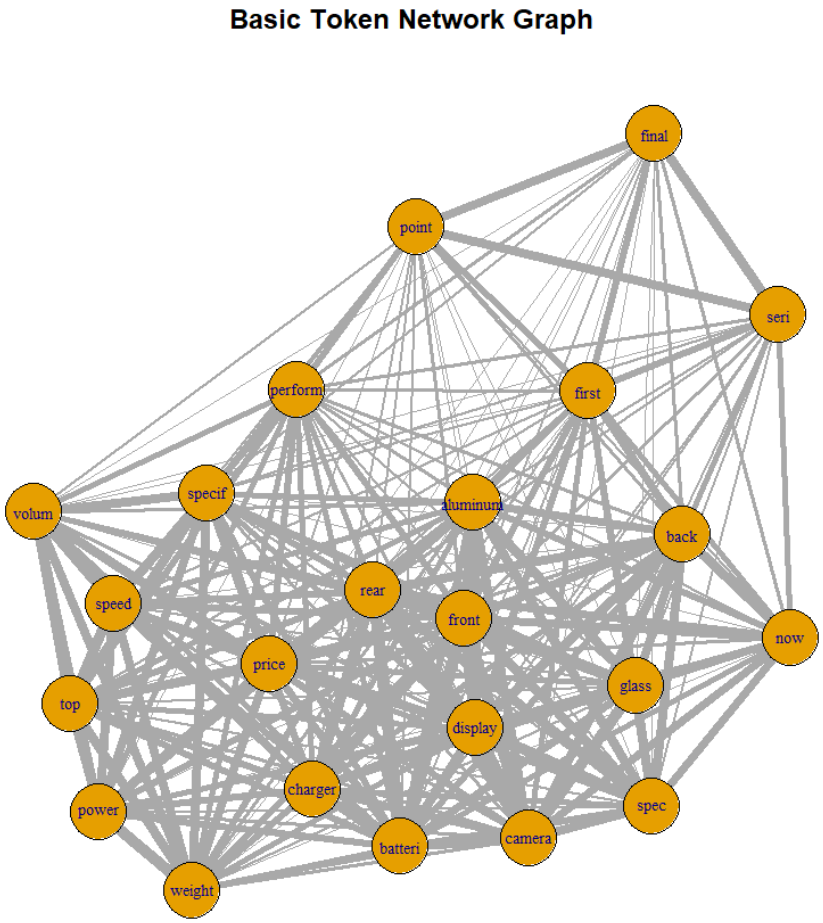
Relative Importance of Nodes:

- Node size was scaled based on degree centrality, representing the number of direct lexical connections a document has with others. This allows viewers to quickly identify central or influential documents in the network. Larger nodes represent documents that share terms with more documents, indicating their importance and representativeness in the corpus.

Observation of the Improved Document Network

The improved document network now clearly displays three distinct colors, each representing a specific genre: Car Reviews (light blue), Smartphone Reviews (orange), and Sports Articles (green). This color coding highlights the genre-based clustering of documents within the network. Nodes are also sized according to their degree centrality, meaning that larger nodes are more important, as they share terms with a greater number of documents.

Q7) Single-Mode Network (Tokens)



Observation

In the token network graph, we can observe two general clusters similar to the pattern seen in the document network graph, although the separation is not very distinct. One cluster at the lower potion of the network appears to be dominated by technical product-related terms such as "display", "camera", "batteri", "charger", and "speed", commonly associated with Car and Smartphone Reviews.

The smaller cluster above the network contains terms like "final", "point", and "seri", which may represent the terms of Sports Articles.

This clustering suggests that tokens tend to group together based on genre-specific vocabulary, reinforcing the network graph observed in the document network. The high density of connections within clusters further highlights the presence of core vocabularies shared within specific types of content.

Most Important Tokens in the Network

From the observation, token “perform”, “aluminium”, “point”, “front” appear to be the most important documents in the network. They are positioned at the center of the graph and are visually connected to many other nodes

```
[1] "Top tokens by Degree Centrality:"
> print(sorted_degree_tokens)
   front  perform    point    price     rear   specif aluminum  display    first     spec   camera
      22       22       22       22       22       22       22       22       22       22       22
   glass  batteri  charger     back    power    volum   weight    speed      top      now    final
      22       21       21       21       20       20       20       19       19       19       17
    seri
      17
```

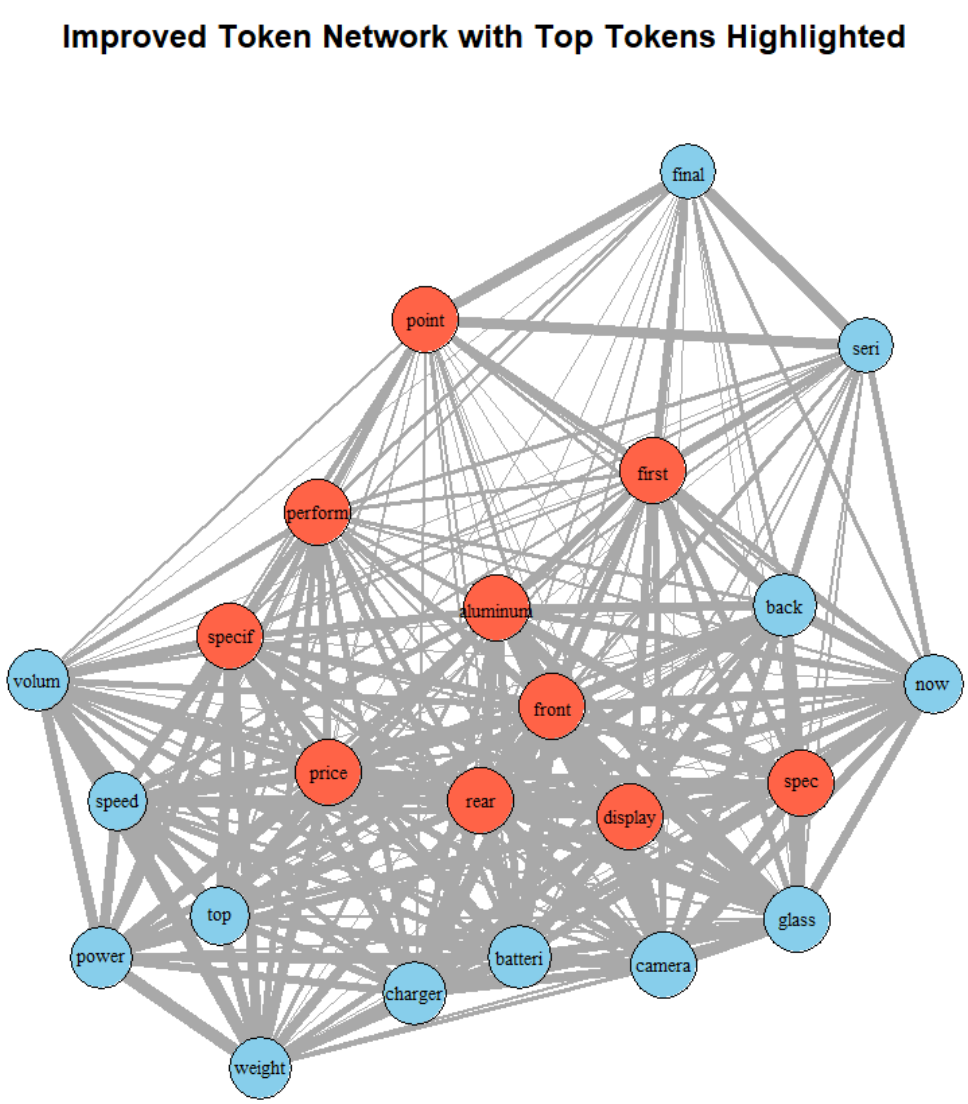
We further supported this observation using Degree Centrality, where “front”, “perform”, “point” etc were found to have the highest degree centrality. This confirms that these documents are the most connected in the network, reinforcing the visual observation of their central and influential positions.

