NLU Assignment-01
Problem-04 : SPORTS OR POLITICS

Sangini Garg (B22CS045)

February 13, 2026

1 Introduction

Text classification is one of the most fundamental tasks in Natural Language Processing (NLP). It involves automatically assigning predefined categories to textual documents. Applications of text classification include spam detection, sentiment analysis, topic categorization, recommendation systems, and information retrieval.

In this project, we design a document classifier that determines whether a given news article belongs to the domain of **SPORT** or **POLITICS**. This is a binary topic classification problem. The objective is to evaluate the effectiveness of classical machine learning algorithms when applied to textual data represented using statistical features.

Unlike modern deep learning approaches that require large computational resources, traditional machine learning methods remain highly effective for sparse high-dimensional text data. Therefore, we compare three widely used supervised learning algorithms:

- Naive Bayes
- Logistic Regression
- Support Vector Machine (SVM)

The models are trained using TF-IDF n-gram features extracted from real news articles. Their performance is evaluated quantitatively using accuracy, precision, recall and F1-score.

2 Data Collection

The dataset used in this project is the **20 Newsgroups dataset**, which is publicly available through the Scikit-learn library. Instead of manually downloading and preprocessing files, the dataset was programmatically loaded using the `fetch_20newsgroups` function. This ensures reproducibility and portability of the project across different systems.

The original dataset contains approximately 20,000 documents distributed across 20 different categories such as computers, religion, science, politics and sports.

For the purpose of this assignment, only four categories were selected:

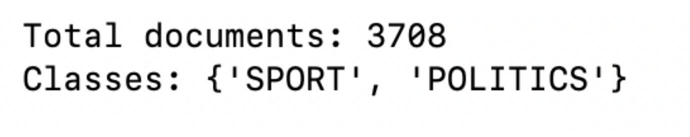
- rec.sport.baseball
- rec.sport.hockey
- talk.politics.mideast
- talk.politics.misc

These categories were further merged into two super-classes:

Original Category	Final Label
rec.sport.baseball	SPORT
rec.sport.hockey	SPORT
talk.politics.mideast	POLITICS
talk.politics.misc	POLITICS

This conversion creates a binary classification problem while preserving real-world textual complexity.

During dataset loading, headers, email signatures and quoted replies were removed. These components often contain metadata rather than linguistic content and may introduce noise into the model.



```
Total documents: 3708
Classes: {'SPORT', 'POLITICS'}
```

Figure 1: Dataset Loaded - Terminal showing Total documents classes

3 Dataset Description and Analysis

The final dataset contained **3708 documents**. The data was split into training and testing sets using an 80-20 ratio.

Characteristics of the Dataset

- Text length varies significantly (short opinions to long discussions)
- Informal language present in sports discussions
- Argumentative tone common in political discussions
- Overlapping vocabulary (e.g., team, government, win, support)

Observations

- Sports articles typically contain: - Team names - Scores - Player names - Match descriptions
- Political articles often contain: - Countries - Policies - Conflicts - Debates and opinions
- However, some words such as support, fight, win, lead, and campaign appear in both domains, making classification non-trivial.

4 Text Preprocessing

Raw text cannot be directly used by machine learning models. Therefore, preprocessing and feature extraction were required before training the classifiers.

The following steps were performed:

- Lowercasing handled automatically by the vectorizer
- Stop-word removal (common English words removed)
- Tokenization performed implicitly by the TF-IDF vectorizer
- Bigram extraction to capture contextual word meaning

Heavy linguistic preprocessing such as stemming or lemmatization was intentionally avoided to preserve interpretability and reduce additional parameters in the system.

5 Feature Representation

We used TF-IDF (Term Frequency – Inverse Document Frequency) representation with unigram and bigram features.

Bag-of-Words counts word frequency but treats all words equally. However, common words such as "the" and "is" appear frequently but carry little meaning.

