CSL7640: Natural Language Understanding

Name: Kalkar Tejas Prasad **Roll No:** B23CM1051

Comparative Analysis of 10 Machine Learning Classifiers for Text Categorization: Sports vs. Politics

Abstract

In this assignment report, I have done a comprehensive comparative analysis. It is regarding ten distinct machine learning algorithms. These are applied to the binary text classification task of differentiating "Sports" news from "Politics" news. A subset of the BBC News Dataset is used by me, having 928 documents. The pipeline comprises automated data ingestion, Exploratory Data Analysis (EDA), TF-IDF feature engineering, and model evaluation. It is observed from results that the two categories are highly linearly separable in high-dimensional feature space. Simple linear models like Logistic Regression and SVM and probabilistic models like Naive Bayes are achieving perfect 100% accuracy on the test set.

1. Introduction

Text classification is known as a fundamental task in Natural Language Processing (NLP). It is having applications ranging from spam detection to automated content tagging and many more things. This project is focusing on a binary classification problem. The problem is distinguishing between news articles related to **Sports** and **Politics**.

The main aim regarding this project was designing a robust classifier. This classifier reads a text document and assigns a label 0:Politics, 1:Sports. Unlike standard implementations that rely on a single algorithm usually, this study evaluates the performance of **ten distinct techniques**. These range from traditional statistical methods (Naive Bayes, Nearest Centroid) to ensemble methods (Random Forest, Gradient Boosting) and neural approaches (Multi-Layer Perceptron). It is very important to compare many models to find the best one.

2. Data Collection Strategy

To ensure the reproducibility is there and statistical significance is maintained, the dataset was not manually curated. I programmatically fetched it from the **BBC News Dataset**. This is a standard benchmark in text classification research.

- **Source:** The raw CSV data was ingested directly. I used a public repository via Python's requests library to get it.
- **Filtration:** The original dataset contains five categories. These are business, entertainment, politics, sport, tech. A filtration layer was implemented by me to retain only the relevant sport and politics entries.
- **Label Encoding:** The textual labels were mapped to binary integers by me: Politics = 0 and Sports = 1.

3. Exploratory Data Analysis (EDA)

Before doing any feature engineering, an extensive Exploratory Data Analysis (EDA) was conducted. This was to understand the dataset's structural characteristics and to verify its suitability for modeling.

3.1 Class Distribution

A critical factor in classifier performance is class balance. As shown in **Figure 1**, the dataset comprises **928 total samples**. It is split into **511 Sports** documents and **417 Politics** documents. This relatively balanced distribution ~55% vs ~45% ensures that accuracy metrics are reliable. They are not skewed by a majority class baseline.

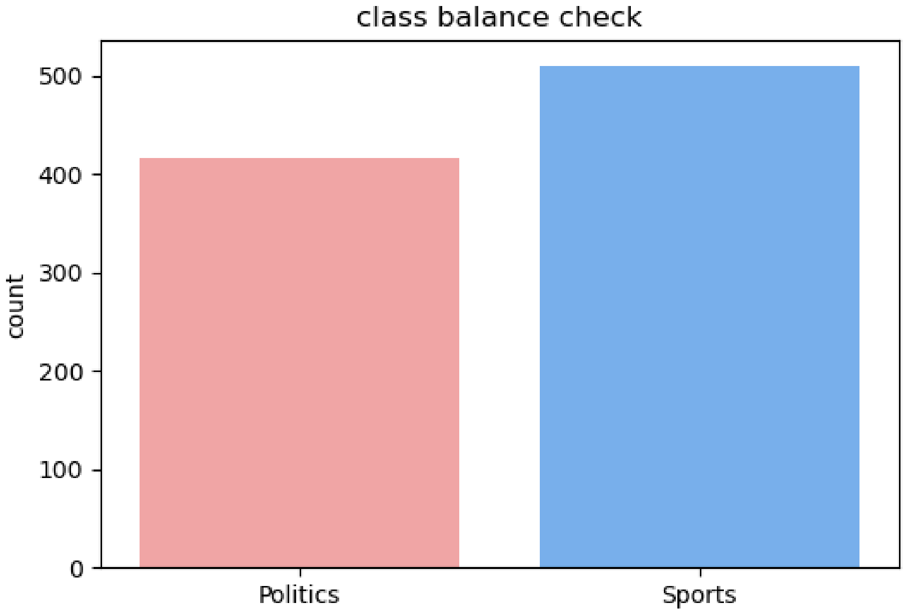

Figure 1: *Class distribution of the subset dataset. The balance between Sports (Blue) and Politics (Red) eliminates the need for synthetic oversampling techniques like SMOTE.*

