Empirical Evaluation of Preprocessing and Balancing Techniques Impact Across Algorithm-Vectorizer Combinations in Sentiment Classification

Nathanael Motta[1], **Ana Claudia Maria de Souza**[1], **Carlo Marcelo Revoredo da Silva**[1], **Cleyton Mario de Oliveira Rodrigues**[1]

[1]Programa de Pós-Graduação em Engenharia da Computação (PPGEC)
Universidade de Pernambuco (UPE)
Recife-PE – Brasil

{nathanael.carauna,ana.cmsouza, marcelo.revoredo, cleyton.rodrigues}@upe.br

Abstract. *This study systematically evaluates the individual impact of six commonly applied preprocessing techniques on sentiment classification performance using 14,224 Steam reviews. Each technique was assessed across four algorithms (SVC, Random Forest, TextBlob, VADER) and three vectorization approaches (TF-IDF, Count Vectorizer, Hashing Vectorizer). Results reveal significant variability in preprocessing effectiveness: four consecutive stages (HTML cleanup, emoji removal, number elimination, punctuation normalization) produced identical results, indicating redundancy for this corpus. Stop word removal and stemming showed mixed effects depending on algorithm-vectorizer combinations. Machine learning approaches demonstrated different sensitivity patterns compared to lexicon-based methods. SVC with TF-IDF showed consistent performance across preprocessing stages. These findings challenge assumptions about universal preprocessing benefits and emphasize the importance of empirical validation for specific domains and algorithm combinations.*

Keywords: Natural Language Processing, Sentiment Analysis, Preprocessing Techniques, Algorithm-Vectorizer Interactions, Empirical Evaluation, Steam Reviews.

1. Introduction

Sentiment analysis in natural language processing relies heavily on preprocessing pipelines that transform raw text into structured features suitable for machine learning algorithms [Guzsvinecz and Szűcs 2023]. The gaming industry, particularly Steam platform with millions of user reviews, presents a rich domain for evaluating preprocessing effectiveness due to its diverse linguistic patterns and user-generated content characteristics [Ruseti et al. 2020].

Traditional natural language processing workflows systematically apply multiple preprocessing techniques under the assumption that each contributes positively to classification performance. Common practices include HTML cleanup, emoji removal, number elimination, punctuation normalization, stop word removal, tokenization, and stemming [Maree et al. 2024, Prastyo et al. 2018]. However, empirical evidence supporting universal effectiveness of these techniques across different domains, algorithms, and vectorization approaches remains limited.

Recent investigations reveal inconsistent patterns regarding preprocessing effectiveness across different domains and contexts. Schoenfeld et al. demonstrated that preprocessing can, on average, harm accuracy in machine learning pipelines, though best-performing pipelines do utilize appropriate preprocessors [Schoenfeld et al. 2018]. Similarly, Amorim et al. showed that inadequate scaling technique selection can be more detrimental than not scaling data at all [Amorim et al. 2022], suggesting that commonly applied techniques may not provide universal benefits.

Despite these findings, a gap exists in systematic empirical evaluation of individual preprocessing technique contributions. Previous studies typically evaluate preprocessing pipelines as complete workflows [Ruseti et al. 2020, Tan et al. 2022], making it difficult to isolate the specific impact of individual techniques. Manda et al. [Manda et al. 2021] identified this limitation in their systematic review, noting the absence of controlled studies that systematically vary individual preprocessing steps while maintaining other experimental conditions constant. Understanding which techniques provide measurable benefits, which have neutral impact, and which may degrade performance across different algorithm-vectorizer combinations is essential for optimizing sentiment classification systems.

The objective of this work is to systematically evaluate the individual impact of six commonly applied preprocessing techniques on sentiment classification performance using Steam game reviews, analyzing how these effects vary across different algorithms and vectorization approaches. This study focuses on classic machine learning models combined with traditional vectorization techniques (TF-IDF, Count, and Hashing vectorizers), providing foundational insights that complement the growing body of research on transformer-based approaches. Three main hypotheses guide this investigation: (H1) commonly applied preprocessing techniques exhibit variable effectiveness, with some traditionally accepted techniques providing minimal or negative impact; (H2) preprocessing technique effectiveness depends significantly on algorithm-vectorizer combinations, requiring domain-specific empirical validation; and (H3) lexicon-based approaches demonstrate different sensitivity patterns to preprocessing compared to machine learning methods.

