AMERICAN INTERNATIONAL UNIVERSITY-BANGLADESH

Choose an item.

Project

Assignment Title:	Data Science Final Project		
Assignment No:	Click here to enter text.	Date of Submission:	25 May 2025
Course Title:	Introduction to Data Science		
Course Code:	Click here to enter text.	Section:	E
Semester:	Spring 2024-25	Course Teacher:	Abdus Salam

Declaration and Statement of Authorship:

1. I/we hold a copy of this Assignment/Case-Study, which can be produced if the original is lost/damaged.
2. This Assignment/Case-Study is my/our original work and no part of it has been copied from any other student's work or from any other source except where due acknowledgement is made.
3. No part of this Assignment/Case-Study has been written for me/us by any other person except where such collaboration has been authorized by the concerned teacher and is clearly acknowledged in the assignment.
4. I/we have not previously submitted or currently submitting this work for any other course/unit.
5. This work may be reproduced, communicated, compared and archived for the purpose of detecting plagiarism.
6. I/we give permission for a copy of my/our marked work to be retained by the faculty for review and comparison, including review by external examiners.
7. I/we understand that Plagiarism is the presentation of the work, idea or creation of another person as though it is your own. It is a formofcheatingandisaveryseriousacademicoffencethatmayleadtoexpulsionfromtheUniversity. Plagiarized material can be drawn from, and presented in, written, graphic and visual form, including electronic data, and oral presentations. Plagiarism occurs when the origin of their arterial used is not appropriately cited.
8. I/we also understand that enabling plagiarism is the act of assisting or allowing another person to plagiarize or to copy my/our work.

* *Student(s) must complete all details except the faculty use part.*
** Please submit all assignments to your course teacher or the office of the concerned teacher.

Group Name/No.:	9

No	Name	ID	Program	Signature
1	SABIHA IQBAL ETU	22-47227-1	BSc [CSE]	
2	MUKSHIT SAFI OWASI	22-47251-1	BSc [CSE]	
3	MAHMUDUS SAMI MAAHI	22-46446-1	BSc [CSE]	
4	MD. JAHIRUL ISLAM NAHID	22-47212-1	BSc [CSE]	

Faculty use only		
FACULTY COMMENTS	Marks Obtained	
	Total Marks	

Introduction:

In this project, we focused on extracting and processing news data from *The Daily Star*, one of Bangladesh's leading English-language news portals. We selected five diverse categories—**Business, Sports, Entertainment, Youth**, and **Life & Living**—and scraped **100 news articles from each,** resulting in a dataset of **500 articles**. For each article, we collected the **category, news link, title, description,** and **publication time.** The primary goal was to apply a comprehensive text preprocessing pipeline to the **description** field to prepare the data for further natural language processing tasks. The steps included **emoji handling** (replacing emojis with text), **text cleaning** (converting to lowercase, removing punctuation, numbers, and extra spaces), **tokenization**, **stopword removal, stemming and lemmatization,** and **basic spell correction.** The cleaned and processed text was stored in a new column called processed_description. This workflow demonstrates how web scraping and text preprocessing can be combined to build a structured, clean dataset suitable for content analysis.

Extract news from the News portal:

Required Library:

```
install.packages("rvest")
library(rvest)
```

The command is install.packages("rvest") installs the rvest package in R, which is used for web scraping, and library(rvest) loads the package so you can use its functions to extract data from web pages.

Select Categories and their URL:

```
categories_list <- list(
  Business = "https://www.thedailystar.net/business",
  Sports = "https://www.thedailystar.net/sports",
  Entertainment = "https://www.thedailystar.net/entertainment",
  LifeLiving = "https://www.thedailystar.net/life-living",
  Youth = "https://www.thedailystar.net/youth"
)
```

This R code defines a named list called categories_list that maps each news category to its corresponding URL on *The Daily Star* website. Each element of the list has a category name (e.g., *Business*, *Sports*, etc.) as the key, and the URL of that category's news page as the value. This list will be useful for looping through each category to scrape news articles from their respective pages.