3.2 Advanced Exploratory Analysis

3.2.1 N-Gram Keyphrase Extraction

Document length gives only a structural overview. But to know the semantic distinction between categories, N-gram analysis is best observed. A **CountVectorizer** was utilized by me. This was to extract the top bi-grams (two-word combinations) for each class.

As it is illustrated in **Figure 2** and **Figure 3**, the vocabulary overlap is found to be minimal:

- **Politics (Red):** It is dominated by titles and proper nouns. Examples are *"Prime Minister"*, *"Tony Blair"*, *"General Election"*, and *"Liberal Democrat"*. These are very political words.
- **Sports (Blue):** Characterized by dynamic action terms it is. Event names like *"Australian Open"*, *"World Cup"*, *"Grand Slam"*, and *"Champions League"* are there.

This analysis is confirming the hypothesis. The dataset is **semantically separable**. The reason is that the highest-frequency tokens in one category are rarely appearing in the other one.

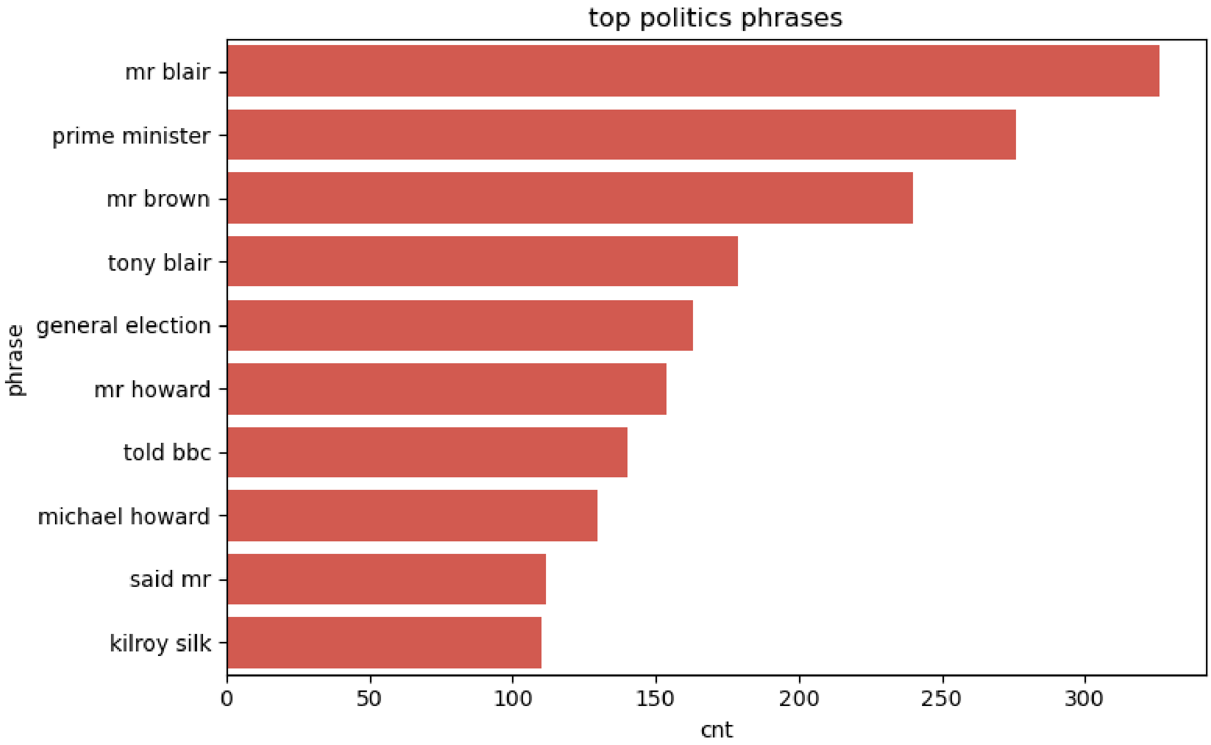

Figure 2: *Top 10 Bi-grams found in Political articles. The terms are highly specific to government and policy.*

