KATHMANDU UNIVERSITY

SCHOOL OF SCIENCE

DEPARTMENT OF MATHEMATICS

A PROJECT REPORT ON

Nepali News Analytics: EDA, Classification, Sentiment, Summarization, and Beyond

Submitted by:

Aaditya Sapkota

Aashish Bam

Supreme Khatiwada

Ashwin Burlakoti

Project Supervisor:

Prof. Dr. Jyoti U. Devkota

Department of Mathematics

Date: 23rd July 2025

Acknowledgement

We want to give a huge thank you to everyone who helped us with our project, “Nepali News Analytics: EDA, Classification, Sentiment, Summarization, and Beyond.”

A special thanks to our guide, Prof. Dr. Jyoti U. Devkota. Her advice and encouragement were incredibly helpful and really shaped our project. We couldn't have done it without her.

We're also grateful to the Department of Mathematics at Kathmandu University's School of Science. They gave us the space, tools, and support we needed to get this work done.

Big thanks to our friends and classmates as well. Their ideas and suggestions during our brainstorming sessions were a huge help.

And finally, a shout-out to our amazing team: Aaditya Sapkota, Aashish Bam, Supreme Khatiwada, and Ashwin Burlakoti. Everyone's hard work and great teamwork made this proposal possible.

We are truly thankful for every single person who played a part, big or small, in this project.

Abstract

Ever feel like there's too much news to keep up with? It's a real challenge, especially in Nepali, since most analysis tools are built for English, leaving a gap for Nepali content. To solve this, we created “Nepali News Analytics.”, built using the R programming language to automatically read, understand, and organize Nepali news articles.

Our tool has several powerful features. It can sort the news automatically, reading an article to figure out if it's about politics, sports, or entertainment, much like an efficient librarian. It also gets the vibe of a story by telling if the tone is positive, negative, or neutral, which we achieved by creating a special dictionary of Nepali words and their emotional scores. If you don't have time for a long read, the tool gives you the short version by pulling out the most important sentences to create a quick summary. Finally, it finds the key players by identifying important people, places, and organizations through an analysis of the story's grammar.

Our goal was to build a simple, useful toolkit for anyone interested in Nepali news—from journalists and researchers to students. It helps us see the bigger picture by tracking trends and finding key information in a sea of text. This project is a first step, proving that we can create powerful and effective tools specifically for the Nepali language without needing giant, expensive AI.

Table of Contents

Chapter 1: Introduction

1.1 Background

In today's digital world, we are surrounded by massive amounts of data, especially in the form of text. While languages like English have many advanced tools and models to analyze this text, languages like **Nepali** still face challenges due to limited resources. There are not enough large datasets, pre-trained models, or efficient tools available for **Natural Language Processing (NLP)** in Nepali. As a result, important insights from Nepali text data—like news articles—often go unexplored. This project aims to bridge that gap by building a full NLP pipeline for **analyzing Nepali news content** using the **R programming language**, which offers strong support for data analysis and text mining.

The goal of this project is to collect Nepali news articles and apply various NLP techniques to extract meaningful information. We will begin with **Exploratory Data Analysis (EDA)** to understand the data and its structure. Then we will use **text classification** to group news by topics or categories, and apply **sentiment analysis** to study the emotions or opinions in the articles. **Text summarization** will help reduce long news into short, meaningful summaries, while **Named Entity Recognition (NER)** will identify names of people, places, and organizations mentioned in the news. Finally, **clustering techniques** will group similar news articles. These processes will not only help us understand patterns, trends, and public opinions in Nepali media but will also support the growth of **NLP tools for low-resource languages** like Nepali, making it useful for researchers, journalists, and data scientists.

1.2 Motivation & Significance

This project holds substantial academic and practical importance for several compelling reasons:

- **Addressing the Low-Resource Language Gap:** It directly contributes to the underrepresented field of NLP for low-resource languages by creating and validating a comprehensive text analysis pipeline for Nepali. This effort can serve as a blueprint and inspire similar initiatives for other languages with limited existing tools and resources.

- **Enhanced Media Understanding:** The analytical insights derived from this project will provide a deeper, data-driven understanding of Nepali media trends, including prevailing narratives, recurring themes, and potential biases in reporting. This is crucial for media monitoring organizations, researchers, policy analysts, and the general public.

- **Practical Applications for Professionals:** The summarization and classification modules can significantly assist journalists and content creators in efficiently grasping the essence of large volumes of news, identifying relevant articles, and tracking topics. Sentiment analysis can aid brand monitoring and public relations.

- **Support for Sociopolitical Research:** The sentiment analysis and NER capabilities can empower sociologists, political scientists, and humanitarian organizations in gauging public

opinion on various issues, tracking the influence of key individuals and organizations, and understanding discourse dynamics.

- **Open-Source Contribution:** By leveraging R's open-source ecosystem, the project will produce well-documented scripts, potentially a custom Nepali sentiment lexicon, and potentially an adaptable TextRank implementation, contributing valuable resources back to the global NLP community.

- **Demonstration of R's Capabilities:** This initiative prominently showcases the power and flexibility of R as a comprehensive tool for end-to-end data science projects, from data acquisition and cleaning to sophisticated machine learning, advanced text analysis, and interactive visualization.

1.3 Objectives

This project aims to achieve the following key objectives: delivering a modular and extensible NLP pipeline for Nepali text.

- **Perform Detailed Exploratory Data Analysis (EDA):** Uncover fundamental patterns and characteristics within the Nepali news dataset, including article and headline length distributions, common word and n-gram usage patterns, and category distributions, to deeply understand the corpus's linguistic and structural features.
- **Build a Supervised News Classification System:** Develop and rigorously evaluate machine learning models (e.g., Naive Bayes, Support Vector Machines (SVM), Random Forest) to automatically categorize Nepali news articles based on their content, aiming for high accuracy, precision, recall, and F1-scores.
- **Conduct Fine-Grained Sentiment Analysis:** Implement a robust sentiment analysis module utilizing a custom-built or carefully adapted Nepali sentiment lexicon. This system will classify articles with positive, negative, or neutral polarity, enabling analysis of sentiment variations across different news categories, sources, and over time.
- **Generate Automated Extractive Summaries:** Create concise 1-3 sentence summaries of longer news articles by implementing unsupervised summarization algorithms, such as TextRank, based on sentence importance and relevance.
- **Execute Named Entity Recognition (NER):** Utilize the UDPipe model to accurately extract and analyze mentions of key entities, specifically persons, locations, and organizations, from the Nepali news text to identify prominent figures, places, and institutions.

Perform News Clustering: Apply unsupervised clustering algorithms (e.g., K-Means, Hierarchical Clustering) to group news articles into thematic clusters, enabling the identification of emerging news topics and trends without prior categorization.

1.4 Scope

Data Domain: Nepali-language digital news articles collected from selected, trusted online sources (dataset will be curated / custom-built; public datasets referenced for comparison and augmentation where appropriate).

Language Focus: Nepali only (current phase). Multilingual expansion is out of scope for this iteration, but the pipeline is designed to be extendable.

Processing Stages Included:

1. Ingestion & cleaning of raw text.
2. Tokenization (words + sentences).
3. Stopword removal (custom Nepali list).
4. Optional lemmatization/stemming (where tools permit).
5. Exploratory statistics & visualizations (counts by category/source; length distributions; top n-grams; lexical diversity; word clouds; histograms; bar charts).
6. Supervised text classification using TF-IDF features and the models listed above.
7. Rule-based sentiment tagging using a Nepali lexicon.
8. Extractive summarization via TextRank.
9. NER approximation via UDPipe PROPN tags + multi-word grouping.

Software Stack: R (tidy text mining workflows; modeling via packages such as *caret*; linguistic annotation via *udpipe*; potential interactive viewing via *shiny* if time permits).

Intended Users: Students, data enthusiasts, researchers, journalists, and others needing structured, code-driven insight from Nepali news text without relying on large pretrained LLM stacks.

1.5 Limitations

Being straight about constraints helps everyone use the results properly.

- **Low-Resource Starting Point:** Nepali lacks large, labeled corpora and robust pretrained transformers; project works within that reality rather than solving it fully.
- **Dataset Coverage:** We work with a finite, curated set of trusted digital sources—results may not generalize to *all* Nepali media ecosystems.
- **Rule-Based Sentiment:** Depends on coverage + quality of the custom Nepali sentiment lexicon; nuance (sarcasm, negation scope beyond simple rules, domain slang) may be missed.
- **Extractive Summaries:** TextRank selects existing sentences; it does not generate new paraphrased text and may include context fragments if source writing is noisy.
- **Coarse NER:** UDPipe provides POS tags; we infer entities from **PROPN** spans. Fine-grained labels (PERSON vs ORG vs LOC) are approximations and may need manual mapping.
- **Tooling Constraints:** Lemmatization/stemming quality in Nepali is limited; preprocessing choices may affect downstream model scores.

2. Methodology

The project will follow a structured, modular approach, with each objective addressed by a dedicated task or phase. The methodology will primarily utilize R's powerful text mining libraries and machine learning capabilities.

2.1 Data Collection and Preprocessing

- **Corpus Acquisition:** The project will leverage a substantial pre-existing dataset of Nepali news articles, ideally sourced from various reputable online portals. Legal and ethical considerations regarding data usage will be strictly observed.
- **Text Cleaning:** A comprehensive text preprocessing pipeline will be implemented to ensure data quality. This includes:
  -Removal of HTML tags, special characters, punctuation, numbers, and extraneous whitespace.
  -Conversion of all text to lowercase.
  -Normalization of character variations (e.g., different forms of a character).

- **Preprocessing:** Further linguistic processing will involve
- **Tokenization:** Breaking down text into individual words (tokens) and sentences.
  -Stopword Removal: Filtering out common Nepali stopwords using a custom-built or adapted Nepali stopword list.
  -Stemming/Lemmatization: Exploring and applying suitable techniques to normalize words to their root form, if effective Nepali-specific tools are available or can be developed.
- **Key Tools:** readr,stringr,tidytext.

2.2 Exploratory Data Analysis (EDA)

2.2.1 Dataset Summary

This project deals with a dataset of **Nepali news articles**, where each article includes:

- **Content** (main body of the news)
- **Heading** (title of the article)
- **Category** (type of news: politics, sports, etc.)
- **Source** (news publisher)

EDA Goals

The goal of this EDA is to **deeply understand the structure and patterns** in the data to help build better NLP models later. Specifically, we explore:

- **Category & source distribution** – How many articles are there per category? Which sources publish the most?
- **Text length analysis** – How long are the headlines and contents? (in characters or words)

- **Common terms & phrases** – What are the most frequently used words?
- **Lexical diversity** – How varied is the vocabulary across articles?

Why This EDA Matters

Understanding this dataset helps:

- **Detect class imbalance** → important for training accurate classifiers
- **Spot long/short texts** → useful for padding/truncation in modeling
- **Highlight key keywords** → can become features for classification
- **Analyze vocabulary richness** → important for understanding complexity in a low-resource language like Nepali

Structure of This Section

The rest of this EDA is presented in sections:

- **Visualizations**
- **Summary stats**
- **Interpretations**
  Each part will explain what we found and **why it matters** for building ML models like:
  - **Text classifiers**
  - **Sentiment analyzers**
  - **Summarizers**

Data Overview & Preprocessing Summary

Before diving into EDA, it's important to understand **how the data was collected, cleaned, and structured**.