5.1 Why TF-IDF

TF-IDF assigns importance using:

- TF: frequency of a word in a document
- IDF: rarity of the word across all documents

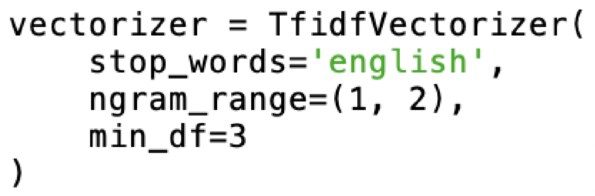
Thus informative words receive higher weight while common words receive lower weight.

5.2 Why Bigrams?

Single words sometimes lack meaning. For example:

- "white" (ambiguous)
- "white house" (clearly political)

Hence, bigrams improve contextual understanding.



```
vectorizer = TfidfVectorizer(
    stop_words='english',
    ngram_range=(1, 2),
    min_df=3
)
```

Figure 2: Vectorizer code snippet

TF-IDF (Term Frequency – Inverse Document Frequency) with unigram and bigram features was used. TF-IDF assigns lower weight to common words and higher weight to informative words.
Bigrams help capture context such as “white house” which strongly indicates politics.

6 Machine Learning Algorithms

6.1 Naive Bayes

Naive Bayes assumes conditional independence between features. Though unrealistic, this assumption works surprisingly well for text because documents are represented as word occurrence patterns.

Advantages: - Fast training - Works well with sparse data - Strong baseline for text classification

6.2 Logistic Regression

Logistic Regression is a linear classifier that learns feature weights using probability optimization.

Advantages: - Interpretable weights - Handles correlated features better than Naive Bayes

6.3 Support Vector Machine

SVM finds a maximum-margin separating hyperplane between classes in high-dimensional space.

Advantages: - Effective for sparse high-dimensional data - Robust against overfitting - Often best traditional method for text classification

