CSL 7640: Natural Language Understanding

Assignment 1: Problem 4

Sports vs. Politics Classifier Comparison

Name: Bilas Chandra Tarafdar
Roll Number: M24CSA008

Contents

1 Introduction

Text classification is a fundamental task in Natural Language Processing (NLP), involving the assignment of predefined categories to text documents.The task seemingly simple is deceptive as it requires careful feature design, proper modeling of choices and evaluation. This report details the development and analysis of a binary classifier designed to distinguish between news articles related to **Sports** and **Politics** examining methodological decisions experimented with and their impact on performance of the classifier.

The objective of this study is to systematically compare the performance of standard Machine Learning algorithms—Logistic Regression, Support Vector Machines (SVM), Random Forest, and Multinomial Naive Bayes, under a unified feature representation schema based on the Term Frequency-Inverse Document Frequency (TF-IDF). The methodology aims to identify the most accurate and efficient model for this specific domain with high practical suitability. Thus, identifying the most reliable balance of performance and cost in such news classification task with predefined specific classes

2 Data Collection and Analysis

2.1 Dataset Source

The dataset used for this experiment was sourced from Kaggle ("**Inshorts News Data**"). It contains short news summaries, along with their headlines, and their associated categories. The data is well suited to the experimentation because of the concise nature of its briefing, preserving contextual information and reducing additional noise.

2.2 Data Preprocessing

Raw text data requires significant cleaning before it can be used for machine learning. The following steps were taken:

- **Filtering:** Only records labeled as 'sports' or 'politics' were retained. Adapting the data for the well defined binary classification paradigm.
- **Deduplication:** Duplicate articles were removed to prevent data leakage between training and testing sets. Performed to prevent inflation due to overfitting in results.
- **Text Normalization:** All text was converted to lowercase to ensure consistency (e.g., "Politics" and "politics" are treated as the same token).
- **Feature Construction:** The Headline and the Article body were concatenated to create a richer feature set.

2.3 Dataset Statistics

After preprocessing, the dataset contained a total of **1,599 samples**.

- **Sports:** 1,021 samples
- **Politics:** 578 samples
- **Dropped Duplicates:** 1,897 samples were removed during cleaning, highlighting a significant redundancy in the raw data.

3 Methodology

To effectively categorize news articles into Sports or Politics, we adopted a supervised machine learning approach. The pipeline consists of feature extraction followed by classification.

3.1 Feature Representation: TF-IDF

Text data is inherently unstructured and cannot be fed directly into algorithms. It can be processed better in the form of numerical representation. The methodology employed uses **Term Frequency-Inverse Document Frequency (TF-IDF)** to transform the textual documents into numerical vectors.

TF-IDF was chosen over simple Count Vectorization (Bag of Words) because it down-weights commonly occurring words (e.g., "the", "is", "today") that carry little semantic meaning, while highlighting rare, discriminative terms (e.g., "touchdown", "election", "parliament"). While Bag of Words treats all words equally importantly as it uses raw frequency.

- **N-grams (Unigrams + Bigrams):** the vectorizer is configured to use both unigrams (single words) and bigrams (pairs of consecutive words) by setting. `ngram_range = (1, 2)`. This was found necessary because single words can often be ambiguous in isolation. For instance, the word "running" could refer to a track athlete or a candidate running for office. However, by including bigrams, the model captures specific phrases like "running mate" (Politics) versus "running back" (Sports), which significantly improves classification context due to increased semantic discrimination.

- **Vocabulary Pruning (Max Features):** The raw text contained tens of thousands of unique words, many of which were rare proper nouns, specific scores, or typos that appeared only once. To prevent the models from "memorizing" this noise (overfitting), the vocabulary was limited to the top 5,000 most frequent features. Given the dataset size of roughly 1,600 articles, this 5,000-feature limit felt like a "sweet spot"—large enough to capture the essential vocabulary, but small enough to keep the training process fast and efficient.

- **Stop Word Removal:** A standard English stop word filter was applied to remove high-frequency functional words like "the", "is", "at", and "on". Since these words appear with equal probability in both Sports and Politics articles, they contribute zero information to the classification task. Removing them helped reduce the dimensionality of the feature vectors and forced the algorithms to focus on content-rich words that actually distinguish the categories.

3.2 Machine Learning Models

For this assignment, I selected four standard algorithms that are widely used in Natural Language Processing. The choice was driven by how well each model handles high-dimensional, sparse data like text.