Function to Scrape News Articles from a Single Category

```
scrape_category <- function(base_url, category_name) {
  all_titles <- c()
  all_links <- c()
  page_number <- 0

  while (length(all_links) < 100) {
    page_url <- paste0(base_url, "?page=", page_number)
    webpage <- tryCatch(read_html(page_url), error = function(e) NULL)
    if (is.null(webpage)) break

    title_nodes <- html_nodes(webpage, ".card-content a")
    titles <- html_text(title_nodes)
    links <- html_attr(title_nodes, "href")
    full_links <- paste0("https://www.thedailystar.net", links)

    new_titles <- titles[!full_links %in% all_links]
    new_links <- full_links[!full_links %in% all_links]

    all_titles <- c(all_titles, new_titles)
    all_links <- c(all_links, new_links)

    page_number <- page_number + 1
    Sys.sleep(1)
  }
```

In this part of the project, we created a function named scrape_category to automate the process of scraping news article titles and their corresponding links from a specific category page on *The Daily Star* website. The function accepts two inputs: the category URL and the category name. It then iteratively loads each paginated section of the category page, extracting the article titles and links using CSS selectors. To ensure only unique articles are stored, it filters out duplicates during each loop iteration. This process continues until 100 unique articles are collected for the given category. Additionally, a 1-second delay is added between requests to follow responsible web scraping practices. The output of this function is a clean and structured list of news titles and links for each selected category, laying the foundation for collecting full news content and metadata in the next steps.

Extracting Descriptions and Publication Times for News Articles

```
  all_titles <- all_titles[1:100]
  all_links <- all_links[1:100]


  get_description = function(link) {
    tryCatch({
      news_page = read_html(link)
      para = html_nodes(news_page, ".clearfix p")
      para_text = html_text(para)
      if (length(para_text) == 0) return(NA)


      para_text <- para_text[nzchar(para_text)]
      description <- paste(head(para_text, 3), collapse = " ")
      return(description)
    }, error = function(e) NA)
  }


  get_time <- function(link) {
    tryCatch({
      page <- read_html(link)
      times <- page %>% html_nodes(".color-iron") %>% html_text(trim = TRUE)
      full_time <- times[grepl("Last update on:|Published on:", times)][1]
      if (is.na(full_time) || full_time == "") {
        full_time <- times[times != ""][1]
      }
      return(full_time)
    }, error = function(e) NA)
  }


  descriptions = sapply(all_links, get_description)
  times = sapply(all_links, get_time)

  data.frame(
    category = category_name,
    news_link = all_links,
    title = all_titles,
    description = descriptions,
    time = times,
    stringsAsFactors = FALSE
  )
}
```

In this part, we trimmed the collected news titles and links to exactly 100, then used two functions to extract the description (by combining up to three meaningful paragraphs) and the publication time from each article page. These were applied to all links using sapply(), and the results—category, news link, title, description, and time—were stored in a structured data frame for further use. The get_time function extracts the publication or update time from the page using the .color-iron selector and filters relevant time-related text using pattern matching (e.g., "Published on:" or "Last update on:").

Scraping All Categories and Saving News Data

```
final_data <- do.call(rbind, lapply(names(categories_list), function(cat) {
  cat("Scraping", cat, "...\n")
  scrape_category(categories_list[[cat]], cat)
}))


write.csv(final_data, "ids_final_project_group_9_news_raw.csv", row.names = FALSE)
cat("Datas saved to 'ids_final_project_group_9_news_raw.csv'\n")
```

In this section, we automated the scraping process for all selected categories by using lapply() to loop through each category in the categories_list. For each category, the scrape_category function is called, and the results are combined into a single data frame using do.call(rbind, ...). This aggregated dataset, containing 500 news articles across five categories, is then saved to a CSV file named ids_final_project_group_9_news_raw.csv using write.csv(). A confirmation message is printed after successful completion.

