Multi-class Classification of Online News Articles using Linear Models

Francesco Cellaura, Daniele Ferrara
Politecnico di Torino
Student IDs: s353978, s353385
{s353978, s353385}@studenti.polito.it

***Abstract*—In this report, we propose a machine learning pipeline for classifying online news articles collected from multiple international news sources. The dataset records are described by the article text content and associated metadata. Since the raw data contains significant noise, the proposed approach focuses on extracting the most relevant semantic information for the prediction task.**

I. Problem Overview

The proposed assignment is a classification problem of online news articles collected from different international news sources. The dataset contains approximately 100,000 records, each corresponding to a single news article, including both metadata and textual content. Articles originate from several sources and span different publication dates. A categorical label is associated with each record. The goal is to assign a unique category to each article.

The categories are: International News, Business, Technology, Entertainment, Sports, General News, and Health.

The dataset is divided into two parts:

- A development set, containing approximately 80,000 articles;
- An evaluation set, containing approximately 20,000 articles.

We will use the development set to build a classification model to correctly label the points in the evaluation set.

An analysis of the category distribution reveals that the dataset is unbalanced, with a prevalence of International News. This skew is expected, as it reflects the natural dominance of these topics within the broader media landscape.

While the textual content is likely the most relevant attribute for solving the classification problem, we decided to explore the other metadata, as they may contain useful information and patterns that could enrich the training data. Each article is associated with a unique *Id*; a *Source* (the publisher or news outlet); a *Title*; the *Article* (textual content); a *PageRank* (a ranking related to the source); and a *TimeStamp* (the publication date and time).

PageRank: we observed that the vast majority of articles share a PageRank of 5. Due to this lack of variance, we expect the feature to offer negligible discriminative power for the model.

Source: the distribution of sources is highly skewed, dominated by a few high-frequency publishers. Generalist sources like Yahoo, CNN, and Reuters appear across all categories, whereas specialized sources are category-specific (e.g., RedNova and CNet for Technology, ESPN for Sports). Additionally, we identified approximately 300 records with missing source information, denoted by the string "\N".

TimeStamp: the Timestamp feature includes both date and time components. Detailed analysis revealed a seasonality in the data, where specific topics and categories recur periodically. We consider this temporal information valuable for the model's predictive performance. However, a significant data quality issue exists: approximately 30% of the articles are missing a timestamp.

Text Content: the 'Article' feature is unstructured and noisy, containing significant HTML markup, hyperlinks, and special symbols. While these technical artifacts are irrelevant for classification, the embedded text provides rich semantic content essential for determining the category and topic. As we observed with the sources, the article text is also missing in some cases. About 2,000 records contain the string "\N" instead of the actual article content.

Title: the 'Title' attribute also exhibits noise. We identified approximately 200 records containing the substring `ADV`. The article content associated with these titles appears incoherent and semantically empty, suggesting these records are likely spam or advertisements rather than genuine news.

The converse, however, does not hold. Records with missing 'Article' content still possess valid metadata (such as Title and Source) that provides semantically relevant information.

All these considerations have been necessary to approach the preprocessing phase in the best way.

II. Proposed approach

A. Preprocessing

We focused on three core preprocessing tasks: handling missing values, engineering new features, and sanitizing the text. This preparation ensures the textual data is structured effectively to maximize the performance of the TF-IDF method.

First thing we did was dropping the 'Id' and the 'PageRank' attributes, since they are not relevant features for classification.

Regarding the textual content, we implemented a three-step preprocessing strategy:

- **Flag Creation** (*is_adv, is_null*): We engineered binary flags to explicitly identify promotional titles and missing article bodies. This improves model robustness by

allowing it to distinguish between genuine news, spam artifacts, and data errors.

- **Noise Reduction** (`clean_text`): We defined a custom function to strip non-semantic noise—including HTML tags, hyperlinks, symbols, and digits. This step ensures the text is optimized for the TF-IDF vectorizer while preserving the core semantic information. [1], [2].
- **Feature Aggregation** (`full_content`): We replaced the missing "\N" values with empty strings and concatenated the *Title* and *Article* into a single feature. This consolidates the textual signal and ensures the model can leverage valid title information even when the article body is missing.

Regarding the *Timestamp* feature, we adopted the following preprocessing strategy:

- **Missing Value Indicator:** We introduced a binary flag to explicitly mark records with missing timestamps, allowing the model to recognize and learn from the absence of temporal data.
- **Temporal Discretization (*year*, *month*):** We extracted *year* and *month* as distinct features. This discretizes the continuous time variable into monthly intervals, enabling the model to capture the seasonal patterns and temporal dependencies identified during the exploratory analysis.

