

ASSIGNMENT 1 - PROBLEM 4:

SPORTS vs. POLITICS TEXT CLASSIFIER REPORT

Github Repository Link -

<https://github.com/Layaa-V/M25CSA017-NLU-A1-prob4>

Google Colab Link -

<https://colab.research.google.com/drive/1081T1pPkmnGsVrYPu1qFTirtAk68plOd?usp=sharing>

I. OBJECTIVE AND WORKFLOW -

The aim of this project is to build a binary classifier that takes a raw text document as input and predicts whether its content belongs to the domain of Sports or Politics.

To do this task, the following procedure was followed -

1. Collection and preprocessing a robust dataset of text documents.
2. Implementation of feature extraction using TF-IDF.
3. Comparing the performance of three supervised learning algorithms: Naive Bayes, Logistic Regression, and SVM.
4. Analyze the quantitative results and select the best-performing model.
5. Verification of model performance, using a user-input new document.

II. DATA COLLECTION AND DESCRIPTION -

We utilized the 20 Newsgroups dataset. This is a benchmark dataset in the machine learning community, comprising approximately 18,000 newsgroup posts on 20 topics. It offers a standardized environment for evaluating text classifiers.

• Data Selection and Filtering -

To create a binary classification problem (Sports vs. Politics), we extracted documents from specific sub-categories within the dataset.

1. Sports Class: Aggregated from *rec.sport.baseball* and *rec.sport.hockey*.
2. Politics Class: Aggregated from *talk.politics.guns*, *talk.politics.mideast*, and *talk.politics.misc*.

The data loading process included a critical cleaning step:

`remove=['headers', 'footers', 'quotes']`.

This ensures that the model learns from the actual body text of the discussion rather than relying on metadata artifacts like email headers or organization names, which could lead to data leakage and artificial inflation of accuracy.

- **Dataset Statistics -**

After loading and cleaning the data (removing empty rows), the final dataset consisted of 4,494 unique documents.

1. Total Documents: 4,494
2. Class Distribution:
 - Politics: 2,561 documents
 - Sports: 1,933 documents
3. While there is a slight class imbalance (with more Politics documents than Sports), the dataset is sufficiently balanced to train robust models.

III. METHODOLOGY -

The following steps were followed to propagate through-

- **Text Preprocessing -**

Raw text cannot be fed directly into machine learning algorithms. We employed the following preprocessing steps:

1. **Stop Word Removal:** Common English words (e.g., "the", "is", "and") were removed using the built-in English stop word list. These words contribute little semantic meaning and add noise to the model.
2. **Vectorization:** We used TF-IDF (Term Frequency-Inverse Document Frequency) vectorization. Unlike a simple Bag-of-Words (count) model, TF-IDF weighs terms based on their importance. A word that appears frequently in a document but rarely across the corpus (e.g., "touchdown" or "filibuster") receives a high weight, whereas a word appearing everywhere receives a low weight.
3. **Feature Selection:** To manage computational efficiency and reduce overfitting, we limited the vocabulary to the top 5,000 features (max_features=5000).

- **Experimental Setup -**

The dataset was split into training and testing sets using a standard 80-20 split:

- **Training Set:** 80% of data (Used to fit the models).
- **Testing Set:** 20% of data (Used to evaluate performance).
- **Random State:** Fixed at 10 to ensure reproducibility of results.

IV. MACHINE LEARNING MODELS -

We evaluated three distinct algorithms, each with different theoretical foundations:

- Multinomial Naive Bayes (MNB)
- Logistic Regression (LR)
- Support Vector Machine (Linear SVM)

• Multinomial Naive Bayes (MNB) -

- **Theory:** Based on Bayes' theorem, this probabilistic classifier assumes that features (words) are independent of each other given the class label. While the independence assumption is often violated in natural language, MNB is empirically known to perform exceptionally well on text classification tasks due to its simplicity and effectiveness with high-dimensional data.
- **Configuration:** Default parameters were used.

• Logistic Regression (LR) -

- **Theory:** A discriminative model that learns the probability of a sample belonging to a class using a logistic function (sigmoid). It explicitly models the boundary between the classes.
- **Configuration:** `max_iter=1000` was set to ensure the solver converges, as text data often results in sparse matrices that can be slower to optimize.

• Support Vector Machine (Linear SVM) -

- **Theory:** SVM constructs a hyperplane in a high-dimensional space that maximally separates the two classes. For text classification, which often results in linearly separable data in high dimensions (5000 features), a Linear Kernel is usually superior to non-linear kernels (RBF).
- **Configuration:** `kernel='linear'`.

V. EXPERIMENTAL RESULTS AND ANALYSIS -

All three models were trained on the same training set and evaluated on the same held-out test set. The metric used for evaluation was Accuracy (percentage of correctly classified documents).

