MACHINE LEARNING PROJECT REPORT

THAPAR INSTITUTE
OF ENGINEERING & TECHNOLOGY
(Deemed to be University)

Submitted By:

Name – Vanshikaa
Roll number -102103580
Batch - 3COE21

Submitted To:

Ms. Suchita Sharma

July 2023 – December 2023

1 Introduction

1.1 Problem Statement

The aim of this project is to develop a robust multiclass text classification system for categorizing news articles into predefined categories based on the content. The goal is to create a model that can accurately classify news articles from the BBC News dataset into various categories, such as business, entertainment, politics, sport, and tech.

1.2 Dataset

The dataset used for this project is the BBC News dataset, which consists of news articles collected from the BBC News website. It comprises articles from 2004 to 2005 and is divided into five categories: business, entertainment, politics, sport, and tech.

2 Data Exploration

2.1 Data Overview

The dataset consists of articles from the BBC News dataset, categorized into five classes: sport, business, politics, tech, and entertainment. Initial observations and summary statistics of the dataset are as follows:

- Data Shape: The dataset contains a total of 2226 rows and 2 columns.
- Column Information: The columns include ’category’ and ’text’.
- Category Distribution: Each article is labeled with one of the five categories mentioned above, indicating the news category it belongs to. The distribution of articles across categories is as follows:
  - Sport: 511 articles
  - Business: 510 articles
  - Politics: 417 articles
  - Tech: 401 articles
  - Entertainment: 386 articles

3 Data Preprocessing

3.1 Handling Missing Values

The ’text’ column was checked for missing values using the fillna() function, replacing any NaN values with an empty string. Verification was done to ensure there are no more missing values in the dataset, as indicated by df.isna().sum() resulting in zeros across both ’category’ and ’text’ columns.

3.2 Text Preprocessing Steps

3.2.1 Lowercasing and Stripping

Lowercasing the text content and stripping unnecessary spaces, replacing newlines, and ensuring uniformity.

3.2.2 Removing Non-Alphabetic Characters

Using regular expressions to remove non-alphabetic characters, ensuring only alphabetical characters remain.

3.2.3 Removing Links

Utilizing regex to eliminate URLs or hyperlinks from the text.

3.2.4 Tokenization and Stopword Removal

Tokenizing the text into words and removing stopwords from the dataset using NLTK's English stopwords list.

3.2.5 Lemmatization

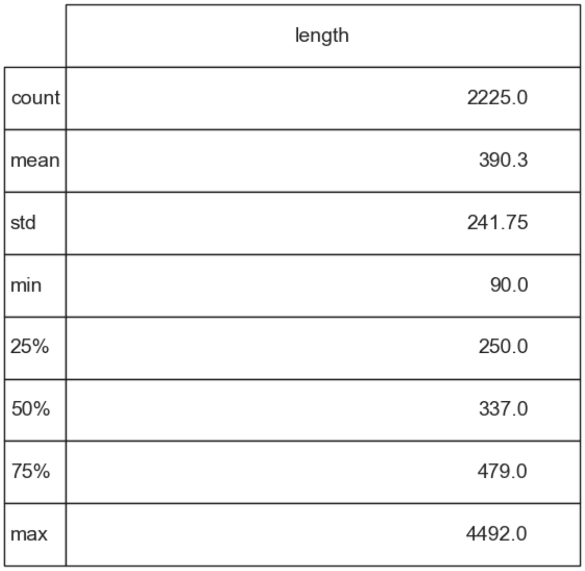
Lemmatizing the remaining words to reduce them to their base forms for better analysis and modeling.

4 Exploratory Data Analysis (EDA)

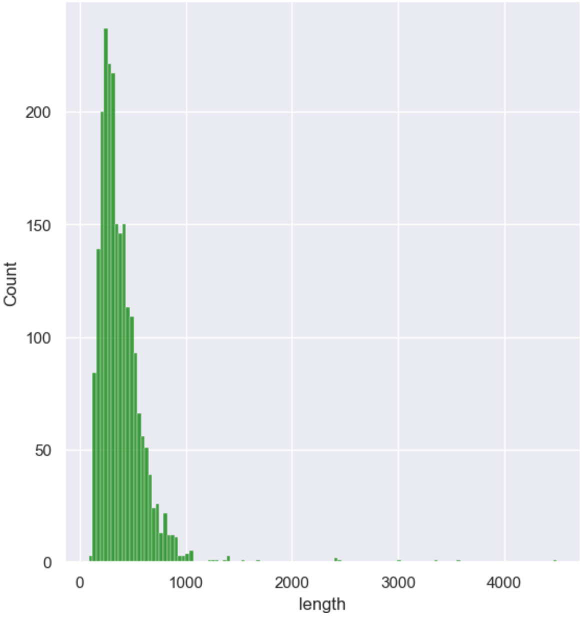
4.1 Text Length Distribution

The histogram visualizes the distribution of text lengths across all articles in the dataset. The x-axis represents the number of words in an article, while the y-axis displays the frequency of articles with a specific word count. This histogram showcases the variability in text lengths, where the majority of texts fall within the range of 200 to 500 words, with a peak around the median length of 337 words. This distribution indicates a diverse range of text lengths present in the dataset.

