Text Mining with Latent Dirichlet Allocation and Random Forest

WEIDEMAN, JOHN-PIERRE

Table of Contents

Introduction

We explore a dataset of text documents categorized into five themes: Politics, Sport, Technology, Entertainment, and Business. We conduct exploratory analysis, topic modeling with Latent Dirichlet Allocation and classification modeling with Random Forest models. The analysis highlights how different analytical techniques complement each other in understanding the dataset's structure and identifying category-specific features. Each step contributes to a comprehensive understanding of the data and its inherent relationships.

1) Data Cleaning

We conduct data cleaning in the Data_cleaning.r script.

Our dataset, df_file.csv, contains 2225 text documents, where each document is classified into one of the five categories:

- Label 0: Politics
- Label 1: Sport
- Label 2: Technology
- Label 3: Entertainment
- Label 4: Business

Each entry in the datasets contains:

- Text: The text content of the entry
- Label: The category label of the entry

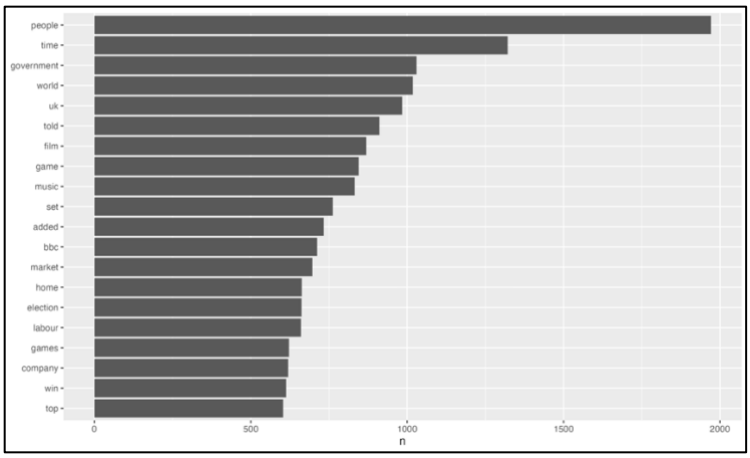
Firstly, we add another column, Text_number, to identify each unique text entry. This is needed to distinguish between the different texts later when we do tokenization for example.

Text tokenization: The textual data was tokenized into individual words using the unnest_tokens() function from the tidytext package. This process split the texts into smaller units (tokens), with one word per row. The context of each text is preserved by maintaining its Text_number and Label. In the same function where we tokenize our data, we also remove the noise in our text as follow:

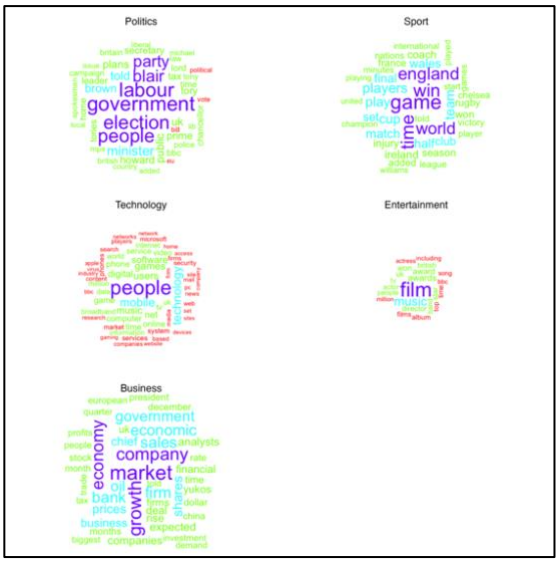
- Stopword Removal: Common, non-informative words (e.g., “the,” “and”) were removed using the anti_join(stop_words) function.
- Number Filtering: Words containing only numeric characters were excluded using a regular expression filter.
- Punctuation Removal: Punctuation marks were removed using a regular expression to focus only on textual content.

Artifact removal: An artifact (Â) frequently appeared in the text. We saw that it appeared before every monetary value and could thus be removed as it doesn’t add any value to the text. We removed it using the str_replace_all() function. Additionally, rows with empty words (resulting from artifact removal) were filtered out.

Word frequency analysis: By conducting a word frequently analysis, we can better understand which words contribute most significantly to the text. It also helps us to detect artifacts in our data (as was the case with the “Â” artifact described above). We create a word frequency plot to visualize the words which appeared most often in our text.