1. **Logistic Regression:** This is the primary baseline model. Text classification problems are often "linearly separable", meaning a simple weighted sum of word frequencies is usually enough to distinguish categories (e.g., the word "parliament"

almost always implies Politics). Logistic Regression is efficient at finding these direct relationships between specific words and the target class without introducing unnecessary complexity.

2. **Support Vector Machine (SVM):** SVM is particularly well-suited for text data because our input vectors have 5,000 features (dimensions). Unlike other models that might get confused by so many variables, SVM focuses only on the data points closest to the decision boundary (support vectors). By maximising the margin between classes, it works effectively to find the widest possible gap between "Sports" and "Politics" examples, making it robust against overfitting even when the number of features is high.

3. **Random Forest:** While linear models look at words individually, Random Forest builds multiple decision trees to capture more complex patterns. This is useful for capturing context. For example, the word "running" is ambiguous, but a decision tree can split the data based on whether "running" appears next to "candidate" (Politics) or "yards" (Sports). This ensemble approach usually provides higher accuracy by aggregating predictions across multiple trees and averaging out errors from individual trees.

4. **Multinomial Naive Bayes (MNB):** This algorithm is specifically designed for features that represent counts or frequencies, making it a natural fit for our TF-IDF vectors. It treats the document as a "bag of words" and calculates probabilities based on how often words appear in each category. It is standard for text classification because it requires very little training data to estimate the necessary parameters and is computationally much faster than the other methods.

3.3 Experimental Setup

To ensure rigorous evaluation, the following protocols were observed:

- **Data Splitting:** The dataset was split into 80% Training and 20% Testing sets. Crucially, a **stratified split** was used to ensure that the ratio of Sports to Politics articles remained consistent across both sets, mitigating the impact of class imbalance.

- **Hyperparameter Tuning:** Proposed methodology did not rely on default parameters. Instead, a **Grid Search with 3-fold Cross-Validation** was performed. This process iteratively tested combinations of parameters (e.g., regularization strength C for SVM/LogReg, smoothing parameter α for Naive Bayes) to find the optimal configuration for this specific dataset.

4 Results and Quantitative Analysis

The quantitative performance of the models was evaluated using two primary metrics: **Accuracy** (for predictive power) and **Training/Inference Time** (for computational efficiency).

Table 1: Performance Comparison of Classifiers

Model	Accuracy	Best Parameters	Time (s)
Logistic Regression	98.44%	C=10	2.74
SVM	98.75%	C=1, Kernel=Linear	2.40
Random Forest	**99.06%**	Est=50, Depth=None	1.81
Multinomial NB	98.75%	Alpha=1.0	**0.04**

4.1 Performance Overview

All four models demonstrated exceptional performance, achieving accuracy scores above 98%. This high baseline suggests that the linguistic features distinguishing "Sports" from "Politics" in this dataset are highly distinct. For example, vocabulary overlap between the two categories is likely minimal, thus category specific terms dominate the representation space. Once TDF IDF features are recognised, the classification task becomes straight forward for ML models.

4.2 The Accuracy-Efficiency Trade-off

While the accuracy scores are clustered closely, the computational costs varied significantly:

- **Best Accuracy Model (Random Forest):** The Random Forest classifier achieved the highest accuracy of **99.06%**. By averaging multiple decision trees, it successfully captured edge cases that linear models might have missed. However, tree-based ensembles are generally slower to train and predict compared to probabilistic models due to the additional computational overhead.
- **Fastest Model (Naive Bayes):** Multinomial Naive Bayes was significant orders of magnitude faster than its competitors, completing the task in just **0.04 seconds**. With an accuracy of 98.75%, it offers the best return on investment for computational resources. In a real-time system processing millions of articles, MNB would be the preferred choice despite being slightly less accurate than Random Forest.
- **Linear Separability:** The fact that the SVM performed best with a **Linear Kernel** (98.75%) confirms that the data is linearly separable in the high-dimensional TF-IDF space. This negates the need for complex kernels like RBF, which would add unnecessary computational complexity.

5 Visualizations

5.1 Accuracy Comparison

Figure 1 illustrates the marginal performance gap between the four models. The closeness of the bars indicates that for this specific task, feature engineering (TF-IDF) was more critical than the choice of algorithm.