Distribution of text length for text.

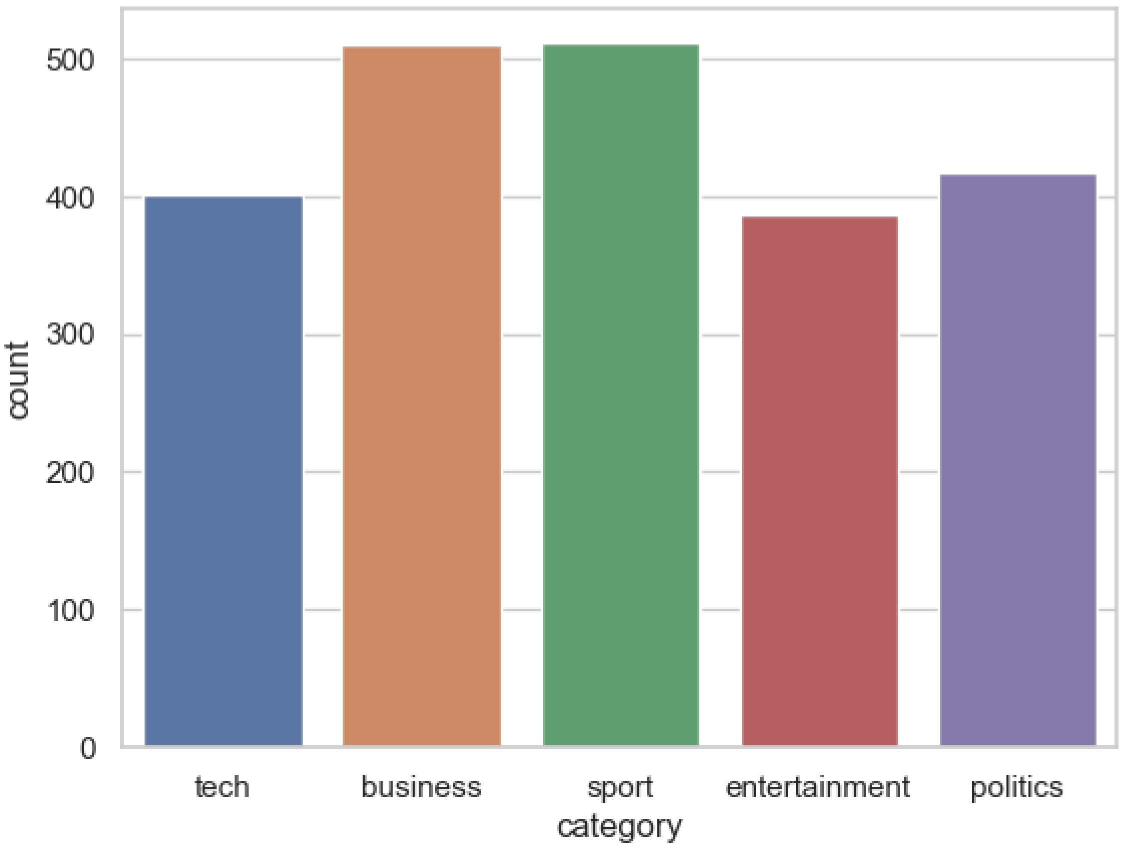


	length
count	2225.0
mean	390.3
std	241.75
min	90.0
25%	250.0
50%	337.0
75%	479.0
max	4492.0



4.2 Category Distribution

The count plot illustrates the distribution of articles across different categories within the dataset. Each bar represents a news category, and the height of the bars indicates the number of articles associated with that specific category. The distribution demonstrates a relatively balanced dataset across the five categories, with ’Sport’ and ’Business’ having the highest number of articles, followed closely by ’Politics,’ ’Tech,’ and ’Entertainment.’