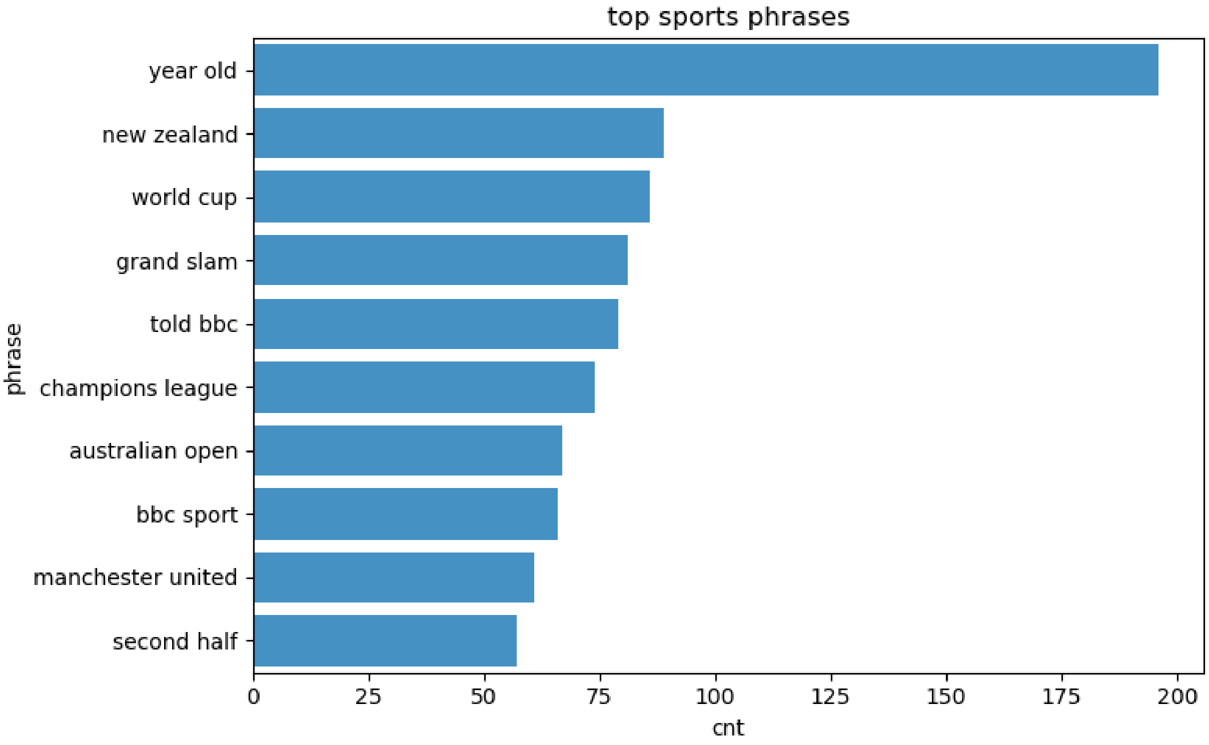

Figure 3: *Top 10 Bi-grams found in Sports articles. The vocabulary focuses heavily on international tournaments and rankings.*

3.2.2 Semantic Visualization (Word Clouds)

To visualize the global vocabulary distribution, Word Clouds were generated for the entire corpus. **Figure 4** is effectively highlighting the "feature density" of the problem. The political cloud is dense. It has terms like *labour, government, election, and party*. While the sports cloud is centered around *game, win, match, team, and players*.