Also, we perform eigenvector centrality — which measures how important a node (token) is not just by how many connections it has, but how important its neighbors are.

```
> central_tokens <- eigen_centrality(TokenGraph)$vector
> # View top 5 most central tokens
> head(sort(central_tokens, decreasing = TRUE), 5)
    front      rear   display   batteri    camera
1.0000000 0.9817834 0.8935122 0.8375195 0.8302575
```

From the result we can see that **"front"** has a centrality score of **1.00** — the highest. This means it is the most influential token in the network. It's not just connected to many tokens, but those tokens are also important.

Overall, tokens such as **“front”, “perform”, and “point”** emerged as highly connected and influential. Their central positions in the graph, along with high Degree Centrality and Eigenvector Centrality scores, confirm their importance.



Improved Token Network Visualization

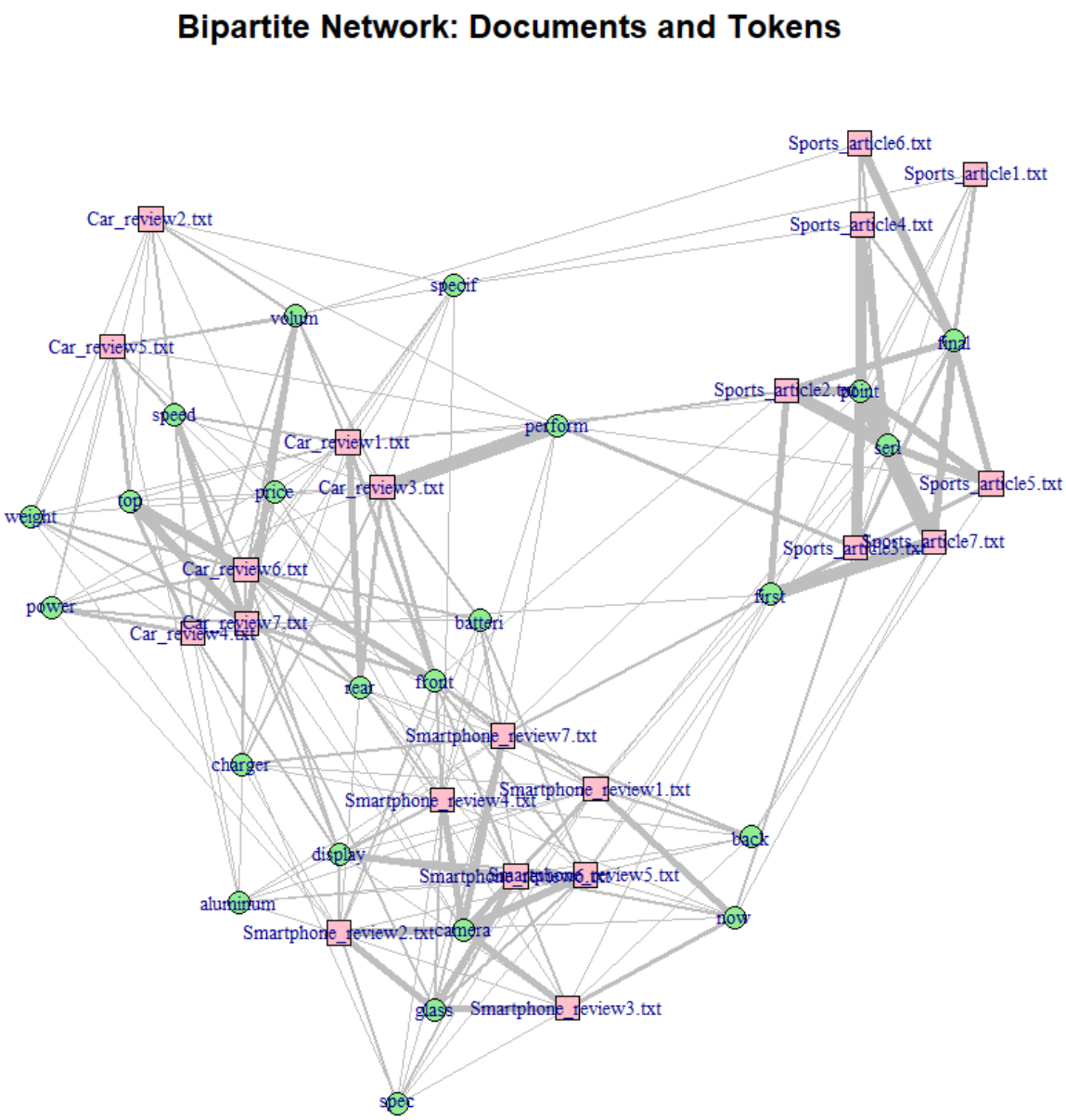
Node Color by Centrality:
To highlight the most influential tokens, the top 10 tokens based on degree centrality were colored in red ("tomato"), while all other tokens were colored light blue. This color distinction draws immediate attention to the most important vocabulary items in the corpus, which act as hubs connecting to many other tokens.

Relative Importance of Nodes:
Node size was scaled using degree centrality, which measures how many documents each token appears in alongside others. Larger nodes indicate tokens that co-occur frequently with many other terms, representing their lexical importance and widespread presence across the corpus.

Observation of the Improved Document Network

The red-highlighted tokens, such as *"front"*, *"perform"*, and *"camera"*, are visually central and more connected, reinforcing their high degree centrality. This confirms their importance as they frequently co-occur with a wide range of other terms in the dataset.

Q8) Bipartite (two-mode) Network



Observation:
Three distinct clusters are clearly visible in the network, corresponding to the three genres:

- Car Reviews are grouped in the left cluster, mostly linked to technical terms such as “speed”, “weight”, and “power”.
- Smartphone Reviews are positioned in the center-lower area, connected to words like “camera”, “display”, “charger”, and “aluminium”.
- Sports Articles are grouped on the right side, connected to terms such as “match”, “final”, and “point”.

Each genre has its own core vocabulary, resulting in tight intra-cluster connections. These shared terms within a genre reflect similar content focus and writing style.