This gives us insights into the topics and the contexts of our text. For example, we can probably infer that the texts are based off news from the UK, since “uk” is the 5th most frequently used word. We can also identify key terms associated with different categories, such as “government” in relation to politics, “film,” “game,” and “music” linked to entertainment, and “company” reflecting topics related to business. To illustrate more clearly the most frequently occurring words in each category we create a word cloud for all the categories.



Larger words indicate higher frequency. From the word clouds we can see that the most dominant words reflect the categories represented by their texts. For instance, “government” and “election” in Category 0 suggest a political focus, while “game” and “film” in Category 1 point to entertainment. Similarly, “market” and “economy” in Category 4 emphasize business and finance.

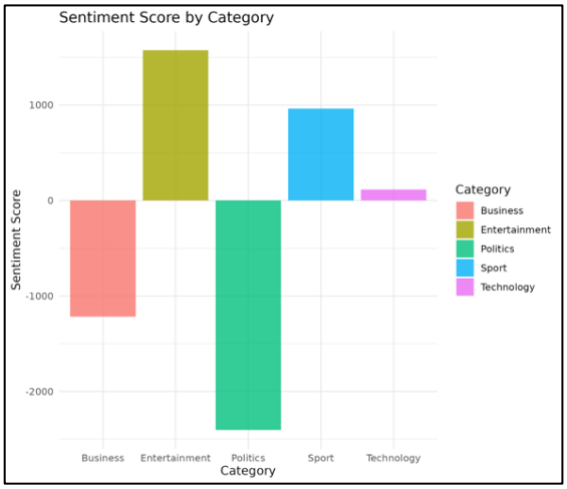
2) Comparative Analysis of Categories

In the script, Category_comparative_analysis.r, we do a comparative analysis of the categories in the dataset. This includes sentiment analysis, readability, relevant words and bigrams, and the distribution of words and characters.

2.1) Sentiment Analysis

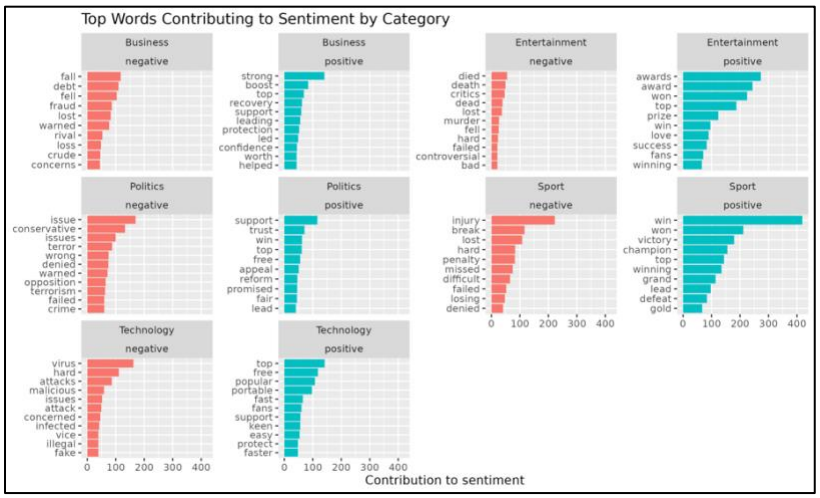
We conduct a sentiment analysis on each category to understand if the texts are predominantly positive or negative. We use the Bing lexicon and qdap package respectively to analyze sentiment of the text.

Using the Bing lexicon, words were classified as either positive or negative. The total count of positive and negative words for each category was calculated, and the net sentiment score was derived by subtracting the number of negative words from the positive words.



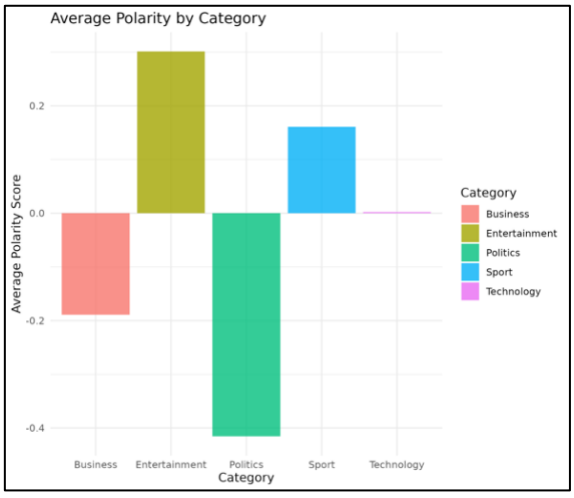
The bar plot visualizes these sentiment scores for the different categories. The plot shows that Entertainment and Sport have predominantly positive sentiment, while Business and Politics are strongly negative. Technology is more neutral, with a more similar number of positive and negative words.