Dataset Info

- ~50,000 **Nepali news articles**

- Each article contains:

  - **Heading**
  - **Content**
  - **Category**
  - **Source**

```
dim(news_data)
```

```
## [1] 50000     4
```

```
head(news_data)
```

```
## # A tibble: 6 × 4
##   source       category      heading                                    content
##   <chr>        <chr>         <chr>                                      <chr>
## 1 Onlinekhabar World         टोकियो ओलम्पिकमा उत्तर कोरियाले भाग नलिने, दक्षिण कोरियालाई पनि झट्का… २४ चेत,…
## 2 Ekantipur    Entertainment रंगीचंगी राखी                             विभिन्न थरी…
## 3 Setopati     Arts          सुनिलको 'केटो अलि कमाउने होस्' चर्चामा    विदेशमा ब…
## 4 Onlinekhabar Sports        एसिया कप महिला क्रिकेटः नेपालले आज पाकिस्तानसँग खेल्ने…… ११ मंसिर…
## 5 Nepalkhabar  Society       लैनचौरमा ३५ लाख लुटपाट गर्ने दुई पक्राउ    नेपाल प्रह…
## 6 Setopati     Automobile    टिभिएसको नयाँ अपाचे आरटिआर-२०० ४भी सार्वजनिक, कति पर्ला मूल्य?… टिभिएस मो…
```

2.2.2 Preprocessing Steps

To ensure clean, usable data for analysis and modeling, the following steps were performed:

- **Missing & duplicate data**
  - Detected and **removed** to maintain integrity
- **Text cleaning** included:
  - Removing **HTML tags**
  - Removing **URLs** and **email addresses**
  - Stripping **extra whitespace**
- **Text normalization**
  - Lowercasing
  - Standardizing punctuation or symbols (if done)

Checking and Removing missing values.

```
colSums(is.na(news_data))
```

```
##   source category  heading  content
##        0        0        0        3
```

```
news_data<-na.omit(news_data)
dim(news_data)
```

```
## [1] 49997     4
```

Checking and Removing Duplicate Rows.

```
sum(duplicated(news_data))
```

```
## [1] 5
```

```
news_data[duplicated(news_data), ]
```

```
## # A tibble: 5 × 4
##   source      category  heading                                          content
##   <chr>       <chr>     <chr>                                            <chr>
## 1 Nepalipatra Lifestyle https://www.nepalipatra.comब्लड प्रेसर लो भएमा...   एजेन्सी
## 2 Nepalipatra Lifestyle https://www.nepalipatra.comयी खानेकुरा खाँदा घट्छ तौल…… काठमाडौं ।…
## 3 Nepalipatra Lifestyle https://www.nepalipatra.comअदुवा-चिया खानुका फाइदा - अदुवामा…
## 4 Nepalipatra Lifestyle https://www.nepalipatra.comहोसियार ! कसले खानुहुँदैन दूध … काठमाडौं ।…
## 5 Nepalipatra Lifestyle https://www.nepalipatra.comशरीरमा बोसो कम गर्न के खाने ?… बोसो कम …
```

```
news_data<-news_data[!duplicated(news_data), ]
```

```
dim(news_data)
```

```
## [1] 49992     4
```

Function for Nepali Text Cleaning

```
clean_nepali_text<-function(text) {
  text<-as.character(text)
  text<-gsub("<.*?>","",text)
  text<-gsub("http\\S+|www\\S+|https\\S+","",text)
  text<-gsub("\\S+@\\S+","",text)
  text<-gsub("\\s+"," ",text)
  text<-trimws(text)
  return(text)
}
```

Cleaning the content and Headline

```
news_data$content<-clean_nepali_text(news_data$content)
news_data$heading<-clean_nepali_text(news_data$heading)
```

Advanced Preprocessing using Corpus Creation

1. **Corpus Creation** – Converts **news_data** into a structured text corpus (49,992 articles) with metadata (source, category).

2. **Tokenization** – Splits text into words, removing punctuation, numbers, symbols, and URLs.

3. **Document-Feature Matrix (DFM)** – Creates a numeric word-frequency matrix.

4. **Trimming Rare Words** – Keeps only terms appearing in ≥1% of documents to reduce noise.

5. **Output** – Cleaned dataset ready for text classification, topic modeling, or sentiment analysis.

6. **Purpose** - Prepares raw text for NLP and machine learning tasks.

Creating a Corpus from Cleaned text

```
news_corpus<-corpus(news_data,text_field = "content")
```

```
docvars(news_corpus,"category")<-news_data$category
docvars(news_corpus,"source")<-news_data$source

news_tokens<-tokens(news_corpus,
                    remove_punct = TRUE,
                    remove_symbols = TRUE,
                    remove_numbers = TRUE,
                    remove_url = TRUE)

news_dfm<-dfm(news_tokens)

news_dfm<-dfm_trim(news_dfm,min_docfreq = 0.01,docfreq_type = "prop")
```

Distribution Analysis: Categories & Sources

Understanding how articles are spread across different **categories** and **sources** helps us:

- Detect **imbalances** in the dataset
- Spot potential **biases**

- Decide how to **train our models** better

2.2.3 News Articles by Category

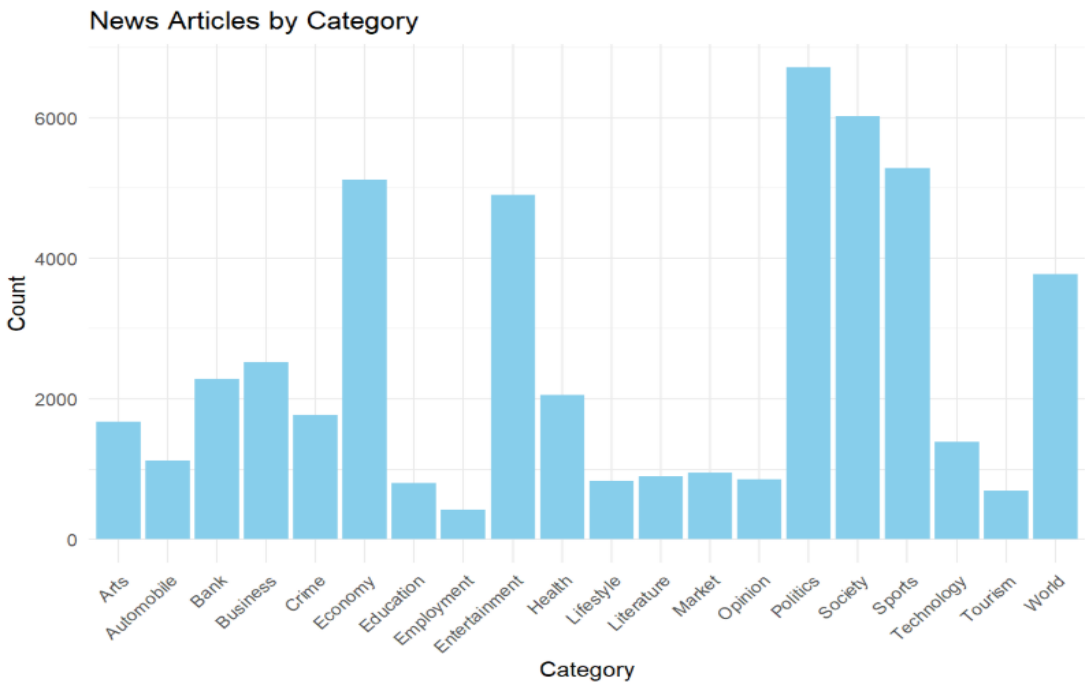
Visual: Bar chart titled *"News Articles by Category."*

What It Shows:

- Each bar = a **news category** (e.g., Politics, Sports, Entertainment)
- Height of bar = **number of articles** in that category

Category Distribution Analysis

```
category_counts <- table(news_data$category)
ggplot(data = as.data.frame(category_counts), aes(x = Var1, y = Freq)) +
  geom_bar(stat = "identity", fill = "skyblue") +
  theme_minimal() +
  labs(title = "News Articles by Category",
       x = "Category",
       y = "Count") +
  theme(axis.text.x = element_text(angle = 45,hjust=1))
```



Key Insights

- **Diverse Focus Areas:**
  - ***Daily life topics***: Sports, Entertainment, Lifestyle
  - ***Finance-related***: Business, Economy, Bank, Market
  - ***Societal issues***: Education, Health, Crime, Opinion

- ***Special interests***: Technology, Tourism, Automobile

- **Category Imbalance Likely:**
  - High-frequency categories such as ***Politics*** likely dominate the dataset.
  - Low-frequency categories such as ***Aids*** and ***Literature*** may be underrepresented.
- **What This Helps With:**
  - Identifying which topics are most commonly covered (e.g., **Sports** vs. **Technology**)
  - Spotting potential gaps (e.g., Is **Health** news underreported compared to others?)

2.2.4 News Articles by Source

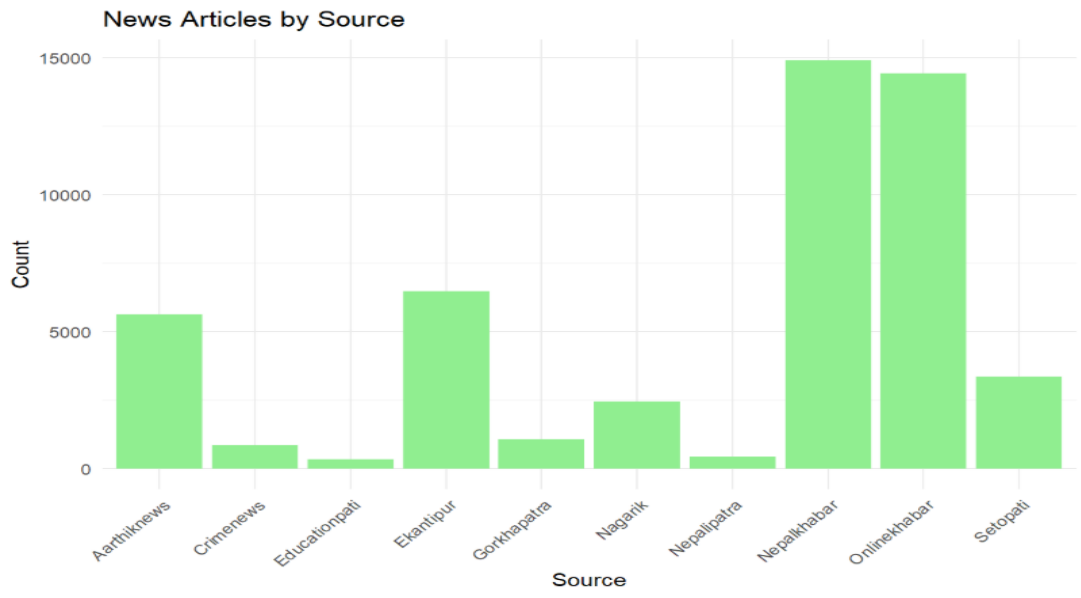
Visual: Bar chart titled *"News Articles by **Source**."*

What It Shows:

- Each bar = a **news publisher/source**
- Height of bar = **how many articles** that source contributed

Source Distribution Analysis

```
source_counts <- table(news_data$source)
ggplot(as.data.frame(source_counts), aes(x = Var1, y = Freq)) +
  geom_bar(stat = "identity", fill = "lightgreen") +
  theme_minimal() +
  labs(title = "News Articles by Source",
       x = "Source",
       y = "Count") +
  theme(axis.text.x = element_text(angle = 45,hjust=1))
```



Key Insights:

1. Top 4 Dominant Sources (by article count):

- **Nepalkhabar** – approximately 15,000 articles
- **Onlinekhabar** – approximately 13,000 articles
- **Ekantipur** – approximately 7,000 articles
- **Aarthiknews** – approximately 6,000 articles

2. Smaller Sources (fewer than 5,000 articles each):

- **Setopati**
- **Nepalipatra**
- **Nagarik**
- **Gorkhapatra**
- **Educationpati**
- **Crimenews**

2.2.5 Text Length Analysis: Headings and Content

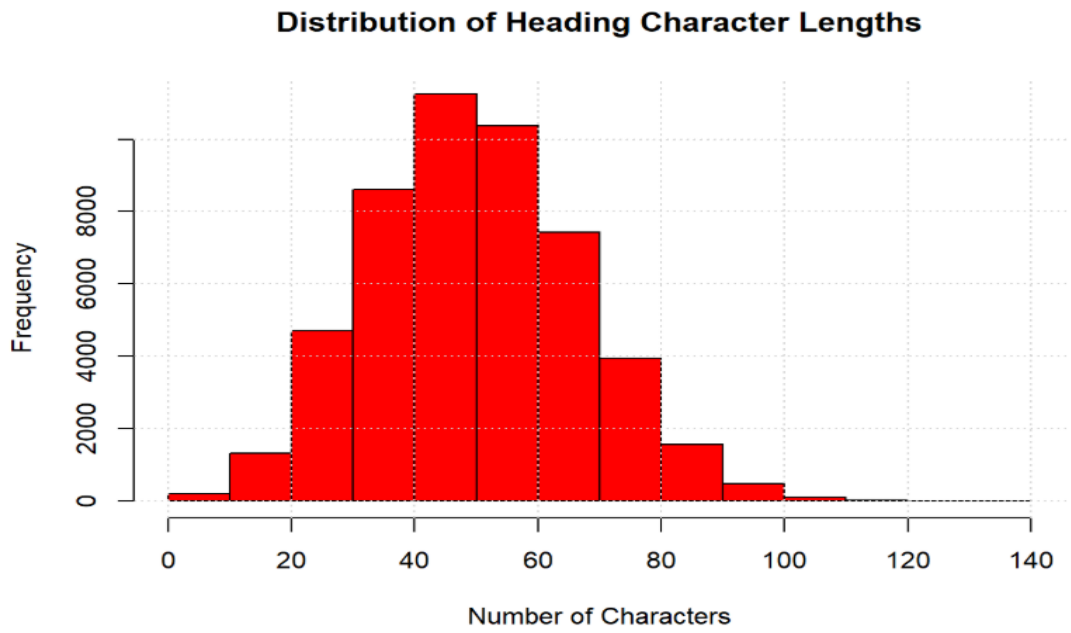
Analyzing the length of headings and article content offers valuable insight into how concise or detailed the news writing is. This information is useful for tasks such as summarization, input truncation for models, and feature engineering (e.g., using length as a predictor).