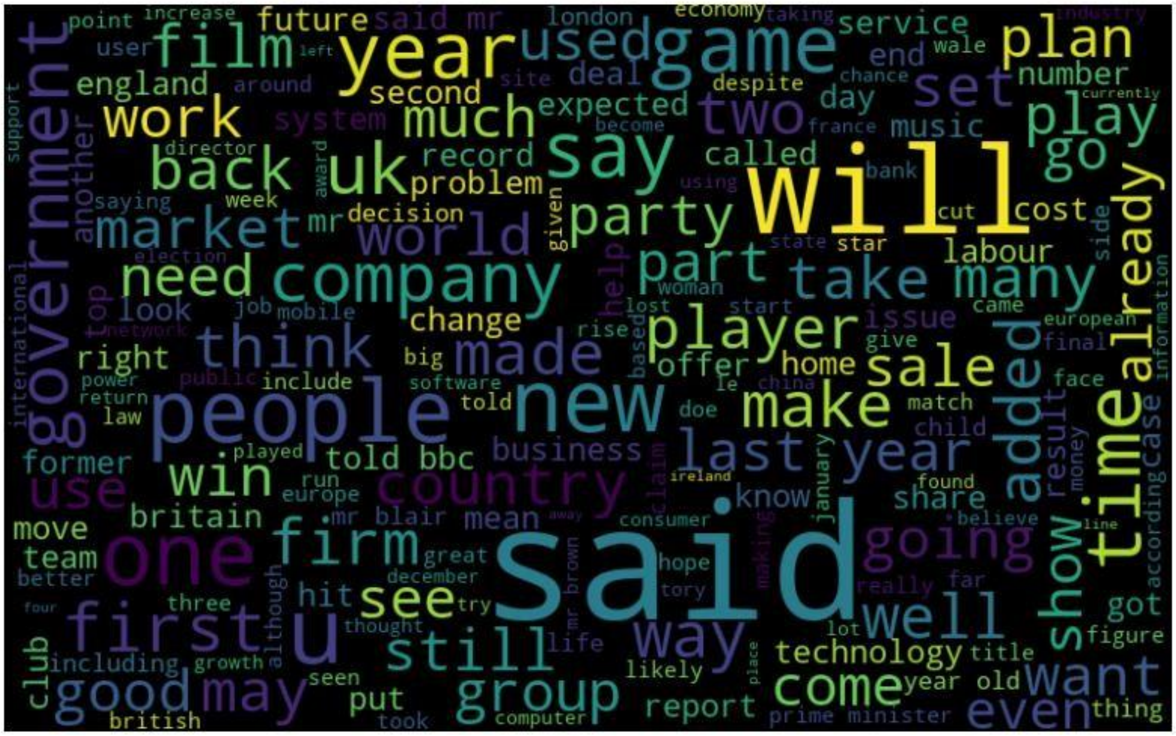


4.3 Word Frequency Analysis

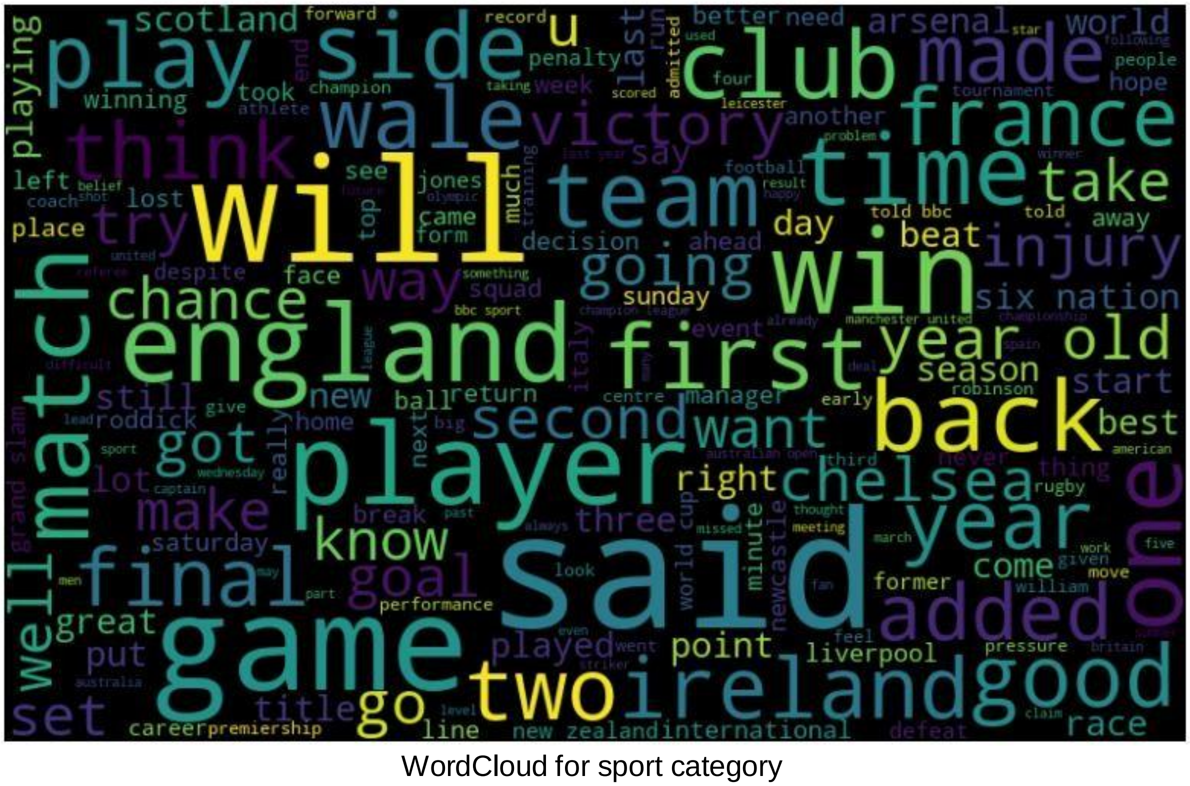
The bar chart highlights the most common words present in the dataset. It showcases the top 20 frequently occurring words across all articles after preprocessing steps such as tokenization, stopword removal, and lemmatization. This visualization provides insight into the prevalent terms present in the overall text corpus, aiding in understanding the dominant themes and topics covered in the articles.