The output of extracting news from the news portal

	category	news_link	title	description	time	
2	Business	https://www	Interest pay	Interest pay	Fri May 16, 2025 12:00 AM	Last update on: Fri May 16, 2025 01:01 AM
3	Business	https://www	‘Govt to act	The governr	Fri May 16, 2025 12:00 AM	Last update on: Fri May 16, 2025 12:52 AM
4	Business	https://www	Economy or	Bangladesh'	Fri May 16, 2025 12:00 AM	Last update on: Fri May 16, 2025 01:35 PM
5	Business	https://www	Analysis / B	Bangladesh'	Fri May 16, 2025 12:00 AM	Last update on: Fri May 16, 2025 02:42 PM
6	Business	https://www	NBR officials	Protesting o	Fri May 16, 2025 12:00 AM	Last update on: Fri May 16, 2025 09:50 AM
7	Business	https://www	Who appoir	A new powe	Fri May 16, 2025 12:30 AM	Last update on: Fri May 16, 2025 02:53 PM
8	Business	https://www	Dollar rema	The US dolla	Fri May 16, 2025 12:00 AM	Last update on: Fri May 16, 2025 09:51 AM
9	Business	https://www	Currency ur	The Banglad	Fri May 16, 2025 12:00 AM	Last update on: Fri May 16, 2025 12:45 AM
10	Business	https://www	Trump: Indi	US Presiden	Fri May 16, 2025 12:00 AM	Last update on: Fri May 16, 2025 12:43 AM
11	Business	https://www	Crisis ignore	If you place	Fri May 16, 2025 12:00 AM	Last update on: Fri May 16, 2025 12:36 AM
12	Business	https://www	Bata’s profit	Bata Shoe C	Fri May 16, 2025 12:00 AM	Last update on: Fri May 16, 2025 09:51 AM
13	Business	https://www	Pubali Bank'	Pubali Bank	Fri May 16, 2025 12:00 AM	Last update on: Fri May 16, 2025 09:51 AM
14	Business	https://www	WB approve	The World E	Fri May 16, 2025 12:00 AM	Last update on: Fri May 16, 2025 09:53 AM
15	Business	https://www	Gold falls to	Gold prices	Fri May 16, 2025 12:00 AM	Last update on: Fri May 16, 2025 09:53 AM
16	Business	https://www	Focus on qu	Bangladesh'	Thu May 15, 2025 09:35 PM	Last update on: Thu May 15, 2025 09:34 PM
17	Business	https://www	NBR moves	The Nationa	Thu May 15, 2025 09:16 PM	Last update on: Thu May 15, 2025 10:28 PM
18	Business	https://www	Padma Oil n	Padma Oil C	Thu May 15, 2025 08:46 PM	Last update on: Thu May 15, 2025 08:50 PM
19	Business	https://www	Tokyo, Osak	Tokyo and (	Thu May 15, 2025 05:54 PM	Last update on: Thu May 15, 2025 05:57 PM
20	Business	https://www	Trump: Indi	US Presiden	Thu May 15, 2025 02:14 PM	Last update on: Thu May 15, 2025 02:16 PM
21	Business	https://www	UK econom	Britain's eco	Thu May 15, 2025 01:07 PM	Last update on: Thu May 15, 2025 01:08 PM
22	Business	https://www	Analysis / N	Nissan's nev	Thu May 15, 2025 11:16 AM	Last update on: Thu May 15, 2025 11:26 AM
23	Business	https://www	Stocks drop	Equities stu	Thu May 15, 2025 11:12 AM	Last update on: Thu May 15, 2025 11:14 AM
24	Business	https://www	China, US sl	The United S	Thu May 15, 2025 12:00 AM	Last update on: Thu May 15, 2025 12:51 AM
25	Business	https://www	US compani	President D	Thu May 15, 2025 12:00 AM	Last update on: Thu May 15, 2025 12:51 AM
26	Business	https://www	Boeing April	Boeing com	Thu May 15, 2025 12:00 AM	Last update on: Thu May 15, 2025 12:45 AM
27	Business	https://www	ICAB delega	A two-mem	Thu May 15, 2025 09:23 PM	Last update on: Thu May 15, 2025 09:21 PM
28	Business	https://www	IPDC Financ	IPDC Financ	Wed May 14, 2025 08:47 PM	Last update on: Wed May 14, 2025 08:48 PM

thedailystarRawData

This is the output we get when extracting the information from the online news portal The Daily Star. We extract 100 news articles and save them (category, news link, title, time) in csv file.

Applying the text processing steps :

Why Preprocessing Is Important -

Raw text data is often messy and inconsistent. It contains contractions ("don't"), emojis, punctuation, HTML tags, numbers, spelling mistakes, and irrelevant words (stopwords) that can negatively affect the accuracy and performance of models or analyses. Preprocessing helps standardize the text, reduce noise, and highlight meaningful content.

The core objective of this work is to perform detailed text preprocessing on the news description field to prepare it for downstream tasks like content classification, sentiment analysis, or summarization. The following text processing steps were applied:

1. **Emoji Handling** – Replaced emojis with their textual descriptions.
2. **Text Cleaning** – Converted text to lowercase, removed punctuation, numbers, and extra whitespace.
3. **Tokenization** – Split text into individual tokens or words.
4. **Stopword Removal** – Eliminated common English stopwords that do not contribute much to meaning.
5. **Stemming and Lemmatization** – Reduced words to their root or base forms to normalize the vocabulary.
6. **Spell Checking** – Applied basic spell correction to improve textual quality, enforcing lowercase consistency.

