News Article Classification Using TF-IDF and Context Injection

Davide Motta
Politecnico di Torino
Student id: s352816
davide.motta@studenti.polito.it

***Abstract*—In this report, we present a machine learning pipeline for automatic news article classification into seven predefined categories. The proposed approach combines TF-IDF text vectorization with a novel context injection technique that embeds metadata directly into the text representation. A LinearSVC classifier with balanced class weights achieves strong performance on the multi-class classification task. The pipeline includes comprehensive preprocessing, feature engineering from both textual and numerical attributes, and hyperparameter tuning via grid search. Results demonstrate that context injection is an effective strategy, outperforming traditional naive text vectorization methods.**

I. Problem Overview

The task involves classifying English-language news articles into seven categories: International News (0), Business (1), Technology (2), Entertainment (3), Sports (4), General News (5), and Health (6). The dataset is divided into two parts:

- *Development set*: $\sim$ 80,000 labeled records for training and validation.
- *Evaluation set*: 20,000 unlabeled records for final evaluation.

An investigation of the development set revealed several data quality issues and challenges that make the news classification task non-trivial. First of all, as depicted in Figure 1 the categories are imbalanced, with *International News* being the most frequent class by a large margin. Furthermore, some categories share vocabulary (like General News and International News), making the distinction even harder. Another problem arises by inspecting the *article* field: there are URLs, HTML artifacts and odd characters that need to be thoroughly cleaned before the model training. Last, the *timestamp* field contains more than 27,000 NaN values. Fortunately, these missing values are distributed proportionally across categories, simplifying their treatment.

By investigating the textual a metadata fields we discovered patterns that could be very discriminative: *page_rank*, *source*, and *article* fields contain effective signals for classification. For instance, the *Technology* category is the only one having a significant amount of *page_rank* records different from 5 (Figure 2b), and it contains very long articles compared to the other categories (Figure 2a). Moving forward, certain sources are exclusive to specific categories: as shown in Figure 2c, "Syfy.com" publishes only *Entertainment* articles, providing perfect classification for those records.

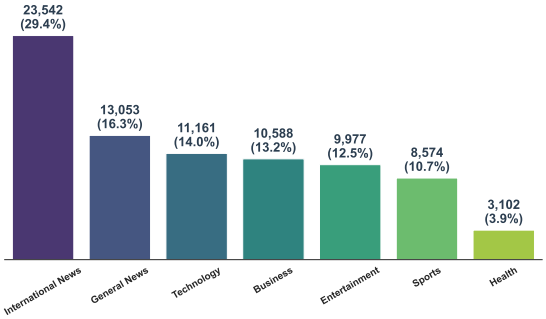


Fig. 1: Category distribution

RSS (100% accuracy)	Category
`/europe /politics /world`	International
`/entertainment`	Entertainment
`/sports`	Sports
`/business`	Business
`/tech /science`	Technology
`/health`	Health
`/cnn_topstories`	General

TABLE I: RSS feeds (9%) fully classify the articles

Another meaningful pattern emerges from URLs within articles. We found 15.5% of articles with at least one URLs. Of these, 58.1% (i.e. 9% of the total dataset) includes the `/rss/[category]` web feeds. These feeds exhibit a perfect correlation with the target labels as shown in Table I.

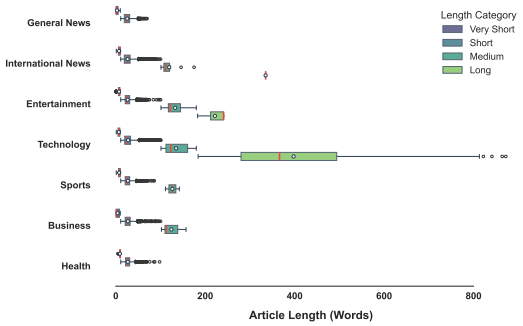
While these information appear highly useful, two concerns must be addressed:

- *Temporal drift*: The dataset contains articles published between 2004 and 2008. Models may learn era-specific patterns (e.g., source distributions, topic prevalence) that do not generalize to contemporary news.
- *Shortcut learning*: Highly discriminative metadata features may cause the model to rely on surface-level correlations rather than learning semantic patterns from the text, harming generalization when such metadata is unavailable.