1. Heading Character Length Distribution

```
heading_character_length <- nchar(news_data$heading)
summary(heading_character_length)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    2.00   38.00   50.00   50.24   62.00  133.00
```

```
hist(
  heading_character_length,
  main = "Distribution of Heading Character Lengths",
  xlab = "Number of Characters",
  ylab = "Frequency",
  col = "red",
  border = "black")
grid()
```



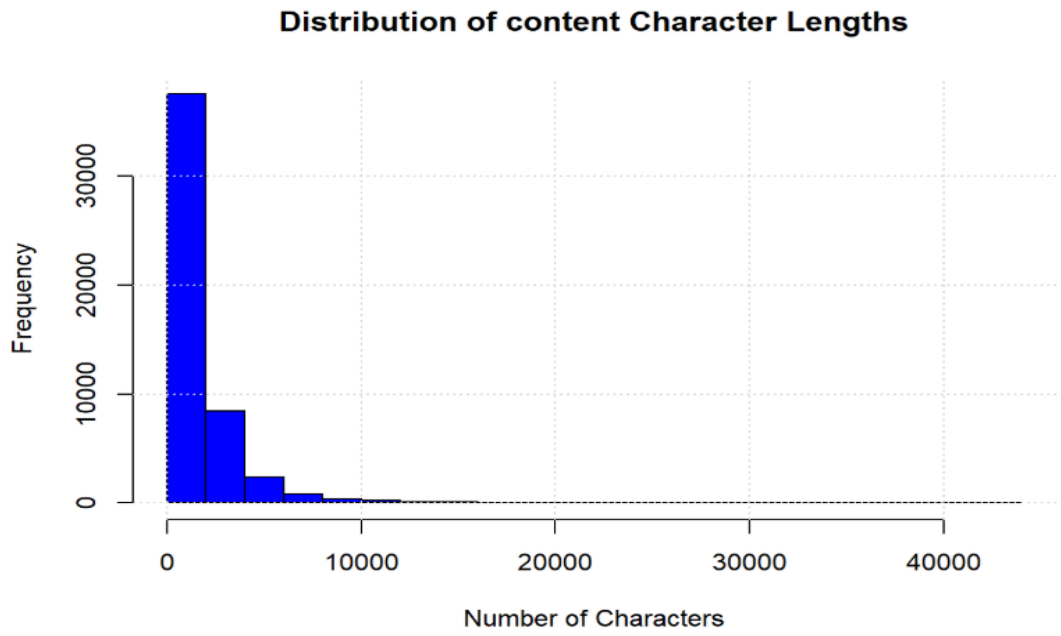
- This histogram shows how many characters each heading contains.
- It helps identify the typical size and style of headlines in the dataset.
- Important patterns to look for include:
    - **Central tendency** (mean or median character length)
    - **Spread** (range or standard deviation)
    - **Outliers** (extremely short or long headings)

2. Content Character Length Distribution

```
content_character_length <- nchar(news_data$content)
summary(content_character_length)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##       5     757    1155    1761    1996   42585
```

```
hist(
  content_character_length,
  main = "Distribution of content Character Lengths",
  xlab = "Number of Characters",
  ylab = "Frequency",
  col = "blue",
  border = "black")
grid()
```



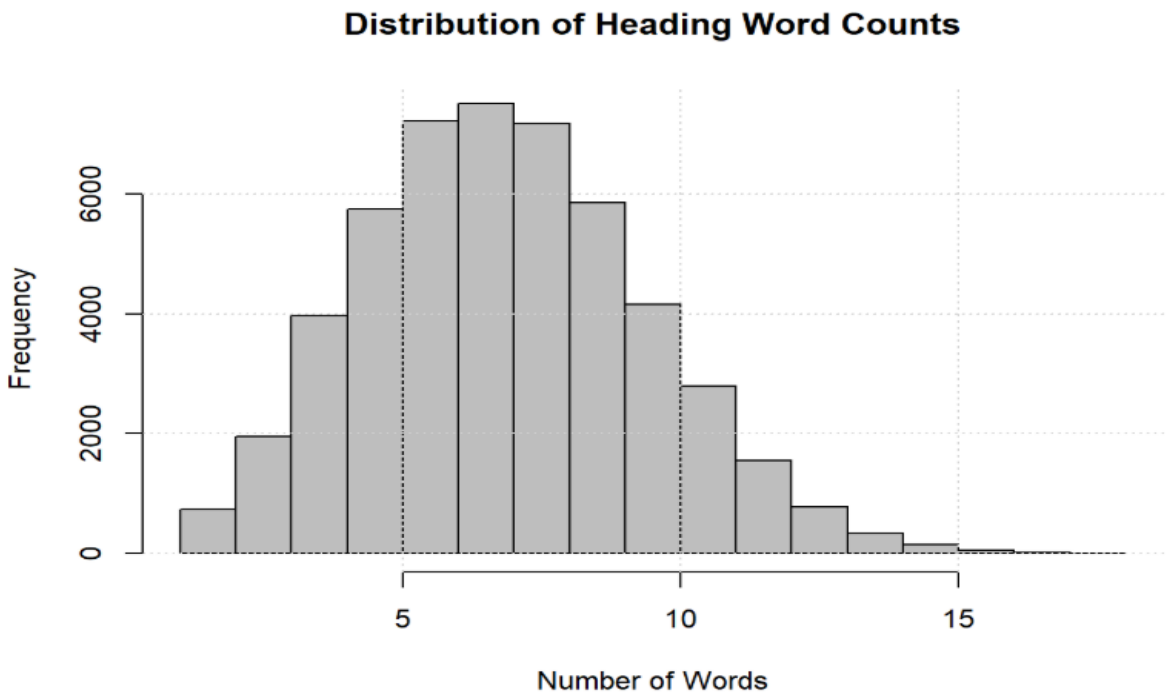
- This histogram reveals how long news articles are, in terms of character count.
- This visual is **right-skewed,** which is common in news datasets.
    - Most articles are short to moderate in length
    - A few articles are much longer (long tail)

3. Heading Word Count Distribution

```
heading_word_count <- str_count(news_data$heading, "\\w+")
summary(heading_word_count)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   1.000   6.000   7.000   7.338   9.000  18.000
```

```
hist(
  heading_word_count,
  main = "Distribution of Heading Word Counts",
  xlab = "Number of Words",
  ylab = "Frequency",
  col = "grey",
  border = "black"
)
grid()
```



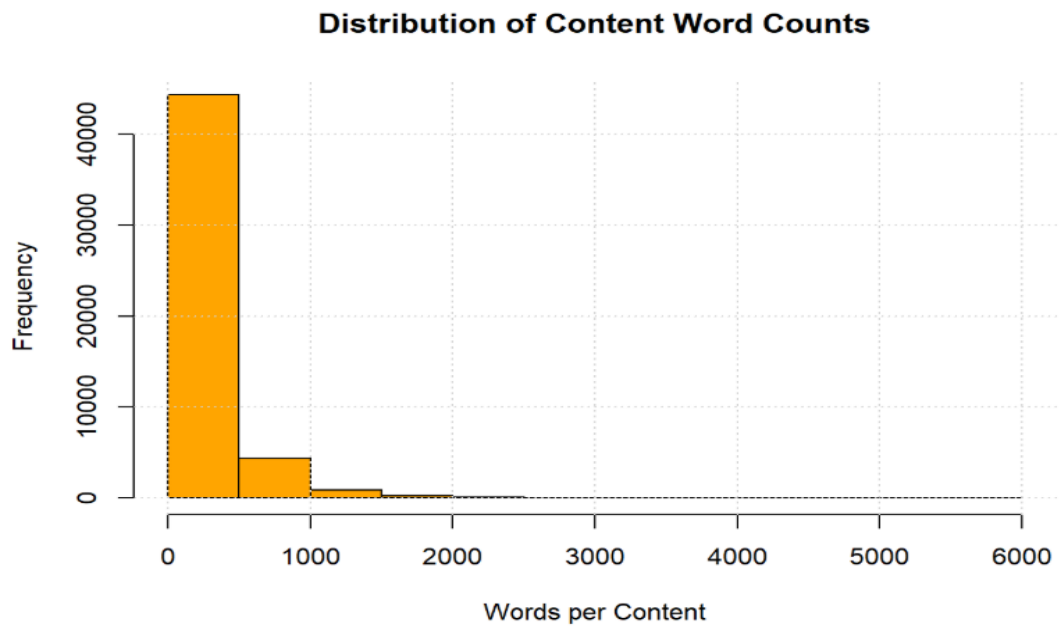
- This histogram focuses on **word counts**, offering a more human-readable view than character counts.
- It reflects how concise or elaborate headlines are, in terms of word use.

4. Content Word Count Distribution

```
content_word_count <- str_count(news_data$content, "\\w+")
summary(content_word_count)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##     1.0   113.0   172.0   261.8   296.0  5808.0
```

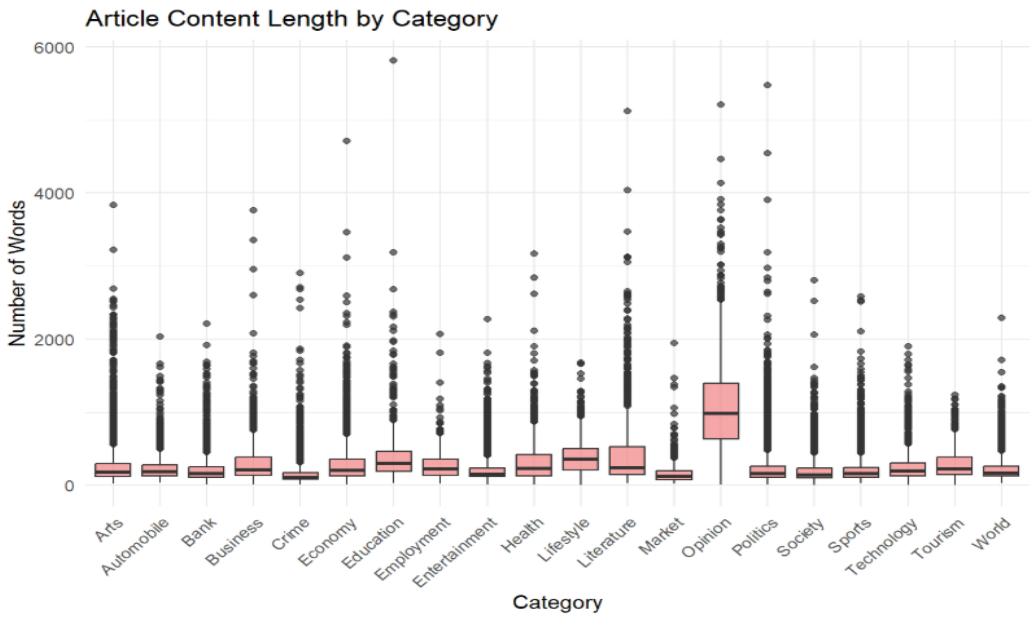
```
hist(
  content_word_count,
  main = "Distribution of Content Word Counts",
  xlab = "Words per Content",
  ylab = "Frequency",
  col = "orange",
  border = "black"
)
grid()
```