(Table on next page)

- **Quantitative Comparison -**

Model	Accuracy Score
Naive Bayes	95.44%
Logistic Regression	94.44%
SVM (Linear)	94.33%

- **Analysis of Result -**

- The Multinomial Naive Bayes model achieved the highest accuracy of 95.44%
- All three models performed exceptionally well. This indicates that the features extracted via TF-IDF (words like "ball", "election", "game", "government") are highly predictive and distinctive for these two categories.
- The slight edge of Naive Bayes suggests that the dataset likely contains strong, independent keyword signals (e.g., the presence of "hockey" almost guarantees Sports) which aligns well with the model's independence assumption.

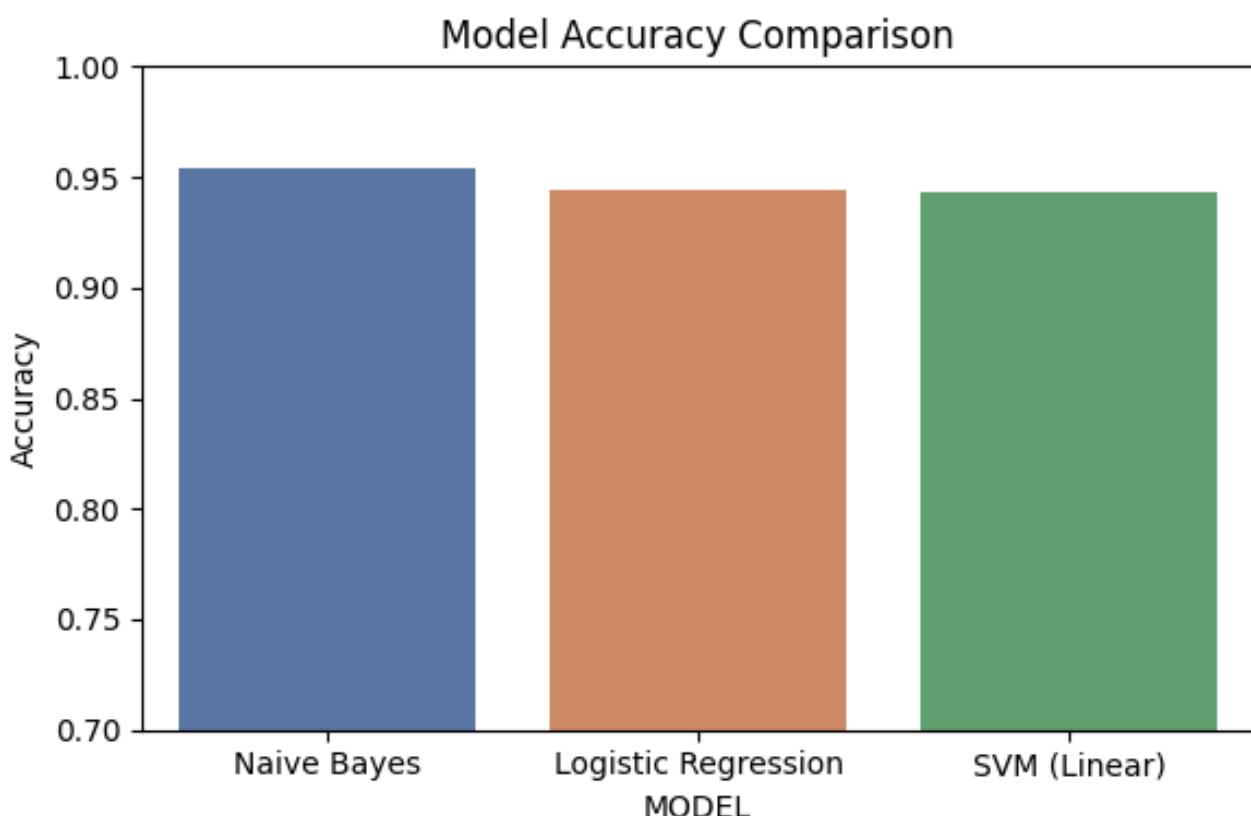


Figure produced using matplotlib in code (link given above in page 1)

VI. TESTING USING USER INPUT TEXT FILE -

The code created a interaction with the user by allowing him to input the filename of a raw text document (e.g., pol.txt, used in the code), which the system then reads, vectorizes (using the fitted TF-IDF vocabulary), and classifies using the best-performing model (Naive Bayes).

Following points show the details of the user-input test case -

- **Input File:** pol.txt (a small txt file which I created on my own by writing random political references).
- **Content:** A text document discussing political themes.
- **System Prediction:** Politics
- **Result:** The model successfully identified the context of the unseen document.

VII. LIMITATIONS -

- **Static Vocabulary:** The model is limited to the top 5,000 words learned during training. If a new document uses evolving slang or entirely new political terminology not present in the training data, the model might miss important context.
- **Ambiguity:** The classifier might struggle with documents that bridge both topics, such as an article about "legislation on sports betting" or "government funding for the Olympics," as these contain vocabulary from both domains.