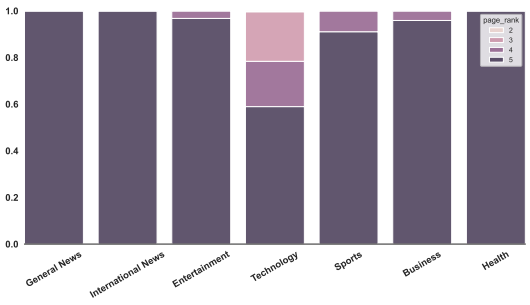
II. Proposed Approach

A. Data Preprocessing

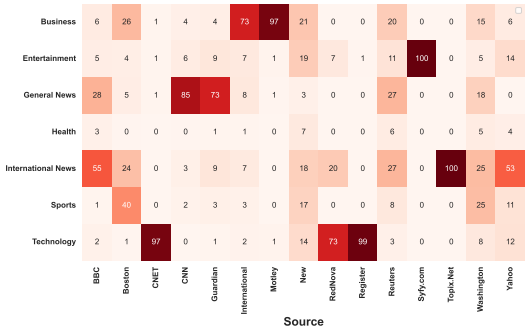
The preprocessing pipeline addresses data quality issues in several steps:



(a) Length Distribution



(b) Page Rank Distribution



(c) Top Sources Distribution

Fig. 2: Features analysis

Duplicate removal: The dataset contains exact duplicates (same source, title, article, and label) and near-duplicates (same content from different sources). We removed these by keeping the most recent instance based on timestamp.

Ambiguous sample removal: Some articles appear with identical text but different labels, indicating annotation inconsistencies. These samples are removed to prevent contradictory training signals.

Text cleaning: Raw text undergoes extensive cleaning:

- HTML entities, URLs and tags removal
- Special character removal (keeping only alphabetic characters)
- Stopword removal (common English words plus domain-specific terms like "said", "says", "year")
- Lowercase conversion
- Lemmatization

Eventually, we dropped the *Id* columns since it brings no information.

Title	Suffix	Category
NEC Shoots to Regain Supercomputer Title (PC World)	PC World	Technology

TABLE II: Suffix example (PC World is only Technology)

B. Feature Engineering

While there are only few fields in the dataset, it is possible to extrapolate effective signals from them. Here the main features we extracted:

- *RSS category*: the RSS web feeds category extracted from articles (e.g `politics`). A complete picture of the categories is shown Table I.
- *Title suffix*: some titles contain extra label that categorize the articles (Table II).
- *Article length*: the binned number of words inside each article (after cleaning). We created four category: long, medium, short, very short.
- *Link counts*: The total count of links, images, ads, and feeds.
- *Cyclic time features*: timestamp is encoded as cyclic features using sine/cosine for hour, day of week and month [1]. We decided not to include the year since the dataset contains a dated and limited time range.

C. Data Pipeline

After the features generation we built the pipeline by applying several techniques.

Context Injection: A key contribution is the context injection technique, where metadata is embedded directly into the text before vectorization. The source name and RSS category (extracted from URLs) are repeated multiple times and prepended to the article text:

$$\text{combined\_text} = \text{source}_{\times 3} \oplus \text{rss}_{\times 5} \oplus \text{title}_{\times 2} \oplus \text{article} \quad (1)$$

where $\oplus$ denotes string concatenation. This allows the TF-IDF vectorizer to capture interactions between metadata tokens and content words within a unified bag-of-words representation.

TF-IDF Vectorization: The primary features come from TF-IDF vectorization [2] of the combined text. Key parameters include:

- Maximum 10,000 features
- N-gram range: unigrams, bigrams, and trigrams (1-2)
- Minimum document frequency: 5
- Maximum document frequency: 70%
- Sublinear TF scaling

χ^2 *Features Selection* To reduce the dimensionality of the TF-IDF feature space while retaining the most informative

terms, we applied the chi-squared (χ^2) statistical test for feature selection [3]. This method evaluates the dependence between each term and the target class. Terms with higher χ^2 scores exhibit stronger associations with specific categories.

CatBoost Encoding: Categorical features (*source*, *first_link_domain*, *title_suffix*, *rss_label*) are encoded using CatBoost target encoding [4]. For each categorical variable x and class c, the encoder computes:

$$\text{enc}(x, c) = \frac{\text{count}(x, c) + \sigma \cdot \text{prior}(c)}{\text{count}(x) + \sigma} \tag{2}$$

where $\sigma = 0.9$ is a smoothing parameter that prevents overfitting on rare categories. This produces 28 features (4 categorical columns $\times$ 7 classes), capturing the conditional probability of each class given the categorical value while avoiding data leakage.

D. Model Selection

We evaluated three model families, each with different characteristics and trade-offs. The primary evaluation metric was Macro F_1 score, which is particularly suitable for multi-class problems with imbalanced class distributions as it gives equal weight to each class regardless of its frequency [5].