- This shows how many words are in the main body of each article.
- Like character count, the distribution often skews right, meaning:
    - Most articles are moderate in length
    - A few are very long

Content Length by Category (Boxplot)

```
ggplot(news_data, aes(x = category, y = content_word_count)) +
  geom_boxplot(fill = "lightcoral", alpha = 0.7) +
  labs(
    title = "Article Content Length by Category",
    x = "Category",
    y = "Number of Words"
  ) +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```



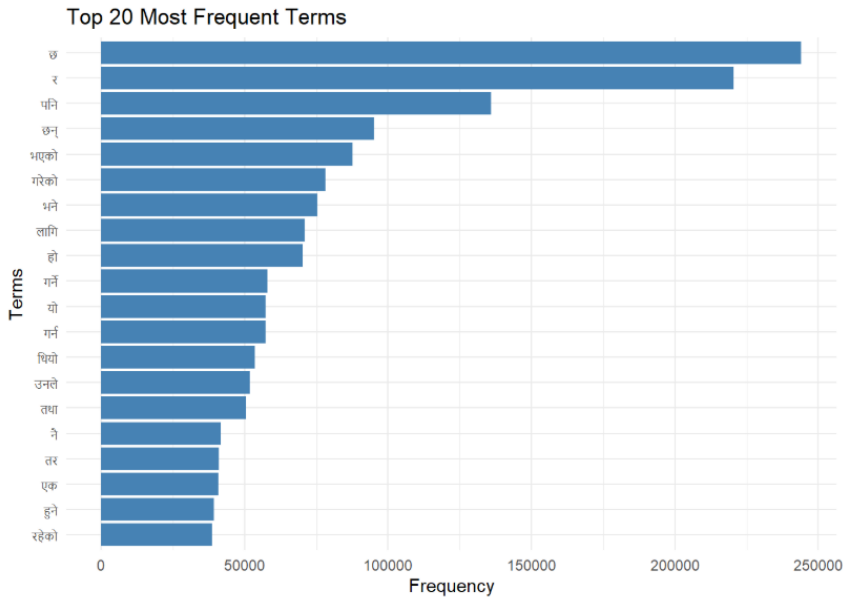
This visualization compares the distribution of article lengths (in word count) across different news categories. Each category is represented by a boxplot, showing how content length varies within and between categories.

2.2.6 Frequency Analysis: Terms and N-grams

Frequency analysis is key to understanding the **core vocabulary, themes, and patterns** in any text dataset. For this Nepali news corpus, we used frequency-based visualizations (bar charts and word cloud) to uncover the most used terms and phrases.

- **Top 20 Most Frequent Terms (Unigrams)**

```
ggplot(head(top_features_df,20),aes(x=reorder(Term,Frequency),
 y= Frequency))+geom_col(fill="steelblue")+coord_flip()+
  labs(title = "Top 20 Most Frequent Terms",
       x="Terms",y="Frequency")+theme_minimal()
```



- This chart displays the **single most used words** in the dataset.
- Each bar shows how often a word appears.
- It gives a quick snapshot of what topics or entities dominate the articles.
- These frequent terms are especially helpful for:
    - **Text classification**
    - **Topic modeling**
    - **Keyword-based summarization**

Word Cloud

```
set.seed(123)
wordcloud(words = top_features_df$Term[1:50],
          freq = top_features_df$Frequency[1:50],
          min.freq = 2,
          max.words = 100,
          random.order = FALSE,
          rot.per = 0.35,
          colors = brewer.pal(8, "Dark2"))
```

- A more visual, artistic way to see word frequency.
- Bigger words = more frequent in the dataset.

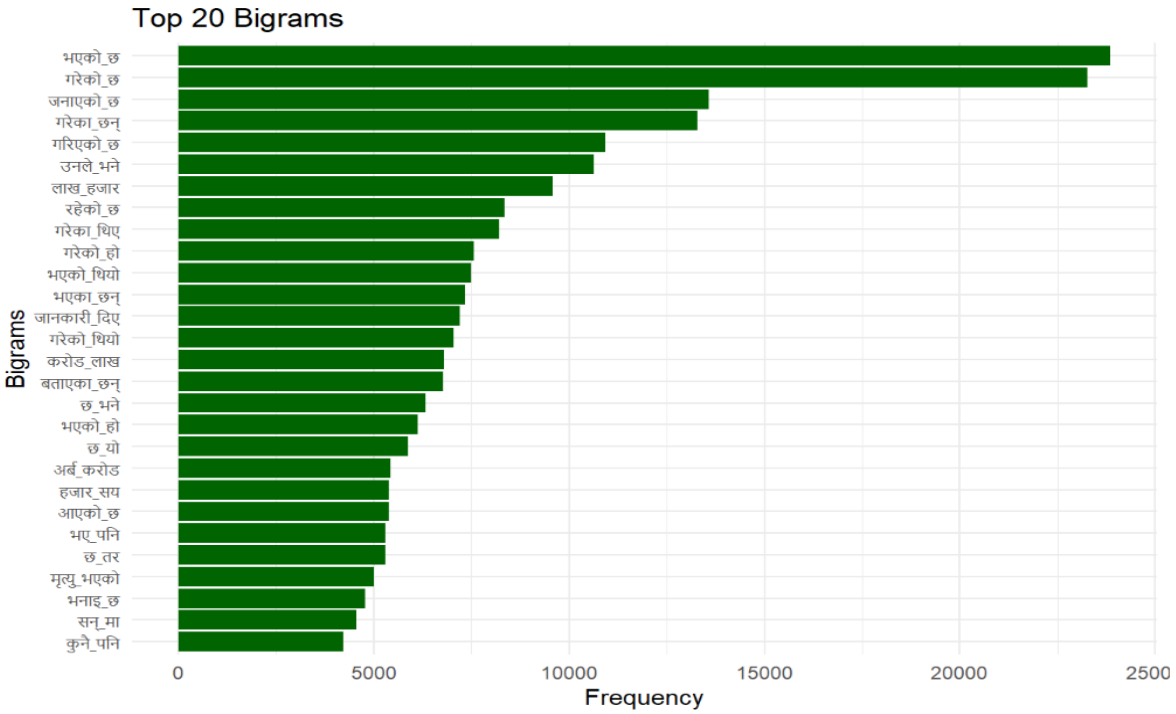
Top 20 Bigrams (2-word phrases)

Bigram Analysis:

```
news_bigrams<-tokens_ngrams(news_tokens,n=2)
bigram_dfm<-dfm(news_bigrams)
top_bigrams<-topfeatures(bigram_dfm,28)
```

Visualize the bigrams

```
bigram_df<-data.frame(
  Bigram=names(top_bigrams),
  Frequency=as.numeric(top_bigrams)
)

ggplot(bigram_df,aes(x=reorder(Bigram,Frequency),y=Frequency))+
  geom_col(fill="darkgreen")+
  coord_flip()+
  labs(title = "Top 20 Bigrams",
       x="Bigrams",y="Frequency")+theme_minimal()
```



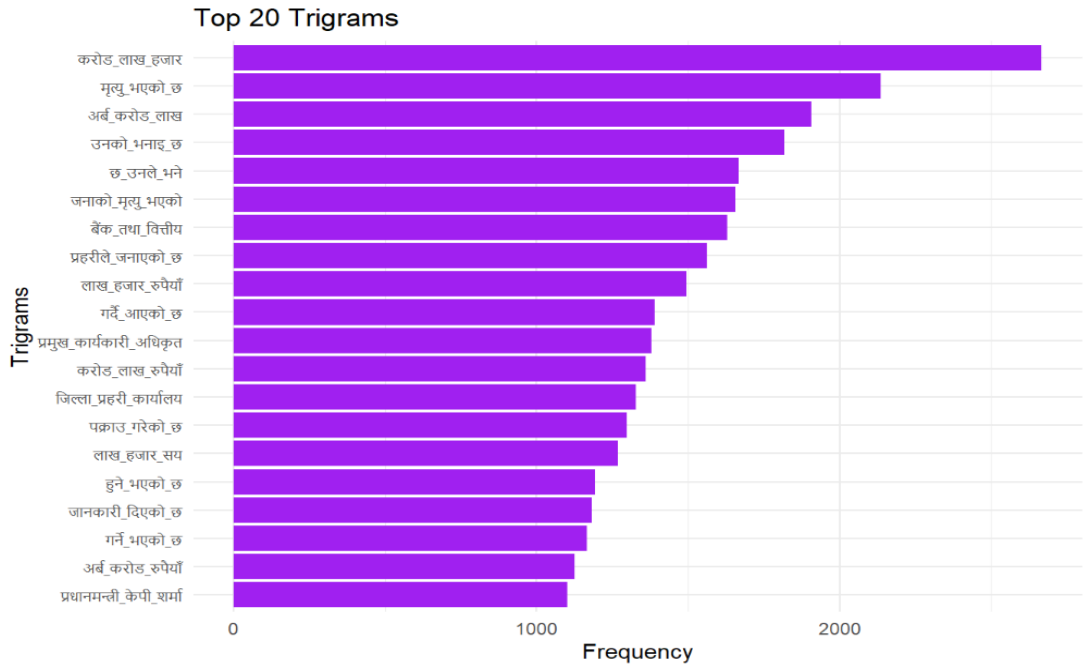
- Shows the most common **word pairs** in the news.
- These are often **more informative** than single words.
- Helps discover:
  - **Common expressions**
  - **Entity names**
  - **Specific events or topics**
- Very useful in:
  - **Topic clustering**
  - **Entity recognition**
  - **Better tokenization strategies**

◇ Top 20 Trigrams (3-word phrases)

```
news_trigrams<-tokens_ngrams(news_tokens,n=3)
trigram_dfm<-dfm(news_trigrams)
top_trigrams<-topfeatures(trigram_dfm,20)
```

Visualize the trigrams

```
trigram_df <- data.frame(
  Trigram = names(top_trigrams),
  Frequency = as.numeric(top_trigrams)
)