The processed content was stored in a new column named `processed_description` and the complete dataset was exported to a CSV file for further analysis or modeling. This pipeline showcases the integration of web scraping with natural language preprocessing, establishing a foundation for robust content mining and NLP-based news analytics.

Required Libraries

```
install.packages("tm")
install.packages("textclean")
install.packages("textstem")
install.packages("tokenizers")
install.packages("hunspell")



library(tm)
library(textclean)
library(textstem)
library(tokenizers)
library(hunspell)
library(dplyr)
```

These packages are used for text processing in R: **tm** handles text cleaning and management; textclean fixes messy or incorrect text; textstem reduces words to their base forms; tokenizers break text into words or sentences; and hunspell checks and corrects spelling errors. Together, they prepare text for analysis or modeling.

Text Preprocessing

```
expand_contractions <- function(text) {
  replace_contraction(text)
}


handle_emojis <- function(text) {
  replace_emoji(text)
}


clean_text <- function(text) {
  text %>%
    tolower() %>%
    gsub("<.*?>", " ", .) %>%
    gsub("[^a-z\\s]", " ", .) %>%
    gsub("\\s+", " ", .) %>%
    trimws()
}


tokenize_text <- function(text) {
  unlist(tokenize_words(text))
}
```

We defined four specific text preprocessing functions:

1. **expand_contractions:** Converts shortened forms like "don't" to "do not" using `replace_contraction()`.
2. **handle_emojis:** Replaces emojis in the text with their descriptive words using `replace_emoji()`.
3. **clean_text:** Converts text to lowercase, removes HTML tags, non-letter characters, extra spaces, and trims whitespace.
4. **tokenize_text:** Splits the cleaned text into individual word tokens using `tokenize_words()`.

Text Preprocessing

```
remove_stopwords <- function(tokens) {
  tokens[!tokens %in% stopwords("en")]
}


stem_and_lemmatize <- function(tokens) {
  lemmatize_words(stem_strings(tokens))
}


spell_check <- function(text) {
  words <- unlist(strsplit(text, "\\s+"))
  corrected_words <- sapply(words, function(w) {
    if (!hunspell_check(w)) {
      sugg <- hunspell_suggest(w)[[1]]
      if (length(sugg) > 0) return(tolower(sugg[1]))
    }
    return(tolower(w))
  })
  paste(corrected_words, collapse = " ")
}
```

Here's what each function does specifically:
5. **remove_stopwords:** Filters out English stopwords from the token vector using `stopwords("en")`.

6. **stem_and_lemmatize:** First stems tokens with `stem_strings()`, then lemmatizes them using `lemmatize_words()` to get their base forms.
7. **spell_check:** Splits the text into words, checks each word's spelling with `hunspell_check()`, replaces misspelled words with the first suggested correction from `hunspell_suggest()`, and returns the corrected lowercase text.

```r
process_text <- function(text_vector) {
  sapply(text_vector, function(text) {
    text %>%
      expand_contractions() %>%
      handle_emojis() %>%
      clean_text() %>%
      {
        tokens <- tokenize_text(.)
        tokens <- remove_stopwords(tokens)
        tokens <- stem_and_lemmatize(tokens)
        paste(tokens, collapse = " ")
      } %>%
      spell_check() %>%
      tolower()
  }, USE.NAMES = FALSE)
}


news_data$processed_description <- process_text(news_data$description)


write.csv(news_data, "E:/DataScience/ids_final_project_group_9_news_clean.csv", row.names = FALSE)

cat("Text preprocessing is done. Saved as 'ids_final_project_group_9_news_clean.csv'\n")
```

