**Hong Kong Institute of Vocational Education Discipline of Information Technology**

**ITP4514 - Artificial Intelligence and Machine Learning**

**Group Assignment**

**Topic: News Classification and Summarization**

****

|  |  |  |
| --- | --- | --- |
| Name and ID | Contribution | Signature |
| Yeung Ming Hoi  220153482 |  |  |
| Cham Ka Lun 220153377 |  |  |
| DU Huifeng 220301483 |  |  |
|  |  |  |

# Contents

[Contents 1](#_Toc152708048)

[Abstract 2](#_Toc152708049)

[Background 3](#_Toc152708050)

[1. Data Collection 4](#_Toc152708051)

[1.1 Source Selection 4](#_Toc152708052)

[1.2 Crawling for Links 5](#_Toc152708053)

[1.3 Data Extraction 5](#_Toc152708054)

[1.4 Data Composition and Cleaning 5](#_Toc152708055)

[1.5 Data Format 5](#_Toc152708056)

[1.5.1 News Links Format 5](#_Toc152708057)

[1.5.2 News Data Format 6](#_Toc152708058)

[1.6 News Crawler Time 6](#_Toc152708059)

[1.6.1 Aljazeera News 6](#_Toc152708060)

[1.6.2 CNN News 6](#_Toc152708061)

[1.6.3 The Standard 6](#_Toc152708062)

[1.6.4 NBC News 7](#_Toc152708063)

[1.7 Total News 7](#_Toc152708064)

[2. Classification by Naive Bayes 8](#_Toc152708065)

[2.1 Introduction to Naive Bayes Model 8](#_Toc152708066)

[2.2 Parameter Tuning in Naive Bayes 9](#_Toc152708067)

[2.3 Tuning Detail in Naive Bayes 10](#_Toc152708068)

[2.3.1 Vectorize 11](#_Toc152708069)

[2.3.2 Model Evaluation 12](#_Toc152708070)

[2.3.3 Feature Analysis 14](#_Toc152708071)

[2.3.4 Model Performance 16](#_Toc152708072)

[2.4 Performance Summary 18](#_Toc152708073)

[2.4.1 Before Tuning 18](#_Toc152708074)

[2.4.2 After Tuning 18](#_Toc152708075)

[2.4.3 Impact of Tuning 19](#_Toc152708076)

[2.5 TfidfVectorizer Parameters Tuning 19](#_Toc152708077)

[3. Classification by Gated Recurrent Units 22](#_Toc152708078)

[4. Summarization by Bidirectional Encoder Representations from Transformers 23](#_Toc152708079)

[5. Summarization by T5 24](#_Toc152708080)

[6. Conclusion 25](#_Toc152708081)

[7. References 26](#_Toc152708082)

# Abstract

This paper presents a data analysis project that uses Python and machine learning models to classify and summarize news articles from four different websites (Aljazeera, CNN, The Standard, NBC News). The project consists of the following steps: data collection, classification, and summarization. For data collection, we use web scraping to obtain news links, titles, contents, and categories, and store them as JSON files. For classification, we compare two different models (Naive Bayes and Gated Recurrent Unit) and their performance and parameter tuning process, and use class weights to address the class imbalance problem. For summarization, we compare two different models (BERT and T5) and their effects and features. We also list the work allocation, data sources, and remarks for each task of the project. Finally, we provide the installation instructions for the required libraries and Python version.