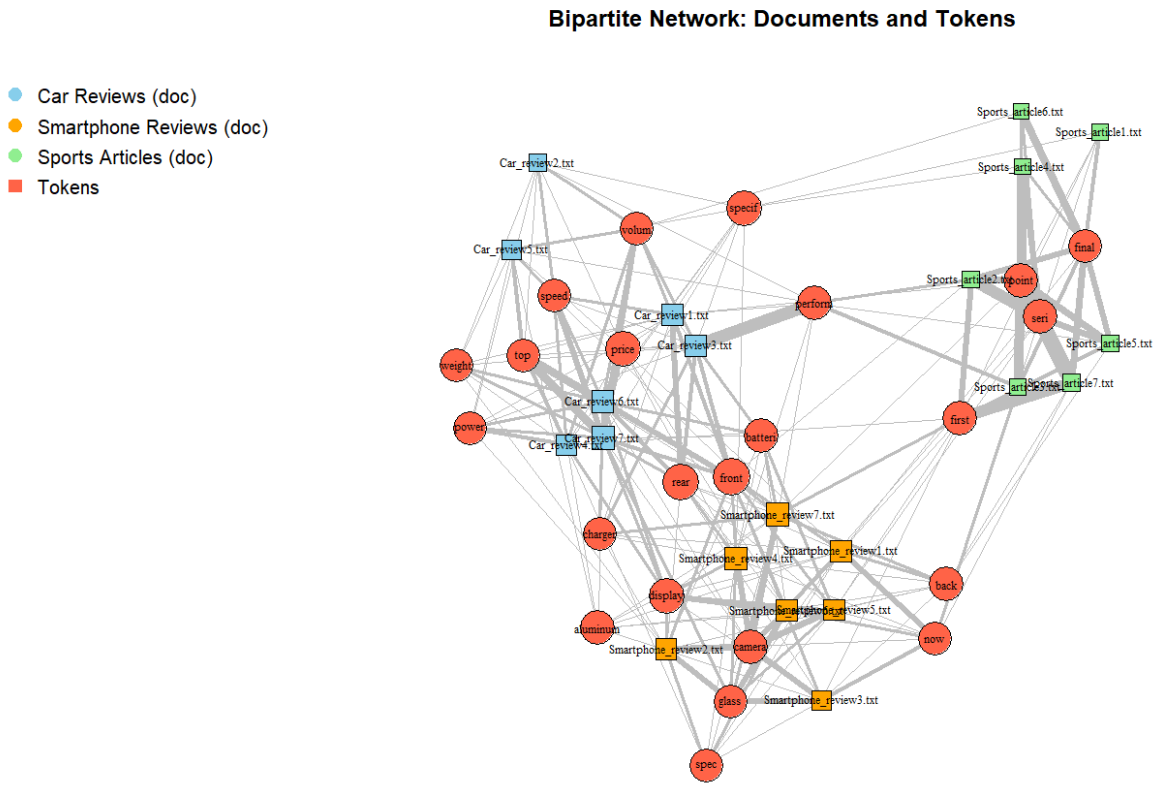
Some tokens are shared across multiple genres, acting as bridges between clusters. For example, the word “perform” appears in Car Reviews, Smartphone Reviews, and Sports Articles. This suggests it is a more general-purpose term used in multiple contexts — such as car performance, device performance, or player performance in sports.

We further identified the most connected tokens using degree centrality, with “perform” emerging as one of the most prominent. This matches our earlier observation that “perform” is an important word used across different genres.

```
> ############################
> # Most Connected Token
> ############################
> # Compute degree centrality
> degree_vals <- degree(g)
> # Get token nodes (the ones NOT ending in ".txt")
> token_nodes <- V(g)[!grepl("\\.txt$", names(V(g)))]
> # Top 10 most connected tokens
> top_tokens <- sort(degree_vals[token_nodes], decreasing = TRUE)[1:10]
> print(top_tokens)
   front     rear    price   specif  display  batteri  perform    point aluminum
      12       11       10       10       10        9        9        9        9
   first
       9
```

We also identified the least connected tokens, such as “charger” and “glass,” which may appear only in the Smartphone Reviews genre. This suggests these terms are more specific and less widely used across other genres.

```
> ############################
> # Least Connected Token
> ############################
> # Filter only document nodes (type == TRUE)
> doc_nodes <- V(g)[V(g)$type == TRUE]
> # Get documents with the lowest degree (least connected)
> low_doc_degrees <- sort(degree_vals[doc_nodes], decreasing = FALSE)[1:10]
> print(low_doc_degrees)
charger   power   speed     top   volum  weight    spec   glass   final     now
      8       8       8       8       8       8       8       8       8       8
```



Improved Bipartite Network Visualization

Node Color by Genre and Type:

To emphasize the structure of the dataset, node colors were assigned based on genre and type:

- Car Reviews were colored skyblue
- Smartphone Reviews were colored orange
- Sports Articles were colored lightgreen
- Tokens (shared words) were colored tomato

This distinction clearly separates the three genres into clusters, making it easy to visually identify genre-specific vocabularies and shared terms across document types.

Relative Importance of Nodes (Node Size):

Node sizes were scaled based on degree centrality—the number of connections a node has. This helps highlight:

- Important documents (connected to many tokens)
- Frequent tokens (appearing in many documents)
  Larger nodes thus signal either widely used vocabulary or documents that contain richer, more varied language.

Legend for Clarity:
A custom legend was added to help interpret the visualization, clearly labeling colors and shapes associated with each document genre and token group.

Observation of the Improved Bipartite Network: Documents and Tokens

Tokens (in tomato red) are shared across documents and act as bridges between genres. Some highly connected tokens such as “perform,” “front,” appear prominently in the center of the network, indicating that they are frequently used across multiple genres.

The relative size of each node reflects how many connections it has—larger nodes are more central or frequently used. For example, tokens like “perform” and “front” have large node sizes because they appear in many documents across different genres. Similarly, documents like “Smartphone_review1.txt” and “Car_review1.txt” are larger because they contain more commonly shared terms, making them more connected within the network.

In contrast, the Sports Articles node are visibly smaller. This is because they are connected to fewer tokens overall and often use more genre-specific or unique vocabulary that is not widely shared with documents from other genres. For example, terms like **“seri”**, **“point”**, and **“final”** are commonly used in Sports Articles but do **not appear** in Car or Smartphone Reviews. These exclusive terms result in fewer shared connections, making the Sports Articles less central in the network.

Q9)

Summary of Results

The analysis identified meaningful patterns in the corpus using both clustering and network analysis. Documents were grouped into three main genres: Car Reviews, Smartphone Reviews, and Sports Articles. These groups were successfully detected using hierarchical clustering based on word usage patterns.

Within the token network, several important words stood out due to their frequent appearance across documents. Tokens like “perform”, “front”, and “power” were highly connected and served as bridges between genres, indicating their general relevance. In contrast, more specific tokens such as “battery” and “camera” were strongly associated with Smartphone Reviews, while “match”, “point”, and “final” were mainly used in Sports Articles.