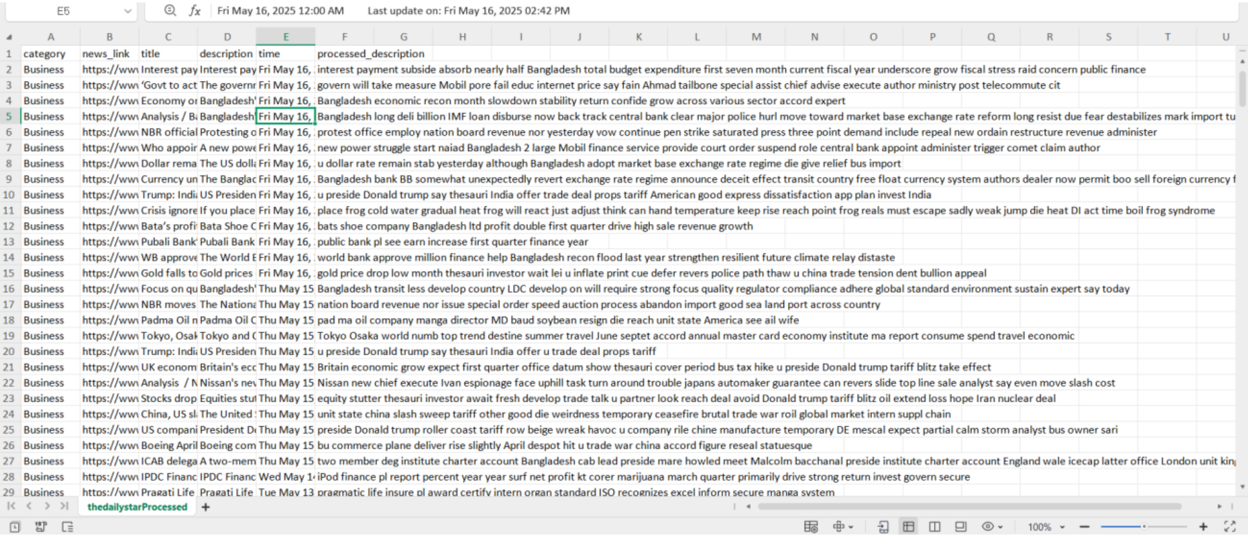
This code defines a main function `process_text` that applies a full text preprocessing pipeline to each element of a text vector. For each text entry, it:

1. Expands contractions (e.g., "don't" → "do not").
2. Replaces emojis with descriptive words.
3. Cleans the text by lowercasing, removing HTML tags, punctuation, numbers, and extra spaces.
4. Tokenizes the cleaned text into words.
5. Removes stopwords from the tokens.
6. Applies stemming and lemmatization to reduce words to their base forms.
7. Joins the processed tokens back into a single string.
8. Performs spell checking and correction.
9. Converts the final output to lowercase for consistency.

Then, it applies this function specifically to the `description` column of the `news_data` dataframe, storing the cleaned text in a new column `processed_description`.

Finally, it saves the updated dataframe with the processed text to a new CSV file at the specified location and prints a confirmation message indicating that preprocessing is complete.