2. Related Work

The literature reveals complex, domain-dependent preprocessing results with inconsistent evaluation methodologies, making individual technique contribution assessment challenging.

2.1. Gaming Domain Sentiment Analysis

Gaming domain studies show varied preprocessing approaches. Guzsvinecz and Szűcs [Guzsvinecz and Szűcs 2023] analyzed 35.9M Steam reviews focusing on temporal patterns, while Ruseti et al. [Ruseti et al. 2020] achieved 61-67% accuracy using fixed preprocessing pipelines. Tan et al. [Tan et al. 2022] demonstrated 91% SVC accuracy but focused on algorithm comparison rather than preprocessing impact.

2.2. Preprocessing Technique Effectiveness Studies

Cross-domain investigations reveal variable preprocessing effectiveness. Maree et al. [Maree et al. 2024] conducted bilingual Arabic-English analysis with individual technique evaluation across four experimental axes, achieving 70% (SVM-ASTD) and 87% (NB-AJGT), demonstrating context-dependent preprocessing impacts.

Prastyo et al. [Prastyo et al. 2018] analyzed Indonesian slang-enriched reviews using systematic normalization, achieving 96% accuracy with SVM/RF after normalization vs 88-89% without, emphasizing preprocessing importance for informal language. Elahi et al. [Elahi et al. 2024] evaluated Bengali noise reduction methods, finding models trained on noisy texts outperformed noise-reduced ones, challenging common preprocessing assumptions.

2.3. Systematic Reviews and Meta-Analyses

Manda et al. [Manda et al. 2021] reviewed preprocessing techniques, concluding no universally optimal technique exists, identifying the gap this study addresses: systematic empirical evaluation of individual technique impacts across algorithm-vectorizer combinations. Table 1 compares previous work with this study's unique systematic individual technique evaluation approach.

Table 1. Comparison between related works and the present study

Ref.	**Domain**	**Preproc.**	**Bal.**	**Alg.**	**Main Contr.**	**Key Diff.**
Guzsvinecz & Szűcs (2023)	35.9M Steam, 11 genres	NRC Emotion Lexicon	None	Statistical (R)	Temporal analysis; neg. reviews longer (40 vs 19)	Temporal focus, not preproc.
Ruseti et al. (2020)	201K Metacritic	Stop-words, lemmatization, content extraction	Balanced	SVM, MNB, DNN	Extensible pipeline 61-67% acc.	Fixed pipeline, no individual eval.
Tan et al. (2022)	Steam + Metacritic	HTML, normalization, stopwords, tokenization	SMOTE	SVC, MLP, XGB, LR, MNB	Alg. comparison 91% SVC acc.	Alg. focus, not preproc. impact
Maree et al. (2024)	Arabic tweets (ASTD, AJGT)	N-gram, Arabic stopwords, stemming	None	LR, RF, NB, SVM	Bilingual analysis 70% (SVM), 87% (NB)	Bilingual comparison, no algo-vectorizer int.
Prastyo et al. (2025)	Indonesian 10K	Slang normalization, TF-IDF	None	NB, SVM, RF	Slang norm. impact 96% acc.	Slang-specific, not full preproc.
Elahi et al. (2024)	Bengali noisy texts	Noise reduction: translation, correction, MLM	Cost-sensitive	SVM, BiLSTM, BERT	Noise reduction 0.75 F1	Noise focus, not standard preproc.
This study	Steam 14K	8 stages: HTML, emoji, numbers, punctuation, stopwords, stemming	Under/ over/ proportional	SVC, RF, TextBlob, VADER	Individual technique impact	Systematic individual preproc. eval.

3. Methodology

This study systematically evaluates the individual impact of six preprocessing techniques on sentiment classification performance using Steam game reviews. The experimental design isolates each preprocessing step to quantify its specific contribution across different algorithm-vectorizer combinations. Performance evaluation employs precision, recall, F1-score, and accuracy for both positive (class 1) and negative (class 0) sentiment classes.

3.1. Dataset

The study was conducted using a sample of 14,224 Steam game reviews randomly selected from a larger Kaggle dataset containing millions of reviews, after duplicate removal. This sample presents severe class imbalance with 11,896 positive reviews (84%) and 2,328 negative reviews (16%), creating a challenging scenario for classification algorithms.