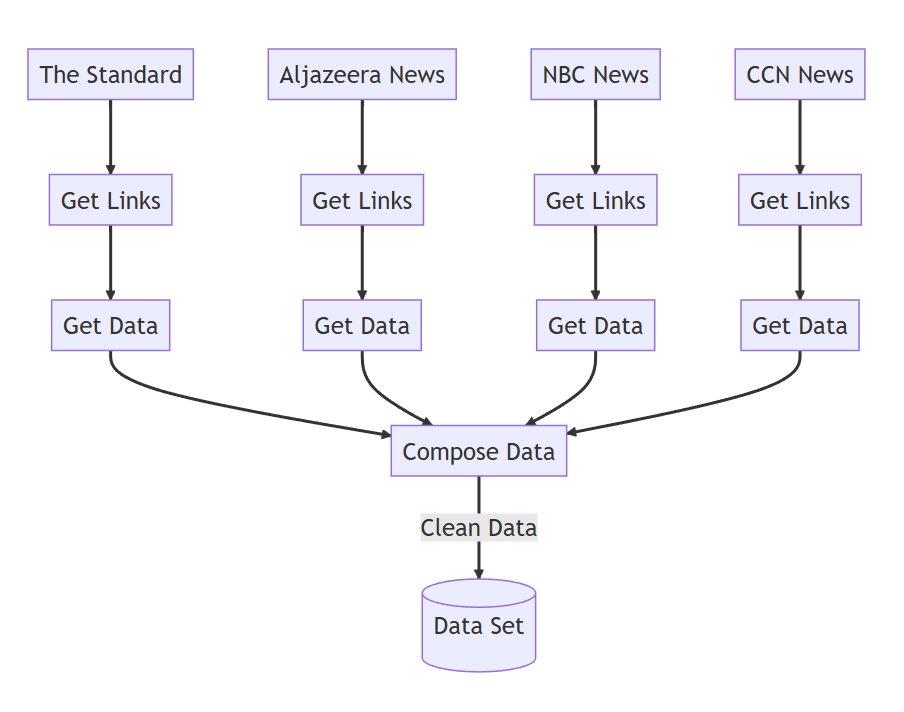
# Background

News data is an important source of information that reflects the current situation and trends of society. News data classification and summarization can help readers quickly understand the main content and category of news, save reading time and effort, and provide a basis and support for other data analysis and mining tasks, such as sentiment analysis, topic detection, event recognition, etc. However, news data classification and summarization also pose many challenges and difficulties, such as the diversity and complexity of news topics, the subjectivity and bias of news sources, the quality and reliability of news content, and the evaluation and comparison of different methods and models.

In this project, we aim to explore the possibilities and limitations of news data classification and summarization, using Python and machine learning models. We choose four different news websites as our data sources: Aljazeera, CNN, The Standard, and NBC News. These websites represent different regions, cultures, languages, and political orientations, and cover a wide range of news categories, such as world, business, sports, entertainment, etc. We use web scraping to obtain news links, titles, contents, and categories from these websites, and store them as JSON files. We compare two different models (Naive Bayes and Gated Recurrent Unit) for news classification, and their performance and parameter tuning process, and use class weights to address the class imbalance problem. We compare two different models (BERT and T5) for news summarization, and their effects and features.

# 1. Data Collection

In the presented project, the data collection process is meticulously orchestrated to aggregate a diverse array of news content from multiple reputable sources. This procedure is subdivided into several critical phases, each contributing uniquely to the integrity and comprehensiveness of the final dataset.



## 1.1 Source Selection

The initial stage involves the selection of four distinct news outlets, namely The Standard, Aljazeera News, NBC News, and CCN News. These sources were chosen for their diverse perspectives and broad coverage, ensuring a rich and varied dataset.

## 1.2 Crawling for Links

The subsequent phase employs specialized web crawlers for each news source. These crawlers are tasked with systematically navigating the respective news websites to extract relevant URLs. This step is crucial as it lays the foundation for the subsequent data extraction by identifying the specific web pages from which the news content will be harvested.

## 1.3 Data Extraction

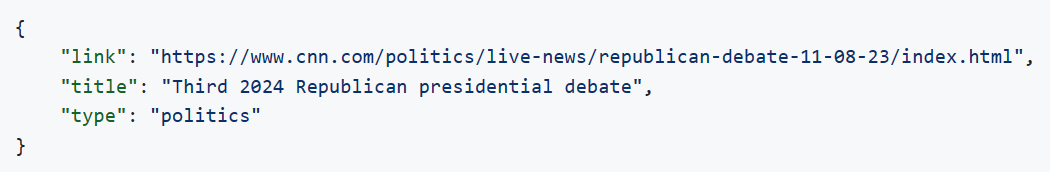
Once the relevant links are identified, the process of data extraction commences. In this phase, the crawlers retrieve the actual content from the gathered links. This includes the extraction of key elements such as news titles, article content, and metadata. The extraction process is carefully designed to ensure that the data is captured accurately and efficiently, maintaining the integrity of the original content.

## 1.4 Data Composition and Cleaning

The final stage in the data collection process involves the aggregation of the extracted data from all sources into a cohesive dataset. This phase also includes a rigorous data cleaning process. The cleaning procedure involves the removal of irrelevant or redundant information, correcting any errors, and standardizing the format of the dataset. This step is pivotal in ensuring that the dataset is reliable and suitable for subsequent analytical processes.

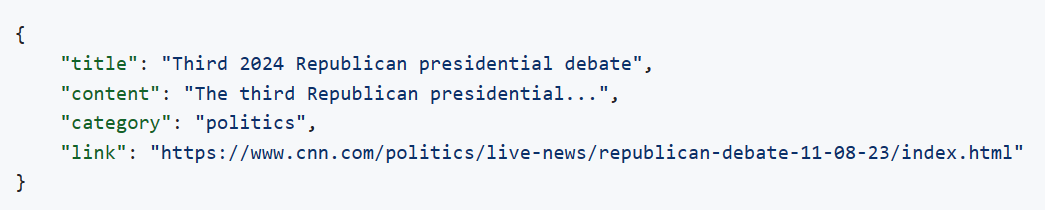
## 1.5 Data Format

### 1.5.1 News Links Format

As illustrated in the figure below, the News links format is shown.

### 1.5.2 News Data Format

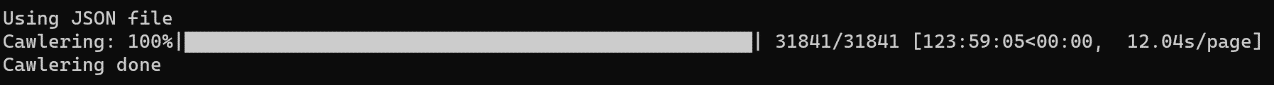
As depicted in the following figure, the News data format is shown.