Final Output after doing Preprocessing on the description column



E5 | Fri May 16, 2025 12:00 AM | Last update on: Fri May 16, 2025 02:42 PM

	category	news_link	title	description	time	processed_description
2	Business	https://www	Interest pay	Interest pay	Fri May 16,	interest payment subside absorb nearly half Bangladesh total budget expenditure first seven month current fiscal year underscore grow fiscal stress raid concern public finance
3	Business	https://www	'Govt to act	The governr	Fri May 16,	govern will take measure Mobil pore fail educ internet price say fain Ahmad tailbone special assist chief advise execute author ministry post telecommute cit
4	Business	https://www	Economy or	Bangladesh'	Fri May 16,	Bangladesh economic recon month slowdown stability return confide grow across various sector accord expert
5	Business	https://www	Analysis / Ba	Bangladesh'	Fri May 16,	Bangladesh long deli billion IMF loan disburse now back track central bank clear major police hurl move toward market base exchange rate reform long resist due fear destabilizes mark import tu
6	Business	https://www	NBR official	Protesting o	Fri May 16,	protest office employ nation board revenue nor yesterday vow continue pen strike saturated press three point demand include repeal new ordain restructure revenue administer
7	Business	https://www	Who appoir	A new powe	Fri May 16,	new power struggle start naiad Bangladesh 2 large Mobil finance service provide court order suspend role central bank appoint administer trigger comet claim author
8	Business	https://www	Dollar rema	The US dolla	Fri May 16,	u dollar rate remain stab yesterday although Bangladesh adopt market base exchange rate regime die give relief bus import
9	Business	https://www	Currency un	The Banglac	Fri May 16,	Bangladesh bank BB somewhat unexpectedly revert exchange rate regime announce deceit effect transit country free float currency system authors dealer now permit boo sell foreign currency f
10	Business	https://www	Trump: India	US Presiden	Fri May 16,	u preside Donald trump say thesauri India offer trade deal props tariff American good express dissatisfaction app plan invest India
11	Business	https://www	Crisis ignore	If you place	Fri May 16,	place frog cold water gradual heat frog will react just adjust think can hand temperature keep rise reach point frog reals must escape sadly weak jump die heat DI act time boil frog syndrome
12	Business	https://www	Bata's profi	Bata Shoe C	Fri May 16,	bats shoe company Bangladesh ltd profit double first quarter drive high sale revenue growth
13	Business	https://www	Pubali Bank'	Pubali Bank	Fri May 16,	public bank pl see earn increase first quarter finance year
14	Business	https://www	WB approve	The World E	Fri May 16,	world bank approve million finance help Bangladesh recon flood last year strengthen resilient future climate relay distaste
15	Business	https://www	Gold falls to	Gold prices	Fri May 16,	gold price drop low month thesauri investor wait lei u inflate print cue defer revers police path thaw u china trade tension dent bullion appeal
16	Business	https://www	Focus on qu	Bangladesh'	Thu May 15	Bangladesh transit less develop country LDC develop on will require strong focus quality regulator compliance adhere global standard environment sustain expert say today
17	Business	https://www	NBR moves	The Nationa	Thu May 15	nation board revenue nor issue special order speed auction process abandon import good sea land port across country
18	Business	https://www	Padma Oil n	Padma Oil C	Thu May 15	pad ma oil company manga director MD baud soybean resign die reach unit state America see ail wife
19	Business	https://www	Tokyo, Osak	Tokyo and (	Thu May 15	Tokyo Osaka world numb top trend destine summer travel June septet accord annual master card economy institute ma report consume spend travel economic
20	Business	https://www	Trump: India	US Presiden	Thu May 15	u preside Donald trump say thesauri India offer u trade deal props tariff
21	Business	https://www	UK econom	Britain's ecc	Thu May 15	Britain economic grow expect first quarter office datum show thesauri cover period bus tax hike u preside Donald trump tariff blitz take effect
22	Business	https://www	Analysis / N	Nissan's nev	Thu May 15	Nissan new chief execute Ivan espionage face uphill task turn around trouble japans automaker guarantee can revers slide top line sale analyst say even move slash cost
23	Business	https://www	Stocks drop	Equities stut	Thu May 15	equity stutter thesauri investor await fresh develop trade talk u partner look reach deal avoid Donald trump tariff blitz oil extend loss hope Iran nuclear deal
24	Business	https://www	China, US sl	The United	Thu May 15	unit state china slash sweep tariff other good die weirdness temporary ceasefire brutal trade war roil global market intern suppl chain
25	Business	https://www	US compani	President D	Thu May 15	preside Donald trump roller coast tariff row beige wreak havoc u company rile chine manufacture temporary DE mescal expect partial calm storm analyst bus owner sari
26	Business	https://www	Boeing April	Boeing com	Thu May 15	bu commerce plane deliver rise slightly April despot hit u trade war china accord figure reseal statuesque
27	Business	https://www	ICAB delega	A two-mem	Thu May 15	two member deg institute charter account Bangladesh cab lead preside mare howled meet Malcolm bacchanal preside institute charter account England wale icecap latter office London unit kin
28	Business	https://www	IPDC Financ	IPDC Financ	Wed May 14	iPod finance pl report percent year year surf net profit kt corer marijuana march quarter primarily drive strong return invest govern secure
29	Business	https://www	Pragati Life	Pragati Life	Tue May 13	pragmatic life insure pl award certify intern organ standard ISO recognizes excel inform secure manga system

thedailystarProcessed

Finally we get the clean text after doing the preprocessing and all other steps the description part is processed and saved in the csv file named ids_final_project_group_9_news_clean.

Applying Top Modelling :

Required Libraries

```
library(readr)
library(tm)
library(topicmodels)
library(dplyr)
library(stringr)
```

These libraries help load data, clean and manage text (tm), build topic models (topicmodels), and handle data frames (dplyr, stringr).

Loading the Preprocessed News Data

```
data <- read_csv("E:/DataScience/ids_final_project_group_9_news_clean.csv")
```

In this step, we load the cleaned dataset into R using the read_csv() function from the readr package. The file named ids_final_project_group_9_news_clean.csv contains the news articles that were collected and preprocessed in the earlier part of the project. This dataset includes multiple columns, but the one that is processed_description, which holds the cleaned version of the news content. This step is essential because it brings our prepared data into the current R environment, making it ready for further analysis. Without loading this file, we would not be able to perform topic modeling on the news descriptions. This step forms the foundation for all the tasks that follow in the topic modeling part of the project.