For the Source feature, we implemented categorical grouping, mapping sources into 9 macro-categories based on their topic dominance and frequency. As shown in Fig. 1, specialized sources are strong predictors of certain topics.

Implementing this mapping as explained has the following benefits:

- **Dimensionality Reduction:** Reducing the cardinality from 1,300 unique values to 9 categories prevents the "curse of dimensionality" inherent in high-dimensional one-hot encoding.
- **Generalization & Overfitting:** Aggregating rare sources eliminates noise and prevents the model from memorizing specific outliers, thereby improving generalization on unseen data.
- **Statistical Robustness:** Transforming sparse labels into dense clusters ensures that every category has a sufficient sample size, providing the model with a stable and interpretable signal.

The preprocessing phase concluded with feature encoding: the *full_content* was transformed via TF-IDF, while categorical features underwent One-Hot Encoding. With these steps complete, the dataset is optimized and ready for training.

B. Model selection

*a) **Logistic Regression**:* This model is a linear classifier that estimates class membership probabilities as a function of a weighted combination of input features. It is particularly well-suited for high-dimensional and sparse representations, such as TF-IDF vectors, due to its computational efficiency and scalability. Furthermore, the inclusion of L2 regularization helps mitigate overfitting in scenarios where the number of features largely exceeds the number of samples, making it a robust candidate for text classification.

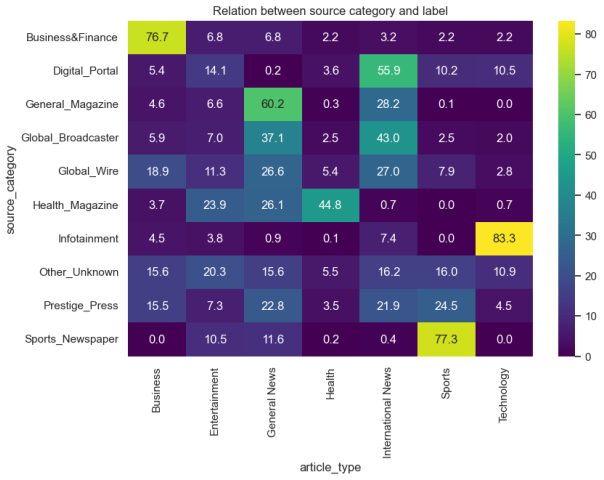

Fig. 1. **Correlation matrix between source category and labels.** The heatmap illustrates the percentage distribution of article types within each source category. High correlations (yellow blocks) indicate that specialized sources (e.g., Sports, Business) are strong predictors of the article topic.

*b) **Linear Support Vector Classifier**:* This model is a linear, margin-based classifier that identifies the maximum-margin separating hyperplane between classes. We selected this model because it is specifically optimized for large-scale sparse feature spaces and is widely recognized as a standard approach in NLP tasks. Its theoretical foundation allows it to handle high-dimensional data effectively by maximizing the distance between decision boundaries.

C. Hyperparameters tuning

For Logistic Regression, the parameter C is varied around 1 to balance under- and over-regularization in a high-dimensional TF-IDF space. We use l2 regularization for stability with correlated features and the liblinear solver with a one-vs-rest strategy, which is well suited for sparse text classification.

For Linear SVC, C controls the trade-off between margin maximization and classification error. Smaller values help reduce overfitting, while larger ones allow more flexible decision boundaries. The squared hinge loss is adopted for smoother optimization and improved stability.

The results of these configurations are summarized in Table I and visualized in Fig. 2.

TABLE I
HYPERPARAMETERS CONSIDERED

Model	Parameter	Values
Logistic Regression	*C*	{0.7, 0.8, 0.9, 1, 2, 3}
	penalty	l2
	solver	lbfgs
	multi_class	{multinomial, ovr}
SVC	*C*	{0.1, 0.3, 0.5, 0.7, 1.0, 2.0}
	loss	{hinge, squared_hing}

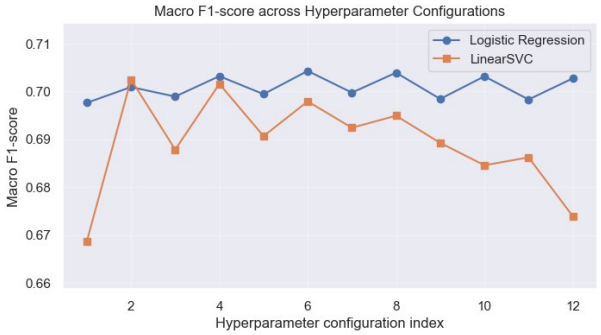

Fig. 2. **Macro F1-score across Hyperparameters Configurations.** The plot shows the macro F1-score across different hyperparameter configurations for Logistic Regression and LinearSVC.