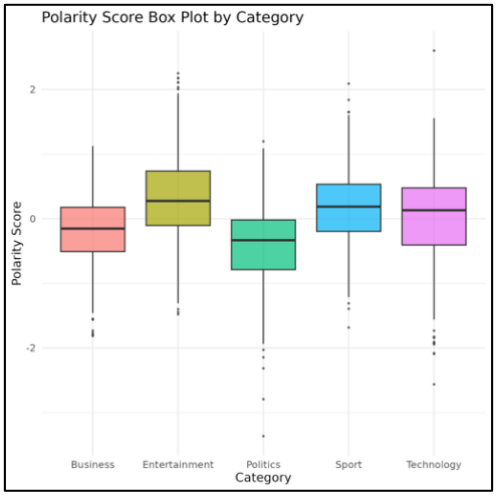
We also identify the top words contributing to sentiment for each category. To do this we again use the Bing lexicon. We can then classify words as positive or negative and calculate the frequency of each sentiment word for every category. The top 10 positive and negative words for each category are visualized below.



Here we can see that for the Business category for example, positive words like “strong”, and “boost” contributed most heavily to positive sentiment, while negative words such as “fail”, and “debt” contributed to negative sentiment.

Next, we calculate the average polarity score for each category using the qdap package. Polarity scores measure the emotional tone of texts, with positive scores indicating positivity and negative scores indicating negativity. We also visualize the distribution of the polarity scores with a box plot.



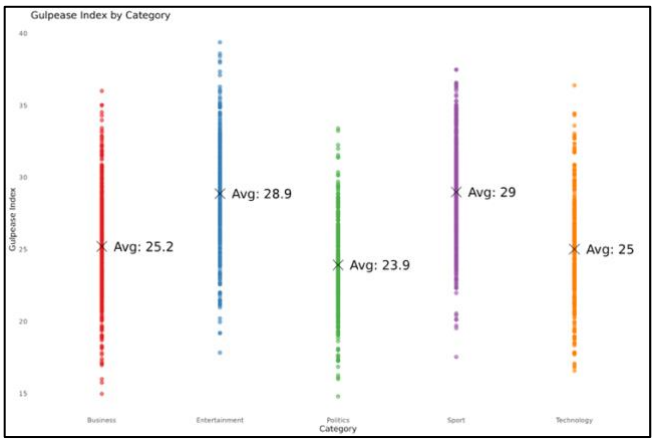


We see a similar trend as before, where Entertainment and Sport have positive sentiment, indicated by the positive polarity scores. Politics and Business shows a negative polarity, indicating negative sentiment. Technology is neutral, with a polarity close to zero.

2.2) Readability analysis

To determine the readability of the texts in each category, we use the Gulpease Index, a readability metric. The Gulpease Index evaluates text readability based on the word length,

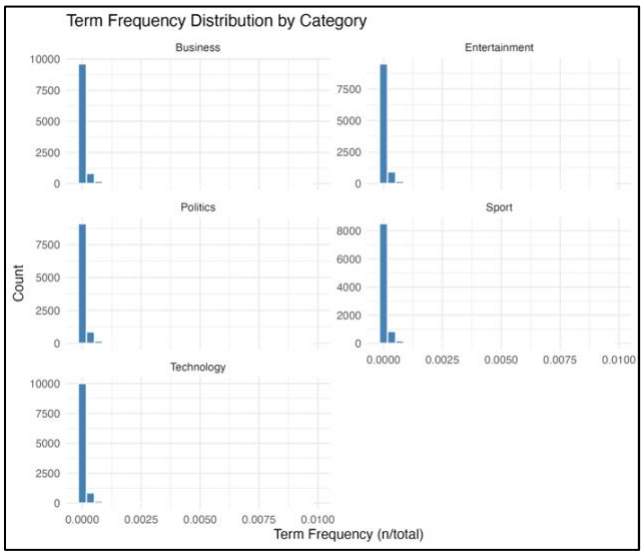
word count, and sentence length. For more information on the calculation of the Gulpease index, see here. We plot the readability scores as well as the averages for each category.



We see that Sport has the highest average readability score (29) and politics has the lowest (23.9). The plot shows the variability of readability across the different categories.

2.3) TF and TF-IDF analysis

Term Frequency (TF) measures how often each word appears in individual documents, highlighting frequently used terms. We plot the TF for each category. To do this we plot the TF (ratio of the term's occurrences to the total word count) against the number of terms that fall into each frequency interval.