Creating a Text Corpus

```
corpus <- VCorpus(VectorSource(data$processed_description))
```

In this part, we create a text corpus using the VCorpus() function from the tm package. A corpus is a structured collection of text documents, and in this case, it is built from the processed_description column of our dataset, which contains the cleaned news article descriptions. We use VectorSource() to tell R that the source of the text is a vector (a column) from our data frame. Creating a corpus is an important step in this project because it prepares the text for further processing and analysis. Once we create the corpus, we can apply various text cleaning functions and later convert it into a document-term matrix, which is needed for topic modeling.

Cleaning the Text Again

```
corpus <- tm_map(corpus, content_transformer(tolower))
corpus <- tm_map(corpus, removePunctuation)
corpus <- tm_map(corpus, removeNumbers)
corpus <- tm_map(corpus, removeWords, custom_stopwords)
corpus <- tm_map(corpus, stripWhitespace)
```

Here, we perform additional text cleaning on the corpus to ensure it is fully ready for topic modeling. Although the text was already cleaned earlier, it is necessary to apply these transformations again after creating the corpus, because tm_map() functions work specifically with corpus objects. First, we convert all text to lowercase using content_transformer(tolower) so that the same words in different cases are treated equally. Then, we remove all punctuation and numbers. We also remove a set of custom stopwords like common words such as “will,” “can,” “say,” and others that do not carry much meaning and can negatively affect model performance. Lastly, we use stripWhitespace to remove any extra spaces left behind. These preprocessing steps clean and normalize the text, making it consistent and reducing noise, which helps the topic modeling algorithm identify more meaningful patterns in the data.

Document-Term Matrix (DTM)

```
dtm <- DocumentTermMatrix(corpus)


dtm <- removeSparseTerms(dtm, 0.99)


row_totals <- apply(dtm, 1, sum)
dtm <- dtm[row_totals > 0, ]
```

In this part, we convert the cleaned corpus into a Document-Term Matrix (DTM) using the DocumentTermMatrix() function. A DTM is a structured table where each row represents a document, and each column represents a unique word. The values in the matrix show how often each word appears in each document. This format is essential for topic modeling, as it allows algorithms like LDA to analyze word usage patterns across documents. After creating the DTM, we remove sparse terms using removeSparseTerms(dtm, 0.99). This means we keep only the words that appear in at least 1% of the documents, which helps remove rare or irrelevant terms. Then we check the row totals using apply(dtm, 1, sum) and remove any rows where the total is zero, which indicates documents that have no remaining words after filtering. This step ensures our DTM is clean, compact, and ready for topic modeling.

The LDA Topic Model

```
set.seed(123)
lda_model <- LDA(dtm, k = 5, control = list(seed = 123))
```

This part of the code sets a starting point for randomness using set.seed(123) so that the results are the same every time we run it. Then, we use LDA model to the document-term matrix (dtm) using the LDA() function. The parameter k = 5 specifies that the model should identify 5 distinct topics within the corpus. The control = list(seed = 123) argument further ensures that the internal random processes within the LDA algorithm are consistent across runs. Essentially, this part trains the topic model to uncover the hidden thematic structure of the text data by grouping words that frequently co-occur into five topics.

Extracting Top Words for Each Topic

```
> top_terms <- terms(lda_model, 10)
> print("Top terms in each topic:")
[1] "Top terms in each topic:"
> print(top_terms)
      Topic 1      Topic 2      Topic 3 Topic 4      Topic 5
 [1,] "year"       "student"    "like"  "win"        "film"
 [2,] "percent"    "universe"   "just"  "match"      "music"
 [3,] "bank"       "bangladesh" "time"  "league"     "year"
 [4,] "bangladesh" "research"   "feel"  "team"       "include"
 [5,] "market"     "work"       "good"  "final"      "song"
 [6,] "rate"       "intern"     "even"  "bangladesh" "culture"
 [7,] "trade"      "stud"       "die"   "good"       "new"
 [8,] "corer"      "office"     "ever"  "first"      "perform"
 [9,] "price"      "expire"     "home"  "plain"      "show"
[10,] "high"       "manga"      "main"  "year"       "feature"
```

This section extracts and displays the most representative words for each topic generated by the LDA model. The function terms(lda_model, 10) is used to retrieve the top 10 terms (words) that have the highest probability of belonging to each topic in the model. These terms are considered the most relevant keywords that characterize the content of each topic. By examining these top terms, we can gain insight into the themes or subjects that the model has discovered within the text data. After extracting these terms, the code prints a message "Top terms in each topic:" to inform the actual lists of the top words for each of the five topics. This output is crucial for interpreting and labeling the topics in a meaningful way, as it provides a summary of the key vocabulary associated with each topic.