4.4 Word Clouds for Each Category

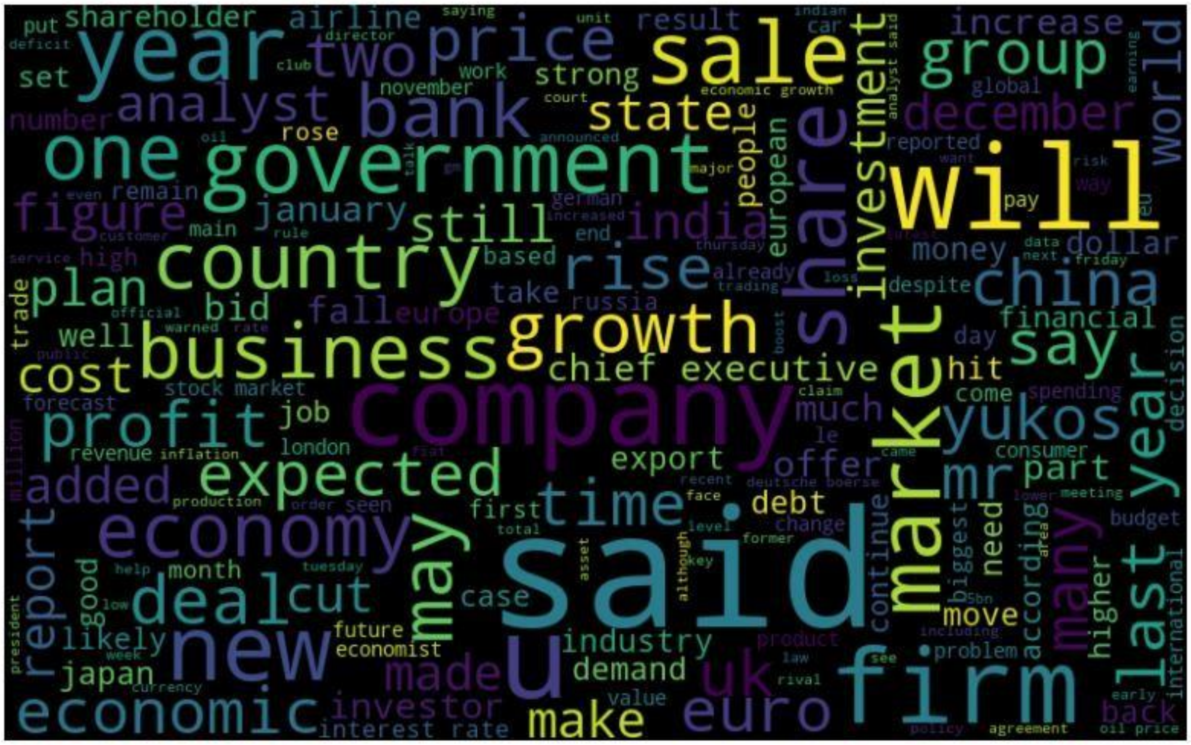
The word clouds generated for each category display the most prominent words in the respective category's articles. The size of each word corresponds to its frequency in the articles within that specific category. These visualizations offer a quick glimpse into the distinct vocabulary and recurring themes within each news category, such as 'Sport,' 'Business,' 'Politics,' 'Tech,' and 'Entertainment.' Common terms within each category can be identified based on the size and prominence of the words in the word clouds.