In terms of documents, files like “Car_review1.txt” and “Smartphone_review1.txt” were identified as central nodes in the network due to their strong connections with widely used terms. Sports articles were smaller and more isolated, reflecting the use of more unique, domain-specific vocabulary.

Together, these findings reveal both the shared language across genres and the distinctive vocabulary that characterises each group.

Comparison of Clustering vs. Network Analysis

Clustering methods, such as hierarchical clustering to document-term matrices, are effective for grouping documents that share similar patterns of word usage. These techniques allow for the quick identification of genre-based groupings, such as Car Reviews, Smartphone Reviews, and Sports Articles. The results are typically easy to interpret, especially when visualized through dendrograms, which provide a clear and structured overview of how documents are related based on their content.

However, clustering comes with limitations. It does not reveal the specific relationships between individual tokens and documents, nor does it consider the centrality or importance of certain words across the entire corpus. Clustering also tends to impose strict group boundaries, which can be problematic when some terms are shared across different genres. For example, a few tokens like “perform” and “front” are used across Car Reviews, Smartphone Reviews, and Sports Articles. However, clustering may still assign documents into fixed groups based on dominant patterns, ignoring these shared terms. As a result, it may produce an oversimplified view of the data and overlook meaningful cross-genre connections

In contrast, network analysis is better at visualizing these connections. By mapping the actual links between documents and the tokens they contain, network analysis makes it easy to identify shared vocabulary across genres and highlight important words like "perform" and "front" that connect different groups. The size of the node and degree centrality further helps in finding the most important tokens and documents. Network analysis also shows overlapping relationships, giving a clearer and more detailed view of how words are used across the texts. Overall, while clustering is helpful for seeing the big picture, network analysis gives a deeper understanding of how words are shared and used across different kinds of documents.

Text Processing Improvement methods

1) Add Named Entity Recognition (NER)

(Encord, 2024) **https://encord.com/blog/named-entity-recognition/**

Named Entity Recognition is an NLP method that identifies and categorizes proper nouns such as product names, brand names, person names, and geographical locations. Integrating NER into the preprocessing pipeline would help isolate domain-specific entities that are strong indicators of genre.

For example, entities like "iPhone" or "Galaxy" would be highly representative of smartphone reviews, while names like "Nadal", "final", or "Champions League" would occur mostly in sports articles. In car reviews, entity mentions such as "Tesla", "engine", or "Model 3" would serve as strong features. Highlighting and retaining these named entities would improve the model's ability to differentiate between genres and allow for more accurate network and sentiment analysis.

2) Add Keyphrase Extraction (e.g., YAKE or KeyBERT)

(Maarten Grootendorst, 2025) **https://maartengr.github.io/KeyBERT/**

Keyphrase extraction methods like YAKE or KeyBERT identify important multi-word expressions (e.g., "fuel efficiency", "grand slam") that represent core ideas in a document. Unlike single-word tokens, keyphrases are more semantically meaningful and specific to the topic.

For example, instead of generic words like "performance" or "final", the network would include precise phrases such as "battery life" or "championship final", allowing for more accurate genre distinction and better clustering between different genre. This improves the graph's interpretability and reduces noise from irrelevant or overly common terms.

3) Add Topic Modeling with BERTopic

(Maarten Grootendorst, 2025) **https://maartengr.github.io/BERTopic/index.html**

BERTopic is a modern text analysis method that finds hidden topics in documents by using BERT (a language model) and clustering. Instead of just counting word frequency, it looks at the meaning of words and how they relate to each other.

Using BERTopic could help organize the documents into meaningful topics like "battery life" or "camera quality" for smartphone reviews, "engine performance" or "fuel efficiency" for car reviews, and "match result" or "player performance" for sports articles. This would make it easier to understand what each document is about, improve how we cluster similar documents together, and support clearer comparisons across genres.

Appendix:

Car_review1: (Andrew Wendler, 2025) https://www.caranddriver.com/reviews/a65011939/2026-gmc-sierra-ev-at4-drive/

Car_review2: (David Gluckman, 2025) https://www.caranddriver.com/reviews/a65009609/2025-audi-q5-sq5-drive/

Car_review3: (Andrew Wendler, 2025) https://www.caranddriver.com/reviews/a64813835/2026-hyundai-ioniq-9-drive/

Car_review4: (Andrew Krok, 2025) https://www.caranddriver.com/reviews/a64927647/2025-toyota-4runner-limited-test/

Car_review5: (Ezra Dyer, 2025) https://www.caranddriver.com/reviews/a64838311/2025-chevrolet-corvette-zr1-drive/

Car_review6: (Dave VanderWerp, 2025) https://www.caranddriver.com/reviews/a60442156/2025-porsche-taycan-turbo-gt-drive/

Car_review7: (Dave VanderWerp, 2025)
https://www.caranddriver.com/reviews/a62989929/2025-volvo-xc90-drive/

Sports_article1: (Tim Reynolds, 2025) https://www.independent.co.uk/news/thunder-nba-game-oklahoma-city-indiana-b2768054.html

Sports_article2: (ESPN, 2025) https://www.espn.com/nba/story/_/id/45445853/2025-nba-finals-biggest-takeaways-thunder-pacers-championship

Sports_article3: (Paul Kasabian, 2025) https://bleacherreport.com/articles/25204984-haliburton-pacers-win-nba-finals-game-3-vs-thunder-fans-praise-mathurin-bench

Sports_article4: (The Athletic NBA Staff, 2025) https://www.nytimes.com/athletic/live-blogs/url-thunder-vs-pacers-live-updates-nba-finals-game-3-score-result/4Yu7CXA9KIDp/

Sports_article5: (Noah Rubin, 2025) https://www.nbcsports.com/nba/news/how-to-watch-oklahoma-city-thunder-vs-indiana-pacers-game-3-tv-stream-info-date-time

Sports_article6: (Ricky O'Donnell, 2025)
https://www.sbnation.com/nba/2025/6/5/24443662/thunder-vs-pacers-nba-finals-predictions-champion-mvp

Sports_article7: (Joe Vardon, 2025)
https://www.nytimes.com/athletic/6420525/2025/06/11/thunder-pacers-nba-finals-bennedict-mathurin-tj-mcconnell/

Smartphone_review1: (GSMArena Team, 2025)
https://www.gsmarena.com/oneplus_13s_handson-review-2838.php

Smartphone_review2: (GSMArena Team, 2025)
https://www.gsmarena.com/samsung_galaxy_s25_edge-review-2834.php

Smartphone_review 3: (GSMArena Team, 2025) https://www.gsmarena.com/sony_xperia_1_vii-review-2828.php

Smartphone_review4: (GSMArena Team, 2025)
https://www.gsmarena.com/google_pixel_9a_handson-review-2821.php

Smartphone_review5: (GSMArena Team, 2025) https://www.gsmarena.com/poco_f7_ultra-review-2810.php

Smartphone_review6: (GSMArena Team, 2025)
https://www.gsmarena.com/samsung_galaxy_a56-review-2812.php

Smartphone_review7: (GSMArena Team, 2025) https://www.gsmarena.com/apple_iphone_16e-review-2805.php