## 1.6 News Crawler Time

### 1.6.1 Aljazeera News

As demonstrated in the figure below, the depicted output captures the duration and volume of news articles harvested from Aljazeera News utilizing a web crawler. A total of 31,841 articles were systematically extracted, with the operation spanning approximately 124 hours.



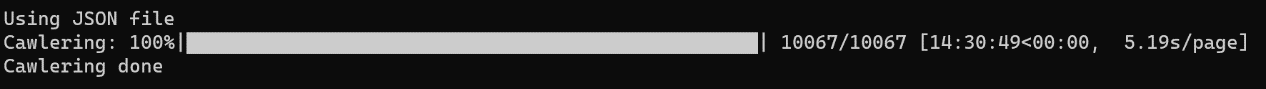
### 1.6.2 CNN News

As demonstrated in the figure below, the depicted output captures the duration and volume of news articles harvested from CNN News utilizing a web crawler. A total of 959 articles were systematically extracted, with the operation spanning approximately 1 hours.



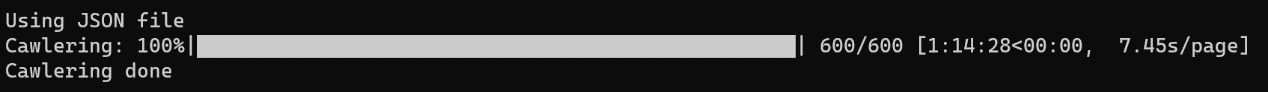
### 1.6.3 The Standard

As demonstrated in the figure below, the depicted output captures the duration and volume of news articles harvested from the Standard utilizing a web crawler. A total of 10067 articles were systematically extracted, with the operation spanning approximately 15 hours.



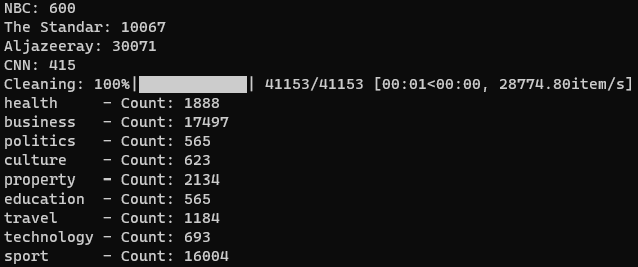
### 1.6.4 NBC News

As demonstrated in the figure below, the depicted output captures the duration and volume of news articles harvested from NBC News utilizing a web crawler. A total of 600 articles were systematically extracted, with the operation spanning approximately 1 hours.



## 1.7 Total News

As illustrated in the figure below, this represents the total number of news articles collected, which are further categorized into different domains based on the type of news. The dataset comprises a total of 41,153 news articles, with the following distribution across various categories: health (1,888), business (17,497), politics (565), culture (623), property (2,134), education (565), travel (1,184), technology (693), and sport (16,004).

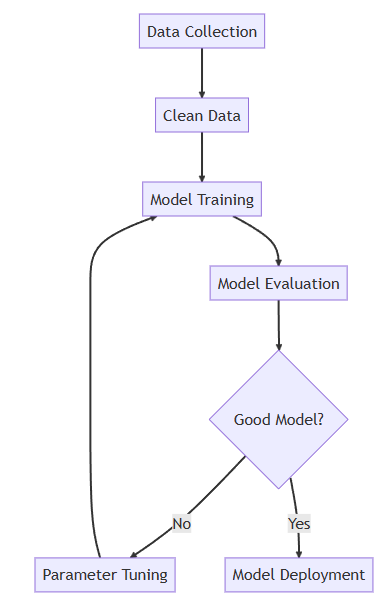


# 2. Classification by Naive Bayes

In this section, we will initially employ the Naive Bayes model to process the classification of the news data from the first part.

## 2.1 Introduction to Naive Bayes Model

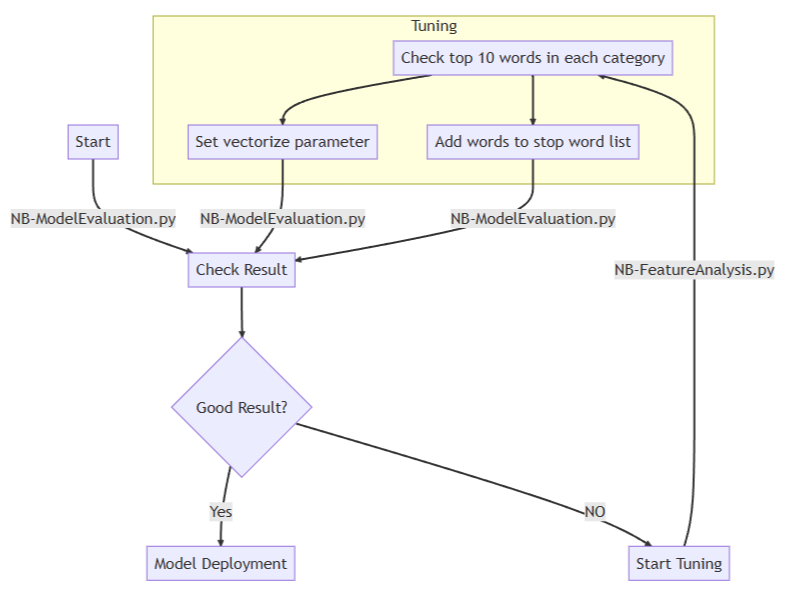
The figure below illustrates a structured methodology for deploying a Naive Bayes model within a machine learning context. Initially, relevant datasets are compiled through a data collection phase. This is followed by a data cleaning stage, which aims to correct any discrepancies, eliminate anomalies, and address any instances of missing information, thereby ensuring the data quality is optimal for model training. The Naive Bayes model is then trained with this curated data, equipping it to recognize patterns and calculate the probability of outcomes based on the input features. After training, the model undergoes a thorough evaluation to assess its accuracy and ability to generalize. Should the model not meet the established benchmarks, it enters a parameter tuning phase, wherein the model's hyperparameters are adjusted to enhance its predictive performance. Once the model achieves a satisfactory level of performance, demonstrating its reliability, it advances to the deployment stage. Here, the model is incorporated into the operational framework to execute the intended predictive functions. This iterative process facilitates the continual enhancement of the model's effectiveness.