3.2. Preprocessing Pipeline

Eight sequential preprocessing stages were systematically evaluated: (1) baseline without preprocessing, (2) HTML markup and encoding removal, (3) emoji removal, (4) number elimination, (5) punctuation/symbol removal, (6) stop word elimination, (7) tokenization and stemming, and (8) dataset balancing. Each stage was designed to isolate individual technique contributions across algorithm-vectorizer combinations.

Three balancing strategies addressed class imbalance: undersampling (reducing majority class to match minority), oversampling (replicating minority class instances), and proportional sampling (creating a balanced dataset by sampling 10,000 instances from each class from the original larger Steam dataset, which contains millions of reviews).

3.3. Models and Vectorization

Four classification algorithms were evaluated: Support Vector Classifier (SVC) for high-dimensional robustness, Random Forest (RF) as an ensemble method, TextBlob (TB) as a lexicon-based baseline, and VADER (VDR) for social media-oriented sentiment analysis. Three vectorization techniques complemented the analysis: TF-IDF for importance weighting, Count Vectorizer for frequency representation, and Hashing Vectorizer for dimensionality efficiency. Each preprocessing stage was evaluated across all algorithm-vectorizer combinations to systematically quantify individual technique impacts.

3.4. Hyperparameters and Configuration

Table 2 presents the hyperparameters and configurations used for each model and vectorization technique to ensure reproducibility and fair comparison.

4. Results and Discussion

The systematic evaluation was conducted through seven sequential preprocessing stages using 14,224 Steam game reviews. This section presents key findings that reveal variable effectiveness of commonly applied preprocessing techniques and their dependence on algorithm-vectorizer combinations.

4.1. Stage 1: Baseline - Data without Preprocessing

The baseline stage established reference performance patterns for four classification algorithms (Random Forest, SVC, TextBlob, and VADER) combined with three vectorization techniques (TF-IDF, Count Vectorizer, and Hashing Vectorizer) applied to raw Steam review data. Results presented in Table 3 reveal fundamental algorithm-vectorizer interaction patterns and provide the foundation for evaluating subsequent preprocessing impacts.

SVC with TF-IDF emerged as the superior configuration, achieving 0.87 accuracy and 0.92 F1-Score for the positive class. Algorithm-vectorizer interactions showed significant variability: SVC performance varied substantially across vectorizers (0.83-0.87

Table 2. Hyperparameters and configurations for models and vectorizers

Method	Parameter	Value/Config.
SVC	kernel	RBF (default)
	probability	True
	random_state	Default
Random Forest	n_estimators	100
	random_state	2021
	Others	Default (scikit-learn)
TextBlob	Sentiment	Polarity-based
	Threshold	>0 (pos.), ≤0 (neg.)
VADER	Sentiment	Compound score
	Threshold	>0 (pos.), ≤0 (neg.)
TF-IDF	max_features	5000
	Others	Default (scikit-learn)
Count Vect.	max_features	5000
	Others	Default (scikit-learn)
Hashing Vect.	n_features	5000
	alt_sign	False
	Others	Default (scikit-learn)
Data Split	Train/Test	80%/20%
	random_state	42

Table 3. Comparative results of models with different vectorizers without textual preprocessing

Mod	Vec	P(0)	P(1)	R(0)	R(1)	F1(0)	F1(1)	Acc
RF	TF-IDF	0.76	0.83	0.10	0.99	0.18	0.91	0.83
RF	Count	0.75	0.84	0.12	0.99	0.20	0.91	0.83
RF	Hash	0.82	0.83	0.07	1.00	0.13	0.90	0.83
SVC	TF-IDF	0.85	0.87	0.32	0.99	0.46	0.92	0.87
SVC	Count	0.90	0.83	0.07	1.00	0.13	0.91	0.83
SVC	Hash	0.88	0.85	0.24	0.99	0.37	0.92	0.86
TB	TF-IDF	0.32	0.89	0.59	0.72	0.41	0.80	0.70
TB	Count	0.32	0.89	0.59	0.72	0.41	0.80	0.70
TB	Hash	0.32	0.89	0.59	0.72	0.41	0.80	0.70
VDR	TF-IDF	0.33	0.87	0.49	0.78	0.40	0.82	0.73
VDR	Count	0.33	0.87	0.49	0.78	0.40	0.82	0.73
VDR	Hash	0.33	0.87	0.49	0.78	0.40	0.82	0.73

accuracy), while Random Forest showed more consistent patterns (0.83 accuracy across all vectorizers). Machine learning algorithms demonstrated class-dependent performance patterns due to corpus imbalance (84% positive, 16% negative).