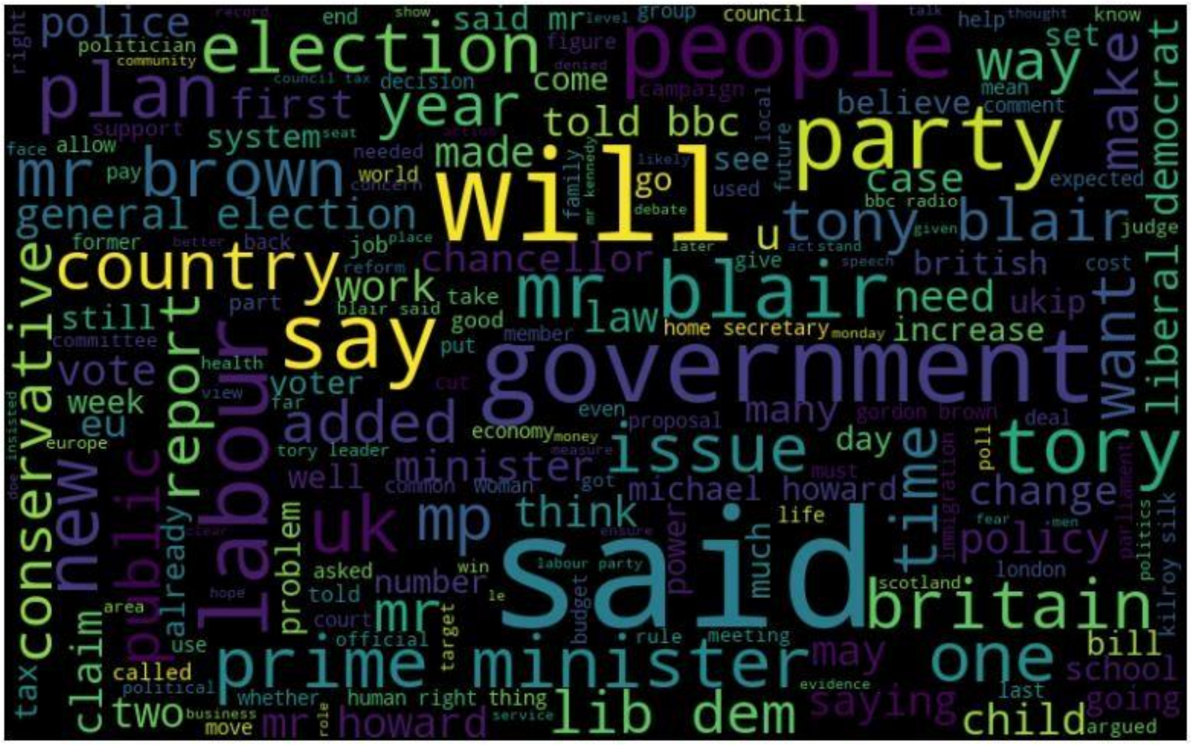
WordCloud for all text articles



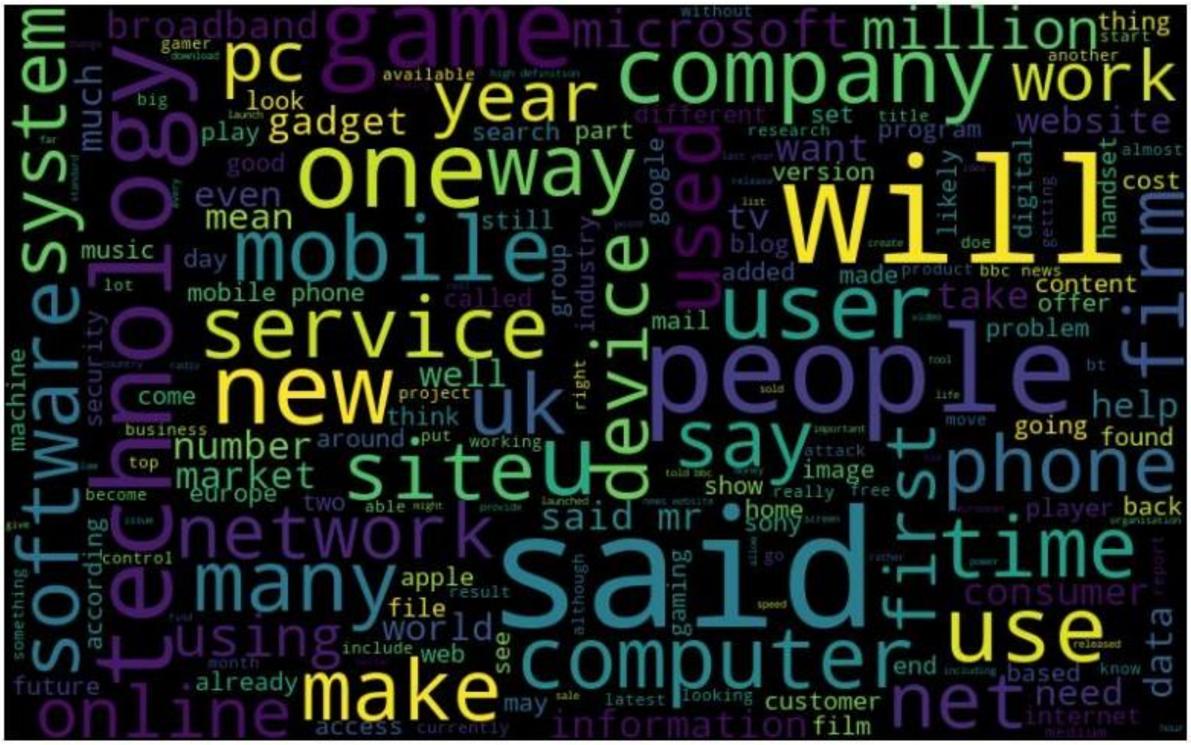
WordCloud for sport category



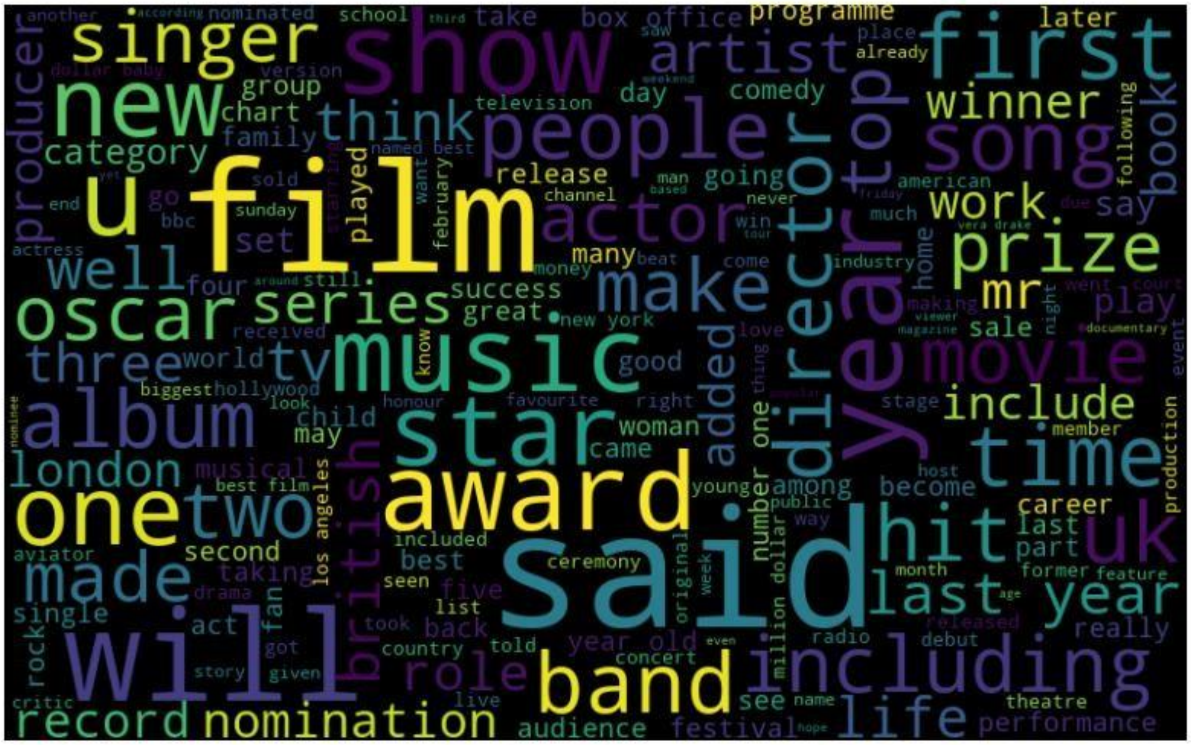
WordCloud for ‘business’ category



WordCloud for ‘politics’ category



WordCloud for ‘tech’ category



WordCloud for ‘entertainment’ category

5 Applying N-gram

N-grams have been applied to the text data for feature extraction and representation. The process involved splitting the text corpus into tokens of contiguous words and generating n-grams (sequences of ’n’ words) to capture contextual information. Key steps taken during this phase include:

5.1 Train-Test Split

The dataset was split into training and test sets with a ratio of 75:25, maintaining a random seed of 42 for reproducibility. The split resulted in:

- Training Set: 1668 samples
- Test Set: 557 samples

5.2 Count Vectorization

Utilizing CountVectorizer with an n-gram range of (1, 2), the text was transformed into a matrix representation, capturing unigrams and bigrams. This generated a feature matrix with a shape of (1668, 284054) for the training set and (557, 284054) for the test set.

5.3 TF-IDF Transformation