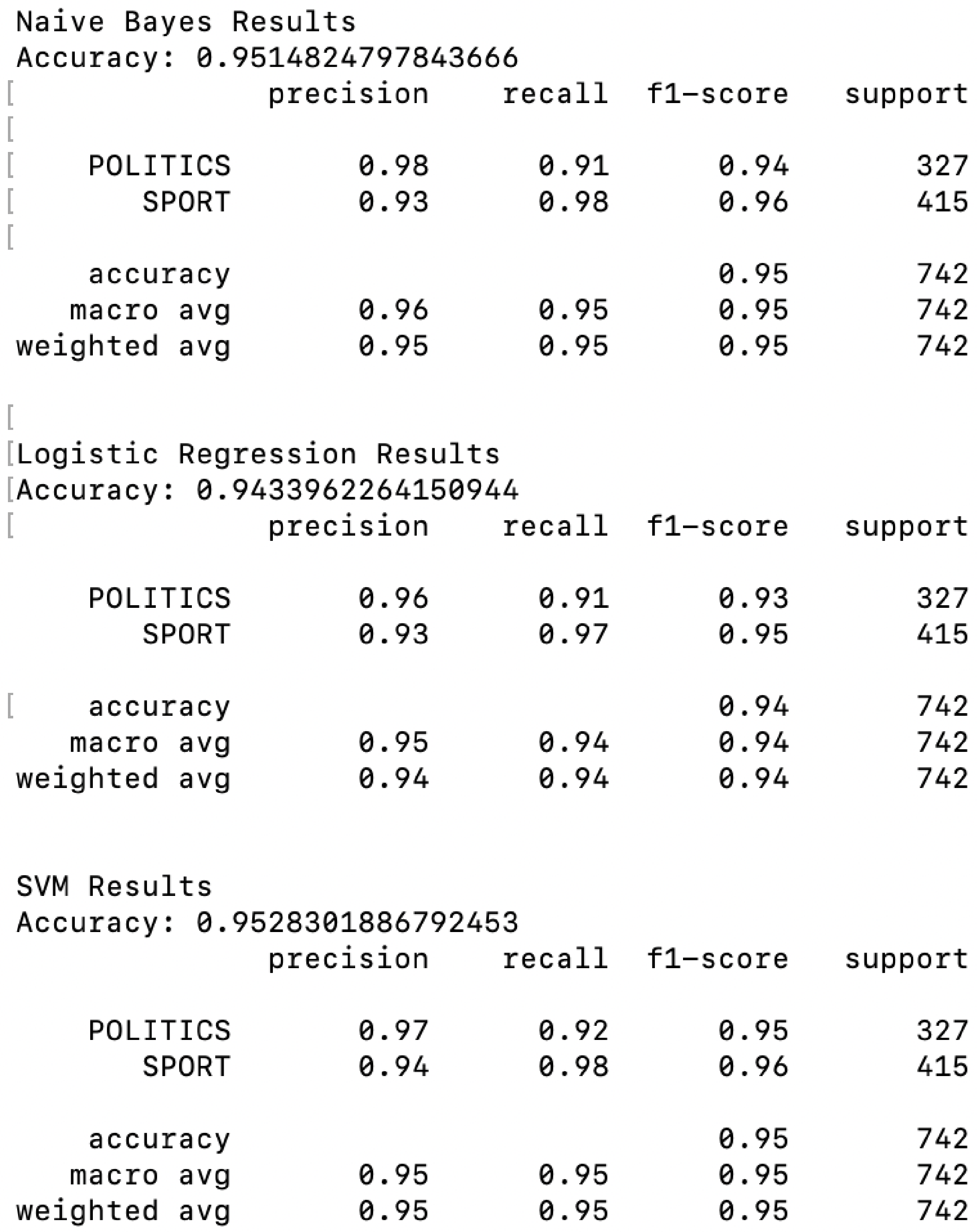
7 Experimental Setup

- Train-test split: 80-20
- Feature extraction: TF-IDF (unigram + bigram)
- Minimum document frequency: 3
- Stop-words removal
- Same features used for all models to ensure fair comparison

Evaluation metrics: Accuracy, Precision, Recall and F1-score.

8 Results and Quantitative Comparison

Model	Accuracy
Naive Bayes	95.15%
Logistic Regression	94.34%
Support Vector Machine	95.28%



```
 Naive Bayes Results
 Accuracy: 0.9514824797843666
[              precision    recall  f1-score   support
[
[    POLITICS       0.98      0.91      0.94       327
[       SPORT       0.93      0.98      0.96       415
[
     accuracy                           0.95       742
    macro avg       0.96      0.95      0.95       742
 weighted avg       0.95      0.95      0.95       742

[
[Logistic Regression Results
[Accuracy: 0.9433962264150944
[              precision    recall  f1-score   support

     POLITICS       0.96      0.91      0.93       327
        SPORT       0.93      0.97      0.95       415

[    accuracy                           0.94       742
    macro avg       0.95      0.94      0.94       742
 weighted avg       0.94      0.94      0.94       742


 SVM Results
 Accuracy: 0.9528301886792453
               precision    recall  f1-score   support

     POLITICS       0.97      0.92      0.95       327
        SPORT       0.94      0.98      0.96       415

     accuracy                           0.95       742
    macro avg       0.95      0.95      0.95       742
 weighted avg       0.95      0.95      0.95       742
```

