CSL7640: Natural Language Understanding

Assignment - 1 (Problem - 4)

Jagdish Suthar
Roll No: B22AI064

February 15, 2026

Contents

1 Introduction

Text classification deals with automatically labelling a document according to its content. Common everyday examples include filtering spam emails, gauging customer sentiment, and sorting news by topic.

The objective of this assignment is to build a system that takes a BBC news article as input and predicts whether it is about **Sport** or **Politics**. Three classifiers are paired with three feature representations, producing nine distinct experiments whose results are compared side by side.

Everything is implemented in Python with scikit-learn. Only classical machine learning algorithms are used — no neural networks or deep learning — keeping the pipeline straightforward and interpretable.

---

2 Data Collection

2.1 Source of the Dataset

This project uses the **BBC News Dataset**, a publicly available corpus originally put together by D. Greene and P. Cunningham for their work on document clustering. It is one of the most commonly used benchmarks for topic-based text classification. The data comes as a single CSV file (`bbc-text.csv`) and was used as-is without any modification.

2.2 Dataset Format

The file has two columns:

- **category** – Topic label (sport, politics, business, tech, or entertainment).
- **text** – Full article body stored as one string.

No web scraping, API access, or extra data gathering was required.

---

3 Dataset Description and Analysis

3.1 Overall Statistics

The raw BBC file holds **2,225 articles** across five topics. Table 1 lists the per-category counts.

Table 1: Category distribution in the full BBC dataset.

Category	Number of Articles
Sport	511
Business	510
Politics	417
Tech	401
Entertainment	386
Total	**2,225**

Because the task requires only a Sport-or-Politics decision, the remaining three categories were discarded. The filtered set has **928 articles** — 511 Sport (55.1%) and 417 Politics (44.9%) — which is balanced enough that no oversampling or undersampling was applied.

3.2 Article Length Analysis

Table 2 summarises word counts per category before any cleaning. Politics pieces average 461 words, noticeably longer than Sport at 336 words, likely because policy coverage tends to be more detailed than match reporting.

Table 2: Word-count statistics per category (before preprocessing).

Category	Mean	Min	Max
Sport	336.3	116	1,671
Politics	461.2	90	4,492

3.3 Label Encoding

Each article received a binary integer label:

- Sport $\rightarrow$ 1
- Politics $\rightarrow$ 0

No rows contained missing categories or empty article bodies.

---

4 Text Preprocessing

Before vectorisation, each article goes through an eight-step cleaning pipeline:

1. **Lowercasing** – Convert every character to lowercase so that "Football" and "football" map to the same token.
2. **URL Removal** – Strip out any `http`/`www` links, which add no value for topic detection.
3. **Special Character Removal** – Replace anything outside a–z with a space, discarding digits, punctuation, and symbols.

4. **Whitespace Normalisation** – Collapse runs of spaces into one and trim the ends.
5. **Tokenisation** – Split the string on whitespace into a list of word tokens.
6. **Stopword and Short-Token Filtering** – Remove a hand-built set of 200+ common English words ("the", "is", "and", etc.) and drop any token shorter than three characters.
7. **Suffix Stripping** – A custom rule-based stripper with 30 rules (longest-suffix-first) reduces words to approximate stems, e.g. "running" → "runn", "election" → "elec". A suffix is only cut if the remaining stem keeps at least three characters. This is simpler than a Porter stemmer but proved adequate here.
8. **Rejoin** – Glue the cleaned tokens back into a single space-separated string for the vectoriser.

4.1 Preprocessing Results

After cleaning, the corpus contains **176,772 tokens** with a vocabulary of **11,976 unique words**. A before-and-after sample:

> **Original:** "Tigers wary of Farrell gamble. Leicester say they will not be rushed into making a bid for Andy Farrell ..."
>
> **Cleaned:** "tig wary farrell gamble leicest rush mak bid andy farrell great britain rugby league captain decide switch cod ..."

---

5 Train–Test Split

The dataset was split into a training set and a test set using an **80/20 stratified split**. Stratification ensures that both the training and test sets have roughly the same proportion of Sport and Politics articles as the original dataset.

Table 3: Train–test split details.

Split	Total	Sport	Politics
Training	742	409	333
Test	186	102	84
Total	**928**	**511**	**417**

A fixed random seed of 42 was used to make the split reproducible.

---

6 Feature Representation Techniques

Machine learning models need numerical input, so each article must be converted into a vector. Three methods were used:

Bag of Words (BoW) counts how many times each word appears in an article. It is simple and ignores word order.