Lexicon-based methods (TextBlob and VADER) exhibited distinctly different characteristics, showing complete independence from vectorization techniques with identical results across all vectorizers. This baseline pattern establishes that preprocessing impacts may vary significantly between machine learning and lexicon-based approaches, providing a crucial reference for subsequent preprocessing evaluations.

4.2. Stage 2: HTML Markup and Special Characters Removal

HTML markup and special character removal revealed differentiated response patterns among algorithm categories. This preprocessing involved replacing HTML tags, normalizing entities, and eliminating line breaks. Results shown in Table 4 demonstrate positive response from machine learning algorithms, particularly in minority class recognition.

Random Forest showed substantial improvement in negative class recall (TF-IDF: 0.10→0.18; Count Vectorizer: 0.12→0.24), with corresponding F1-Score increases.

Table 4. Comparative results of models with different vectorizers after HTML markup and special characters removal

Mod	Vec	P(0)	P(1)	R(0)	R(1)	F1(0)	F1(1)	Acc
RF	TF-IDF	0.75	0.84	0.18	0.99	0.29	0.91	0.84
RF	Count	0.70	0.85	0.24	0.98	0.35	0.91	0.84
RF	Hash	0.70	0.83	0.12	0.99	0.20	0.91	0.83
SVC	TF-IDF	0.83	0.86	0.28	0.99	0.42	0.92	0.86
SVC	Count	0.94	0.84	0.14	1.00	0.24	0.91	0.84
SVC	Hash	0.82	0.86	0.27	0.99	0.41	0.92	0.86
TB	TF-IDF	0.25	0.89	0.71	0.53	0.37	0.66	0.56
TB	Count	0.25	0.89	0.71	0.53	0.37	0.66	0.56
TB	Hash	0.25	0.89	0.71	0.53	0.37	0.66	0.56
VDR	TF-IDF	0.29	0.86	0.48	0.74	0.36	0.80	0.69
VDR	Count	0.29	0.86	0.48	0.74	0.36	0.80	0.69
VDR	Hash	0.29	0.86	0.48	0.74	0.36	0.80	0.69

SVC maintained superiority while presenting slight variations in recall patterns. Conversely, lexicon-based methods experienced systematic degradation - TextBlob's accuracy dropped from 0.70 to 0.56, and Vader's from 0.73 to 0.69, demonstrating negative sensitivity to textual normalization.

4.3. Stages 3-5: Emoji, Number, and Punctuation/Symbol Removal

The subsequent three preprocessing stages (emoji removal, number removal, and punctuation/symbol removal) revealed identical results to Stage 2, demonstrating technical redundancy. This equivalence suggests Steam game reviews contain naturally minimal emojis, numbers, and special symbols.

The identical correspondence across all metrics for all algorithms and vectorizers indicates these techniques produced no measurable corpus alteration. This pattern suggests Steam game reviews present naturally ”clean” characteristics with low incidence of emojis, numbers, and special symbols. The absence of impact contrasts with theoretical expectations and questions universal application of standard preprocessing protocols, revealing domain-specific corpus characteristics that naturally facilitate automated processing.

4.4. Stage 6: Stop Word Removal

Stop word removal definitively broke the redundancy pattern observed in the four previous stages, representing the first significant inflection point since baseline. Results in Table 5 demonstrate that preprocessing techniques can produce differentiated impacts when directed at linguistic elements with specific relevance for vectorial representation.

Machine learning algorithms exhibited complex response patterns with SVC with TF-IDF showing slight degradation (negative recall $0.28 \rightarrow 0.27$), while Random Forest with TF-IDF also experienced deterioration ($0.18 \rightarrow 0.16$). Lexicon-based methods maintained unchanged results, confirming their independence from stop word presence.

4.5. Stage 7: Tokenization and Stemming

Tokenization and stemming demonstrated compensatory effects, partially reversing degradations from stop word removal. Results in Table 6 show morphological normalization can mitigate previous stage limitations through strategic sequencing of techniques.

