B23CM1016 — Problem 4 Report

Sports vs Politics — A Comparative Study of Feature Representations and Classifiers

Repository Link:
https://github.com/Faizimdad07/NLU_Assignment1

Abstract

This report describes the design, implementation, and evaluation of a text classification system that discriminates between sports and politics articles. The dataset used is a subset of the BBC Full Text dataset (Kaggle), where only the "politics" and "sport" categories were extracted. I implemented three feature representations (Bag-of-Words, TF-IDF, and n-grams) and compared multiple classification techniques implemented from scratch: Multinomial Naive Bayes, k-Nearest Neighbors (k-NN using cosine similarity), and a simple Logistic Regression trained with stochastic gradient descent (SGD). All code is self-contained in `B23CM1016_prob4.py`. Experimental results are reported and analyzed; the report concludes with limitations, reproducibility instructions, and recommendations for further work.

1. Introduction

Automatic classification of text into topical categories is a central problem in natural language processing. The aim for this assignment was to compare different feature extraction methods and classifiers on a binary topic classification task: Sports vs Politics. The project emphasizes implementing standard algorithms from first principles (no external ML/NLP libraries), following standard preprocessing, and producing a reproducible pipeline and a written analysis.

2. Data Collection and Dataset Description

Source: The original dataset was taken from the BBC Full Text dataset on Kaggle. From that dataset, I extracted documents whose category was "politics" or "sport" and stored them in the workspace under:

- `dataset/train/politics`
- `dataset/train/sports`
- `dataset/test/politics`
- `dataset/test/sports`

Counts: The test set used in the experiments contains **83 politics documents** and **83 sports documents** (total = 166). The training set contains the remaining articles.

Data format: Each article is stored as a single plain-text file.

3. Preprocessing

All preprocessing was implemented in the script:

- **Lowercasing**
- **Punctuation removal**
- **Whitespace tokenization**
- **Stopword removal** (small manually defined list)

Rationale: Keeps pipeline simple and avoids removing discriminative words.

4. Feature Engineering

4.1 Bag-of-Words (BoW)

Sparse vector of token counts using vocabulary (frequency ≥ 2).

4.2 TF-IDF

[
idf(t) = log((N + 1) / (df(t) + 1)) + 1
]

Vectors are **L2 normalized**.

4.3 N-grams (Unigram + Bigram)

Bigrams created using underscore joining:

- Example: `prime_minister`, `world_cup`

5. Models and Training

5.1 Multinomial Naive Bayes (MNB)

- Laplace smoothing (α = 1.0)
- Uses log-probabilities

5.2 k-Nearest Neighbors (k-NN)

- Cosine similarity

- k = 7
- Majority voting

5.3 Logistic Regression (SGD)

- Sigmoid activation
- Cross-entropy loss
- Fixed learning rate and epochs

6. Experimental Setup

- **Vocabulary cutoff:** min_freq = 2
- **Models:**
    - MNB (α=1.0)
    - k-NN (k=7)
    - SGD Logistic (lr=0.5, epochs=5)
- **Metric:** Accuracy

Outputs saved in:

- `results_prob4.txt`
- `B23CM1016_report.txt`

7. Results (Summary)

Feature + Model	Accuracy
BoW + MNB	**1.0000**
BoW + k-NN	**0.9639**
N-gram + MNB	**1.0000**
TF-IDF + SGD	**0.5000**
TF-IDF + k-NN	**0.9940**

8. Analysis and Discussion

8.1 Perfect Accuracy Explanation

- Strong domain vocabulary separation
- Small dataset
- Bigrams capture strong topical phrases

8.2 Poor Logistic Regression Performance

- No regularization
- Poor hyperparameters
- Sparse high-dimensional features

8.3 Overfitting Risk

- Possible lexical memorization
- No cross-validation
- Needs shuffled splits

8.4 Classifier Trade-offs

Model	Strength	Weakness
MNB	Fast, effective	Assumes independence
k-NN	Intuitive, strong TF-IDF	Slow at test time
SGD Logistic	Generalizable	Needs tuning

9. Limitations

- No hyperparameter tuning
- Simplified preprocessing
- No cross-validation
- No class imbalance handling
- Dataset dependency

10. Reproducibility

Files

- `B23CM1016_prob4.py`
- `results_prob4.txt`
- `B23CM1016_report.txt`

Run

python3 B23CM1016_prob4.py

11. Future Work

- Add lemmatization
- Cross-validation

- Regularization
- Feature selection
- Advanced classifiers (SVM, Transformers)
- Out-of-domain evaluation

12. Conclusions

Simple models like **Naive Bayes** and **k-NN** perform extremely well on clearly separable text datasets. Logistic regression requires better tuning. The study highlights the importance of preprocessing, feature engineering, and model selection.

Appendix A — Quantitative Summary

- BOW + MNB: **1.0000**
- BOW + kNN: **0.9639**
- NGRAM + MNB: **1.0000**
- TFIDF + SGD: **0.5000**
- TFIDF + kNN: **0.9940**

Appendix B — Suggested GitHub Structure

README.md
data/
code/
results/
report/

Acknowledgements

This work uses the BBC Full Text dataset available on Kaggle. All models were implemented from scratch using Python standard libraries.