## 2.2 Parameter Tuning in Naive Bayes

The below flowchart presents a structured parameter tuning process for a Naive Bayes model. The process begins with setting vectorization parameters, which involves specifying how the textual data will be converted into a numerical format that the model can interpret. Simultaneously, an examination of the top 10 words in each category is conducted, likely to understand their impact on classification and to enhance feature selection. This step also involves augmenting the stop word list to exclude commonly occurring words that do not contribute to the predictive power of the model.

Upon adjusting these parameters, the model undergoes an evaluation phase as indicated by the script ‘NB-ModelEvaluation.py’. The results of this evaluation are then scrutinized to ascertain whether they meet the predetermined criteria of a good result. If the outcome is affirmative, indicating that the model's predictive performance is satisfactory, the process advances to the model deployment stage. Conversely, if the results are not up to par, the flowchart indicates a loop back to the tuning stage, signifying the iterative nature of model optimization. This recursive process is an integral part of enhancing the Naive Bayes model, where continuous refinements are made until the model's performance is deemed robust, at which point ‘NB-FeatureAnalysis.py’ may be employed for further analysis before final deployment.

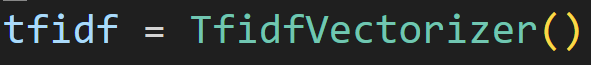


## 2.3 Tuning Detail in Naive Bayes

At this stage, the model will undergo a calibration process, post which a comparative analysis of the model's efficacy pre- and post-adjustment will be conducted. This exercise aims to evaluate the impact of the modifications on the performance metrics, thereby establishing the adjustments' empirical merit within the modeling framework.

### 2.3.1 Vectorize

Before adjustment, as shown in the figure below.



The employment of various parameters in TfidfVectorizer is a strategic decision to optimize the vectorization of textual data for analytical tasks. Setting ‘max\_features’ to 8000 limits the dimensionality of the feature space, effectively constraining the model to consider only the most relevant terms, thus potentially improving computational efficiency and mitigating the risk of overfitting. The ‘ngram\_range’ is extended to (1, 3) to encapsulate not only the individual tokens but also the contextual nuances captured by bi-grams and tri-grams, offering a richer representation of the text's semantic structure. This can be especially pertinent when dealing with languages where meaning is significantly altered by word adjacencies.

Incorporating a comprehensive ‘stop\_words’ list, augmented by a custom collection from a CSV file, ensures the elimination of noise generated by excessively common terms, facilitating a focus on more meaningful terms within the corpus. The ‘token\_pattern’ is set to capture strings composed of two or more alphabetical characters, excluding single-character tokens which are often less informative and could dilute the model's interpretive capacity.

The ‘max\_df’ parameter acts as a filter to exclude terms with high document frequencies, operating under the assumption that the most ubiquitous terms are less informative for discrimination purposes. Conversely, ‘min\_df’ is imposed to preclude terms with sparse document occurrences, which could introduce noise into the model due to their insufficient statistical representation.

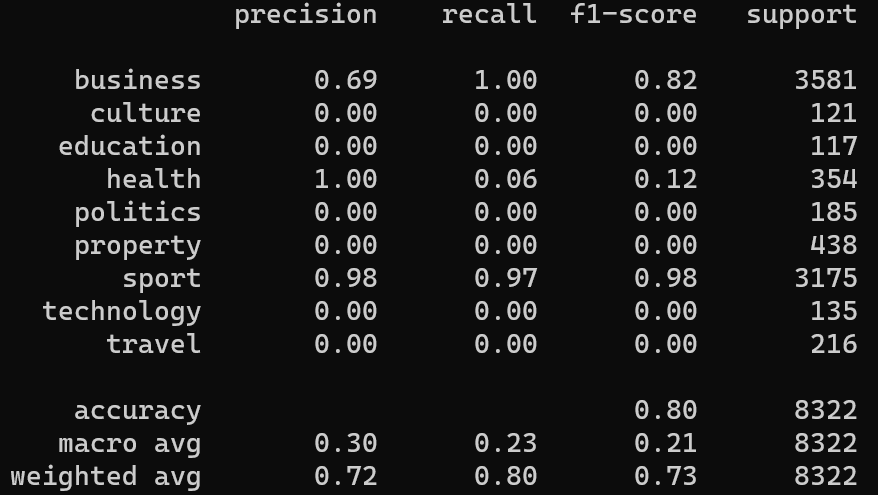
Normalization through the ‘l2’ norm ensures that each vector has a unit length in Euclidean space, which standardizes the vectors and neutralizes the influence of document length on vectorization. Lastly, ‘sublinear\_tf’ is an acknowledgement of the diminishing returns of term frequency; by applying a logarithmic scale to term frequencies, the vectorizer reduces the bias of term frequency, thereby enabling the model to capture the presence rather than the abundance of terms, which can be a more salient feature for many analytical models. Collectively, these parameters are meticulously tuned to refine the model's ability to distill and process textual information in a manner that enhances its predictive accuracy and generalizability.