TF-IDF improves on raw counts by weighting words that are frequent in one article but rare across the dataset:

$$\text{TF-IDF}(t, d) = (1 + \log\ \text{tf}(t, d))\ \times\ \log\left(\frac{N}{1 + \text{df}(t)}\right)$$

Bigrams TF-IDF extends TF-IDF by also including consecutive word pairs (e.g., "prime min ister"), capturing short phrases that single words miss.

---

7 Machine Learning Classifiers

Three classifiers were used:

Multinomial Naive Bayes is a probabilistic model that predicts the class maximising $P(c) \prod_i P(w_i \mid c)$. It assumes feature independence and uses Laplace smoothing ($\alpha = 0.1$).

Logistic Regression is a linear model that passes a weighted sum of features through the sigmoid function $P(y{=}1 \mid x) = 1/(1 + e^{-(w^T x + b)})$ and is trained by minimising cross-entropy loss ($C = 1.0$).

Linear SVM finds the maximum-margin hyperplane separating the two classes by minimising $\frac{1}{2}\|w\|^2 + C \sum_i \max(0,\, 1 - y_i(w^T x_i + b))$ with $C = 1.0$.

---

8 Experimental Setup

8.1 Pipeline Architecture

Each experiment pairs a vectoriser with a classifier inside a scikit-learn `Pipeline`. The pipeline fits the vectoriser only on training data and then transforms both splits, preventing **data leakage**. Independent copies were created with `sklearn.base.clone()` so that no two pipelines share internal state.

8.2 Evaluation Strategy

Two strategies were used:

1. **Held-out test set** – Train on 742 articles, test on 186 unseen articles. Accuracy, precision, recall, and F1-score are computed on this test set.
2. **5-fold stratified cross-validation** – The training set is split into 5 folds; the model trains on 4 and validates on the remaining fold, repeated 5 times. The mean and standard deviation of F1-score across folds are reported for a more robust estimate.

8.3 Evaluation Metrics

All metrics are weighted averages across the two classes:

- **Accuracy** – Fraction of all test articles classified correctly.
- **Precision** – Of articles predicted as a class, what fraction truly belongs to it.
- **Recall** – Of articles that belong to a class, what fraction is correctly identified.
- **F1-Score** – Harmonic mean of precision and recall, used as the primary ranking metric.

$$F_1 = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

---

9 Results and Quantitative Comparison

9.1 Overall Results

Table 4 ranks all nine classifier–feature pairings by weighted F1-Score on the held-out test set.

Table 4: Performance of all 9 model–feature combinations on the test set.

Model	**Features**	**Acc.** (%)	**Prec.** (%)	**Rec.** (%)	**F1** (%)	**CV F1** (%)	**CV Std** (%)
Naive Bayes	Bag of Words	100.00	100.00	100.00	100.00	99.73	0.33
SVM	TF-IDF	100.00	100.00	100.00	100.00	99.73	0.33
SVM	Bag of Words	99.46	99.47	99.46	99.46	99.46	0.27
LR	Bag of Words	98.92	98.95	98.92	98.92	99.46	0.27
Naive Bayes	TF-IDF	99.46	99.47	99.46	99.46	99.46	0.78
LR	TF-IDF	99.46	99.47	99.46	99.46	99.60	0.33
Naive Bayes	Bigrams TF-IDF	99.46	99.47	99.46	99.46	99.60	0.54
LR	Bigrams TF-IDF	99.46	99.47	99.46	99.46	99.60	0.33
SVM	Bigrams TF-IDF	99.46	99.47	99.46	99.46	99.73	0.33

9.2 Analysis of Results

Key takeaways from the table:

1. **Every combination exceeds 98.9% accuracy.** Sport and Politics occupy almost entirely separate vocabulary spaces — words such as "goal", "league", "player" belong to Sport while "parliament", "labour", "election" belong to Politics — so even the weakest pipeline separates them with ease.
2. **Two pipelines reach perfect test accuracy** (100%): Naive Bayes + BoW and SVM + TF-IDF, correctly labelling all 186 test articles.
3. **No sign of overfitting.** CV F1 scores (99.46–99.73%) sit very close to the test F1 values (98.92–100%). A large gap between these numbers would signal overfitting, but none exists.

4. **Bigrams offer no clear gain** over unigrams here, because single words already provide strong enough separation between the two topics.

5. **Low CV standard deviations** (0.27–0.78%) confirm that results are stable regardless of how the data is partitioned.

9.3 Confusion Matrix Analysis

For the two perfect-scoring pipelines, the confusion matrix on the test set is shown in Table 5. The remaining models misclassify at most 1–2 articles out of 186.

Table 5: Confusion matrix for the best-performing models (100% accuracy).