ggplot(trigram_df, aes(x = reorder(Trigram, Frequency), y = Frequency)) +
  geom_col(fill = "purple") +
  coord_flip() +
  labs(
    title = "Top 20 Trigrams",
    x = "Trigrams",
    y = "Frequency"
  ) +
  theme_minimal()
```



- Lists the most frequent **three-word combinations**.
- Super helpful for:
    - **Information extraction**
    - **Deep NLP modeling**
    - **Entity and event detection**

2.2.7 Lexical Diversity Analysis

Lexical diversity is a measure of how **rich and varied the vocabulary** is in a given text. It tells us how often new words are introduced instead of repeating the same ones. In this project, lexical diversity was analyzed across news categories using **Type-Token Ratio (TTR)**, a common metric in text analysis.

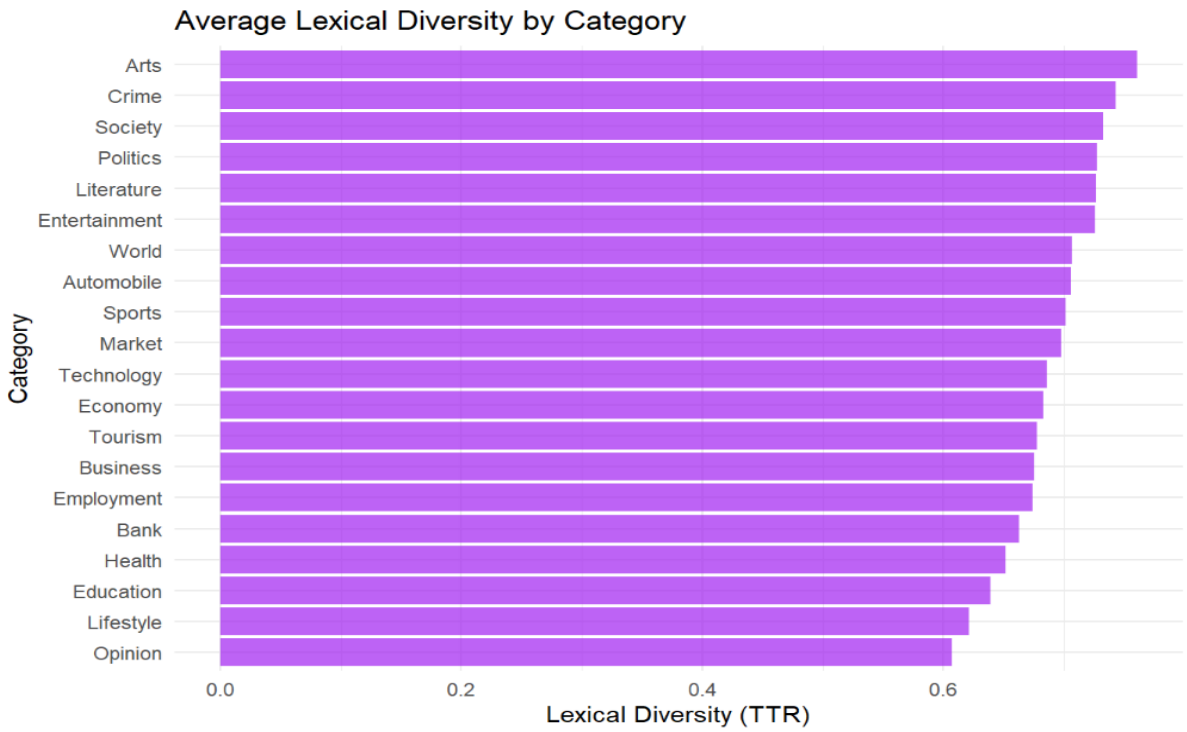
- **Average Lexical Diversity by Category**

Calculate lexical diversity (Type-Token Ratio)

```
calculate_lexical_diversity <- function(tokens) {
types <- length(unique(unlist(tokens)))
tokens_total <- length(unlist(tokens))
return(types / tokens_total)
}
```

Visualize lexical diversity

```
ggplot(lexical_diversity, aes(x = reorder(category, avg_lexical_diversity),
y = avg_lexical_diversity)) +geom_col(fill = "purple",
alpha = 0.7) +coord_flip() +
labs(title = "Average Lexical Diversity by Category",
x = "Category", y = "Lexical Diversity (TTR)") +
theme_minimal()
```



- Each bar represents the **average TTR** for one news category.
- TTR = (Number of unique words) ÷ (Total number of words).
- Higher TTR means more **vocabulary variety**.
- Lower TTR means **more repetition** and limited vocabulary.

Why It Matters:

- **Category-specific Modeling:**
  - Knowing the TTR helps us build better **language models** for each category.
  - A political news model may not need as much vocabulary flexibility, while an opinion piece model might.
- **Feature for Classification:**
  - Lexical diversity can be used as a **feature** in machine learning.
  - Helps a model distinguish between repetitive styles (like news reporting) and expressive writing (like editorials).

2.3 News Classification

- **Feature Engineering:** Convert the cleaned and preprocessed text into numerical features suitable for machine learning models using **TF-IDF (Term Frequency-Inverse Document Frequency)** vectorization. This technique weights words based on their importance in a document relative to the entire corpus.
-

```
# Create DFMs
train_dfm <- dfm(train_tokens)
test_dfm <- dfm(test_tokens)

# Apply feature trimming
train_dfm <- dfm_trim(train_dfm, min_docfreq = 0.01, docfreq_type = "prop")
test_dfm <- dfm_match(test_dfm, features = featnames(train_dfm))
```

```
# Apply TF-IDF weighting
train_tfidf <- dfm_tfidf(train_dfm)
test_tfidf <- dfm_tfidf(test_dfm)
```

```
# Check dimensions
cat("Training DFM dimensions:", dim(train_dfm), "\n")
```

```
## Training DFM dimensions: 16012 2390
```

```
cat("Test DFM dimensions:", dim(test_dfm), "\n")
```

```
## Test DFM dimensions: 3993 2390
```

- **Model Selection and Training:** The dataset will be split into training (e.g., 80%) and testing (e.g., 20%) sets.
-

```
# Set seed for reproducibility
set.seed(123)

# Create training and testing sets from 20K stratified subset (80-20 split)
trainIndex <- createDataPartition(news_data_20k$category, p = 0.8, list = FALSE)
train_data <- news_data_20k[trainIndex, ]  # Using 20K subset
test_data <- news_data_20k[-trainIndex, ]   # Using 20K subset
```

```
# Continue with the rest of your classification pipeline
train_corpus <- corpus(train_data, text_field = "content")
test_corpus <- corpus(test_data, text_field = "content")

train_tokens <- tokens(train_corpus, remove_punct = TRUE, remove_symbols = TRUE, remove_numbers = TRUE)
test_tokens <- tokens(test_corpus, remove_punct = TRUE, remove_symbols = TRUE, remove_numbers = TRUE)
```

Several supervised machine learning models will be explored and trained:

Naive Bayes : A probabilistic classifier well-suited for text categorization. Naive Bayes works in a few simple steps. First, it learns from past data by counting how many times each word appears in each category. For example, it might be observed that the word "election" appears frequently in political news, while "goal" appears often in sports articles. Next, it calculates the probabilities of each word belonging to a particular category based on this data. Finally, when a new headline comes in, it checks which category has the highest total probability based on the words in that headline and assigns the headline to that category. This way, it makes predictions using simple counting and probability.

```
# Model 1: Naive Bayes using quanteda.textmodels (sparse-aware)
cat("Training Naive Bayes model...\n")
```

```
## Training Nai
```

```
nb_model <- text
nb_predictions
nb_confusion <-

cat("=== NAIVE
```

```
## === NAIVE BA
```

```
# Model 2: SVM using e1071 with sparse matrices
cat("Training SVM model...\n")
```

```
## Training SVM model...
```

```
# Convert to sparse format for e1071
train_sparse <- as(train_tfidf, "dgCMatrix")
test_sparse <- as(test_tfidf, "dgCMatrix")

# **KEY FIX**: Ensure categories are factors for classification
train_data$category <- as.factor(train_data$category)
test_data$category <- as.factor(test_data$category)

svm_model <- svm(train_sparse, train_data$category,
                 kernel = "linear", cost = 1)
svm_predictions <- predict(svm_model, test_sparse)
svm_confusion <- confusionMatrix(svm_predictions, as.factor(test_data$category))

cat("\n=== SVM RESULTS ===\n")
```

```
##
## === SVM RESULTS ===
```

```
## Overall Statistics
##
##                Accuracy : 0.6574
##                  95% CI : (0.6424, 0.6721)
##     No Information Rate : 0.1342
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.6304
##
##  Mcnemar's Test P-Value : NA
```

Support Vector Machines (SVM): Effective for high-dimensional data common in text analysis.Support Vector Machines (SVM) work by finding the best boundary, called a hyperplane, that separates different categories of data. In the case of text classification, each document or headline is turned into a set of numbers (like word counts or TF-IDF scores), creating a high-dimensional space where each dimension represents a word. SVM looks for the line or surface that best separates one category from another with the widest possible gap between them. When a new headline is given, SVM checks which side of the boundary it falls on and classifies it accordingly. This makes SVM very effective for text analysis, where data often has many features.

```
## Overall Statistics
##
##                  Accuracy : 0.6897
##                    95% CI : (0.6751, 0.704)
##       No Information Rate : 0.1342
##       P-Value [Acc > NIR] : < 2.2e-16
##
##                     Kappa : 0.662
##
##  Mcnemar's Test P-Value : NA
```

Random Forest: An ensemble method known for its robustness, accuracy, and ability to handle complex relationships.Random Forest works by building a large number of decision trees and combining their results to make a final prediction. Each decision tree is trained on a random part of the data and selects random features (like words in text classification) to make decisions. When a new headline needs to be classified, each tree gives its own prediction, and the Random Forest takes a majority vote to decide the final category. This method helps reduce errors from individual trees, making the overall model more accurate and stable. It's especially good at handling complex patterns and noisy data, which is common in real-world text analysis.

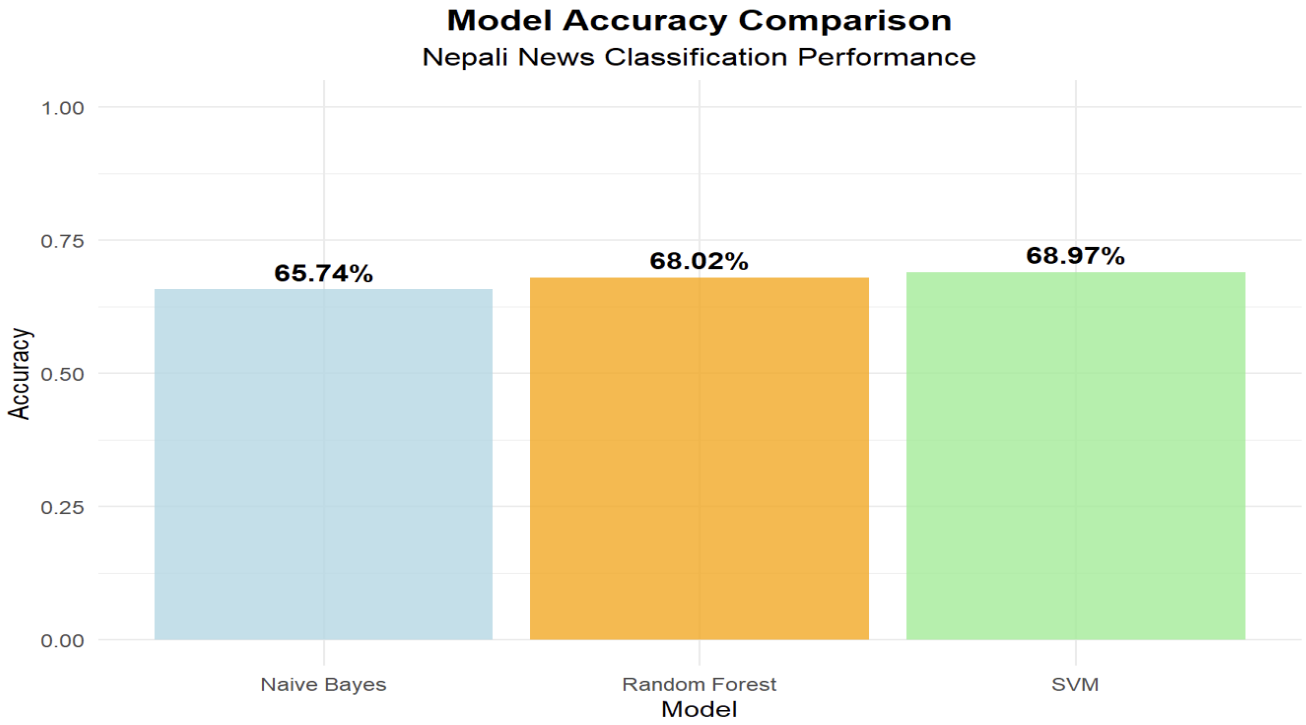
```
# Train Random Forest using ranger (much faster)
rf_model <- ranger(category ~ .,
                   data = train_features,
                   num.trees = 50,
                   num.threads = parallel::detectCores() - 1,
                   verbose = TRUE,
                   seed = 123)

