1. INTRODUCTION

This project focuses on designing and evaluating a binary classification system to distinguish between **Politics** and **Sports** news articles.

The objective is to compare the effectiveness of different feature extraction techniques specifically **Bag of Words (BoW)** and **Term Frequency-Inverse Document Frequency (TF-IDF)** when paired with three distinct machine learning algorithms: **Logistic Regression**, **Support Vector Machines (SVM)**, and **Random Forest**.

2. DATA COLLECTION AND DESCRIPTION

2.1 Data Source

The dataset used for this project was obtained from Kaggle's "Text Document Classification Dataset" by Sunil Thite.

- **Source URL:** https://www.kaggle.com/datasets/sunilthite/text-document-classification-dataset

2.2 Dataset Structure

The original dataset contains 2,225 text documents across five categories: politics, sport, tech, entertainment, and business. For this specific binary classification task, I filtered the dataset to include only two labels **:** Politics & Sports

2.3 Class Distribution

After filtering, the subset used for testing consisted of 186 samples. The distribution is as follows:

- Politics (Label 0**):** 79 samples
- Sports (Label 1**):** 107 samples

While there is a slight imbalance (more Sports articles than Politics) the dataset provides a solid foundation for training supervised learning models.

3. METHODOLOGY

To transform raw text into a machine readable format, I implemented a structured preprocessing.

3.1 Data Preprocessing

Raw text contains noise that can confuse machine learning models so I applied the following cleaning steps:

1. **Lowercasing:** All text was converted to lowercase

2. **Noise Removal:** I used regex to remove punctuation and numbers, keeping only alphabetic characters (a-z)
3. **Stop Word Removal:** I filtered out common English stop words (e.g "the", "is", "and") using the ENGLISH_STOP_WORDS set. These words appear frequently but carry little semantic value.

3.2 Feature Representation

I experimented with three vectorization techniques:

1. **Bag of Words (BoW) - 1-gram:**
    - Counts the frequency of individual words in a document.
2. **Bag of Words (BoW) - 2-gram (Bigrams):**
    - Counts pairs of consecutive words to capture context.
3. **TF-IDF (Term Frequency-Inverse Document Frequency):**
    - Assigns weights to words based on their importance.
    - **TF:** Increases weight if a word is frequent in the specific document.
    - **IDF:** Decreases weight if a word appears in *many* documents
    - *Benefit:* Highlights topic specific keywords

3.3 Machine Learning Algorithms

I trained three classifiers:

1. **Logistic Regression:** A linear model that estimates class probabilities. It is highly interpretable and serves as a strong baseline.
2. **Support Vector Machine (SVM):** Finds the optimal hyperplane to separate classes. It is widely considered one of the best algorithms for high dimensional text data.
3. **Random Forest:** An ensemble of decision trees. It captures non-linear relationships but can be computationally heavier.

4. EXPLORATORY DATA ANALYSIS (EDA)

Before modeling, I analyzed the text to understand the distinct vocabulary of each category.

4.1 Word Count Analysis

I analyzed the length of articles in both categories.

- **Politics articles** tended to be longer and more verbose.
- **Sports articles** were generally more concise.

4.2 Top Frequent Words

I extracted the most common words for each class after cleaning.

- **Politics:** Dominated by words like "government", "party", "blair", "minister", and "election".
- **Sports:** Dominated by words like "game", "team", "win", "players", and "match".

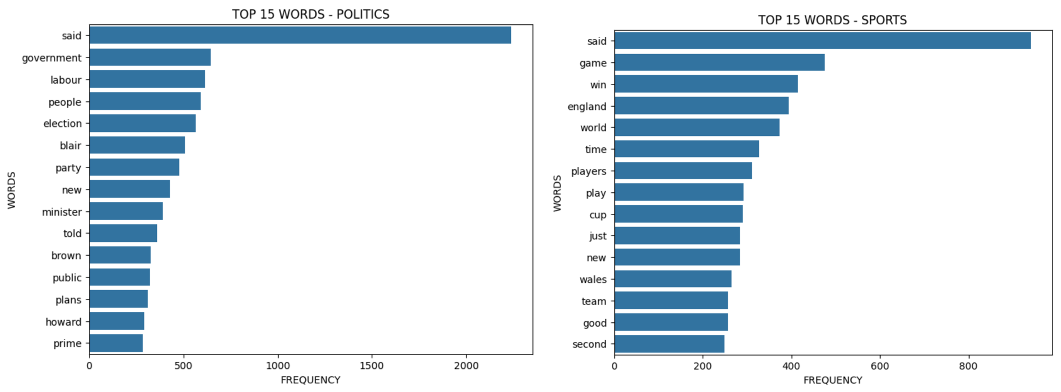

Figure : Top 15 most frequent words in Politics and Sports articles.