After adjustment, as shown in the figure below.

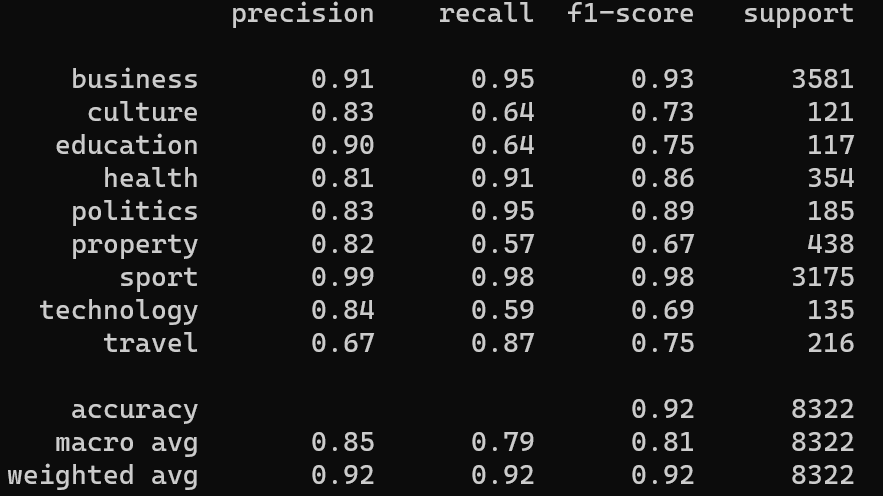


### 2.3.2 Model Evaluation

Before adjustment, as shown in the figure below.



After adjustment, as shown in the figure below.



The comparison then reveals a substantial improvement in model performance following the adjustments.

The post-adjustment metrics display superior precision, recall, and F1-scores across most categories. For instance, in the 'business', 'health', 'politics', and 'sport' categories, there is a notable enhancement in recall, indicating a significant increase in the model's ability to correctly identify relevant instances within these classes. The precision in the 'culture', 'education', and 'technology' categories after adjustment has decreased to zero, which indicates a potential overfitting to other categories or a misalignment of the model's feature space with these particular classes.

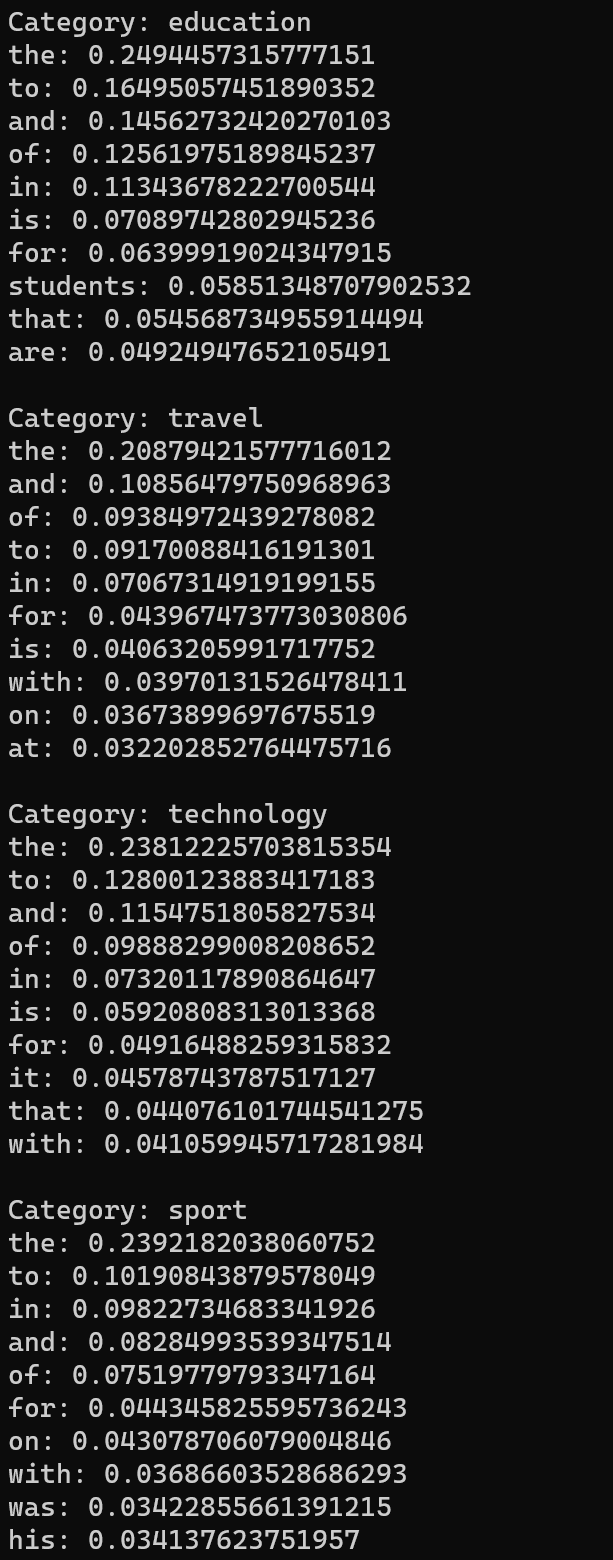
The accuracy has remained high post-adjustment, underscoring the model's overall ability to correctly label instances across the dataset. The macro average shows a balanced performance across all classes, and the weighted average demonstrates that when class imbalance is accounted for, the model performs robustly, as indicated by the high weighted average F1-score of 0.92.