# Make predictions
rf_predictions <- predict(rf_model, test_features)$predictions

# Evaluate performance
rf_confusion <- confusionMatrix(rf_predictions, test_features$category)

cat("\n=== RANDOM FOREST RESULTS (Ranger Package) ===\n")
```

```
##
## === RANDOM FOREST RESULTS (Ranger Package) ===
```

```
print(rf_confusion)
```

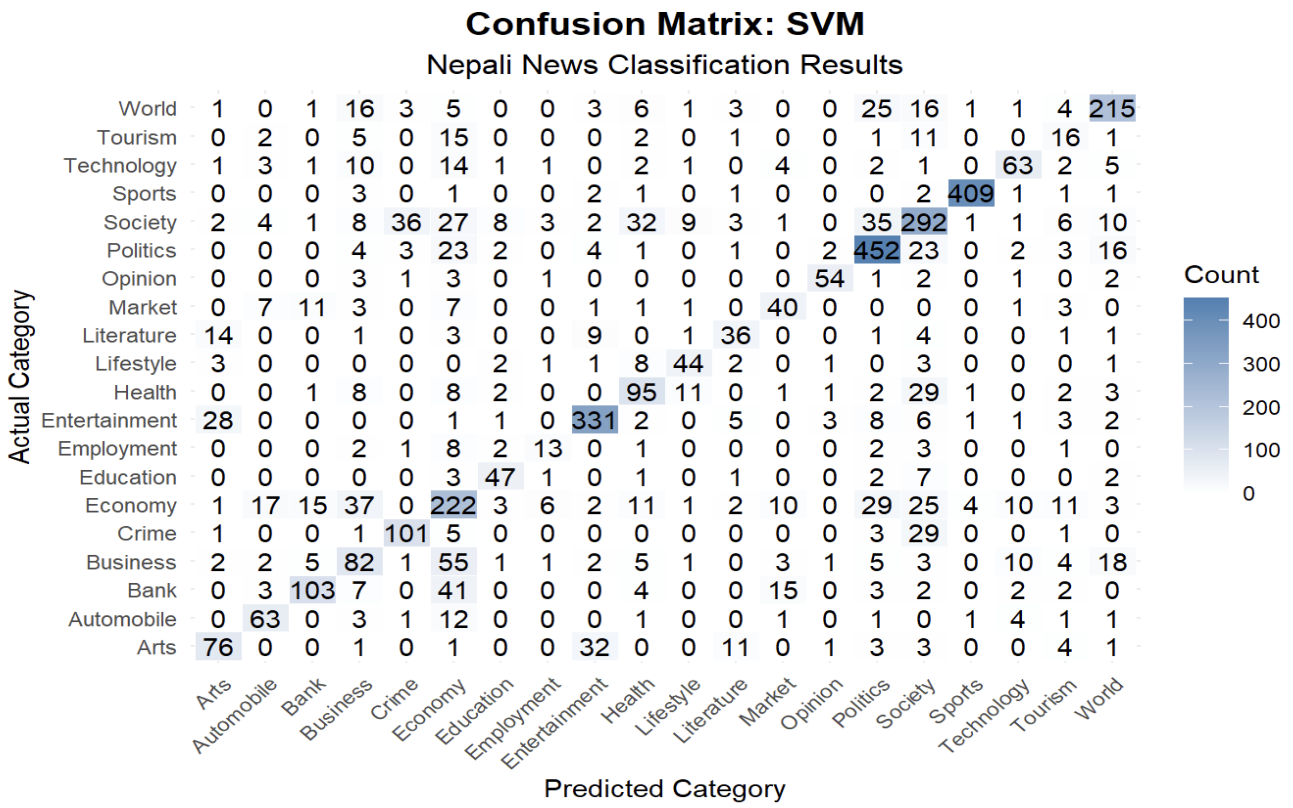
```
## Overall Statistics
##
##                  Accuracy : 0.6802
##                    95% CI : (0.6655, 0.6946)
##       No Information Rate : 0.1342
##       P-Value [Acc > NIR] : < 2.2e-16
##
##                     Kappa : 0.6479
##
##  Mcnemar's Test P-Value : NA
```

- **Evaluation Metrics:** Model performance on the unseen test set will be rigorously assessed using:
- Accuracy:



- Precision, Recall, and F1-score (especially crucial for potentially imbalanced classes) :
- Confusion Matrix (to understand misclassifications)

```
# Create confusion matrix for the best performing model
if (best_model == "Naive Bayes") {
  best_confusion <- nb_confusion
} else if (best_model == "SVM") {
  best_confusion <- svm_confusion
} else {
  best_confusion <- rf_confusion
}

# Convert confusion matrix to data frame for visualization
cm_data <- as.data.frame(best_confusion$table)

# Create confusion matrix heatmap
cm_plot <- ggplot(cm_data, aes(x = Prediction, y = Reference, fill = Freq)) +
  geom_tile(color = "white") +
  scale_fill_gradient(low = "white", high = "steelblue") +
  geom_text(aes(label = Freq), vjust = 0.5) +
  theme_minimal() +
  labs(
    title = paste("Confusion Matrix:", best_model),
    subtitle = "Nepali News Classification Results",
    x = "Predicted Category",
    y = "Actual Category",
    fill = "Count"
  ) +
  theme(
    plot.title = element_text(hjust = 0.5, size = 14, face = "bold"),
    plot.subtitle = element_text(hjust = 0.5, size = 12),
    axis.text.x = element_text(angle = 45, hjust = 1),
    axis.text.y = element_text(angle = 0)
  )

print(cm_plot)
```



- **Key Tools:** caret,e1071,randomForest, RTextTools.

2.4 Sentiment Analysis

Dataset: https://www.kaggle.com/datasets/aayamoza/nepali-sentiment-analysis

- A **CSV file** containing thousands of **Nepali news headlines**.
- Each headline was labeled with sentiment:
  - **-1 = Negative**
  - **0 = Neutral**
  - **1 = Positive**

```
# STEP 1: LOAD DATA
data <- read_csv("nepali_sentiment_lexicon.csv")
```

```
## New names:
## Rows: 35789 Columns: 3
## ── Column specification
## ──────────────────────────────────────────────────────── Delimiter: "," chr
## (1): Sentences dbl (2): ...1, Sentiment
## ℹ Use `spec()` to retrieve the full column specification for this data. ℹ
## Specify the column types or set `show_col_types = FALSE` to quiet this message.
## • `` -> `...1`
```

```r
# STEP 3: SPLIT DATA
set.seed(123)
split <- createDataPartition(data$Sentiment, p = 0.8, list = FALSE)
train <- data[split, ]
test <- data[-split, ]
```

```r
# STEP 4: CREATE DTM (Document-Term Matrix)
it_train <- itoken(train$clean_text)
vocab <- create_vocabulary(it_train)
vectorizer <- vocab_vectorizer(vocab)
dtm_train <- create_dtm(it_train, vectorizer)

it_test <- itoken(test$clean_text)
dtm_test <- create_dtm(it_test, vectorizer)
```

```r
# STEP 5: TRAIN MODEL
model <- cv.glmnet(x = dtm_train, y = train$Sentiment, family = "multinomial", type.measure = "class")
```

```r
# STEP 6: EVALUATE MODEL
pred <- predict(model, dtm_test, s = "lambda.min", type = "class")
confusionMatrix(as.factor(pred), as.factor(test$Sentiment))
```

```
## Confusion Matrix and Statistics
##
##            Reference
## Prediction   -1    0    1
##         -1 1563  468 1143
##         0     7    8    2
##         1  1363  572 2031
##
## Overall Statistics
##
##                Accuracy : 0.5033
##                  95% CI : (0.4916, 0.5149)
##     No Information Rate : 0.4438
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.1316
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
## Statistics by Class:
##
##                      Class: -1 Class: 0 Class: 1
## Sensitivity             0.5329 0.007634   0.6395
## Specificity             0.6186 0.998527   0.5139
## Pos Pred Value          0.4924 0.470588   0.5121
## Neg Pred Value          0.6560 0.854342   0.6412
## Prevalence              0.4098 0.146430   0.4438
## Detection Rate          0.2184 0.001118   0.2838
## Detection Prevalence    0.4435 0.002375   0.5541
## Balanced Accuracy       0.5758 0.503080   0.5767
```

- **Polarity Scoring:** Assign sentiment scores to individual words based on the lexicon and aggregate these scores to determine the overall polarity (positive, negative, or neutral) of each news article.

- **Comparative Analysis:** Conduct an in-depth analysis comparing sentiment across different news sources, categories, and over time to identify prevailing emotional tones and potential biases.

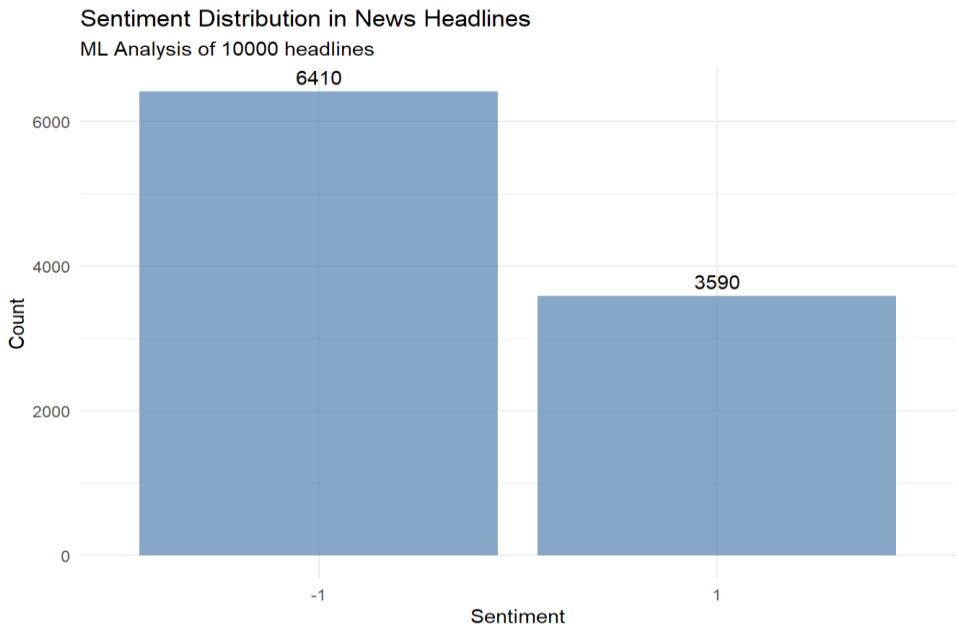
```r
# Load your main news dataset
news_data <- read_csv("50k_news_dataset.csv", show_col_types = FALSE)

# Take a sample for sentiment analysis (adjust size as needed)
set.seed(123)
sample_size <- 10000
sample_indices <- sample(1:nrow(news_data), sample_size)
news_sample <- news_data[sample_indices, ]


# Analyze sentiment of sample headlines
headline_sentiment_results <- predict_sentiment(news_sample$heading, model, vectorizer)

# Add results to the sample dataset
news_sample$predicted_sentiment <- headline_sentiment_results$predicted_sentiment
news_sample$sentiment_confidence <- headline_sentiment_results$confidence
```

```r
# Display sample results
sample_display <- news_sample[1:10, c("category", "predicted_sentiment", "sentiment_confidence")]
print(sample_display)
```

```
## # A tibble: 10 × 3
##    category      predicted_sentiment sentiment_confidence
##    <chr>         <chr>                             <dbl>
##  1 Sports        1                                 0.405
##  2 Sports        -1                                0.4
##  3 Literature    -1                                0.418
##  4 Society       -1                                0.4
##  5 Sports        -1                                0.418
##  6 Bank          1                                 0.417
##  7 Politics      -1                                0.423
##  8 Sports        -1                                0.437
##  9 Crime         -1                                0.418
## 10 Entertainment 1                                 0.468
```

- **Visualization:** Create informative visualizations such as sentiment distribution charts, trend plots, and heatmaps.

```r
p1 <- ggplot(news_sample, aes(x = predicted_sentiment)) +
  geom_bar(fill = "steelblue", alpha = 0.7) +
  labs(title = "Sentiment Distribution in News Headlines",
       subtitle = paste("ML Analysis of", sample_size, "headlines"),
       x = "Sentiment", y = "Count") +
  theme_minimal() +
  geom_text(stat = 'count', aes(label = after_stat(count)), vjust = -0.5)

print(p1)
```



```r
# Model confidence distribution (optional but valuable)
p3 <- ggplot(news_sample, aes(x = sentiment_confidence)) +
  geom_histogram(bins = 20, fill = "steelblue", alpha = 0.7) +
  labs(title = "Model Confidence Distribution",
       x = "Confidence Score", y = "Frequency") +
  theme_minimal() +
  geom_vline(xintercept = mean(news_sample$sentiment_confidence),
             color = "red", linetype = "dashed")

print(p3)
```

2.5 Text Summarization

For summarization, the project employs the unsupervised TextRank algorithm, which extracts key sentences from each article. Articles are segmented into sentences and structured into a graph, where edges represent sentence similarity. The algorithm ranks sentences based on their centrality within the graph structure, and the top-ranked sentences are selected to form the summary. This method produces concise yet informative summaries, reducing the article length while retaining core information, and does not depend on any labelled dataset or language-specific deep learning models.