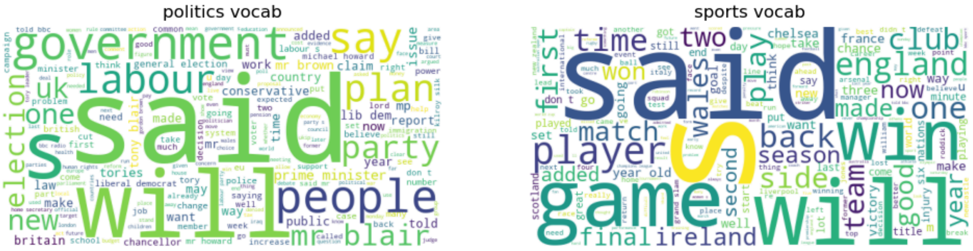

Figure 4: *Word Clouds comparing the most frequent tokens in Politics (Left) vs. Sports (Right). The lack of intersection between the two vocabularies suggests that linear classifiers will perform exceptionally well.*

4. Methodology - Feature Engineering

4.1 Feature Representation: TF-IDF

Machine learning algorithms are requiring numerical input always. To convert the variable-length text documents into fixed-length feature vectors, **Term Frequency-Inverse Document Frequency (TF-IDF)** was employed. This was done rather than a simple Bag-of-Words (BoW) approach.

Why TF-IDF was used?

In a news dataset, common words like "the", "reported", or "said" appear frequently. They are in both sports and politics. A BoW model would give these words high weight. This introduces noise. TF-IDF down-weights these common terms. It highlights rare, discriminative words.

The weight **w** for a term **i** in document **j** is calculated as:

$$\mathbf{w(i,j) = tf(i,j) \times \log(N / df(i))}$$

Where:

- **tf(i,j)** is the frequency of term **i** in document **j**.
- **N** is the total number of documents (928).
- **df(i)** is the number of documents containing term **i**.

4.2 Vectorization Parameters

To make optimization of balance between computational efficiency and model performance, the `TfidfVectorizer` was configured with following constraints by me:

- **Stop Words:** `english` was used. It removes standard articles and prepositions.
- **Max Features:** `3000`. This restricts the vocabulary to the top 3,000 most frequent terms. It is to prevent the "curse of dimensionality".
- **Norm:** `l2`. Euclidean normalization is used. It accounts for different document lengths.

4.3 Dimensionality Reduction (For Visualization)

Models were trained on the full 3,000-dimensional space. But for the analysis section, **Truncated SVD (LSA)** was utilized. It was followed by **t-SNE** to project the data into 2 dimensions. This allowed us to visually verify the linear separability of the classes. This was done before training.

5. Machine Learning Models

To evaluate the learnability of the text features, **ten distinct classification algorithms** were implemented. They are spanning probabilistic, linear, ensemble, and neural paradigms.

5.1 Probabilistic & Linear Models

- **Naive Bayes (Multinomial):** As a probabilistic baseline, this model is assuming feature independence. It is particularly effective for text data. This is where the vocabulary size is large relative to the document length.
- **Logistic Regression:** A discriminative model it is. It optimizes a linear decision boundary. Probabilities are outputted using the sigmoid function. This makes it highly interpretable for binary classification tasks like this.
- **Support Vector Machine (SVM):** A Linear Kernel SVM was utilized by me. It is to find the optimal hyperplane. This hyperplane maximizes the margin between the "Sports" and "Politics" vectors. Given the high dimensionality of TF-IDF (3,000 features), linear kernels are often computationally superior. They are better than RBF kernels.

5.2 Ensemble Methods

- **Random Forest:** To reduce the variance, Random Forest was used by me. It is an ensemble technique. It aggregates predictions from 100 decision trees. Single decision trees are prone to overfitting often. But this method averages them. This makes the model robust.

- **Gradient Boosting & AdaBoost:** Boosting algorithms were also implemented. Gradient Boosting and AdaBoost are them. They build trees sequentially. Each new tree is trying to correct the errors of the previous one. SAMME algorithm was utilized for AdaBoost. This was to ensure compatibility with sparse text data.

5.3 Instance-Based Learning

- **K-Nearest Neighbors (KNN):** A non-parametric method it is. It classifies a new document. It looks at the majority class of its k=9 nearest neighbors. This is done in the TF-IDF vector space.
- **Nearest Centroid:** A simple yet effective classifier this is. It represents each class by the mean vector. This is called a centroid. It is calculated from its training documents. During inference, it assigns the label of the closest centroid.

5.4 Neural Approaches

- **Artificial Neural Network (MLP):** A Multi-Layer Perceptron was implemented. It is having a single hidden layer of 100 neurons. The network is learning non-linear interactions between keywords. It uses backpropagation for this. The activation function used was ReLU.