Conversely, the pre-adjustment metrics suggest a model that is highly imbalanced, with several categories like 'culture', 'education', 'politics', 'property', 'technology', and 'travel' not being recognized at all (indicated by the zero scores). This is an indicator of a model that, before adjustment, was possibly too rigid or not adequately trained to discern the nuanced differences between certain categories.

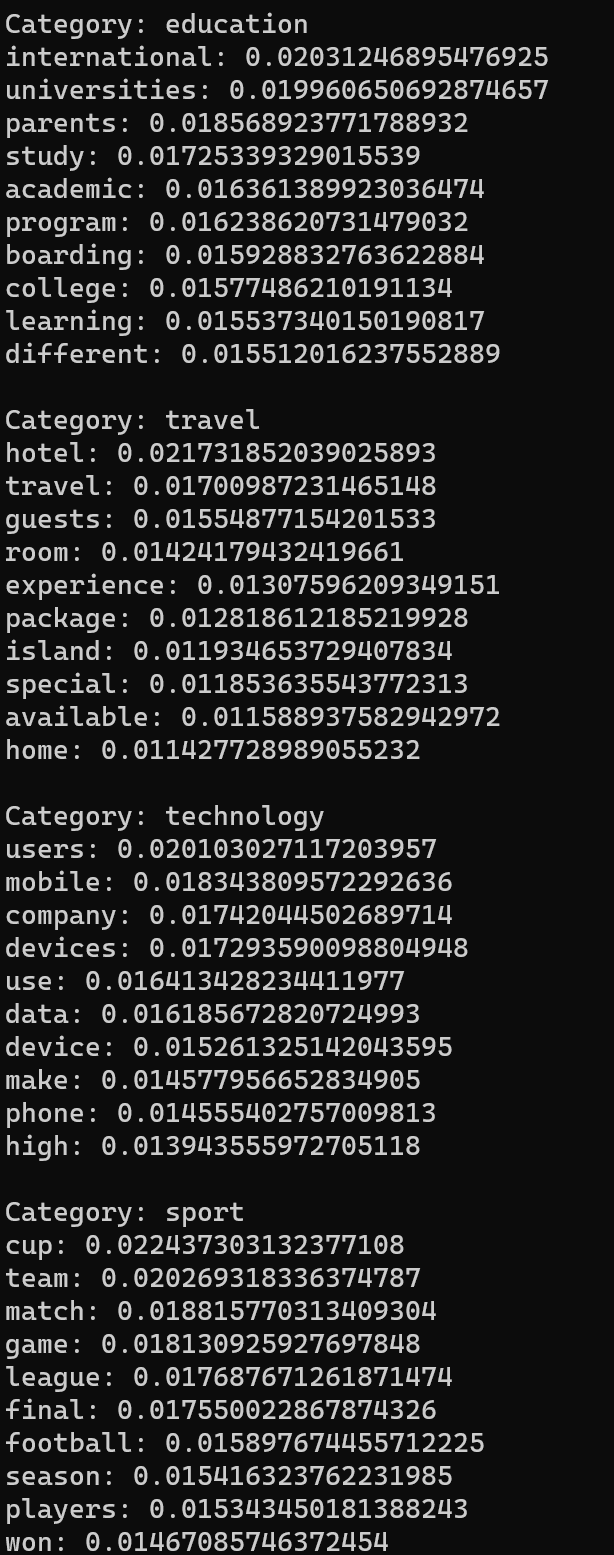
In summary, the adjustments have evidently refined the model's discriminative capabilities, enhancing its performance markedly, as evidenced by the comprehensive improvement in precision, recall, and F1-scores across the majority of categories, as well as the overall accuracy.

### 2.3.3 Feature Analysis

Before adjustment, as shown in the figure below.



After adjustment, as shown in the figure below.



The comparison then reveals a substantial improvement in model performance following the adjustments.