All categories exhibit a similar, highly skewed distribution of term frequencies. Most words occur very infrequently, while a small number of words appear much more often.

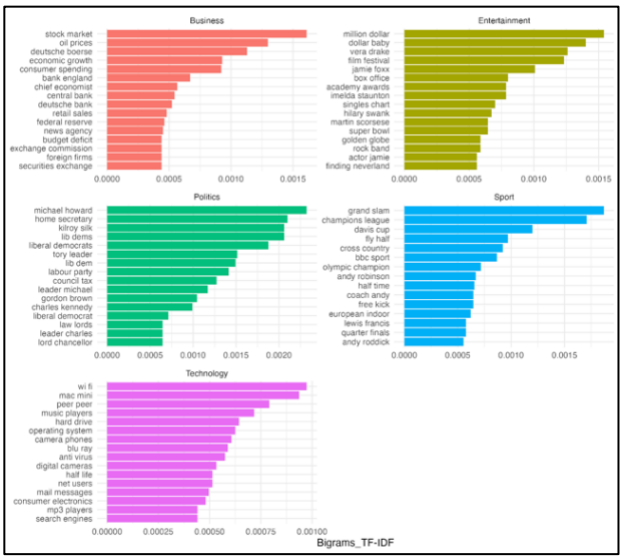
Frequent terms, however, may not always be the most informative across a collection of documents (categories in our case). To address this, Term Frequency–Inverse Document

Frequency (TF-IDF) assigns higher weights to terms that appear often in a specific category but are relatively rare across all documents, making them better indicators of the category's unique content. We calculate the TF-IDF for each category using the bind_tf_idf function from the tidytext package. We then proceed to plot the 15 words with the highest TF-IDF in each category.

The higher a word's TF-IDF, the more uniquely it appears in that category relative to others. The plotted terms thus give insight into each category's specific language focus, indicating which words most distinctly characterize the content in each category.

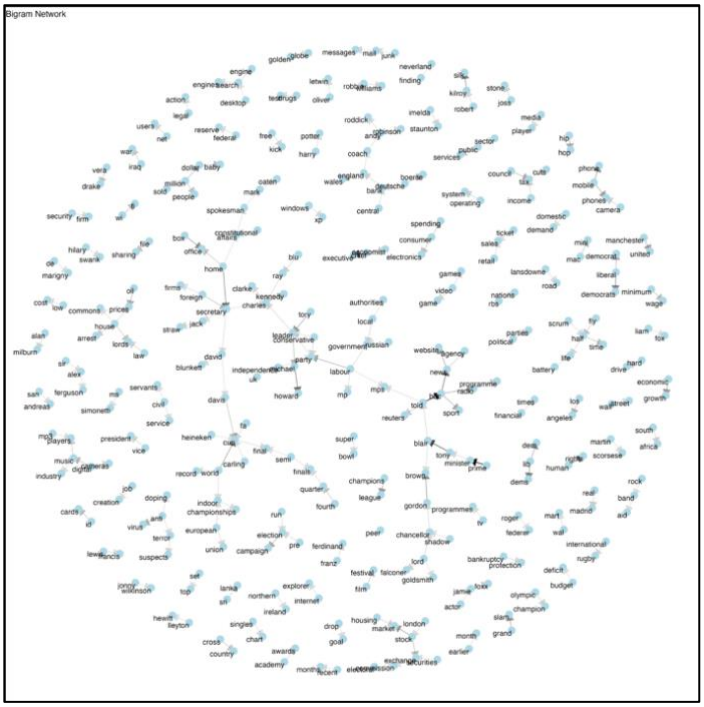
2.4) Most Relevant Word and Bigrams

In addition to the single-word (unigram) analysis which we did, we also analyze pairs of consecutive words (bigrams). Analyzing bigrams helps us to understand context and relationships between words that unigrams cannot provide. For example, bigrams reveal meaningful phrases such as "stock market" in the Business category or "grand slam" in Sport, which adds depth to the analysis. As before, we can identify frequent and unique bigrams using TF-IDF.