Figure 3: Full accuracy output of 3 models

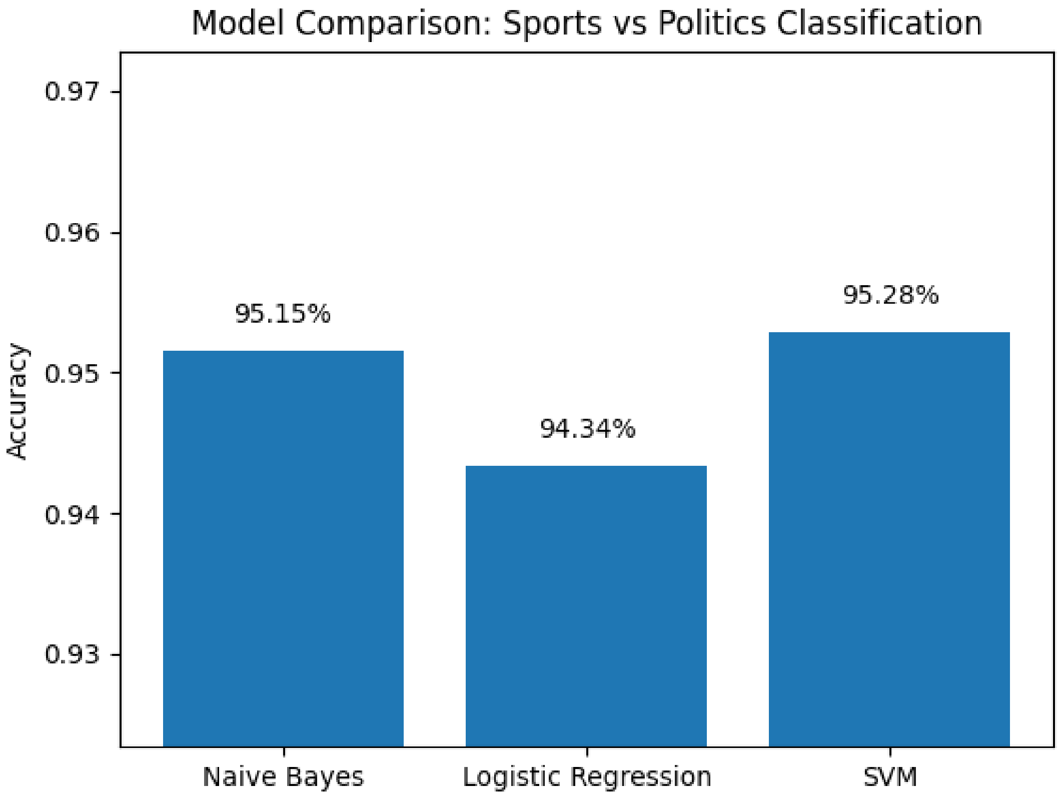


Figure 4: Generated graph - model comparison

Interpretation

- **Overall ranking of models**
  - Support Vector Machine (95.28%) achieved the highest accuracy
  - Naive Bayes (95.15%) performed very close to SVM
  - Logistic Regression (94.34%) slightly lower but consistent

9 Observations:

- **Model-wise observations**
  - SVM benefits from margin maximization in high-dimensional sparse feature space
  - Naive Bayes works well because word occurrence probabilities strongly correlate with topic
  - Logistic Regression handles feature relationships but is slightly affected by overlapping vocabulary
- **Class-wise behaviour**

  - SPORT documents classified more accurately
  - POLITICS documents show slightly lower precision and recall
  - Sports vocabulary contains specific terms such as player names, teams and scores
  - Political text shares common vocabulary with general discussions

- **Reason for confusion**
  - Overlapping words such as support, lead, win and campaign
  - Opinion-based political discussions lack clear domain keywords
  - Short documents provide insufficient context

10 Analysis

Naive Bayes performs well because word distributions in documents approximately follow independent occurrence patterns. TF-IDF enhances informative features, improving performance.

Logistic Regression considers feature correlations but struggles slightly when decision boundaries are strongly separable in sparse space.

SVM performs best because high-dimensional sparse data benefits from margin maximization. Text datasets naturally satisfy SVM assumptions, making it highly effective.

11 Limitations

- The model relies only on statistical patterns and does not understand meaning.
- Sarcasm and figurative language cannot be captured.
- Articles discussing sports politics or political controversies in sports may confuse the classifier.
- Domain shift (e.g., social media slang) reduces accuracy.
- Dataset bias affects predictions.

12 Conclusion

All models achieved above 94% accuracy using simple TF-IDF features.
The results show feature representation is often more important than model complexity.
Future work may include deep learning models such as LSTM or BERT.