```
# Function to apply TextRank summarization
summarize_with_textrank <- function(text, n_sentences = 3) {
  # Preprocess text for TextRank
  sentences <- unlist(strsplit(text, "\\.|\\!|\\?"))
  sentences <- trimws(sentences)
  sentences <- sentences[sentences != ""]

  if (length(sentences) <= n_sentences) {
    return(paste(sentences, collapse = " "))
  }

  # Create data frame for TextRank
  sentences_df <- data.frame(
    sentence_id = 1:length(sentences),
    sentence = sentences,
    stringsAsFactors = FALSE
  )

  # Create terminology (words) for each sentence
  terminology <- data.frame()
  for (i in 1:length(sentences)) {
    words <- unlist(strsplit(sentences[i], "\\s+"))
    words <- words[nchar(words) > 2]  # Filter short words
    if (length(words) > 0) {
      terminology <- rbind(terminology,
                           data.frame(sentence_id = i,
                                      word = words,
                                      stringsAsFactors = FALSE))
    }
  }

  if (nrow(terminology) == 0) {
    return(paste(sentences[1:min(n_sentences, length(sentences))], collapse = " "))
  }

  # Apply TextRank
  tr <- textrank_sentences(data = sentences_df,
                           terminology = terminology)

  # Get top sentences
  top_sentences <- summary(tr, n = n_sentences, keep.sentence.order = TRUE)

  return(paste(top_sentences, collapse = " "))
}

# Apply summarization to a subset of articles
sample_indices <- sample(1:nrow(news_data), min(100, nrow(news_data)))
news_data$summary <- NA

for (i in sample_indices) {
  tryCatch({
    news_data$summary[i] <- summarize_with_textrank(news_data$content[i], 3)
  }, error = function(e) {
    news_data$summary[i] <- substr(news_data$content[i], 1, 200)
  })
}

# Display sample summaries
for (i in 1:3) {
  cat("Original:", substr(news_data$content[sample_indices[i]], 1, 300), "...\n\n")
  cat("Summary:", news_data$summary[sample_indices[i]], "\n\n")
  cat("---\n\n")
}
```

```
## Original: १३ वैशाख, वीरगञ्ज । विभिन्न पार्टी निकट मजदुर संगठनले श्रमिकको पारिश्रमिक भुक्तानी गर्न माग गरेका छन् । नेपाल ट्रेड युनियन महासंघ, अखिल ने
पाल ट्रेड युनियन महासंघ र नेपाल ट्रेड युनियन कांग्रेसका प्रदेश अध्यक्षहरुले संयुक्त विज्ञप्ति जारी गर्दै त्यस्तो माग गरेका हुन् । सरकारले २०७६ चैत १६ गते औपचारिक
...
##
## Summary: १३ वैशाख, वीरगञ्ज । विभिन्न पार्टी निकट मजदुर संगठनले श्रमिकको पारिश्रमिक भुक्तानी गर्न माग गरेका छन् । नेपाल ट्रेड युनियन महासंघ, अखिल नेपा
ल ट्रेड युनियन महासंघ र नेपाल ट्रेड युनियन कांग्रेसका प्रदेश अध्यक्षहरुले संयुक्त विज्ञप्ति जारी गर्दै त्यस्तो माग गरेका हुन् । सरकारले २०७६ चैत १६ गते औपचारिक क्षेत्रमा
कार्यरत श्रमिकहरुले खाइपाइ आएको सम्पूर्ण पारिश्रमिक उपलब्ध गराउनु पर्ने निर्णय गरेको भन्दै उनीहरुले श्रमिकको पारिश्रमिक उपलब्ध गराउन माग गरेका हुन् । सोही अनुसार
श्रम कार्यालय वीरगञ्जले सबै प्रतिष्ठानहरुमा परिपत्र गरी सके  पनि हालसम्म अधिकांश प्रतिष्ठानहरुले तलब भुक्तानी नगरेको विज्ञप्तिमा उल्लेख छ । संकट अवस्थाबाट गुज्रीरहे
को उद्योग व्यवसायलाई पूर्ववत अवस्थामा ल्याउन सहमतिका साथ सहकार्य गर्न तत्पर रहने प्रतिबद्धताका साथ श्रमिकहरुको हालसम्मको खाईपाई आएको सम्पूर्ण तलब सुविधा भु
क्तानीको लागि विज्ञप्तिमा माग गरिएको छ । त्यस्तै अत्यावश्यक सेवा अन्तर्गतका उद्योग प्रतिष्ठानहरुमा विश्व स्वास्थ्य संगठनद्वारा निर्धारित करोना माहामारी रोकथामको लागि अ
पनाउनु पर्ने प्रावधानका साथ सामाजिक दुरी कायम राख्दै उत्पादन कार्यमा संलग्न रहन विज्ञप्तिमा अपिल गरिएको छ । वीरगञ्जको नारायणी अस्पतालमा कार्यरत करिब २०० ज
ना कर्मचारीहरुलाई सेवाबाट हटाइएको विषयमा पनि ट्रेड युनियनले ध्यानाकर्षण भएको जनाएका छन् । कर्मचारीलाई विषम परिस्थितिमा बेरोजगार बनाइएको कार्यप्रति घोर भर्त्स
ना गर्दै सेवाबाट हटाइएका ती कर्मचारीहरुलाई स–सम्मान पुर्नबहाली गरी तलब भता उपलब्ध गराउन माग गरिएको छ । नेपाल ट्रेड युनियन महासंघका अध्यक्ष कमलेश झा, अ
खिल नेपाल ट्रेड युनियन महासंघका अध्यक्ष अर्जुन चिमरिया र नेपाल ट्रेड युनियन कांग्रेस तथा संयुक्त ट्रेड युनियन समन्वय केन्द्र प्रदेश कमिटिका अध्यक्ष लक्ष्मणप्रसाद कुर्मीले
विज्ञप्तिमा हस्ताक्षर गरेका छन् । अनावश्यक भ्रम सिर्जना गरी श्रमिकको पारिश्रमिक तथा अन्य सुबिधा कटौती गरे औद्योगिक वातावरणमा खलल पुग्ने र त्यसको जिम्मेवार सम्ब
न्धित पक्ष नै हुने समन्वय केन्द्र प्रदेश कमिटिका अध्यक्ष कुर्मीले बताए ।
##
## ---
##
## Original: २९ फागुन, काठमाडौं । प्रहरीले विशेष सुराकीको आधारमा काठमाडौंको लाजिम्पाटमा रहेको द इम्परियल क्याफेमा चेकजाँच गर्दा दुई जना लागूऔषध चरेश
सहित पक्राउ परेका छन् । पक्राउ पर्नेमा रोल्पा तालावाङ–५ घर भइ नवलपरासीको गैडाको बस्दै आएका ३२ वर्षीय विजय घर्ती मगर र म्यादीको बाबियाचौर–४, घर भइ हाल
कास्कीको नदिपुर ...
##
## Summary: २९ फागुन, काठमाडौं । प्रहरीले विशेष सुराकीको आधारमा काठमाडौंको लाजिम्पाटमा रहेको द इम्परियल क्याफेमा चेकजाँच गर्दा दुई जना लागूऔषध चरेशस
हित पक्राउ परेका छन् । पक्राउ पर्नेमा रोल्पा तालावाङ–५ घर भइ नवलपरासीको गैडाको बस्दै आएका ३२ वर्षीय विजय घर्ती मगर र म्यादीको बाबियाचौर–४, घर भइ हाल
कास्कीको नदिपुर बस्ने ३० वर्षीय सुमन रोक्का मगर छन् । भारतबाट रक्सौल र अन्य नाका हँदै काठमाडौंमा लागूऔषध भित्रिने गरेको छ । र, राजधानीका विभिन्न होटल, रे
स्टुरेन्टमा बसेर सेवन र बेचविखन हुने गरेको प्रहरीको भनाइ छ । पक्राउ परेकाहरुमाथि थप अनुसन्धान भइरहेको छ ।
##
## ---
##
## Original: ११ माघ, काठमाडौं । बालिकालाई जबरजस्तीकरणी गरेको आरोपमा पक्राउ परेका नेपाली सेनाका मेजर प्रभवविक्रम शाहलाई पुर्पक्षका लागि जेल चलान ग
रिएको छ । उनी बागलुङको २३ नम्बर बाहिनीमा कार्यरत् थिए । काठमाडौं जिल्ला अदालतले बिहीबार दिएको आदेश अनुरुप उनलाई बिहीबार जेल चलान गरिएको महानगरीय
प्रहरी वृत्त, बानेश्वरक ...
##
## Summary: ११ माघ, काठमाडौं । बालिकालाई जबरजस्तीकरणी गरेको आरोपमा पक्राउ परेका नेपाली सेनाका मेजर प्रभवविक्रम शाहलाई पुर्पक्षका लागि जेल चलान गरि
एको छ । उनी बागलुङको २३ नम्बर बाहिनीमा कार्यरत् थिए । काठमाडौं जिल्ला अदालतले बिहीबार दिएको आदेश अनुरुप उनलाई बिहीबार जेल चलान गरिएको महानगरीय प्र
हरी वृत्त, बानेश्वरका प्रहरी नायब उपरीक्षक (डिएसपी) रवि पौडेलले अनलाइनखबरलाई बताए । गत १४ पुसमा बुद्धनगरस्थित आफन्तको घरमा १४ वर्षीय बालिकालाई जबरज
स्तीकरणी गरेको पीडितको उजुरीका आधारमा उनलाई पक्राउ गरिएको थियो । शाहबिरुद्ध जबरजस्तीकरणी मुद्दा अन्तरगत बाल यौन दुराचार सम्बन्धी कसुरमा मुद्दा चलाइएको थि
यो ।
##
## ---
```

2.6 Named Entity Recognition (NER)

Named entity recognition is performed using the UDPipe pre-trained Nepali model,which provides part-of-speech tagging and basic linguistic annotation. We have used the Hindi UDPipe model since the UDPipe doesn’t consists of pretrained Nepali UDPipe Model. The model is used to identify tokens tagged as proper nouns (PROPN), which are assumed to represent named entities such as people, organizations, and places. Multi-word named entities are formed by grouping consecutive proper nouns. While the model does not provide detailed entity type labels, this approach allows us to approximate and extract meaningful named entities from the text, suitable for trend analysis and information extraction.