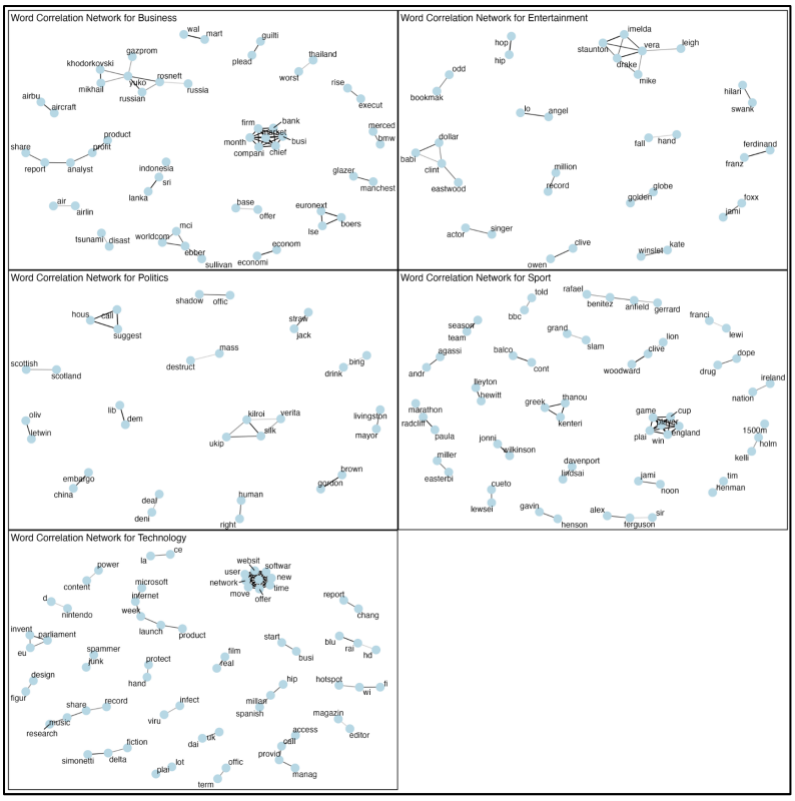
The plot shows the bigrams with the highest TF-IDF scores for each category. It shows the two-word phrases that are most distinctive for each category. For example, “wi-fi” dominate in Business, while “million-dollar” and “film festival” dominate in Entertainment, capturing the category-specific language. From this we can understand how bigrams are able to better capture the context of texts compared to unigrams.

We also represent the bigrams as a network, which represents relationships between words as a graph. Nodes are individual words, and edges connect words that frequently appear together as bigrams. Edges are directional arrows, where darker shading reflects higher bigram frequency.



The bigram network graph is a visualization of word relationships based on their co-occurrence. Words that frequently appear together are closely positioned, forming clusters. We can see for example that a cluster forms around the word “cup”. The cluster around “cup” includes terms like “FA” “Heineken” and “Carling”, reflecting a context related to sports competitions. The network also reveals overlaps between categories, such as the word “home,” which co-occurs with terms like “secretary” and “office”. “Home office” is a topic that probably occurs often in Business news, whereas “home secretary” is most likely discussed in the Politics category.

We are also interested in words that tend to co-occur within the categories, even if they don’t occur next to each other (as in the bigram analysis). This can be represented in word correlation networks.

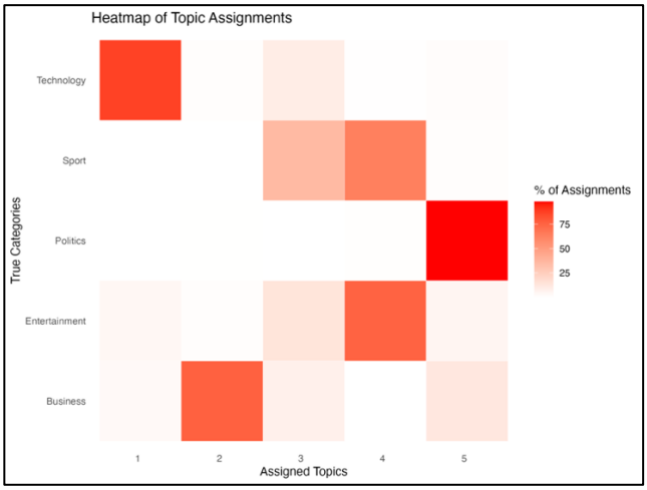


The word correlation network shows how words co-occur and correlate within the different categories. We see words frequently co-occurring in the same context forming clusters. For example, in Business, we see a cluster forming with words like “bank”, “firm”, “market”, etcetera, highlight financial discussions.

3) Latent Dirichlet Allocation

In the LDA.r script we do topic modeling using a Latent Dirichlet Allocation (LDA) model. LDA is used to distinguish between topics in text data. It assumes that each document is made up of multiple topics and that each topic consists of a mixture of words (that overlaps between topics).