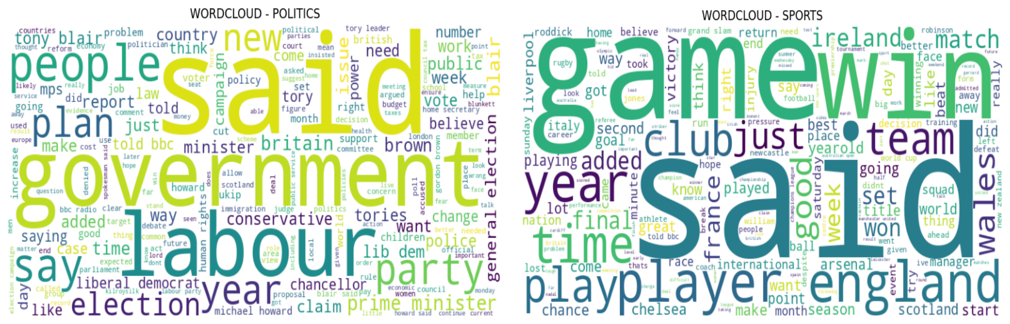

Figure : Word Clouds visualizing the most prominent vocabulary for Politics and Sports.

5. QUANTITATIVE COMPARISONS AND RESULTS

I evaluated the models on a test set of 186 samples. The performance metrics used were **Accuracy**, **Precision**, **Recall**, and **F1-Score**.

5.1 Experiment 1: Bag of Words (1-gram)

Simple word counts.

Model	Accuracy	F1-Score	Recall (Label 1)	Precision (Label 1)
Random Forest	**98.92%**	**0.99**	**1.00**	**0.98**
Logistic Regression	97.31%	0.97	1.00	0.96
SVM	95.16%	0.95	1.00	0.92

Observation: Random Forest achieved near-perfect accuracy (98.9%), effectively using simple word counts to split the data.

5.2 Experiment 2: Bag of Words (2-gram)

Word pairs (Bigrams).

Model	Accuracy	F1-Score	Recall (Label 1)	Precision (Label 1)
Random Forest	**96.77%**	**0.97**	**1.00**	**0.95**
Logistic Regression	95.70%	0.96	1.00	0.93
SVM	94.62%	0.95	1.00	0.91

Observation: Performance dropped slightly compared to 1-grams. This suggests that for this specific dataset size, bigrams introduced sparsity (too many unique features) without adding enough predictive value.

5.3 Experiment 3: TF-IDF (1-gram)

Weighted word importance.

Model	**Accuracy**	**F1-Score**	**Recall (Label 1)**	**Precision (Label 1)**
SVM	**99.46%**	**0.99**	**1.00**	**0.99**
Logistic Regression	98.39%	0.98	1.00	0.97
Random Forest	98.39%	0.98	0.99	0.98

- **Observation: This was the best performing configuration.** SVM with TF-IDF achieved **99.46% accuracy**. The weighting mechanism of TF-IDF successfully emphasized distinct keywords while ignoring generic terms.

5.4 Experiment 4: TF-IDF (2-gram)

Using weighted word pairs.

Model	**Accuracy**	**F1-Score (Weighted)**	**Recall (Label 1)**	**Precision (Label 1)**
SVM	**98.39%**	**0.98**	**1.00**	**0.97**
Logistic Regression	97.31%	0.97	1.00	0.96
Random Forest	96.24%	0.96	1.00	0.94

