International Journal of Multidisciplinary Research and Publications
ISSN (Online): 2581-6187

# Optimized XGBoost for Big Data in Cybersecurity: The Impact of the scale_pos_weight Parameter on Stealth Threat Detection

Patrick NZUAU NZINGA[1], Trésor KALONZO MUSENGA[2]
[1]Secrétariat Général, Ministère de l'Economie Numérique, Kinshasa, R.D. du Congo
[2]Math-Informatique, ISP Popokabaka, Kwango, R.D. du Congo
Email address: patricknzuau@gmail.com[1], tresormusenga@gmail.com[2]

***Abstract***— *The detection of stealthy cyber threats in Big Data environments constitutes a major challenge due to the extreme imbalance between normal activities and rare attacks. XGBoost, a reference algorithm for such data volumes, offers the scale_pos_weight parameter to adapt its learning process to these imbalanced contexts. This article proposes an in-depth technical analysis of the impact of this parameter on detection performance. Through an empirical evaluation on the NSL-KDD and CIC-IDS2018 datasets, we demonstrate that an optimization of scale_pos_weight significantly improves the recall of the most stealthy threats (up to +54 points for U2R attacks), while maintaining the computational efficiency required by Big Data. We analyze the underlying mechanisms of this weighting, quantify the trade-off between recall and precision, and propose a concrete optimization framework for operational deployment in Security Operations Centers (SOC). Our results show that this approach outperforms resampling techniques like SMOTE by offering a better balance between detection performance and training time.*

***Keywords***— *XGBoost, Cybersecurity, Big Data, Intrusion Detection, Stealth Threats, Class Imbalance, scale_pos_weight, Machine Learning.*

## I. Introduction

In the contemporary digital ecosystem, *cyber threat detection* faces a dual challenge: the explosion of data volume (Big Data) and the increasing sophistication of *stealth attacks*. The latter, characterized by their low frequency and high discretion, generate faint signals drowned out in massive flows of legitimate activity. Traditional intrusion detection systems struggle to identify these anomalies in environments where *data is inherently imbalanced*: the ratio between malicious and normal activities can reach extreme proportions in certain industrial networks.

Faced with this problem, machine learning approaches, and particularly *gradient boosting* algorithms, have shown promising capabilities. *XGBoost* (eXtreme Gradient Boosting) has established itself as a reference due to its computational efficiency, its capacity to handle high-dimensional data, and its parametric flexibility. Among its hyperparameters, scale_pos_weight plays a crucial role for imbalanced binary classification problems by dynamically adjusting the relative importance of positive examples (threats) compared to negative ones (normal activities) during training.

This article presents an *in-depth technical analysis* of the impact of the scale_pos_weight parameter on stealth threat detection in Big Data environments. We examine the mathematical foundations of this mechanism, empirically evaluate its performance on realistic cybernetic datasets, and propose *optimization methodologies* adapted to the operational constraints of cybersecurity systems.

## II. Mathematical Foundations and XGBoost Mechanisms

### *a)* XGBoost Architecture and Objective Function

XGBoost relies on an *additive architecture of decision trees* built sequentially. Unlike random forests that use bagging, XGBoost implements an optimized version of *gradient boosting* where each new tree corrects the residual errors of the previous ones.

The regularized objective function is expressed as follows:

$L(t)= \sum_{i=1}^{n} l(y_i, \hat{y}_i^{(t-1)} f_t(x_i)) + \Omega(f_t)$ (Function 1)

where $l$ represents the differentiable loss function (log-loss for binary classification), $\hat{y}_i^{(t-1)}$ the prediction after $t-1$ iterations, $f_t$ the tree added at iteration t, and $\Omega$ the regularization term that controls the model's complexity.

### *b) Integration of scale_pos_weight in Optimization*

The scale_pos_weight parameter fundamentally modifies the *learning dynamics* by rescaling the gradients for the minority class. In a cybersecurity context where the positive class (threat) is underrepresented, this modification is expressed by a weighting of the contribution to the loss.

$$L_{weighted}^{(t)} = \sum_{i=1}^{n} w_i . l(y_i, \hat{y}_i^{(t-1)} f_t(x_i)) + \Omega(f_t)$$

(Function 2)

With $w_i = \begin{cases} \alpha \text{ si } y_i = 1 \\ 1 \text{ si } y_i = 0 \end{cases}$ incorporating

$\alpha = scale\_pos\_weight$.

The default value recommended for imbalanced problems is the ratio between negative and positive instances:

$$\alpha = \frac{Nne'gatif}{Npositif}$$ (Function 3).



Patrick NZUAU NZINGA and Trésor KALONZO MUSENGA, "Optimized XGBoost for Big Data in Cybersecurity: The Impact of the scale_pos_weight Parameter on Stealth Threat Detection," *International Journal of Multidisciplinary Research and Publications (IJMRAP)*, Volume 8, Issue 9, pp. 56-59, 2026.