Document-Term Matrix (DTM) after removing sparse terms

	aluminum	back	batteri	camera	charger	display	final	first	front	glass	now	perform	point	power	price	rear	seri	spec	specif	speed	top	volum	weight
Car_review1.txt	0	0	1	0	1	0	0	0	3	0	0	1	1	1	1	4	0	0	1	2	1	2	1
Car_review2.txt	0	0	0	0	0	0	0	0	0	0	0	1	0	1	1	0	0	0	1	2	1	2	1
Car_review3.txt	0	0	2	0	2	0	0	0	3	0	0	9	0	1	1	3	0	0	1	1	1	2	1
Car_review4.txt	1	0	0	0	0	2	0	1	0	0	0	0	0	3	1	0	0	1	1	2	3	2	1
Car_review5.txt	1	0	0	0	0	0	0	0	0	0	0	1	0	1	1	0	0	0	1	2	3	2	1
Car_review6.txt	0	0	2	1	2	2	0	0	4	0	0	0	0	2	2	3	0	0	1	2	6	2	2
Car_review7.txt	1	0	1	1	1	2	0	0	3	2	0	0	0	2	2	2	0	0	1	4	6	4	2
Smartphone_review1.txt	1	2	0	3	1	1	1	1	2	1	4	0	0	0	1	1	0	1	0	0	0	0	0
Smartphone_review2.txt	0	1	1	5	1	2	0	0	2	4	1	0	0	1	0	1	0	2	0	0	0	0	1
Smartphone_review3.txt	1	1	0	4	0	1	0	1	2	4	3	0	0	0	0	1	0	1	0	0	0	0	0
Smartphone_review4.txt	1	1	1	4	1	2	0	0	2	2	1	1	0	0	1	2	0	1	1	0	0	0	0
Smartphone_review5.txt	1	1	2	5	0	2	0	1	2	2	1	0	0	0	0	1	1	1	0	0	0	0	0
Smartphone_review6.txt	1	1	1	4	0	4	0	0	2	4	2	0	0	0	1	1	1	2	0	0	0	0	0
Smartphone_review7.txt	1	2	2	4	2	1	0	2	2	2	0	1	0	0	1	1	0	1	0	1	0	0	0
Sports_article1.txt	0	0	0	0	0	0	3	1	0	0	0	0	1	0	0	0	1	0	1	0	0	0	0
Sports_article2.txt	0	0	0	0	0	0	4	4	1	0	0	2	5	0	0	0	8	0	0	0	0	0	0
Sports_article3.txt	0	0	0	0	0	0	3	0	0	0	2	3	7	0	0	0	1	0	0	0	0	0	0
Sports_article4.txt	0	0	0	0	0	0	2	0	0	0	0	0	7	0	0	0	4	0	1	0	0	0	0
Sports_article5.txt	0	1	0	0	0	0	4	3	0	0	0	1	5	0	0	0	5	0	0	0	0	0	0
Sports_article6.txt	0	0	0	0	0	0	6	0	0	0	0	0	2	0	0	0	2	0	0	0	0	1	0
Sports_article7.txt	0	1	0	0	0	0	5	8	0	0	1	0	11	0	0	0	6	0	0	0	0	0	0