We build an LDA model with 5 topics. This allows us to determine whether the LDA model's topic assignments align with the known categories. This is a way to test the effectiveness of topic modeling for categorizing our data.



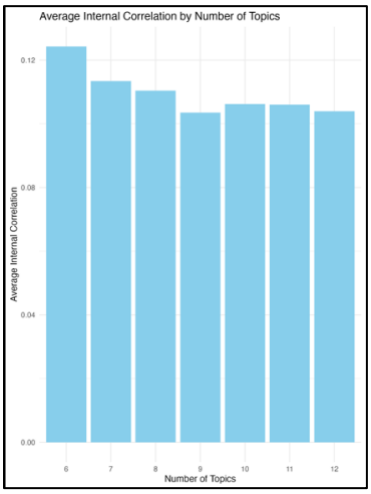
The heatmap shows the relationship between the model's assigned topics (x-axis) and the true categories (y-axis). Each cell represents the percentage of documents from a specific true category assigned to a particular topic, with darker shades of red indicating higher percentages. Each true category predominantly corresponds to a single assigned topic, as seen with the darkest red cells in each topic.

- Technology corresponds to Topic 1.
- Business corresponds to Topic 2.
- Sport corresponds to Topic 3.
- Entertainment corresponds to Topic 4.
- Politics corresponds to Topics 5.

Some overlap exists where documents from a category are assigned to multiple topics. For instance, Sport shows an overlap between Topics 3 and 4, indicating that the model finds similar themes across the Sport category. The LDA model, nevertheless, successfully identifies distinct topics for the categories. However, some overlap between topics and categories suggests the need for further refinement.

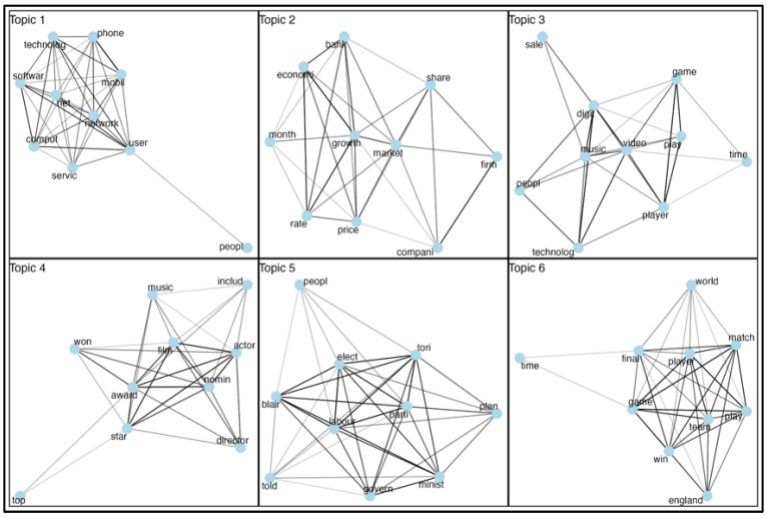
We will look at optimizing the number of topics (k) to ensure that the model generates meaningful topics. The choice of k affects the granularity of the topics. A value too low may merge different themes into a single topic, while a value too high may split coherent topics into smaller, less meaningful ones. To optimize k, we first consider the Beta values of words

in the topics. The Beta values represent the strength of association between words and topics. We consider the top words in each topic (words with highest Beta values). We then proceed to determine the pairwise correlations among the top words (within respective topics). This is our measure of coherence withing topics. High correlations indicate that the top words in a topic tend to co-occur in our text, ensuring that the topic is semantically cohesive. The correlation data for all k values is aggregated, and the average correlation for each k is computed. The optimal value of k is the one that produces the highest average internal correlation, reflecting the greatest coherence in topics.



We see that k=6 is the optimal number of Topics.

Using the model with optimized k, we represent the top 10 words of the identified topics as a network.



Each network shows the most relevant words for a topic and their relationships, giving us an insight into the semantic structure of the topics:

- Topic 1: Words like “technology,” “software,” and “phone” suggest a focus on technological themes.
- Topic 2: Words like “economy,” “market,” and “growth” indicate a business theme.
- Topic 3: Words like “game” and “player”, and words like “digital”, and “technology”, suggests a mix of sports and entertainment-related topics.
- Topic 4: Words like “director”, “actor” and “film”, suggest an entertainment theme.
- Topic 5: Words like “party”, “govern”, and “elect” is indicative of a political topic.
- Topic 6: Words like “team”, “game”, and “final” shows a sport related theme.