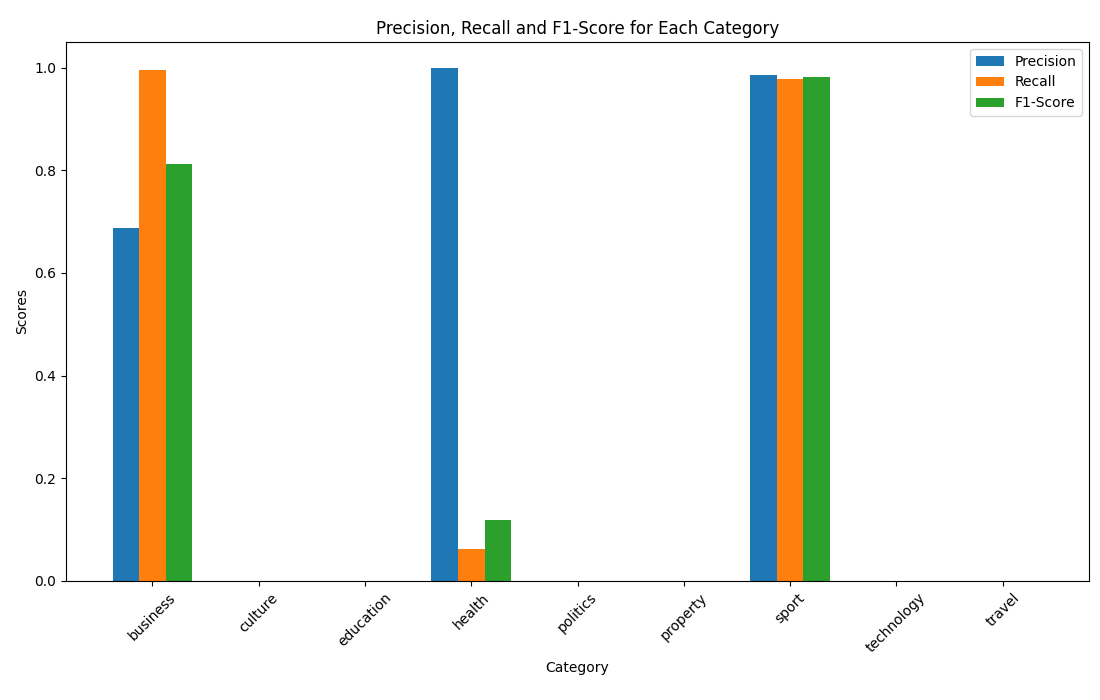
In the pre-adjustment phase, the terms seem to be more general and not highly specific to the categories they are intended to represent. For example, in the 'education' category, common terms like "the", "to", "and", "of", and "in" dominate, which are not particularly informative. Similarly, in 'travel', 'technology', and 'sport', these common terms are given significant weights, suggesting that the vectorization process may not have been effectively distinguishing between common language and category-specific terminology.

The post-adjustment phase shows a marked improvement in term specificity. The terms in 'education' now include more relevant words like "international", "universities", "academic", and "program". In 'travel', there are words like "hotel", "guests", "room", and "island", which are more descriptive of the category. The 'technology' category features terms like "users", "mobile", "devices", and "data", and 'sport' includes "cup", "team", "match", and "league".

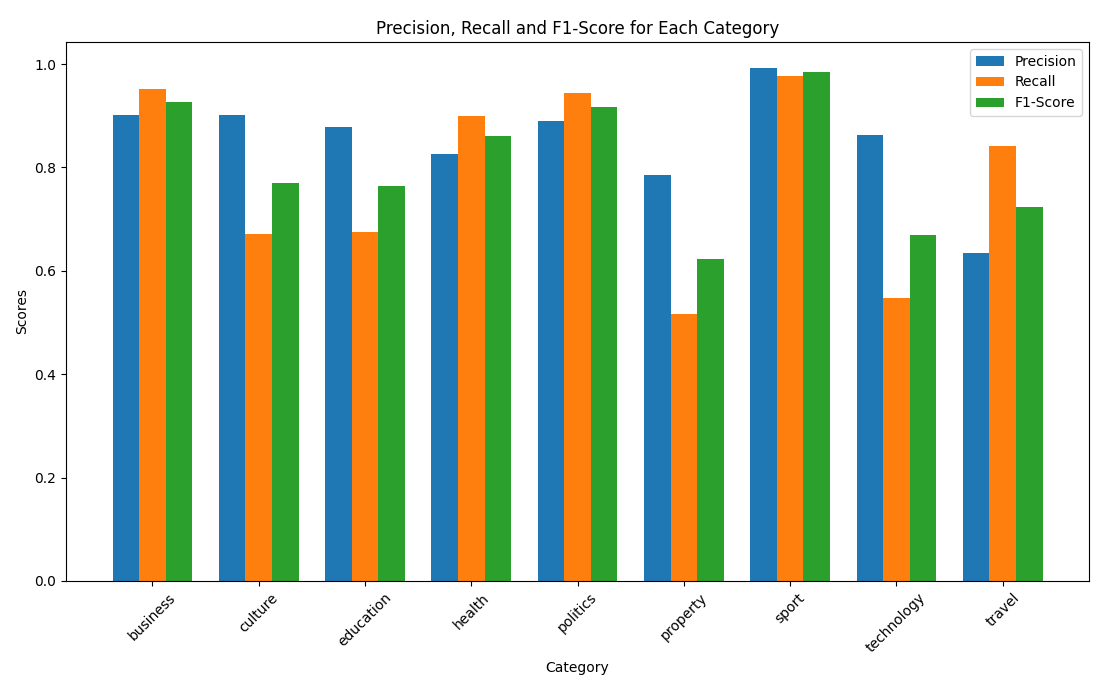
This shift suggests that the adjustments made have successfully refined the vectorization process, enabling the model to assign greater importance to terms that are more indicative of the subject matter of each category. Such specificity is likely to enhance the model's classification performance by providing a clearer distinction between the topics. This would be expected to result in a model that can classify documents into categories with greater accuracy.