Table 5. Comparative results of models with different vectorizers after stop word removal

Mod	Vec	P(0)	P(1)	R(0)	R(1)	F1(0)	F1(1)	Acc
RF	TF-IDF	0.70	0.84	0.16	0.98	0.27	0.91	0.84
RF	Count	0.68	0.85	0.21	0.98	0.32	0.91	0.84
RF	Hash	0.74	0.84	0.14	0.99	0.24	0.91	0.84
SVC	TF-IDF	0.82	0.86	0.27	0.99	0.40	0.92	0.86
SVC	Count	0.94	0.84	0.14	1.00	0.24	0.91	0.84
SVC	Hash	0.82	0.86	0.25	0.99	0.38	0.92	0.85
TB	TF-IDF	0.25	0.89	0.72	0.53	0.37	0.66	0.56
TB	Count	0.25	0.89	0.72	0.53	0.37	0.66	0.56
TB	Hash	0.25	0.89	0.72	0.53	0.37	0.66	0.56
VDR	TF-IDF	0.29	0.86	0.47	0.74	0.36	0.80	0.69
VDR	Count	0.29	0.86	0.47	0.74	0.36	0.80	0.69
VDR	Hash	0.29	0.86	0.47	0.74	0.36	0.80	0.69

Table 6. Comparative results of models with different vectorizers after tokenization and stemming

Mod	Vec	P(0)	P(1)	R(0)	R(1)	F1(0)	F1(1)	Acc
RF	TF-IDF	0.76	0.85	0.19	0.99	0.30	0.91	0.84
RF	Count	0.72	0.85	0.22	0.98	0.34	0.91	0.84
RF	Hash	0.66	0.83	0.11	0.99	0.19	0.90	0.83
SVC	TF-IDF	0.82	0.86	0.27	0.99	0.41	0.92	0.86
SVC	Count	0.94	0.84	0.14	1.00	0.24	0.91	0.84
SVC	Hash	0.82	0.86	0.25	0.99	0.38	0.92	0.85
TB	TF-IDF	0.25	0.89	0.72	0.53	0.37	0.66	0.56
TB	Count	0.25	0.89	0.72	0.53	0.37	0.66	0.56
TB	Hash	0.25	0.89	0.72	0.53	0.37	0.66	0.56
VDR	TF-IDF	0.29	0.86	0.47	0.74	0.36	0.80	0.69
VDR	Count	0.29	0.86	0.47	0.74	0.36	0.80	0.69
VDR	Hash	0.29	0.86	0.47	0.74	0.36	0.80	0.69

SVC showed improvement in negative F1-Score with TF-IDF (0.40→0.41), while Random Forest demonstrated pronounced recovery patterns (negative recall 0.16→0.19, F1-Score 0.27→0.30). Decision tree-based algorithms benefited more from morphological grouping.

4.6. Preprocessing Techniques Impact Summary

The systematic evaluation of six preprocessing techniques revealed significant variability in effectiveness across algorithm-vectorizer combinations. Four consecutive techniques (HTML cleanup, emoji removal, number elimination, punctuation normalization) produced identical results, indicating corpus-specific redundancy. Stop word removal and stemming showed mixed effects with algorithm-dependent benefits. These findings challenge universal application assumptions and demonstrate the importance of empirical validation for specific domains.

5. Impact of Dataset Balancing Techniques

While preprocessing techniques showed limited and variable impacts, dataset balancing approaches demonstrated substantial effects on classification performance, particularly for addressing class imbalance issues inherent in the Steam review corpus. Three balancing strategies were evaluated to understand their differential impacts across algorithms and vectorizers.

5.1. Undersampling Impact Analysis

Undersampling demonstrated algorithm-dependent effects with substantial performance variations. Results in Table 7 reveal differential algorithm responses to reduced dataset size. Machine learning algorithms showed dramatic improvements in minority class recognition, while lexicon-based methods remained unaffected, confirming their independence from training data distribution.