This approach is equivalent to a *cost-sensitive optimization* where false negatives (undetected threats) are penalized α times more severely than false positives (false alarms).

*c) Implications for Stealth Threats*

Stealth attacks present characteristics that exacerbate the challenges of imbalance:

- *Extreme rarity*: occurrences can represent less than 0.01% of events.
- *Morphological variability*: polymorphism of attack signatures.
- *Limited contamination*: low proportion of examples available for training.

In this context, scale_pos_weight allows *amplifying the learning signal* originating from the rare positive examples without requiring resampling, which could introduce bias or remove critical information from the majority data. The mechanical impact manifests in the calculation of gain similarities for tree splits, favoring divisions that improve minority classification.

## III. Experimental Methodology

*a) Datasets and Preprocessing*

Our evaluation uses two datasets representative of modern cybersecurity challenges:

Tab. 1. Datasets used and characteristics

| Dataset | Samples | Features | Classe Minoritaire (%) | Type de Menaces |
|---|---|---|---|---|
| NSL-KDD | 148 517 | 41 | 19.69% | DOS, Probe, R2L, U2R |
| CIC-IDS2018 | 16 000 000+ | 80 | 0.25% | Bruteforce, Infiltration, Botnet |

Preprocessing includes *normalization of numerical features*, encoding of categorical variables, and *dimensionality reduction* via principal component analysis for highly correlated features. To manage Big Data volumes, we apply *stratified sampling* that preserves the original class distribution in the training and testing subsets.

*b) Experimental Setup*

Our implementation uses *XGBoost 3.1.1* with the following baseline parameters:

- objective = 'binary:logistic'
- max_depth = 6
- learning_rate = 0.1
- subsample = 0.8
- colsample_bytree = 0.8
- tree_method = 'hist' (optimized for large datasets)

The key variable of our study, scale_pos_weight, is tested using a *logarithmic grid* of values: [1, 5, 10, 50, 100, 500, 1000]. We also compare alternative imbalance management approaches: SMOTE resampling, class weighting, and the Class Weighted XGBoost algorithm.

*c) Evaluation Metrics*

In the critical context of cybersecurity, where the *costs of false negatives* (undetected threats) far exceed those of false positives (unnecessary alerts), we utilize a comprehensive set of metrics:

- *Recall (Sensitivity):* ability to detect all threats.
- *Precision*: proportion of correct alerts among triggered alerts.
- *F1-Score*: harmonic mean between precision and recall.
- *AUC-ROC:* overall discriminative capacity.
- *FPR (False Positive Rate):* rate of false alarms.
- *Score Fβ* with β=2: metric favoring recall.

## IV. Results and Analysis

*a) Comparative Performance of Configurations*

Tab. 2. Performance according to different values of scale_pos_weight on NSL-KDD

| scale_pos_weight | Recall | Précision | F1-Score | AUC-ROC | FPR (%) |
|---|---|---|---|---|---|
| 1 (défaut) | 0.721 | 0.893 | 0.798 | 0.912 | 1.34 |
| 10 | 0.845 | 0.862 | 0.853 | 0.941 | 1.98 |
| 50 | 0.912 | 0.801 | 0.853 | 0.928 | 3.45 |
| 100 | 0.938 | 0.742 | 0.829 | 0.919 | 5.12 |
| 500 | 0.961 | 0.603 | 0.741 | 0.891 | 9.87 |

The results demonstrate a *non-monotonic relationship* between scale_pos_weight and overall performance. An optimal value around 50 yields the best balance between recall (91.2%) and precision (80.1%). Beyond 100, we observe a *significant degradation in precision* with only marginal gains in recall, illustrating the fundamental trade-off between comprehensive detection and false alarms.

*b) Stealth Threat Analysis*

For the most subtle attack categories (U2R and R2L in NSL-KDD), the impact of scale_pos_weight is particularly pronounced:

Tab. 3. Stealth threat detection by category

| Type d'Attaque | scale_pos_weight=1 | scale_pos_weight=50 | Amélioration |
|---|---|---|---|
| User-to-Root (U2R) | 14.3% | 68.7% | +54.4 points |
| Remote-to-Local (R2L) | 22.1% | 73.9% | +51.8 points |
| Probing | 85.4% | 94.2% | +8.8 points |
| Denial of Service | 96.7% | 98.1% | +1.4 points |

*Stealth threats* (U2R, R2L) disproportionately benefit from the adjustment, with detection improvements exceeding 50 points. This is explained by their extreme rarity in the training data and their low linear separability from normal activities.