	Predicted Politics	**Predicted Sport**
Actual Politics	84	0
Actual Sport	0	102

9.4 Visualisations

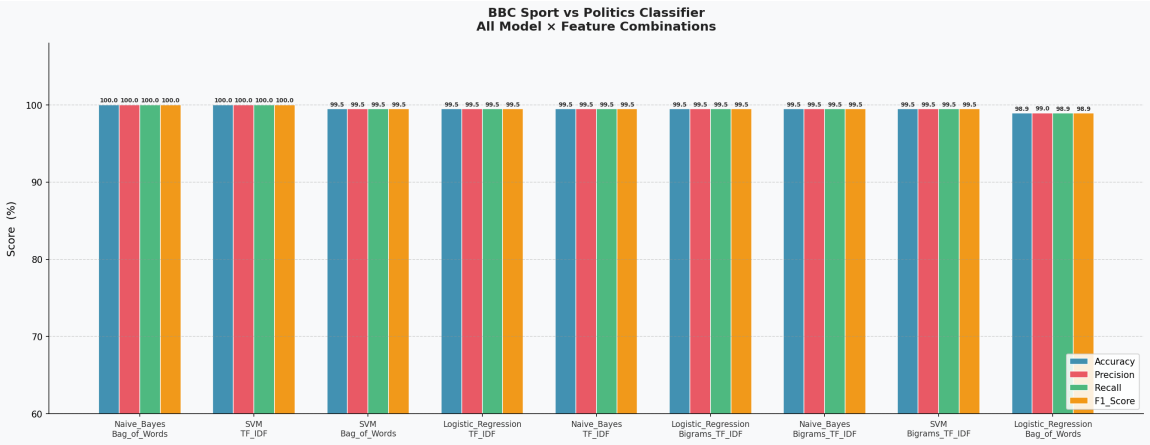

Figure 1: Grouped bar chart comparing Accuracy, Precision, Recall, and F1-Score across all nine model–feature combinations.