Table 7. Comparative results of models with undersampling

Mod	Vec	P(0)	P(1)	R(0)	R(1)	F1(0)	F1(1)	Acc
RF	TF-IDF	0.40	0.93	0.77	0.74	0.52	0.83	0.75
RF	Count	0.40	0.93	0.76	0.74	0.52	0.83	0.75
RF	Hash	0.38	0.94	0.79	0.71	0.51	0.81	0.73
SVC	TF-IDF	0.44	0.94	0.78	0.78	0.56	0.85	0.78
SVC	Count	0.40	0.93	0.73	0.76	0.52	0.84	0.76
SVC	Hash	0.41	0.94	0.77	0.75	0.53	0.84	0.76
TB	TF-IDF	0.25	0.89	0.72	0.53	0.37	0.66	0.56
TB	Count	0.25	0.89	0.72	0.53	0.37	0.66	0.56
TB	Hash	0.25	0.89	0.72	0.53	0.37	0.66	0.56
VDR	TF-IDF	0.29	0.86	0.47	0.74	0.36	0.80	0.69
VDR	Count	0.29	0.86	0.47	0.74	0.36	0.80	0.69
VDR	Hash	0.29	0.86	0.47	0.74	0.36	0.80	0.69

5.2. Oversampling Impact Analysis

Oversampling revealed different algorithm sensitivity patterns compared to undersampling. Results in Table 8 demonstrate more conservative improvements with algorithm-specific response variations. SVC showed superior adaptation to artificially increased data volume, while Random Forest exhibited more modest improvements.

Table 8. Comparative results of models with oversampling

Mod	Vec	P(0)	P(1)	R(0)	R(1)	F1(0)	F1(1)	Acc
RF	TF-IDF	0.55	0.88	0.44	0.92	0.49	0.90	0.83
RF	Count	0.53	0.88	0.43	0.92	0.48	0.90	0.83
RF	Hash	0.54	0.88	0.45	0.91	0.49	0.90	0.83
SVC	TF-IDF	0.59	0.89	0.51	0.92	0.55	0.91	0.85
SVC	Count	0.50	0.90	0.57	0.87	0.53	0.89	0.82
SVC	Hash	0.55	0.90	0.54	0.90	0.54	0.90	0.83
TB	TF-IDF	0.25	0.89	0.72	0.53	0.37	0.66	0.56
TB	Count	0.25	0.89	0.72	0.53	0.37	0.66	0.56
TB	Hash	0.25	0.89	0.72	0.53	0.37	0.66	0.56
VDR	TF-IDF	0.29	0.86	0.47	0.74	0.36	0.80	0.69
VDR	Count	0.29	0.86	0.47	0.74	0.36	0.80	0.69
VDR	Hash	0.29	0.86	0.47	0.74	0.36	0.80	0.69

5.3. Proportional Sampling Impact Analysis

Proportional sampling demonstrated the most balanced algorithm performance across metrics. Results in Table 9 reveal optimal algorithm-vectorizer combinations and consistent performance patterns. This approach showed superior effectiveness for maintaining both accuracy and balanced class recognition across different algorithm types.

Table 9. Comparative results of models with proportional sampling balancing

Mod	Vec	P(0)	P(1)	R(0)	R(1)	F1(0)	F1(1)	Acc
RF	TF-IDF	0.81	0.85	0.86	0.80	0.83	0.82	0.83
RF	Count	0.81	0.86	0.87	0.80	0.83	0.83	0.83
RF	Hash	0.78	0.83	0.84	0.77	0.81	0.80	0.81
SVC	TF-IDF	0.83	0.87	0.87	0.83	0.85	0.85	0.85
SVC	Count	0.81	0.83	0.83	0.81	0.82	0.82	0.82
SVC	Hash	0.82	0.85	0.86	0.81	0.84	0.83	0.83
TB	TF-IDF	0.70	0.70	0.69	0.70	0.69	0.70	0.70
TB	Count	0.70	0.70	0.69	0.70	0.69	0.70	0.70
TB	Hash	0.70	0.70	0.69	0.70	0.69	0.70	0.70
VDR	TF-IDF	0.71	0.63	0.53	0.79	0.61	0.70	0.66
VDR	Count	0.71	0.63	0.53	0.79	0.61	0.70	0.66
VDR	Hash	0.71	0.63	0.53	0.79	0.61	0.70	0.66

5.4. Balancing Techniques Comparative Analysis

The evaluation of three balancing approaches revealed distinct algorithm-dependent effectiveness patterns. Undersampling achieved dramatic improvements in minority class recognition (170-316% improvement) but with accuracy trade-offs that varied significantly across algorithms. SVC demonstrated superior robustness to data reduction compared to Random Forest. Oversampling showed more conservative but consistent improvements across algorithm-vectorizer combinations. Proportional sampling provided optimal balance with algorithm-independent stability.