- *LinearSVC*: Support Vector Machines with linear kernels have been extensively studied for text classification and consistently achieve state-of-the-art performance [6]. In text classification, the number of features (vocabulary terms) often exceeds the number of training samples, a regime where linear models tend to generalize well. Furthermore, the L_2 regularization inherent in SVMs helps prevent overfitting on noisy features.
- *LightGBM*: Gradient Boosting Decision Trees (GBDT) have shown remarkable performance on structured/tabular data [7]. LightGBM, in particular, offers efficient handling of categorical features and fast training times. We included this model to assess whether the combination of textual (TF-IDF) and metadata features (numerical and categorical) could benefit from a non-linear model. However, this model seems to suffer overfitting while keeping comparable performance to LinearSVC on test set.
- *Logistic Regression*: Multinomial Logistic Regression using the softmax function is a well-established baseline for multi-class classification [8]. With L_2 regularization and the L-BFGS optimizer, it provides a probabilistic interpretation of predictions and serves as a computationally efficient baseline. While highly interpretable, preliminary experiments showed consistently lower performance compared to LinearSVC, leading us to exclude it from detailed hyperparameter tuning.

E. Hyperparameter Tuning

The proposed pipeline involves numerous hyperparameters spanning both preprocessing and model configuration. Performing a joint grid search over all parameters would be computationally prohibitive. Following a similar approach to [9], we adopted a two-stage tuning strategy based on the assumption that preprocessing and model hyperparameters are approximately orthogonal—that is, the optimal preprocessing configuration does not strongly depend on the specific model hyperparameters, and vice versa.

1) Preprocessing Configuration: We fixed the model hyperparameters to their default values and systematically explored the preprocessing configuration space. We considered the following preprocessing hyperparameters:

- *Context injection*: We evaluated both its presence/absence and the repetition weights for each metadata field. Higher weights increase the influence of metadata in the final representation.
- *TF-IDF parameters*: We explored different vocabulary sizes, n-gram ranges, and the effect of sublinear term frequency scaling, which dampens the impact of very frequent terms.
- *CatBoost encoding*: We evaluated its presence/absence and the smoothing parameter σ, which controls the regularization strength.
- *Chi-squared feature selection*: We evaluated its presence/absence and the different values of k (number of features to keep).

To reduce the search space, we made the simplifying assumption that each preprocessing component can be evaluated semi-independently. We first established a baseline using only TF-IDF features, then incrementally added components and measured their impact. This approach, while not exhaustive, allowed us to efficiently explore the configuration space.

2) Model Hyperparameter Tuning: After fixing the preprocessing configuration, we proceeded to tune model-specific hyperparameters. We employed 5-fold stratified cross-validation on 80/20 train-test split.

LinearSVC was tuned via exhaustive Grid Search over the main hyperparameters listed in Table.

LightGBM was tuned via Randomized Search with 40 iterations, as the hyperparameter space is larger and exhaustive search would be computationally prohibitive. Key parameters include the number of estimators (trees), maximum tree depth, and learning rate.

Table III summarizes the searching space of hyperparameters.

III. Results

The tuning of the preprocessing parameters proved the presence of both Context Injection and CatBoost Encoding significantly enhance the baseline, while the usage of χ^2 feature selection shows relevant improvement for LightGBM only. In Figure IV the contributions calculated throughout ablation test. Furthemore we found the optimal weights for source, rss, title, article to be, 3,5,2,1 respectively. For TF-IDF *max features=15,000* and *sublinear=True* with *n-grams=(1,3)*. Catboost exhibited a *sigma=0.9* while *k=10,000* for χ^2.

The best configuration for LinearSVC was found with *C=0.05* and *dual=False* with *maxiter=2000* (Macro $F_1 \approx$

Model	Params	Values
Context Inj.	weights	$[1, 5]$
	presence	$\{T, F\}$
TF-IDF	max features	$\{10000, 15000, 20000\}$
	sublinear	$\{T, F\}$
	n-grams	$\{(1, 2), (1, 3)\}$
CatBoost	sigma	$\{0, 0.5, 0.9, 1\}$
	presence	$\{T, F\}$
χ^2	k	$\{5000, 10000\}$
	presence	$\{T, F\}$
LSVC	C	$\{0.01, 0.02, 0.05, 0.1, 0.2\}$
	maxiter	$\{2000, 1000, 3000\}$
	dual	$\{T, F\}$
LGBM	n estimators	$\{100, 200, 300, 500\}$
	max depth	$\{5, 7, 10, 15, -1\}$
	learning rate	$\{0.01, 0.05, 0.1, 0.2\}$
	num leaves	$\{31, 50, 70, 100\}$
	min child samples	$\{20, 50, 100\}$
	subsample	$\{0.7, 0.8, 1.0\}$
	colsample bytree	$\{0.7, 0.8, 1.0\}$