augmented Document-Term Matrix

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	AA
1		aluminum	back	batteri	camera	charger	display	final	first	front	glass	now	perform	point	power	price	rear	seri	spec	specif	speed	top	volum	weight	Genre	Genre	Cluster
2	Car_review1.txt	0	0	1	0	1	0	0	0	3	0	0	1	1	1	1	4	0	0	1	2	1	2	1	Car_Reviews	Car_Reviews	1
3	Car_review2.txt	0	0	0	0	0	0	0	0	0	0	0	1	0	1	1	0	0	0	1	2	1	2	1	Car_Reviews	Car_Reviews	1
4	Car_review3.txt	0	0	2	0	2	0	0	0	3	0	0	9	0	1	1	3	0	0	1	1	1	2	1	Car_Reviews	Car_Reviews	1
5	Car_review4.txt	1	0	0	0	0	2	0	1	0	0	0	0	0	3	1	0	0	1	1	2	3	2	1	Car_Reviews	Car_Reviews	1
6	Car_review5.txt	1	0	0	0	0	0	0	0	0	0	0	1	0	1	1	0	0	0	1	2	3	2	1	Car_Reviews	Car_Reviews	1
7	Car_review6.txt	0	0	2	1	2	2	0	0	4	0	0	0	0	2	2	3	0	0	1	2	6	2	2	Car_Reviews	Car_Reviews	1
8	Car_review7.txt	1	0	1	1	1	2	0	0	3	2	0	0	0	2	2	2	0	0	1	4	6	4	2	Car_Reviews	Car_Reviews	1
9	Smartphone_review1.txt	1	2	0	3	1	1	1	1	2	1	4	0	0	0	1	1	0	1	0	0	0	0	0	Smartphone_Reviews	Smartphone_Reviews	2
10	Smartphone_review2.txt	0	1	1	5	1	2	0	0	2	4	1	0	0	1	0	1	0	2	0	0	0	0	1	Smartphone_Reviews	Smartphone_Reviews	2
11	Smartphone_review3.txt	1	1	0	4	0	1	0	1	2	4	3	0	0	0	0	1	0	1	0	0	0	0	0	Smartphone_Reviews	Smartphone_Reviews	2
12	Smartphone_review4.txt	1	1	1	4	1	2	0	0	2	2	1	1	0	0	1	2	0	1	1	0	0	0	0	Smartphone_Reviews	Smartphone_Reviews	2
13	Smartphone_review5.txt	1	1	2	5	0	2	0	1	2	2	1	0	0	0	0	1	1	1	0	0	0	0	0	Smartphone_Reviews	Smartphone_Reviews	2
14	Smartphone_review6.txt	1	1	1	4	0	4	0	0	2	4	2	0	0	0	1	1	1	2	0	0	0	0	0	Smartphone_Reviews	Smartphone_Reviews	2
15	Smartphone_review7.txt	1	2	2	4	2	1	0	2	2	2	0	1	1	0	0	1	0	1	0	1	1	0	0	Smartphone_Reviews	Smartphone_Reviews	2
16	Sports_article1.txt	0	0	0	0	0	0	3	1	0	0	0	0	1	0	0	0	1	0	1	0	0	0	0	Sports_Articles	Sports_Articles	3
17	Sports_article2.txt	0	0	0	0	0	0	4	4	1	0	0	2	5	0	0	0	8	0	0	0	0	0	0	Sports_Articles	Sports_Articles	3
18	Sports_article3.txt	0	0	0	0	0	0	3	0	0	0	2	3	7	0	0	0	1	0	0	0	0	0	0	Sports_Articles	Sports_Articles	3
19	Sports_article4.txt	0	0	0	0	0	0	2	0	0	0	0	0	7	0	0	0	4	0	1	0	0	0	0	Sports_Articles	Sports_Articles	3
20	Sports_article5.txt	0	1	0	0	0	0	4	3	0	0	0	1	5	0	0	0	5	0	0	0	0	0	0	Sports_Articles	Sports_Articles	3
21	Sports_article6.txt	0	0	0	0	0	0	6	0	0	0	0	0	2	0	0	0	2	0	0	0	0	1	0	Sports_Articles	Sports_Articles	3
22	Sports_article7.txt	0	1	0	0	0	0	5	8	0	0	1	0	11	0	0	0	6	0	0	0	0	0	0	Sports_Articles	Sports_Articles	3

```
# ------------------------------------------------
# R script: FIT3152_Ass3
# Project: FIT3152 Assignment3
#
# Date: 14/5/2025
# Time: 1:55AM
# Author: CHAI SHOU ZHENG
# File: "Assignment3_Corpus2"
# ------------------------------------------------


# clean up the environment before starting
rm(list = ls())

# Load libraries
library(slam)
library(tm)
library(SnowballC)
```

```
library(proxy)
library(dplyr)
library(SentimentAnalysis)
library(igraph)


setwd("FIT3152/FIT3152 Ass3")
getwd()


# ------------------------------------------------
# Q3. Document-Term Matrix (DTM)
# ------------------------------------------------
cname = file.path(".", "Assignment3_Corpus2") # Set path to the text corpus folder
print(dir(cname))  # Print file names in the folder

text_corpus = Corpus(DirSource(cname, recursive = TRUE)) # Load all text files into a text corpus
summary(text_corpus) # Show summary of the corpus (number of docs, etc.)


# ---------- Text Cleaning and Preprocessing ----------

# Tokenisation
text_corpus <- tm_map(text_corpus, removeNumbers)                # Remove numbers
text_corpus <- tm_map(text_corpus, removePunctuation)             # Remove punctuation
text_corpus <- tm_map(text_corpus, content_transformer(tolower))      # Convert all text to lowercase

# Filter words
# Define extra stopwords to remove less informative words
extra_stopwords <- c("use", "get", "make", "need", "just", "can",
                     "also", "one", "will", "time", "like", "may",
                     "much", "more", "that", "with",
                     "arent", "doesnt", "dont", "even", "lot",
                     "might", "often", "wont", "cant", "still")
# Combine built-in stopwords with custom ones
all_stopwords <- c(stopwords("english"), extra_stopwords)

text_corpus <- tm_map(text_corpus, removeWords, all_stopwords)       # Remove common stopwords like "the", "and", "is"
text_corpus <- tm_map(text_corpus, stripWhitespace)               # Remove extra whitespace
text_corpus <- tm_map(text_corpus, stemDocument, language = "english") # Apply stemming (reduce words to root form)



# ---------- Create and Inspect Document-Term Matrix ----------
dtm <- DocumentTermMatrix(text_corpus) # Create DTM from cleaned corpus
dtm                          # View DTM
dim(dtm)                         # Print dimensions: number of documents x number of terms
```

```
inspect(dtm)                    # Show part of the DTM

# ----------- Trim Sparse Terms -----------
dtm_trimmed <- removeSparseTerms(dtm, 0.65) # Remove sparse terms
dtm_trimmed         # View trimmed DTM

dim(dtm_trimmed)     # Show dimensions after trimming
inspect(dtm_trimmed)  # View sample of the trimmed DTM

colnames(as.matrix(dtm_trimmed)) # Show the remaining terms after trimming

# Find terms that appear in at least 10 documents
findFreqTerms(dtm_trimmed, lowfreq = 10)

# Convert DTM to a data frame for inspection/analysis
dtm_df <- as.data.frame(as.matrix(dtm_trimmed))   # Convert DTM to data frame
dtm_df <- dtm_df[, order(colnames(dtm_df))]      # Sort columns
print(dtm_df)  # Print final DTM data frame


# ------------------------------------------------
# Q4. Hierarchical clustering and Dendrogram
# ------------------------------------------------
set.seed(34035958)

# Convert the trimmed DTM to a matrix
dtm_matrix <- as.matrix(dtm_trimmed)

# Create cosine distance matrix using the trimmed dtm with 23 tokens
distmatrix <- proxy::dist(dtm_matrix, method = "cosine")

# Perform hierarchical clustering using Ward's method
fit <- hclust(distmatrix, method = "ward.D")

# Plot the dendrogram
plot(fit,
   hang = -1,
   main = "Hierarchical Clustering of Assignment3_Corpus2 (Using Trimmed DTM with 23
Tokens)",
   xlab = "Documents",
   sub = "",
   cex = 0.8)

# Draw rectangles around k clusters (e.g., k = 3 for 3 genres)
rect.hclust(fit, k = 3, border = "blue")


# ------------------------------------------------
# Evaluate Clustering Accuracy
# ------------------------------------------------
# Create actual genre labels: 7 Car, 7 Smartphone, 7 Sports
```

```
topics <- c(rep("Car", 7), rep("Smartphone", 7), rep("Sports", 7))

# Cut dendrogram into 3 groups (clusters)
groups <- cutree(fit, k = 3)

# Compare predicted clusters with actual genre labels
conf_matrix <- table(GroupNames = topics, Clusters = groups)
print(conf_matrix)

# Reorganize confusion matrix and calculate accuracy
TA <- as.data.frame.matrix(conf_matrix)
TA <- TA[, c(2, 1, 3)]          # Rearrange columns for better alignment