III. Results

The hyperparameter tuning process reveals that both models achieve comparable peak performance. However, as illustrated in Fig. 2, Logistic Regression maintains a more stable predictive power across various settings. In contrast, the LinearSVC exhibits variability and a general performance degradation as configurations change, highlighting a higher sensitivity to hyperparameter selection.

The best configuration for Logistic Regression was found for {*'max_iter': 2000, 'class_weights': 'balanced', 'C': 0.9, 'multi_class': 'ovr', 'penalty': 'l2', 'solver': 'lbfgs'*} (F1-score = 0.70) whereas the best configuration for LinearSVC was found for {*'max_iter': 2000, 'class_weights': 'balanced', 'C': 0.1, 'loss': 'squared_hinge'*}.

The two classifiers achieve satisfactory and comparable results under these optimized settings.

Table II presents the final performance metrics, including precision, recall, and F1-score for each class, as evaluated on the independent test set.

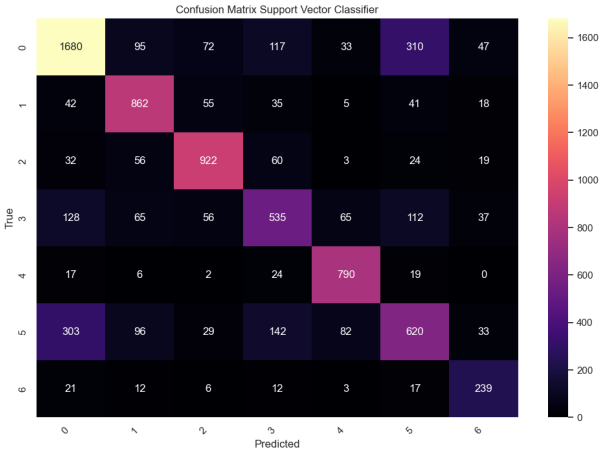

Fig. 3. **Confusion matrix (SVC)** The heatmap shows the confusion matrix for the Support Vector Classifier.

We trained the best performing logistic regressor and linear support vector classifier on all available development data. Then the models have been used to label the evaluation set. The public score obtained is of 0.712 for the logistic regressor and of 0.716 for the SVC. Based on these results, we pick the SVC to be our final model. The confusion matrix for our final SVC model is shown in Fig. 3.

Since these are the first scores obtained using the evaluation data, there should be no overfitting and we can reasonably assume that similar results can be obtained on the private score.

TABLE II
Classification Report (LinearSVC)

	precision	recall	f1-score	support
0	0.77	0.72	0.74	2355
1	0.73	0.83	0.78	1059
2	0.79	0.84	0.81	1116
3	0.63	0.49	0.55	997
4	0.76	0.96	0.85	857
5	0.58	0.47	0.52	1306
6	0.57	0.88	0.69	310
accuracy			0.71	8000
macro avg	0.69	0.74	0.71	8000
weighted avg	0.71	0.71	0.71	8000

IV. Discussion

Despite the overall positive performance, the Entertainment (Label 3) and General News (Label 5) categories consistently show lower scores compared to the other classes. This behavior can be attributed to a combination of class imbalance and significant semantic overlap with neighboring categories, which makes them particularly difficult to separate within a TF-IDF feature space. These classes exhibit systematically lower precision and recall, suggesting they are less semantically distinctive or underrepresented in the training data.

In particular, the textual content of these labels often shares common terminology with other categories, leading to increased confusion for linear classifiers. While we could have implemented more advanced linguistic techniques such as lemmatization, stemming, or part-of-speech tagging [1], we opted for a more standard preprocessing pipeline to maintain computational efficiency and simplicity.

Potential future improvements to address these bottlenecks include class-specific weighting, data augmentation, or the adoption of contextual embeddings—such as BERT—to better capture subtle semantic differences. Ultimately, while these challenges highlight the limits of linear boundaries in bag-of-words representations, the high macro F1-score achieved confirms that the current pipeline is robust and highly effective for the majority of the news spectrum.

References

[1] T. D. Science, "The ultimate preprocessing pipeline for your NLP models," 2023. Accessed: 2026-02-01.
[2] GeeksforGeeks, "Natural language processing (NLP) pipeline," 2023. Accessed: 2026-02-01.