TABLE III: Hyperparameters space

Component	Macro $F_1 \approx$	
	LSVC	LGBM
Base TF-IDF	0.66	0.65
+ Context Inj.	0.72	0.71
+ Catboost	**0.73**	0.72
+ χ^2	0.72	**0.73**

TABLE IV: Ablation Test (on test set)

0.726) whereas the best configuration for the LightGBM was found for *n_estimators=500 max_depth=7 learning_rate=0.05 num_leaves=31 min_child_samples=50*, *subsample=0.7*, and *colsample_bytree=0.6* (Macro $F_1 \approx 0.727$)

We trained the best performing LSVC and LGBM on all available development data. Then the models have been used to label the evaluation set. The public score obtained is of 0.725 for the SVC and of 0.722 for the LGBM. It is worth noticing that LightGBM model seems to suffer of overfitting (Train Macro $F_1 > 0.82$) and this could explain the worse performance on evaluation set with respect to LinearSVC.

Analyzing each category locally, we noticed both models are struggling in labelling General News and Enterteinment articles ($F_1 < 0.6$). Usually, Internationl News was predicted insted of those.

IV. Discussion

The proposed approach achieves satisfactory results on the news article classification task. The ablation study demonstrates that context injection is the most impactful contribution, improving the baseline TF-IDF performance by approximately 6 percentage points. This technique effectively bridges the gap between metadata and textual content by allowing the vectorizer to capture joint patterns.

We have empirically shown that LinearSVC and LightGBM perform similarly on this task, though LinearSVC exhibits better generalization as evidenced by the smaller train-test gap. The overfitting observed in LightGBM (training Macro $F_1 > 0.82$ vs. test ≈ 0.72) suggests that gradient boosting models may be prone to memorizing spurious correlations in high-dimensional sparse feature spaces.

The following aspects might be worth considering to further improve the obtained results:

- *Advanced text representations*: Pre-trained language models such as BERT [10] or sentence transformers could capture semantic relationships that TF-IDF cannot. However, the computational cost and the age of the dataset (2004–2008) may limit their effectiveness.
- *Ensemble methods*: Combining LinearSVC and LightGBM predictions through voting or stacking could leverage their complementary strengths—LinearSVC's robustness on sparse text features and LightGBM's ability to model non-linear interactions among metadata features.
- *Class-specific analysis*: A detailed per-class error analysis could reveal systematic misclassifications. For instance, distinguishing between *General News* and *International News* remains challenging due to vocabulary overlap, and targeted feature engineering for these classes could yield improvements.

Despite these potential improvements, the results obtained are already competitive. The main limitation lies in the strong dependence on metadata features, which raises concerns about generalization to news sources not present in the training data. Nevertheless, for the specific task at hand—where such metadata is consistently available—the proposed pipeline offers an effective and interpretable solution.

References

[1] Scikit-learn Developers, "Cyclical feature engineering." https://scikit-learn.org/stable/auto_examples/applications/plot_cyclical_feature_engineering.html, 2024. Accessed: 2026-01-30.

[2] G. Salton and C. Buckley, "Term-weighting approaches in automatic text retrieval," *Information processing & management*, vol. 24, no. 5, pp. 513–523, 1988.

[3] G. Forman, "An extensive empirical study of feature selection metrics for text classification," *Journal of Machine Learning Research*, vol. 3, pp. 1289–1305, 2003.

[4] L. Prokhorenkova *et al.*, "Catboost: unbiased boosting with categorical features," *Advances in Neural Information Processing Systems*, vol. 31, 2018.

[5] M. Sokolova and G. Lapalme, "A systematic analysis of performance measures for classification tasks," *Information Processing & Management*, vol. 45, no. 4, pp. 427–437, 2009.

[6] T. Joachims, "Text categorization with support vector machines: Learning with many relevant features," in *European conference on machine learning*, pp. 137–142, Springer, 1998.

[7] G. Ke, Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, Q. Ye, and T.-Y. Liu, "Lightgbm: A highly efficient gradient boosting decision tree," in *Advances in neural information processing systems*, vol. 30, 2017.

[8] D. W. Hosmer Jr, S. Lemeshow, and R. X. Sturdivant, *Applied Logistic Regression*. Hoboken, NJ: John Wiley & Sons, 3rd ed., 2013.

[9] F. Giobergia, "Free spoken digit dataset classification problem." Data Science Lab Course Material, Politecnico di Torino, 2024.

[10] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2019.