```
model_download <- udpipe_download_model(language = "hindi", model_dir = "models/")
```

```
## Downloading udpipe model from https://raw.githubusercontent.com/jwijffels/udpipe.models.ud.2.5/master/inst/udp
ipe-ud-2.5-191206/hindi-hdtb-ud-2.5-191206.udpipe to models//hindi-hdtb-ud-2.5-191206.udpipe
```

```
extract_named_entities <- function(text, model) {

  result <- udpipe_annotate(model, x = text)
  result_df <- as.data.frame(result)

  entities <- result_df[result_df$upos == "PROPN", "token"]

  return(unique(entities[entities != ""]))
}

sample_size <- 300
sample_articles <- sample(1:nrow(news_data_20k), sample_size)

cat("Extracting named entities from", sample_size, "articles...\n")
```

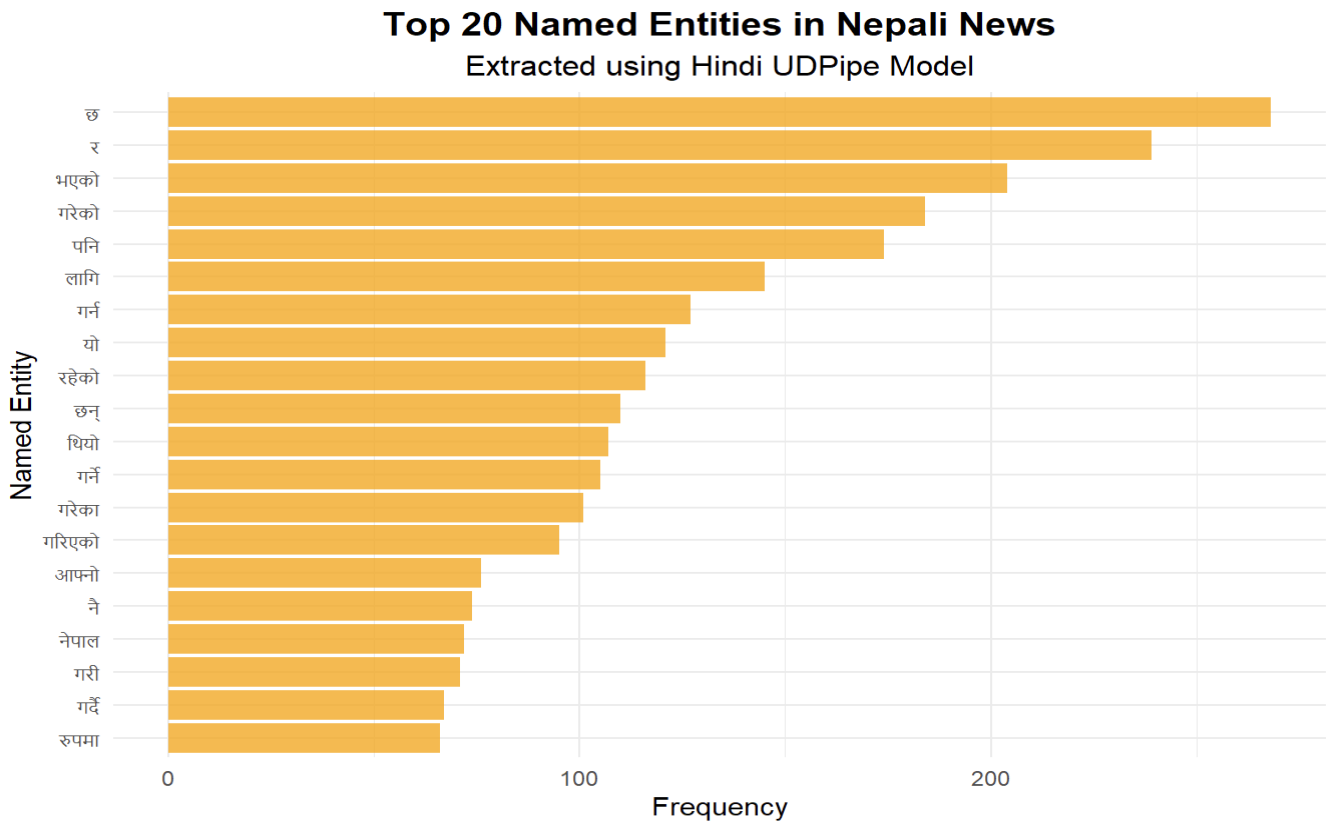
```
## Extracting named entities from 300 articles...
```

```
entity_counts <- table(all_entities)
top_entities <- sort(entity_counts, decreasing = TRUE)[1:30]

cat("\n=== TOP 30 NAMED ENTITIES ===\n")
```

```
##
## === TOP 30 NAMED ENTITIES ===
```

```
print(top_entities)
```

```
## all_entities
##       छ       र    भएको    गरेको     पनि    लागि     गर्न       यो    रहेको      छन्    थियो
##     268     239     204     184     174     145     127     121     116     110     107
##    गर्ने   गरेका गरिएको    आफ्नो      नै   नेपाल    गरी    गर्दै   रुपमा     हुन    केही
##     105     101      95      76      74      72      71      67      66      63      61
##     तर     दुई    भएका    गरेर   आएको   रहेका   कारण     हुन्
##      60      60      56      54      53      52      51      51
```



```
category_entities <- list()

for(category in unique(news_data_20k$category)) {

    cat_articles <- which(news_data_20k$category == category)
  cat_sample <- intersect(sample_articles, cat_articles)

  if(length(cat_sample) > 0) {
    cat_entities <- c()
    for(i in cat_sample) {
      entities <- extract_named_entities(news_data_20k$content[i], hindi_model)
      cat_entities <- c(cat_entities, entities)
    }

    if(length(cat_entities) > 0) {
      cat_freq <- table(cat_entities)
      top_cat_entities <- head(sort(cat_freq, decreasing = TRUE), 5)
      category_entities[[category]] <- top_cat_entities
    }
  }
}

cat("\n=== TOP ENTITIES BY CATEGORY ===\n")
```

```
##
## === TOP ENTITIES BY CATEGORY ===
```

```
for(category in names(category_entities)) {
  if(length(category_entities[[category]]) > 0) {
    cat("\n", toupper(category), ":\n")
    print(category_entities[[category]])
  }
}
```

```
##
##  ENTERTAINMENT :
## cat_entities
##     छ     र   पनि    छन् भएको
##    33    30    26    24    23
##
##  SPORTS :
## cat_entities
##     छ     र भएको गरेको थियो
##    26    25    23    18    18
##
##  AUTOMOBILE :
## cat_entities
## भएको     र गरेको  गर्ने     छ
##     4     4     3     3     3
##
##  POLITICS :
## cat_entities
##     र     छ भएको  पनि रहेको
##    39    34    31    28    25
##
##  HEALTH :
## cat_entities
##       छ कोरोना    भएको        र    थियो
##      13      11      11      11       8
##
##  ECONOMY :
## cat_entities
##     छ भएको     र गरेको  पनि
##    33    28    28    23    23
##
##  CRIME :
## cat_entities
##     छ गरेको     र  गरी    छन्
##     8     7     7     5     5
##
##  BUSINESS :
## cat_entities
## गरेको     छ    यो रहेको  पनि
##     7     7     7     7     6
##
##  TOURISM :
## cat_entities
##           छ     नेपालको     उडानको एयरलाइन्सले         गर्ने
##           3           3           2           2           2
##
##  WORLD :
## cat_entities
##     छ गरेको भएको    छन् थियो
##    22    18    16    15    15
##
```

3. Result and Discussion

3.1 Features

The developed NLP pipeline for Nepali news analytics integrates several significant features that contribute to both linguistic understanding and practical application. Firstly, the system handles pre-processing of raw Nepali text including tokenization, stopword removal, and normalization—tailored for Nepali language structures. It supports exploratory data analysis (EDA), allowing insights into word usage patterns, category distribution, and text length variations. A news classification feature powered by machine learning (Naive Bayes, SVM, and Random Forest) enables automated categorization of articles with evaluation via precision, recall, F1-score, and confusion matrix. The sentiment analysis module, using a rule-based approach with a custom Nepali lexicon, allows determination of public mood on various topics. Additionally, TextRank-based summarization and Named Entity Recognition (NER) using UDPipe provide concise article summaries and extract names of people, places, and organizations, respectively. Each of these features was carefully built to address the limitations of low-resource languages like Nepali.

3.2 Project Execution

The execution of the project followed a modular and iterative approach, ensuring robust results and flexibility. The data collection phase involved curating Nepali news articles from reliable sources, followed by comprehensive text cleaning and pre-processing to prepare the data for analysis. During the EDA phase, a variety of visual and statistical tools were employed to understand dataset characteristics and identify potential biases or imbalances. For the classification module, several models were implemented and compared, with TF-IDF vectorization used to transform text into features. The model with the highest accuracy and balanced performance was selected for final use. Sentiment analysis was executed without machine learning, making it simple and interpretable. Text summarization and NER were added as final modules to demonstrate the extensibility of the pipeline. Regular testing and validation helped maintain performance and consistency across different stages of the pipeline.

3.3 Learning Outcomes

The project provided significant technical and analytical learning experiences. From a technical perspective, the team gained hands-on experience in text mining, machine learning, and linguistic annotation using R programming in a low-resource language setting. Implementing TF-IDF, classification algorithms, and rule-based lexicons provided insights into how different models interpret text data. The importance of customizing tools for Nepali—such as manually curated stopwords and lexicons—was a key takeaway, as general tools for high-resource languages did not suffice. The execution of UDPipe for linguistic annotation demonstrated the practical use of pre-trained models. Moreover, the project highlighted the importance of modular pipeline development, allowing easy upgrades or component replacements. Beyond technical skills, the project enhanced our team collaboration, project planning, research referencing, and reporting abilities, which are critical for academic and professional success in data science and NLP.

4. Conclusions and Recommendations

4.1 Conclusion

This project successfully demonstrated the design and implementation of a modular, interpretable NLP pipeline tailored for Nepali news analytics using the R programming language. Despite working with a low-resource language, the project effectively tackled key NLP tasks including data pre-processing, exploratory data analysis (EDA), news category classification, sentiment analysis, extractive summarization, and named entity recognition (NER). Each module was developed with flexibility and scalability in mind, leveraging rule-based and statistical approaches suited to Nepali language constraints. The outcomes validate the feasibility of performing structured textual analysis on Nepali data and provide a strong foundation for further linguistic, journalistic, or academic research.

4.2 Recommendation

For teams or researchers working on similar low-resource language projects, it is recommended to begin with domain-specific data collection and manual lexicon development when language resources are unavailable. Emphasis should be placed on interpretable and modular systems over black-box deep learning models, especially when computational or data limitations exist. Additionally, using well-documented and flexible platforms like R with packages such as tm, caret, udpipe, and ggplot2 enables reproducibility and ease of collaboration. It is also advised to evaluate multiple models during classification to ensure performance balance across different categories. Lastly, meaningful visualizations and basic statistical summaries should accompany all NLP stages to aid understanding and interpretation.

4.3 Future Enhancement

While the current pipeline performs well within its scope, several enhancements can further improve performance and applicability. Future work can focus on integrating deep learning techniques such as LSTM or transformers using frameworks like keras or torch (via R or Python) for improved classification and sentiment accuracy. Building or fine-tuning pre-trained word embeddings or large language models specifically for Nepali could significantly boost performance in sentiment and NER tasks. Additionally, developing a labeled NER dataset for Nepali would allow for training supervised models that can classify entities into types (person, organization, location).Finally, expanding the pipeline to include topic modeling and fake news detection would make the system more robust and relevant in the context of modern news media.

References

Additional academic papers on NLP for low-resource languages, specific algorithms (e.g., TF-IDF, Naive Bayes, SVM, Random Forest), and relevant R text mining packages will be cited as the project progresses.

https://www.researchgate.net/publication/358720378_A_review_on_machine_learning_techniques_for_text_classification

Chang, W., Cheng, J., Allaire, J. J., Xie, Y., & McPherson, J. (2021). Shiny: Web Application Framework for R. R package version 1.7.1. https://cran.r-project.org/web/packages/shiny/index.html

Kuhn, M. (2008). Building Predictive Models in R Using the caret Package. *Journal of Statistical Software*, 28(5), 1-26. https://www.researchgate.net/publication/26568624_Building_Predictive_Models_in_R_Using_the_caret_Package

Mihalcea, R., & Tarau, P. (2004). TextRank: Bringing Order into Texts. *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*. https://www.researchgate.net/publication/200042361_TextRank_Bringing_Order_into_Text

News Category Classification Dataset Link: https://huggingface.co/datasets/Suyogyart/np20ng

Nepali Lexicon Dataset Link:

https://www.kaggle.com/datasets/aayamoza/nepali-sentiment-analysis

Silge, J., & Robinson, D. (2017). *Text Mining with R: A Tidy Approach*. O'Reilly Media. https://www.oreilly.com/library/view/text-mining-with/9781491981641/

Wijffels, J. (2021). udpipe: Tokenization, Parts of Speech Tagging, Lemmatization, and Dependency Parsing. R package version 0.8.9. https://rdrr.io/cran/udpipe/

OpenAI. (2023). ChatGPT (Mar 23 version) [Large language model]. https://chat.openai.com/

Google. (2024). Gemini: A Family of Highly Capable Multimodal Models. Google AI. https://deepmind.google/technologies/gemini/