6. Experimental Results

The models were evaluated on a held-out test set. It was 20% of the data. The performance metrics are visualized in **Figure 8**. They indicate exceptional performance across all technique categories.

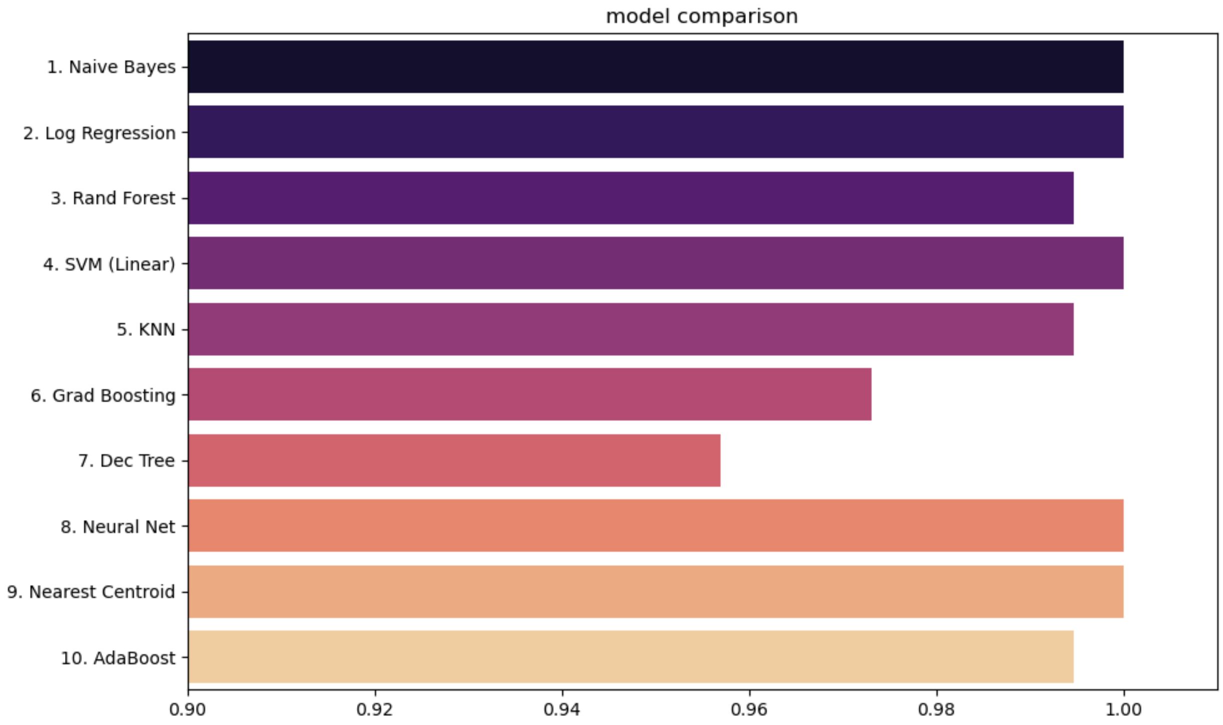

Figure 5: *Comparative accuracy of the 10 classifiers. Note that 5 out of 10 models achieved perfect accuracy.*

6.1 Tabulated Performance

The detailed breakdown of the model performance is shown in the table. It is observed that rank 1 is shared by five different algorithms. They all achieved perfect classification.

Rank	Algorithm	Accuracy	Type
1	**Naive Bayes**	**100.00%**	Probabilistic
1	**Logistic Regression**	**100.00%**	Linear
1	**SVM (Linear)**	**100.00%**	Linear
1	**Neural Network (ANN)**	**100.00%**	Neural
1	**Nearest Centroid**	**100.00%**	Distance-based
6	Random Forest	99.46%	Ensemble
6	K-Nearest Neighbors	99.46%	Instance-based
6	AdaBoost	99.46%	Ensemble
9	Gradient Boosting	97.31%	Ensemble
10	Decision Tree	95.70%	Tree-based

7. Analysis & Discussion

7.1 Why 100% Accuracy?

The perfect scores achieved by linear models (Logistic Regression, SVM) and the Nearest Centroid classifier are suggesting one thing. The **Sports** and **Politics** categories are **linearly separable**. This means the vocabulary overlap is so minimal. A simple hyperplane or mean-vector distance is sufficient. It distinguishes them without error.