The term frequency-inverse document frequency (TF-IDF) transformation was applied to the count matrix using TfidfTransformer. This transformation normalized the count matrix and assigned weights to the terms based on their importance in the corpus.

5.4 Model Serialization

The fitted TF-IDF transformer and CountVectorizer were serialized using joblib.dump() to store the transformer and vectorizer objects as ’transformer. pkl’ and ’countvect.pkl,’ respectively, for future use in model training and prediction.

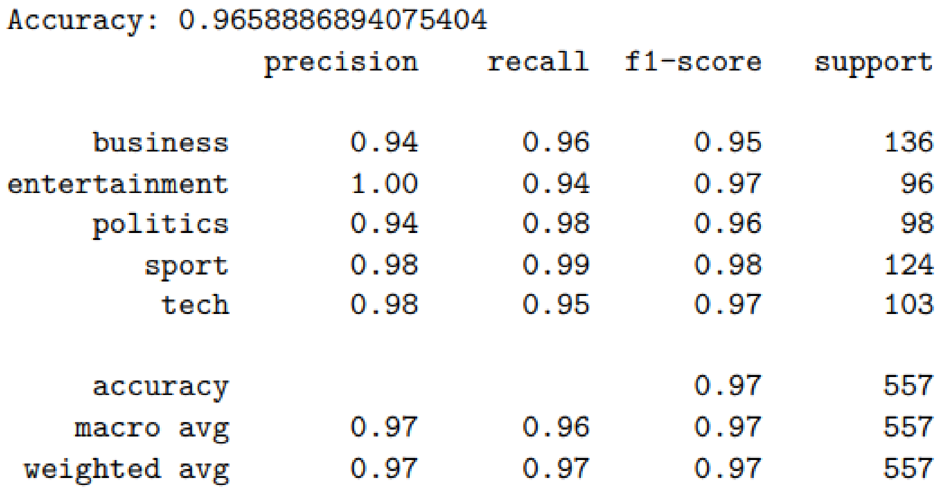
This phase highlights the utilization of n-grams, specifically unigrams and bigrams, to capture both individual words and word sequences within the text corpus. The serialized objects ’transformer.pkl’ and ’countvect.pkl’ can be loaded and used in subsequent model development stages.

6 Machine Learning Models

6.1 Logistic Regression

The Logistic Regression model was trained and evaluated for text classification tasks, yielding promising results:

- The accuracy per category ranges from 94% to 98%, indicating the model’s effectiveness in classifying articles into distinct categories.
- The model was cross-validated using 10-fold cross-validation, demonstrating consistent performance with an average accuracy of 96.6%.
- The serialized model was saved as ’Text_LR.pkl’ for future use.