Lexicon-based methods (TextBlob and VADER) remained completely unaffected by all balancing approaches, confirming their fundamental independence from training data characteristics. This pattern distinguishes them from machine learning approaches and suggests that balancing technique selection should consider algorithm type as a primary factor.

6. Conclusions

This work systematically evaluated the individual impact of six preprocessing techniques and three balancing strategies on sentiment classification performance using 14,224 Steam game reviews. The experimental design revealed significant variability in effectiveness that challenges common assumptions about universal preprocessing benefits.

The evaluation highlighted highly specific interactions between techniques, algorithms, and the corpus. Four preprocessing steps proved redundant, while stop word removal and stemming had mixed, algorithm-dependent effects. Machine learning models demonstrated high sensitivity to these changes, in contrast to lexicon-based methods which remained consistently independent. Similarly, no single dataset balancing strategy was universally optimal, with techniques like undersampling and oversampling showing distinct effectiveness patterns across different algorithms. These results confirm that traditional preprocessing pipelines may contain counterproductive steps and require empirical verification.

This study presents important limitations that define its scope and suggest directions for future research. The exclusive focus on classic machine learning models does not address the preprocessing requirements of modern transformer-based architectures like BERT, which represent the current state-of-the-art. Future work should therefore extend this systematic methodology to transformer models, investigate advanced preprocessing techniques, and assess cross-domain generalizability to establish broader empirical frameworks for optimization.

The systematic methodology presented may also offer insights for the Brazilian NLP community. Given that Portuguese text preprocessing faces similar challenges, our findings suggest that empirical validation is crucial, rather than relying on techniques validated solely in English literature. Such investigations could contribute to the development of more effective Portuguese NLP systems tailored to local linguistic characteristics.

References

Amorim, E., Cai, J., Kadav, A., Cui, L., Das, S., Singh, M., and Chen, J. (2022). The choice of scaling technique matters for classification performance. *arXiv preprint arXiv:2212.12343*. Comprehensive evaluation of scaling techniques across 82 datasets and 20 classification algorithms.

Elahi, K. T., Rahman, T. B., Shahriar, S., Sarker, S., Shawon, M. T. R., and Shahariar, G. M. (2024). A comparative analysis of noise reduction methods in sentiment analysis on noisy bangla texts. *arXiv preprint arXiv:2401.14360*. Accepted in The 9th Workshop on Noisy and User-generated Text (W-NUT), 18th Conference of the European Chapter of the Association for Computational Linguistics (EACL 2024).

Guzsvinecz, T. and Szűcs, J. (2023). Length and sentiment analysis of reviews about top-level video game genres on the steam platform. *Computers in Human Behavior*, 149:107955. Available online 12 September 2023.

Manda, R., Sharma, P., and Singh, R. (2021). A systematic literature review on preprocessing techniques for sentiment analysis. *Expert Systems with Applications*, 168:114432.

Maree, M., Eleyat, M., and Mesqali, E. (2024). Optimizing machine learning-based sentiment analysis accuracy in bilingual sentences via preprocessing techniques. *The International Arab Journal of Information Technology (IAJIT)*, 21(02):257–270.

Prastyo, P. A., Berlilana, and Tahyudin, I. (2018). Sentiment analysis of indonesian slang reviews using machine learning. *Journal of Applied Data Sciences*, 3(1):45–58.

Ruseti, S., Dascalu, D., Calin, M., Dascalu, M., Trausan-Matu, S., and Militaru, G. (2020). *Comprehensive Exploration of Game Reviews Extraction and Opinion Mining Using NLP Techniques*, pages 323–331.

Schoenfeld, M., Zimmermann, A., and Crémilleux, B. (2018). Preprocessor selection for machine learning pipelines. *arXiv preprint arXiv:1810.09942*. Submitted to Machine Learning journal.

Tan, J. Y., Chow, A. S. K., and Tan, C. W. (2022). A comparative study of machine learning algorithms for sentiment analysis of game reviews. *The Journal of The Institution of Engineers, Malaysia*, 82(3):63–68. Special Edition: International Conference on Digital Transformation and Applications 2021 (ICDXA 2021).