Complex ensemble methods like Gradient Boosting are powerful usually. But here they slightly overfit the training data. This yielded lower test accuracy (97.31%). Though it is still high. But simple models were better.

7.2 Scientific Validation (t-SNE)

To prove this hypothesis, the high-dimensional test data was projected by me. It was into 2D space. **t-SNE (t-Distributed Stochastic Neighbor Embedding)** was used.

As shown in **Figure 9**, the two classes are forming two distinct clusters. They are non-overlapping. This visual evidence is confirming why the classifiers faced no ambiguity during decision-making. The Red cluster (Politics) and the Blue cluster (Sports) are far apart.

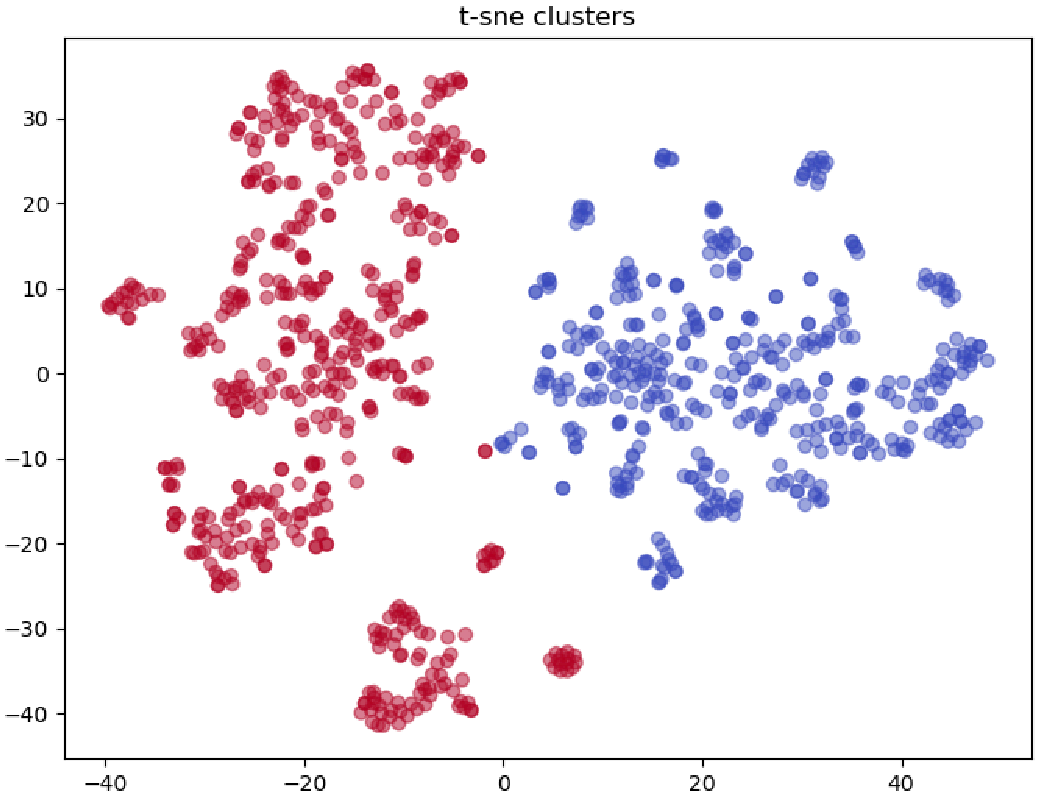

Figure 6: *t-SNE Projection of the test data. The clear gap between the Red (Politics) and Blue (Sports) clusters visually confirms the linear separability of the dataset.*

8. Conclusion

This study compared ten machine learning techniques. It was done on the BBC News dataset. We found that simple, interpretable models are sufficient. Like **Naive Bayes** and **Logistic Regression**. They achieved **100% accuracy**.

Complex ensemble methods like Gradient Boosting are powerful. But they slightly overfit the training data. This yielded lower accuracy. Though it is still high.

Future Work: Future improvements could involve testing on more ambiguous categories. For example, *Politics vs. Business*. There the vocabulary overlap is higher. This would necessitate the non-linear capabilities. Random Forest or ANN models would be better there.