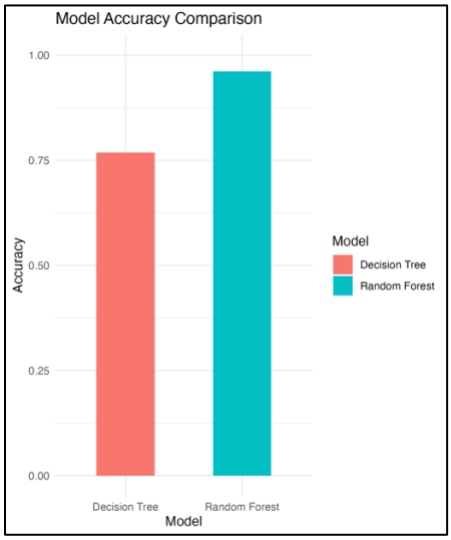
The networks show how words within a topic are related. Strongly connected clusters like in Topic 1, suggest well-defined themes, while sparser connections like Topic 3 indicate broader or less cohesive topics. These relationships helps us to better understand the dominant themes and their semantic connections within each topic.

4) Random Forest

In RF.r , we implement, optimize and evaluate a Random Forest (RF) model to classify the text data into their respective categories. RF is an ensemble machine learning algorithm. It builds multiple decision trees and combines their outputs to classify data. The results are usually more robust compared to single decision trees as we will see.

In building our RF model, we start by calculating the TF-IDF scores for words in the training dataset to identify the top 500 most informative words per category. Then, we filter the dataset to include only these words. This is done to reduce noise and dimensionality and to focus on the most relevant features in our texts. For training of our model. We split our data into a 80%-20% train-test split.

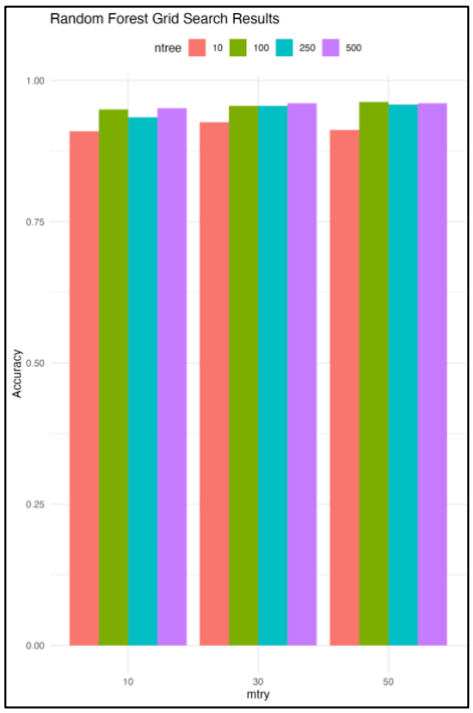
Initially we build our RF model using the default parameters. We also build a single decision tree model for comparison.



We see that the random forest, which is an ensemble of 500 trees in this model, considerably outperforms a single decision tree.

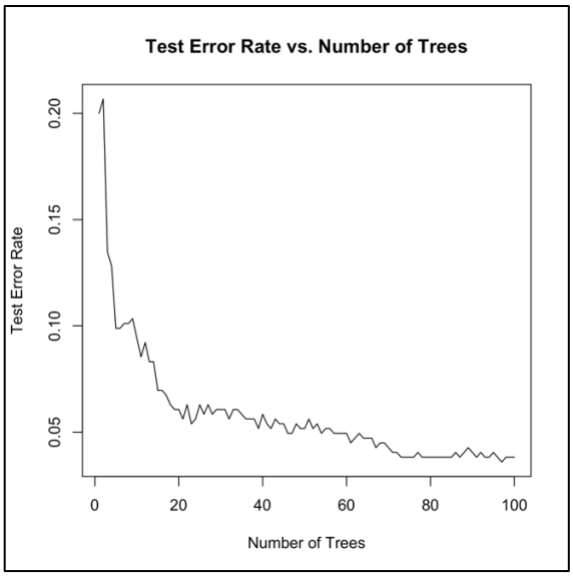
Next, we try to optimize our RF model. This is done by performing a grid search to identify the optimal hyperparameters for the RF model. We specifically aim to optimize the number of trees (ntree) and the number of features considered at each split (mtry). By iterating over a range of values for both parameters, we train models and evaluate their accuracy on the test set to try and find the optimal parameter combination. We define the following ranges to be explored by our grid search:

- ntree: [10, 100, 250, 500]
- mtry: [10, 30, 50]