Interpret Topics

```
interpret_topic <- function(top_terms_vec) {
  keywords <- tolower(top_terms_vec)

  categories <- list(
    "Economic & Business News" = c(
      "market", "stock", "business", "economy", "growth", "price", "trade", "bank", "dollar", "investment",
      "percent", "rate", "corer", "high", "tax", "inflation", "finance", "budget", "profit", "loss"
    ),

    "Sports News" = c(
      "match", "player", "score", "team", "game", "win", "coach", "tournament", "goal", "league",
      "final", "season", "championship", "cricket", "football", "cup", "run", "bat", "ball"
    ),

    "Entertainment News" = c(
      "movie", "actor", "film", "music", "celebrity", "show", "award", "drama", "director", "release",
      "song", "feature", "performance", "cinema", "scene", "album", "entertainment", "star", "role"
    ),

    "Lifestyle & Living" = c(
      "health", "life", "living", "environment", "travel", "food", "family", "fashion", "fitness", "climate",
      "feel", "home", "habit", "culture", "experience", "emotion", "beauty", "diet", "wellness"
    ),

    "Youth & Education" = c(
      "youth", "education", "student", "career", "social", "event", "community", "school", "university", "teacher",
      "research", "study", "intern", "exam", "campus", "degree", "academic", "learn", "graduate"
    )
  )
```

```
  match_counts <- sapply(categories, function(keywords_list) {
    sum(keywords %in% keywords_list)
  })

  best_category <- names(which.max(match_counts))
  return(best_category)
}
```

The interpret_topic function is designed to assign a meaningful label to each topic identified by the LDA model based on its top words. It takes a list of important words for a topic and converts them all to lowercase to ensure consistent matching. The function then compares these words against predefined keyword lists for five different news categories: Economic & Business News, Sports News, Entertainment News, Lifestyle & Living, and Youth & Education. For each category, it counts how many of the topic's words appear in that category's keyword list. Finally, it selects the category with the highest number of matching words as the best description for that topic and returns its name. This helps interpret the topics in human-readable terms by linking the model's output to familiar subject areas.

```
+ }
> for (i in 1:5) {
+   terms_vec <- top_terms[, i]
+   interpretation <- interpret_topic(terms_vec)
+
+   cat(paste0("\nTopic ", i, " contains: [", paste(terms_vec, collapse = ", "), "]\n"))
+   cat(paste0("Interpreted as: *", interpretation, "*\n"))
+   cat(strrep("-", 50), "\n")
+ }

Topic 1 contains: [year, percent, bank, bangladesh, market, rate, trade, corer, price, high]
Interpreted as: *Economic & Business News*
--------------------------------------------------

Topic 2 contains: [student, universe, bangladesh, research, work, intern, stud, office, expire, manga]
Interpreted as: *Youth & Education*
--------------------------------------------------

Topic 3 contains: [like, just, time, feel, good, even, die, ever, home, main]
Interpreted as: *Lifestyle & Living*
--------------------------------------------------

Topic 4 contains: [win, match, league, team, final, bangladesh, good, first, plain, year]
Interpreted as: *Sports News*
--------------------------------------------------

Topic 5 contains: [film, music, year, include, song, culture, new, perform, show, feature]
Interpreted as: *Entertainment News*
--------------------------------------------------
> |
```

Here the loop iterates over the five topics generated by the LDA model, processing each topic sequentially. For every topic index i from 1 to 5, it first extracts the corresponding column of top terms from the top_terms matrix, which represents the ten most significant words associated with that topic. These terms are then passed to the interpret_topic function, which analyzes the words and determines the most appropriate category label for the topic based on predefined keyword lists. After obtaining the interpreted category, the loop prints a formatted message that includes the topic number, the list of top words for that topic, and the category label as an interpretation of the topic's theme. Overall, this loop automates the process of summarizing the key terms for each topic and providing a human-readable label that helps in understanding and communicating the results of the topic modeling.