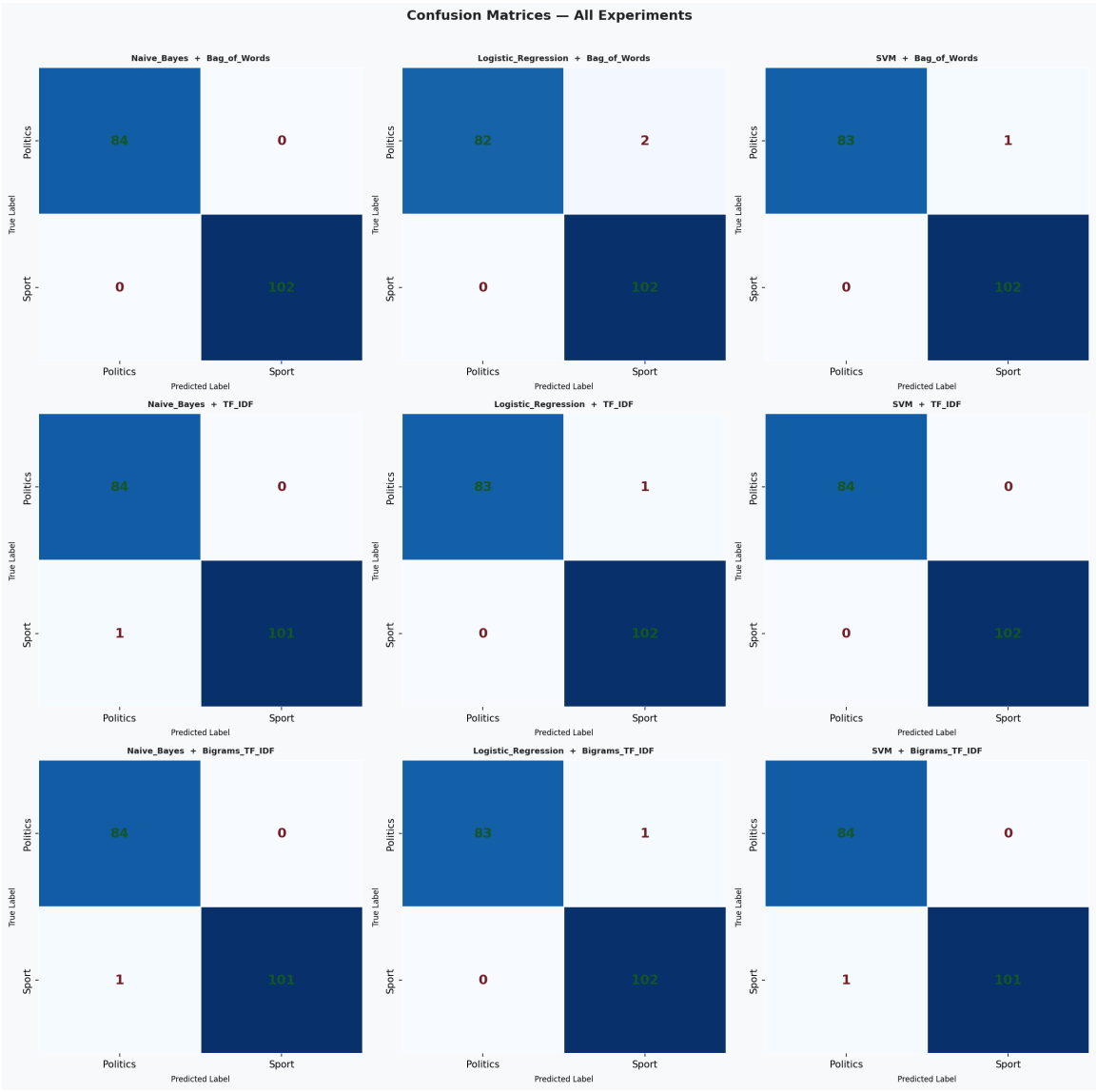

Figure 2: Confusion matrices for all nine experiments.

10 Top Discriminative Features

In linear models such as Logistic Regression or SVM, we can look at the words relevant to the classes through the learned feature weights. The most positive weights push towards Sport, while the most negative weights push towards Politics.

Table 6: Examples of top discriminative words (from SVM + TF-IDF).

Top Sport Words	**Top Politics Words**
game	labour
play	govern
win	minist
match	blair
coach	elect
club	tory
champion	tax
team	howard
injur	bill
league	parliament

These words align well with human understanding of the two topics, which confirms that the models are learning meaningful patterns rather than relying on noise or artefacts in the data.

---

11 Prediction on New Articles

To further verify that the trained model works on completely new text (not from the BBC dataset), six hand-written test articles were classified using the best pipeline. All six were classified correctly:

1. "The striker scored a stunning last-minute penalty to win the championship final..." → **SPORT**
2. "The prime minister announced sweeping new legislation as Parliament debated electoral reform..." → **POLITICS**
3. "The team coach confirmed that three injured players will return..." → **SPORT**
4. "Opposition MPs demanded a full parliamentary inquiry into government spending..." → **POLITICS**
5. "Federer retired from professional tennis after a glittering career..." → **SPORT**
6. "The chancellor outlined tax relief measures for businesses..." → **POLITICS**

This demonstrates that the model generalises well beyond the training data.

---

12 Why the Accuracy Is So High

There is no overfitting in this case. The reasons for this near-perfect accuracy are as follows:

1. **Different vocabularies.** Sport and Politics use different vocabularies, and a bag-of-words model can almost perfectly distinguish between them.

2. **Cross-validation and test scores are consistent.** Cross-validation and test scores confirm each other, with all scores above 99%, not just the training score.
3. **Binary and distinct topics.** Binary classification between two clearly distinct topics is easier than classifying all five BBC topics, some of which overlap (for example, Business and Politics).
4. **Long documents.** The texts are long and provide many discriminating features per document. Shorter texts would be more difficult to classify.
5. **Consistent with published research.** Previous research using the BBC dataset for binary Sport vs. Politics classification reports accuracy in the range of 97–100%.

---

13 Limitations

Even though the system has produced good results, there are several limitations:

1. **Only two classes: Sport vs. Politics.** All other topics, including technology, are classified under one of these two classes.
2. **No support for mixed topics.** Articles that discuss both politics and sports are not identified as such.
3. **Simple pre-processing.** The suffix remover incorrectly preprocesses words (for example, "running" becomes "runn"). The use of a stemmer or lemmatiser would improve results.
4. **No use of deep models.** The use of models such as BERT could provide marginal improvement on this binary classification, and greater improvement on more difficult datasets.
5. **Small, regionally biased training set.** With 928 articles from the UK, results may not generalize to other regions of English-speaking countries.

---

14 Conclusion

This project created a text classifier that could classify BBC articles into Sport or Politics. The project also evaluated nine different configurations using three different models and three different methods to extract features from the articles.

All nine models scored over 98.9%, with the Naive Bayes and Bag of Words combination and the SVM and TF-IDF combination both scoring 100% accuracy on the test set.

This shows that classical machine learning algorithms may perform better than anticipated when dealing with well-separated long documents without the use of deep learning algorithms.