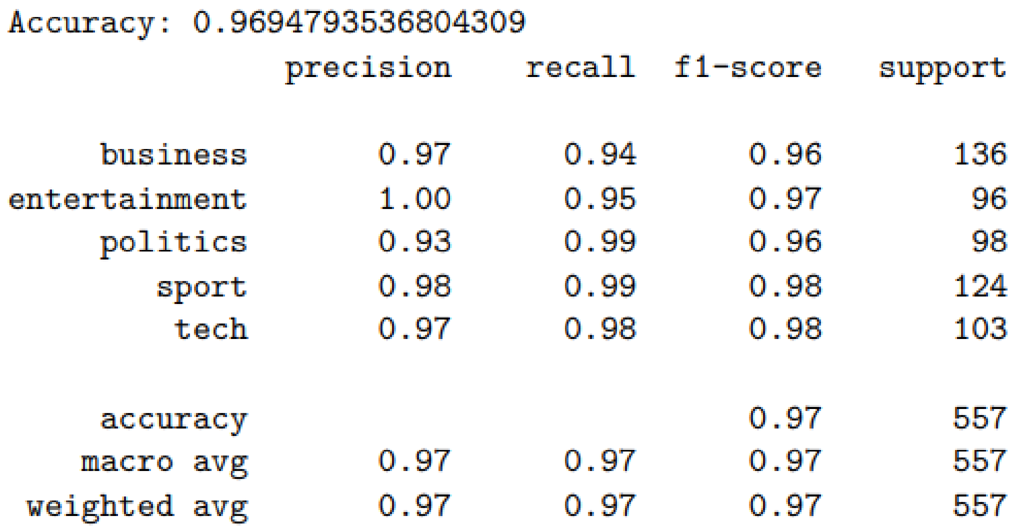


```
Accuracy: 0.9658886894075404
               precision    recall  f1-score   support

     business       0.94      0.96      0.95       136
entertainment       1.00      0.94      0.97        96
     politics       0.94      0.98      0.96        98
        sport       0.98      0.99      0.98       124
         tech       0.98      0.95      0.97       103

     accuracy                           0.97       557
    macro avg       0.97      0.96      0.97       557
 weighted avg       0.97      0.97      0.97       557
```

6.2 Support Vector Machine (SVM)

The Linear Support Vector Machine model exhibited strong performance in text classification:

- Achieving accuracy between 93% to 98% for each category signifies its effectiveness in classifying articles.
- The cross-validation score of the SVM model was 97.8% on average, indicating robustness and stability in its performance.
- The serialized model was saved as ’Text_SVM.pkl’ for future use.

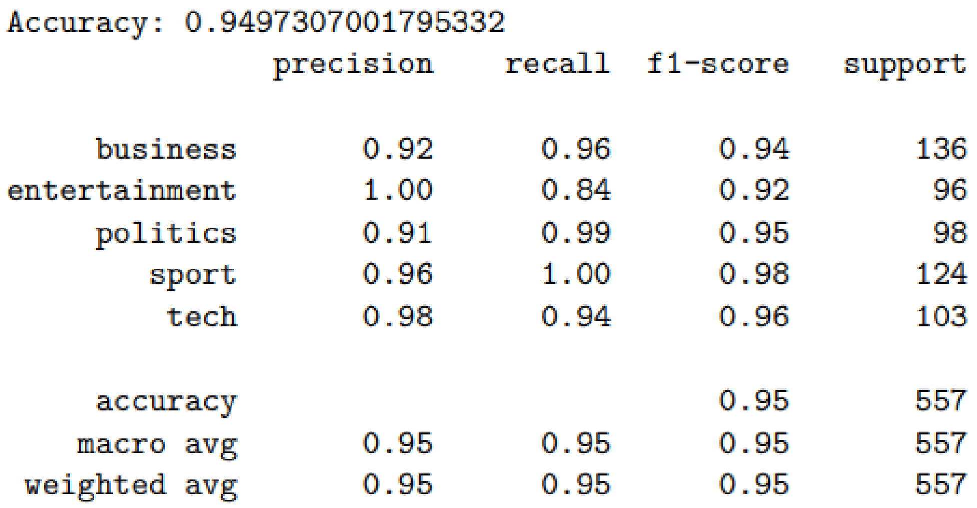


```
Accuracy: 0.9694793536804309
               precision    recall  f1-score   support

     business       0.97      0.94      0.96       136
entertainment       1.00      0.95      0.97        96
     politics       0.93      0.99      0.96        98
        sport       0.98      0.99      0.98       124
         tech       0.97      0.98      0.98       103

     accuracy                           0.97       557
    macro avg       0.97      0.97      0.97       557
 weighted avg       0.97      0.97      0.97       557
```

6.3 Naive Bayes (Multinomial)

The Multinomial Naive Bayes model showcased respectable performance:

- Achieving accuracy ranging from 91% to 98% indicates its capability in classifying articles into relevant categories.
- The cross-validation score averaged at 94.9%, demonstrating consistent performance across the folds