correct <- sum(apply(TA, 1, max))  # Largest value per row
total <- sum(TA)                # Total documents
accuracy <- correct / total      # Clustering accuracy
print(paste("Clustering Accuracy:", round(accuracy * 100, 2), "%"))


# ------------------------------------------------
# Analyze Most Frequent Terms per Genre
# ------------------------------------------------
# Add genre labels to the DTM
genres <- c(rep("Car_Reviews", 7), rep("Smartphone_Reviews", 7), rep("Sports_Articles", 7))
dtm_df$Genre <- genres

# Sum term frequencies by genre
genre_term_freq <- dtm_df %>%
 group_by(Genre) %>%
 summarise(across(everything(), sum))

# Function to get top N terms for a genre
top_n_terms <- function(data, genre_label, n = 10) {
 # Get the row that matches the genre
 row_vector <- as.numeric(data[data$Genre == genre_label, -1])  # remove Genre column
 names(row_vector) <- colnames(data)[-1]  # assign term names
 sorted <- sort(row_vector, decreasing = TRUE)
 return(head(sorted, n))
}

# Print top 10 terms for each genre
top_n_terms(genre_term_freq, "Car_Reviews")
top_n_terms(genre_term_freq, "Smartphone_Reviews")
top_n_terms(genre_term_freq, "Sports_Articles")

# ------------------------------------------------
# Export Augmented DTM
# ------------------------------------------------
# Add Cluster labels to the DTM data
aug_dtms <- cbind(dtm_df, Genre = genres, Cluster = groups)

# Sort by Cluster
```

```
aug_dtms <- aug_dtms[order(aug_dtms$Cluster), ]

# Save the result to CSV
write.csv(aug_dtms, "Augmented_DTM.csv", row.names = TRUE)


# ------------------------------------------------
# Q5. Sentiment Analysis
# ------------------------------------------------
cname = file.path(".", "Assignment3_Corpus2")

text_corpus = Corpus(DirSource(cname, recursive = TRUE))

# Perform sentiment analysis directly on corpus
SentimentA <- analyzeSentiment(text_corpus)
SentimentA$Genre <- c(rep("Car_Reviews", 7), rep("Smartphone_Reviews", 7),
rep("Sports_Articles", 7))

# Set layout to 1 row, 3 columns
par(mfrow = c(1, 3))

# Plot the three GI-based sentiment measures by Genre
boxplot(SentimentGI ~ Genre, data = SentimentA, main = "SentimentGI by Genre")
boxplot(PositivityGI ~ Genre, data = SentimentA, main = "PositivityGI by Genre")
boxplot(NegativityGI ~ Genre, data = SentimentA, main = "NegativityGI by Genre")


# Set up 2x2 plot layout
par(mfrow = c(2, 2))

# Plot sentiment metrics grouped by Genre
boxplot(WordCount ~ Genre, data = SentimentA, frame = TRUE, main = "Word Count by Genre")
boxplot(SentimentQDAP ~ Genre, data = SentimentA, frame = TRUE, main = "Sentiment (QDAP)
by Genre")
boxplot(PositivityQDAP ~ Genre, data = SentimentA, frame = TRUE, main = "Positivity (QDAP) by
Genre")
boxplot(RatioUncertaintyLM ~ Genre, data = SentimentA, frame = TRUE, main = "Uncertainty
Ratio (LM) by Genre")


############################
# t-test SentimentQDAP
############################

# Car_Reviews vs Smartphone_Reviews
lhs = SentimentA[SentimentA$Genre == "Car_Reviews", "SentimentQDAP"]
rhs = SentimentA[SentimentA$Genre == "Smartphone_Reviews", "SentimentQDAP"]
t.test(lhs, rhs, alternative = "greater")

# Car_Reviews vs Sports_Articles
lhs = SentimentA[SentimentA$Genre == "Car_Reviews", "SentimentQDAP"]
```

```
rhs = SentimentA[SentimentA$Genre == "Sports_Articles", "SentimentQDAP"]
t.test(lhs, rhs, alternative = "greater")

# Smartphone_Reviews vs Sports_Articles
lhs = SentimentA[SentimentA$Genre == "Smartphone_Reviews", "SentimentQDAP"]
rhs = SentimentA[SentimentA$Genre == "Sports_Articles", "SentimentQDAP"]
t.test(lhs, rhs, alternative = "greater")


############################
# t-test NegativityGI
############################

# Car_Reviews vs Smartphone_Reviews
lhs = SentimentA[SentimentA$Genre == "Car_Reviews", "NegativityGI"]
rhs = SentimentA[SentimentA$Genre == "Smartphone_Reviews", "NegativityGI"]
t.test(lhs, rhs, alternative = "greater")

# Car_Reviews vs Sports_Articles
lhs = SentimentA[SentimentA$Genre == "Car_Reviews", "NegativityGI"]
rhs = SentimentA[SentimentA$Genre == "Sports_Articles", "NegativityGI"]
t.test(lhs, rhs, alternative = "greater")

# Smartphone_Reviews vs Sports_Articles
lhs = SentimentA[SentimentA$Genre == "Smartphone_Reviews", "NegativityGI"]
rhs = SentimentA[SentimentA$Genre == "Sports_Articles", "NegativityGI"]
t.test(lhs, rhs, alternative = "greater")



# ------------------------------------------------
# Q6. Create a single-mode network (documents)
# ------------------------------------------------
############################
# plot basic network graph
############################

set.seed(34035958)

# Start with trimmed DTM matrix
dtmsx <- as.matrix(dtm_trimmed)

# Convert to binary matrix (1 if word appears in document, 0 otherwise)
dtmsx_binary <- (dtmsx > 0) + 0

# Create adjacency matrix by multiplying with transpose
ByAbsMatrix <- dtmsx_binary %*% t(dtmsx_binary)

# Remove self-links (no edge from document to itself)
diag(ByAbsMatrix) <- 0
```

```
# Create an undirected, weighted graph
ByAbsGraph <- graph_from_adjacency_matrix(ByAbsMatrix, mode = "undirected", weighted =
TRUE)

# Plot the graph
plot(ByAbsGraph, vertex.label.cex = 0.8, edge.width = E(ByAbsGraph)$weight)


#####################################################
# Identify Most Connected Documents (Degree Centrality)
#####################################################

degree_centrality <- degree(ByAbsGraph)
sorted_degree <- sort(degree_centrality, decreasing = TRUE)
print("Top documents by Degree Centrality:")
print(sorted_degree)


################################
# plot improved network graph
################################

set.seed(34035958)

# Create matrix from trimmed DTM
dtmsx <- as.matrix(dtm_trimmed)

# Convert to binary matrix (1 if word appears, else 0)
dtmsx_binary <- (dtmsx > 0) + 0

# Create adjacency matrix by multiplying with transpose
ByAbsMatrix <- dtmsx_binary %*% t(dtmsx_binary)

# Remove self-connections
diag(ByAbsMatrix) <- 0

# Create undirected, weighted graph
ByAbsGraph <- graph_from_adjacency_matrix(ByAbsMatrix, mode = "undirected", weighted =
TRUE)

# Assign genre based on document names
genre <- ifelse(grepl("Car_review", rownames(ByAbsMatrix)), "Car",
         ifelse(grepl("Smartphone_review", rownames(ByAbsMatrix)), "Smartphone", "Sports"))

# Assign colors to genres
genre_colors <- c("Car" = "skyblue", "Smartphone" = "orange", "Sports" = "green")
vertex_colors <- genre_colors[genre]

# Scale node size based on degree centrality
degree_vals <- degree(ByAbsGraph)
vertex_sizes <- 5 + (degree_vals / max(degree_vals)) * 10  # Scale from 5 to 15
```

```
# Fix layout with set.seed for reproducibility
layout_orig <- layout_with_fr(ByAbsGraph)

# Plot improved graph
plot(ByAbsGraph,
   layout = layout_orig,
   vertex.color = vertex_colors,
   vertex.size = vertex_sizes,
   vertex.label.cex = 0.7,
   vertex.label.color = "black",
   edge.width = E(ByAbsGraph)$weight,
   main = "Improved Document Network by Genre")