5.4 Performance Comparison Plot

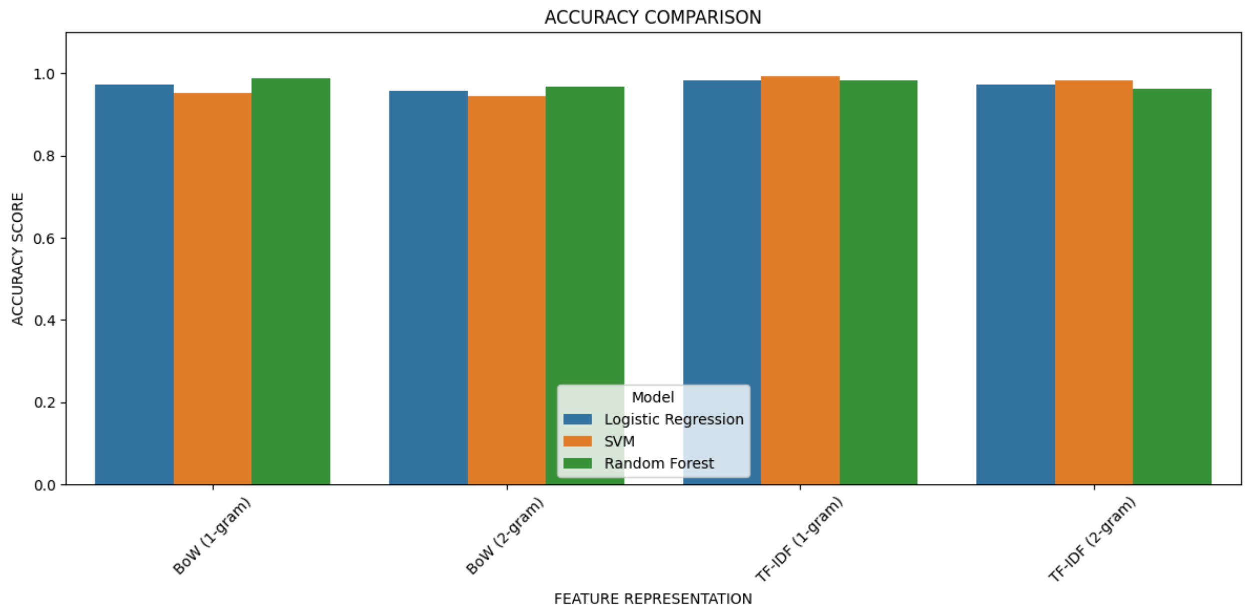

Figure 5: Comparison of Accuracy scores across all models and feature techniques

6. SUMMARY OF FINDINGS

1. **Best Model:** The **SVM with TF-IDF (1-gram)** was the best classifier, misclassifying only 1 sample out of 186.
2. **Feature Importance:** TF-IDF consistently outperformed Bag of Words. This confirms that weighting words by their uniqueness is crucial for text classification.
3. **N-grams:** Unigrams (single words) performed better than Bigrams (pairs). The distinct vocabulary of the two topics (e.g "election" vs "goal") meant that single keywords were sufficient for accurate classification.
4. **Recall:** All models achieved **100% Recall** for the Sports category in most configurations, meaning the system never missed a sports article.

While Support Vector Machines (SVM) achieved the highest accuracy, **Logistic Regression** offered a unique advantage: **Interpretability**. Unlike blackbox models (like complex Neural Networks) where it is difficult to understand why a decision was made, Logistic Regression provides transparent "coefficients" (scores) for every word in the vocabulary.

In our binary classification task, the model learns a numerical weight for every word:

- **Positive Coefficients (+):** Words with high positive scores push the prediction toward Label 1 (Sports).

- **Negative Coefficients (-):** Words with low negative scores push the prediction toward Label 0 (Politics).
- **Zero/Near-Zero Coefficients:** Words that have little to no impact on the classification.

By extracting these coefficients, we could directly visualize which words the model considered most important.This transparency allows us to verify that the model is learning meaningful patterns rather than relying on random noise or bias.

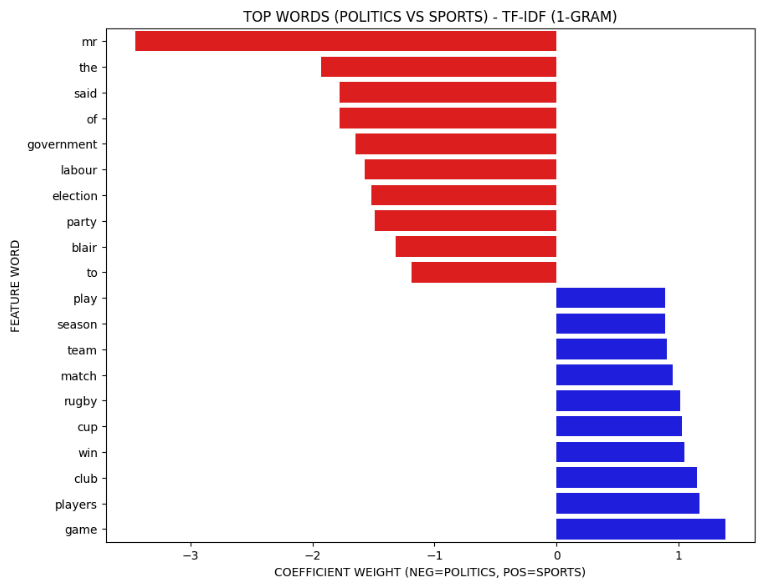

Figure The most influential words for the Logistic Regression model. Red bars indicate words strongly associated with Politics, while Blue bars indicate words strongly associated with Sports.

7. LIMITATIONS OF THE SYSTEM

1. **Dataset Size:** The high accuracy (99%) is partly due to the distinct nature of the topics and the limited size of the test set (186 samples). On a massive real-world stream of news, edge cases (e.g an article about a politician attending a sports game) might cause errors.
2. **Static Vocabulary:** The model is trained on historical data. It may fail to recognize new proper nouns (e.g new athlete names) that appear in the future.
3. **Context:** BoW and TF-IDF ignore word order. Sentences like "The team did not win" and "The team did win" might have similar vector representations depending on how stop words are handled, potentially leading to confusion in sentiment-heavy tasks

8. REFERENCES

1. Thite, S. (2023). *Text Document Classification Dataset*. Kaggle. Available at: https://www.kaggle.com/datasets/sunilthite/text-document-classification-dataset.
2. https://scikit-learn.org/1.4/tutorial/text_analytics/working_with_text_data.html
3. https://doi.org/10.21105/joss.03021