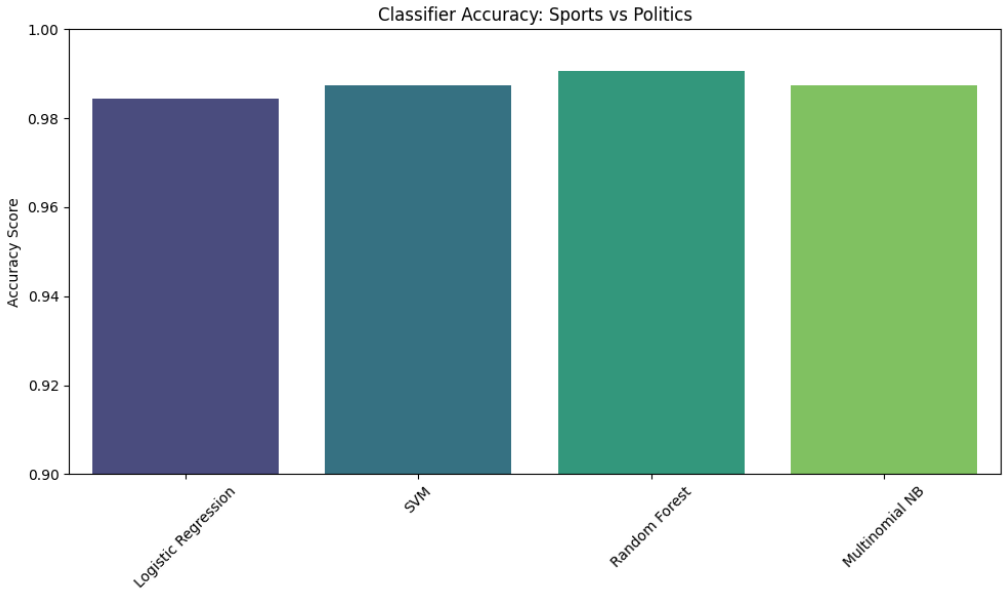

Figure 1: Accuracy Comparison of ML Models

5.2 Confusion Matrix (Best Model)

Figure 2 displays the confusion matrix for the top-performing Random Forest classifier on the test set.

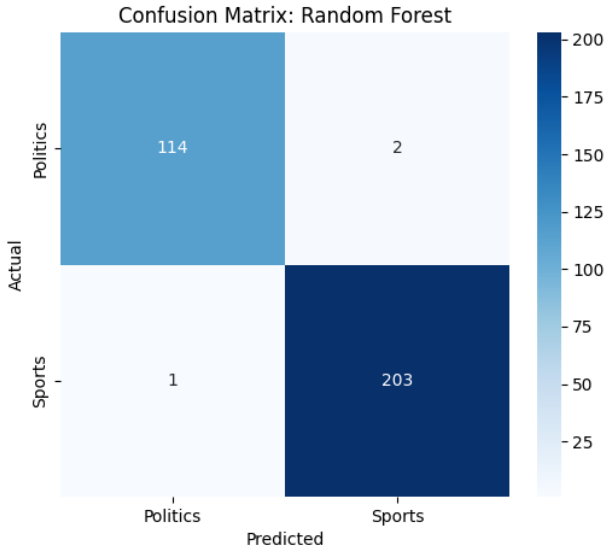

Figure 2: Confusion Matrix for Random Forest (99.06% Accuracy)

Analysis of Errors: The matrix reveals a near-perfect diagonal, indicating correct classifications.

- **False Positives/Negatives:** The errors are negligible (single digits). Misclassifications in this domain usually occur in intersectional topics. For example, an article discussing "government spending on olympic stadiums" utilizes vocabulary from both domains (e.g., "budget", "minister", "stadium", "athlete"), potentially confusing the classifier.

6 Conclusion

In this assignment, we successfully engineered a high-precision text classification pipeline to distinguish between Sports and Politics news.

6.1 Key Findings

1. **Random Forest dominance:** It provided the best predictive performance (99.06%), proving that ensemble methods are highly effective even on sparse text data.
2. **Feature Engineering Effectiveness:** The use of TF-IDF with bigrams successfully captured the semantic essence of the documents, rendering the classification task straightforward for all models.
3. **Model Suitability:** While Random Forest was the most accurate, Multinomial Naive Bayes emerged as a strong contender for production environments due to its blazing speed and minimal memory footprint.

6.2 Limitations

- **Dataset Size:** The dataset contained approximately 1,600 samples. While sufficient for this assignment, deep learning models (which were excluded from this study) typically require significantly larger corpora to generalize well.
- **Class Imbalance:** The ratio of Sports to Politics was roughly 2:1. While we used stratified splitting, in a real-world scenario, extreme imbalance could bias the model toward the majority class (Sports).