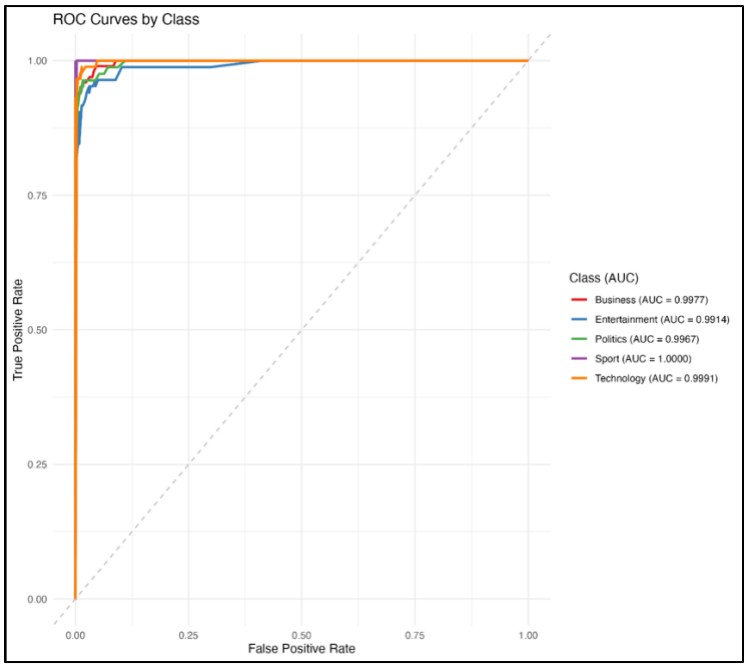
The above plot shows the grid search results for the accuracy of the RF model for different combinations. We see that generally speaking a low number of trees (10) underperforms models with more trees. However, for models with a sufficient number of trees the accuracy is, relatively speaking, high for all combinations. The optimal hyperparameter combination is the ntree=100 and mtry=50.

We visualize the test error rate as a function of the number of trees in our optimal RF model.



The plot shows how the test error decreases as more trees are added, demonstrating the ensemble effect of RF.

In our analysis of the model's performance, we determine which category is the easiest to predict by evaluating sensitivity (true positive rate) and specificity (true negative rate) for each class. Using Receiver Operating Characteristic curves (ROC curves) and their corresponding Area Under the Curve (AUC) values, we assess how well the model distinguishes between categories and identify the class with the highest predictive accuracy. For a detailed explanation of these concepts, see this paper.



The ROC curves visualize the trade-off between sensitivity and 1-specificity for our different categories. A curve closer to the top-left corner indicates better classification performance, as it demonstrates a high combination of sensitivity and specificity. The AUC quantifies overall performance. From the ROC curves and AUC values in the plot, we see that Sport is the easiest category to predict. It has an AUC of 1, indicating perfect classification with no misclassifications. All the Other categories also have high AUC values close to 1, meaning the model performs very well across all the categories. Entertainment is the hardest category to predict with an AUC of 0.9914.

Results

The results obtained from the exploratory analysis, the classification model, and the unsupervised topic modeling show consistency, demonstrating that the dataset contains patterns which enabled us to distinguish between categories.

From the exploratory analysis, word frequency and TF-IDF revealed the most relevant terms for each category. From this analysis we could see the differences in vocabulary and themes present in the different categories. Sentiment analysis also showed distinctions, showing positive sentiment in categories like Sport and Entertainment, while Politics and Business exhibited more negative sentiment. These patterns suggest thematic differences between categories, giving us a good foundation for classification.

Topic modeling using LDA further revealed the structure of our data. Topics from the model aligned closely to the categories. Technology, Politics and Business were seen as distinct topics. However, there was some overlap in topics between Sport and Entertainment. This is consistent with the exploratory analysis where there was overlap in vocabulary such as “win” and “game” for example.

The classification model built using RF achieved high accuracy, with AUC values close to 1 across all categories, and perfect classification for Sport. The overlap between topics like Sport and Entertainment which we saw in the prior analyses also shows in our RF model as slightly lower AUC values.

The RF model achieved high accuracy, with AUC values close to 1 across all categories, and perfect classification for Sport (AUC = 1). However, Entertainment had the lowest AUC, despite prior analyses revealing some overlap between Sport and Entertainment. This indicates that while Sport has a distinct and well-defined vocabulary, making it easier to classify, Entertainment shares linguistic features with other categories, which makes its classification more challenging.

Overall, the results from the three analyses are consistent, but each analysis allows us to gain deeper insights into our data structure.