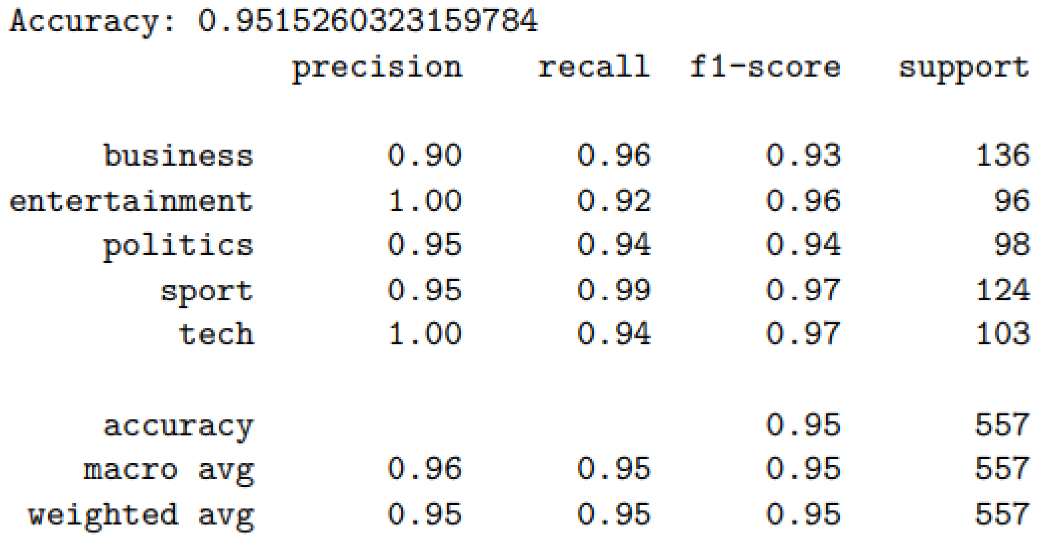
```
Accuracy: 0.9497307001795332
               precision    recall  f1-score   support

     business       0.92      0.96      0.94       136
entertainment       1.00      0.84      0.92        96
     politics       0.91      0.99      0.95        98
        sport       0.96      1.00      0.98       124
         tech       0.98      0.94      0.96       103

     accuracy                           0.95       557
    macro avg       0.95      0.95      0.95       557
 weighted avg       0.95      0.95      0.95       557
```

6.4 Random Forest Classifier

The Random Forest Classifier exhibited competitive performance in text classification:

- The cross-validation score of the Random Forest Classifier averaged at 96%, showcasing stability and reliability in performance.

These machine learning models were effective in categorizing news articles into relevant topics, each demonstrating strengths in various aspects of text classification. Adjustments or ensemble strategies could be explored to further enhance model performance based on specific requirements or preferences.



```
Accuracy: 0.9515260323159784
               precision    recall  f1-score   support

     business       0.90      0.96      0.93       136
entertainment       1.00      0.92      0.96        96
     politics       0.95      0.94      0.94        98
        sport       0.95      0.99      0.97       124
         tech       1.00      0.94      0.97       103

     accuracy                           0.95       557
    macro avg       0.96      0.95      0.95       557
 weighted avg       0.95      0.95      0.95       557
```

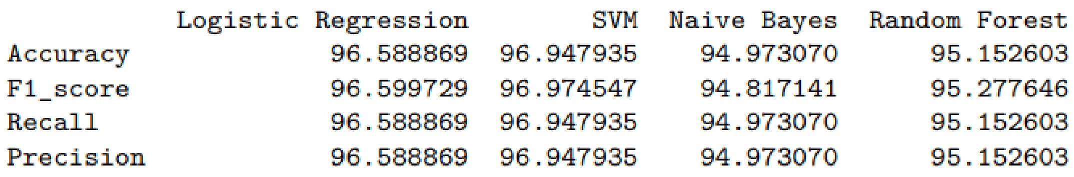
7 Training Strategy

7.1 Model Development Approach

The machine learning models were trained following a supervised learning approach. The dataset was preprocessed, including steps like text cleaning, tokenization, removing stop words, and lemmatization, to prepare the text data for model ingestion. The text content was vectorized using n-gram representation (both unigrams and bigrams) to capture the contextual information present in the articles. Four different classifiers—Logistic Regression, Support Vector Machine, Naive Bayes (Multinomial), and Random Forest Classifier were employed and fine-tuned for optimal performance.

8 Results

8.1 Model Evaluation



```
           Logistic Regression        SVM  Naive Bayes  Random Forest
Accuracy             96.588869  96.947935    94.973070      95.152603
F1_score             96.599729  96.974547    94.817141      95.277646
Recall               96.588869  96.947935    94.973070      95.152603
Precision            96.588869  96.947935    94.973070      95.152603
```

The Logistic Regression and SVM models emerged as top performers, showcasing exceptional accuracy rates of approximately 96.59% and 96.95%, respectively. Both models demonstrated consistency across key metrics such as accuracy, F1-score, recall, and precision, affirming their reliability in effectively categorizing news articles into predefined topics.

Naive Bayes also exhibited respectable performance, achieving an accuracy of around 94.97% alongside comparable F1-scores, recall, and precision metrics. While slightly trailing behind Logistic Regression and SVM, its consistent performance indicates its capability in accurate classification tasks.

Moreover, the Random Forest Classifier, with an accuracy of approximately 95.15%, proved its competitiveness in the text classification domain. Though marginally below the top-performing models, its stability and reliability align with industry standards, showcasing its potential for effective categorization.