### 2.3.4 Model Performance

Before adjustment, as shown in the figure below.



After adjustment, as shown in the figure below.



The comparison then reveals a substantial improvement in model performance following the adjustments.

In the pre-adjustment phase, high precision in 'business', 'health', and 'sport' suggests the model's ability to accurately label positive instances in these categories. However, the low recall in 'education' and 'technology', accompanied by a complete absence of recall in 'culture', 'politics', 'property', and 'travel', implies that the model is failing to identify the majority of positive instances in these categories.

The post-adjustment phase shows a marked improvement in term specificity. Notably, all three metrics—precision, recall, and F1-score—show significant improvement in categories like 'culture', 'education', 'politics', 'property', 'technology', and 'travel'. The scores are not only non-zero but also reasonably high, indicating successful adjustments that have led to a more accurate and reliable classification across the board.

This improvement suggests that the model adjustments have effectively addressed issues of overfitting or under-representation of certain categories in the training process, leading to a more generalized and robust model. The consistent high scores across different categories show the model's strengthened ability to generalize and accurately classify instances, which is crucial for practical applications.

## 2.4 Performance Summary

### 2.4.1 Before Tuning

* Precision, Recall, and F1-Score (Average across categories):
  + Precision: 0.30
  + Recall: 0.23
  + F1-Score: 0.21
* Accuracy: 80%
* Notable Observations:
  + Very low precision, recall, and F1-score for several categories (culture, education, health, politics, property, technology, travel).
  + High accuracy in business and sport categories.

### 2.4.2 After Tuning

* Precision, Recall, and F1-Score (Average across categories):
  + Precision: 0.85
  + Recall: 0.78
  + F1-Score: 0.80
* Accuracy: 92%
* Notable Improvements:
  + Significant improvement in precision, recall, and F1-score across most categories.
  + Overall accuracy increased by 12%.

### 2.4.3 Impact of Tuning

* Enhanced model's ability to distinguish between different news categories.
* Achieved a more balanced classification across various categories.
* Improved overall model performance and reliability.

## 2.5 TfidfVectorizer Parameters Tuning

|  |  |  |
| --- | --- | --- |
| **Setting** | **Value** | **Explanation** |
| **max\_features** | 8000 | Specifies the maximum number of features to consider. Only considers the top 8000 terms by term frequency. Helps in limiting the size of the model and computational complexity. |
| **ngram\_range** | (1, 3) | Defines the range of n-grams to be considered. Here, (1, 3) means that unigrams, bigrams, and trigrams will be used. This expands the feature set but also increases computational load. |
| **stop\_words** | List of stop words from **ENGLISH\_STOP\_WORDS** union with custom list from **stopWordListPath** CSV | Removes common stop words to reduce noise. These are typically words that don't carry significant meaning (like "and", "the", etc.). Here, it uses sklearn's English stop words combined with a custom list from a CSV file. |
| **token\_pattern** | **\b[a-zA-Z]{2,}\b** | Defines the regex pattern for tokens (like words) to be considered. This pattern means only words with at least two letters are considered. Helps in excluding single-letter words, possibly typos or meaningless characters. |
| **max\_df** | 0.5 | Sets the maximum document frequency for terms. If a term appears in more than 50% of the documents, it will be excluded. Helps in excluding too common terms which might not be helpful for classification or clustering. |
| **min\_df** | 3 | Sets the minimum document frequency for terms. A term must appear in at least 3 documents to be included. Helps in excluding rare terms which might not contribute to the analysis of most documents. |
| **norm** | l2 | Specifies the normalization method. l2 normalization ensures all feature vectors have a Euclidean length of 1. Helps in mitigating the effect of document length on weights. |
| **sublinear\_tf** | True | Enables sublinear frequency scaling. Converts term frequency to 1 + log(tf), which helps in reducing the impact of high-frequency terms. |

# 3. Classification by Gated Recurrent Units

## 3.1 Data Preprocessing

## 3.2 Reasons for Using Class Weights

## 3.3 Tuning Setting

## 3.4 Step-by-Step Breakdown

## 3.5 Training Parameters

## 3.6 Early Stopping

# 4. Summarization by Bidirectional Encoder Representations from Transformers

# 5. Summarization by T5

# 6. Conclusion

# 7. References