# Add legend
legend("topright",
     legend = c("Car Reviews", "Smartphone Reviews", "Sports Articles"),
     col = genre_colors,
     pch = 19,
     pt.cex = 2,
     bty = "n")


# ------------------------------------------------
# Q7. Create a single-mode network (tokens)
# ------------------------------------------------
############################
# plot basic network graph
############################

set.seed(123456)

# Convert DTM to matrix
dtmsx <- as.matrix(dtm_trimmed)

# Convert to binary matrix (1 = token appears in doc, 0 = does not)
dtmsx_binary <- (dtmsx > 0) + 0

# Create Token-Token adjacency matrix
ByTokenMatrix <- t(dtmsx_binary) %*% dtmsx_binary

# Remove self-links (no term connected to itself)
diag(ByTokenMatrix) <- 0

# Create undirected, weighted token graph
TokenGraph <- graph_from_adjacency_matrix(ByTokenMatrix, mode = "undirected", weighted =
TRUE)

# Basic plot of the token network
plot(TokenGraph,
   vertex.label.cex = 0.7,
```

```
    edge.width = E(TokenGraph)$weight,
    main = "Basic Token Network Graph")



#####################################################
# Identify Most Connected Token (Degree Centrality)
#####################################################

# Calculate Degree Centrality for tokens
degree_centrality_tokens <- degree(TokenGraph)

# Step 4: Sort and print top tokens
sorted_degree_tokens <- sort(degree_centrality_tokens, decreasing = TRUE)
print("Top tokens by Degree Centrality:")
print(sorted_degree_tokens)


#########################################################
# Identify Most Connected Token (eigenvector centrality)
#########################################################

# Calculate eigenvector centrality
central_tokens <- eigen_centrality(TokenGraph)$vector

# View top 5 most central tokens
head(sort(central_tokens, decreasing = TRUE), 5)



################################
# plot improved network graph
################################
set.seed(1234)

# Convert DTM to matrix
dtmsx <- as.matrix(dtm_trimmed)

# Convert to binary matrix
dtmsx_binary <- (dtmsx > 0) + 0

# Create Token-Token adjacency matrix
ByTokenMatrix <- t(dtmsx_binary) %*% dtmsx_binary

# Remove self-links
diag(ByTokenMatrix) <- 0

# Create undirected, weighted token graph
TokenGraph <- graph_from_adjacency_matrix(ByTokenMatrix, mode = "undirected", weighted =
TRUE)
```

```
# Compute degree centrality
degree_vals <- degree(TokenGraph)

# Highlight top 10 tokens by degree in red
top_tokens <- names(sort(degree_vals, decreasing = TRUE)[1:10])
vertex_colors <- ifelse(V(TokenGraph)$name %in% top_tokens, "tomato", "skyblue")

# Scale node sizes by degree
vertex_sizes <- 5 + (degree_vals / max(degree_vals)) * 10  # Scale 5–15

# Use a consistent layout
layout_token <- layout_with_fr(TokenGraph)

# Plot the improved graph
plot(TokenGraph,
    layout = layout_token,
    vertex.color = vertex_colors,
    vertex.size = vertex_sizes,
    vertex.label.cex = 0.7,
    vertex.label.color = "black",
    edge.width = E(TokenGraph)$weight,
    main = "Improved Token Network with Top Tokens Highlighted")



# ------------------------------------------------
# Q8. Create a bipartite (two-mode) network
# ------------------------------------------------

############################
# Transform data format
############################

# Convert DTM to data frame and add document IDs
dtmsa <- as.data.frame(as.matrix(dtm_trimmed))
dtmsa$doc_id <- rownames(dtmsa)

# Reshape to long format
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)
  }
}
colnames(dtmsb) <- c("weight", "doc_id", "token")

# Remove zero-weight entries
dtmsc <- dtmsb[dtmsb$weight != 0, ]
dtmsc <- dtmsc[, c("doc_id", "token", "weight")]
```

```
################################
# Plot the basic bipartite graph
################################

# Create igraph bipartite network
g <- graph.data.frame(dtmsc, directed = FALSE)
V(g)$type <- bipartite_mapping(g)$type
V(g)$color <- ifelse(V(g)$type, "lightgreen", "pink")  # Docs vs Tokens
V(g)$shape <- ifelse(V(g)$type, "circle", "square")
E(g)$color <- "grey"

# Plot the basic bipartite graph
set.seed(123)

plot(g,
   vertex.label.cex = 0.7,
   vertex.size = 5,
   edge.width = E(g)$weight,
   main = "Bipartite Network:Documents and Tokens")


#####################################################
# Identify Most Connected Token (Degree Centrality)
#####################################################

# Compute degree centrality
degree_vals <- degree(g)

# Get token nodes (the ones NOT ending in ".txt")
token_nodes <- V(g)[!grepl("\\.txt$", names(V(g)))]

# Top 10 most connected tokens
top_tokens <- sort(degree_vals[token_nodes], decreasing = TRUE)[1:10]
print(top_tokens)


################################
# Identify Least Connected Token
################################
# Filter only document nodes (type == TRUE)
doc_nodes <- V(g)[V(g)$type == TRUE]

# Get documents with the lowest degree (least connected)
low_doc_degrees <- sort(degree_vals[doc_nodes], decreasing = FALSE)[1:10]
print(low_doc_degrees)


################################
# Plot the improved bipartite graph
################################
```

```
# Create igraph bipartite network
g <- graph.data.frame(dtmsc, directed = FALSE)

# Define node types: TRUE = document, FALSE = token
V(g)$type <- bipartite_mapping(g)$type

# Assign genres for document nodes
genres <- rep("Token", vcount(g))  # default to "Token"
names(genres) <- V(g)$name        # match by name!

# Assign genre based on name pattern
genres[grepl("^Car_review.*\\.txt$", V(g)$name)] <- "Car"
genres[grepl("^Smartphone_review.*\\.txt$", V(g)$name)] <- "Smartphone"
genres[grepl("^Sports_article.*\\.txt$", V(g)$name)] <- "Sports"

# Assign colors to genres
genre_colors <- c("Car" = "skyblue", "Smartphone" = "orange", "Sports" = "lightgreen", "Token" =
"tomato")
V(g)$color <- genre_colors[genres[V(g)$name]]

# Assign shape by type
V(g)$shape <- ifelse(V(g)$type, "circle", "square")  # docs = circle, tokens = square

# Size by degree
degree_vals <- degree(g)
V(g)$size <- ifelse(V(g)$type, 8 + degree_vals * 0.3, 4 + degree_vals * 0.2)

# Label setup
V(g)$label <- V(g)$name
V(g)$label.cex <- 0.6

# Edge styling
E(g)$color <- "grey"
E(g)$width <- as.numeric(E(g)$weight) * 1.2

# Plot the graph
set.seed(123)
plot(g,
    layout = layout_with_fr(g),
    vertex.label = V(g)$label,
    vertex.label.cex = V(g)$label.cex,
    vertex.label.color = "black",
    vertex.size = V(g)$size,
    vertex.color = V(g)$color,
    vertex.shape = V(g)$shape,
    edge.color = E(g)$color,
    edge.width = E(g)$width,
    main = "Bipartite Network: Documents and Tokens")

# Add legend
```

```
legend("topleft",
    legend = c("Car Reviews (doc)", "Smartphone Reviews (doc)", "Sports Articles (doc)",
"Tokens"),
    col = c("skyblue", "orange", "lightgreen", "tomato"),
    pch = c(19, 19, 19, 15),  # 19 = circle (doc), 15 = square (token)
    pt.cex = 1.5,
    bty = "n")
```