*c) Scalability and Big Data Performance*

The evaluation on CIC-IDS2018 (16M+ instances) reveals the *computational efficiency* of the approach. The use of tree_method='hist' combined with parallelization across 32 CPU cores allows a model to be trained in 42 minutes compared to 3 hours and 20 minutes for a naive configuration. Memory consumption exhibits *sub-linear growth* thanks to gradient histogram optimization.



Patrick NZUAU NZINGA and Trésor KALONZO MUSENGA, "Optimized XGBoost for Big Data in Cybersecurity: The Impact of the scale_pos_weight Parameter on Stealth Threat Detection," *International Journal of Multidisciplinary Research and Publications (IJMRAP)*, Volume 8, Issue 9, pp. 56-59, 2026.

The impact of scale_pos_weight on computational resources is negligible (< 2% overhead), as the weighting is integrated into the already parallelized gradient calculation phase. This *preserved efficiency* makes the technique applicable in near real-time cyber data streaming environments.

*d) Comparison with Alternatives*

Tab. 4. Imbalance management methods (average values over 5 runs)

| Méthode | Recall | Précision | F1-Score | Temps Entraînement |
|---|---|---|---|---|
| scale_pos_weight = 50 | 0.912 | 0.801 | 0.853 | 42 min |
| SMOTE + XGBoost | 0.884 | 0.823 | 0.852 | 67 min |
| Class Weighted | 0.901 | 0.795 | 0.845 | 45 min |
| Under-sampling | 0.933 | 0.652 | 0.768 | 28 min |

Our scale_pos_weight configuration outperforms alternatives in the *performance-time balance*. SMOTE demonstrates slightly superior precision but at the cost of a *significant computational overhead* (+60%) and a risk of generating unrealistic synthetic examples within the cyber threat feature space.

## V. Discussion

*a) Underlying Mechanisms for Improvement*

The effectiveness of scale_pos_weight in stealth threat detection is explained by several interdependent mechanisms, notably:

- *Gradient rebalancing*: By amplifying the gradients of positive examples, the algorithm pays closer attention to feature space regions harboring rare threats.
- *Information preservation*: Unlike under-sampling, all majority data is retained, preserving the informational richness of normal behaviors.
- *Dynamic adaptation:* The weighting applies throughout the training process, enabling a progressive correction of learning biases.

*b) Implications for Cybersecurity Systems*

The optimization of scale_pos_weight offers practical advantages for SOCs (Security Operations Centers), including:

- *Detection time reduction*: Improved recall on stealth threats decreases the average attacker *dwell time* within the network.
- *Analyst resource optimization*: A better recall-precision balance mitigates alert fatigue without compromising security.
- *Adaptability*: The parameter can be dynamically adjusted based on the criticality of protected assets.

However, our study also highlights *important limitations:*

- Output probability calibration is affected, necessitating a post-transformation step for interpretable risk scores.
- The optimality of scale_pos_weight is highly dependent on specific data distributions, requiring rigorous validation over representative timeframes.
- *Zero-day* attacks unrepresented in the training data do not benefit from this adjustment.

*c) Recommendations for Operational Deployment*

For production environment implementations, we propose the following optimization framework:

1. *Preliminary analysis*: Quantify the context-specific class imbalance (threat/normal activity ratio).
2. *Contextual grid search*: Test scale_pos_weight within the interval $[0.5 \times ratio, 2 \times ratio]$.
3. *Temporal validation*: Evaluate performance over distinct periods to identify concept drifts.
4. *Continuous adjustment*: Implement a periodic re-evaluation mechanism using recent data.

For environments with *strict latency constraints*, a value slightly below the theoretical ratio (e.g., $0.7 \times ratio$) may offer a better operational compromise by curbing false alarms.

## VI. Conclusion

This study demonstrates the significant impact of the XGBoost scale_pos_weight parameter on stealth threat detection within Big Data cybersecurity environments. Our technical analysis reveals that an *optimized value* (typically nearing the inverse ratio of class prevalence) substantially bolsters the detection of rare and subtle attacks, yielding gains exceeding 50 percentage points for certain categories.

The approach offers *distinct advantages* over alternative methods: preserved computational efficiency, the absence of resampling artifacts, and seamless integration into existing pipelines. These characteristics establish it as a viable solution for intrusion detection systems tasked with processing growing data volumes while sustaining high sensitivity to emerging threats.

*Research perspectives* include adapting similar mechanisms for multi-class problems (categorized threats), integration with deep learning techniques for unstructured features, and the development of dynamic self-adjusting weighting methods in response to evolving threat landscapes. In a context where cyber adversary sophistication advances more rapidly than defenses, the fine-tuning of machine learning algorithms remains an essential lever for securing a defensive advantage.

## REFERENCES

[1] Alenazi, M., & Mishra, S. (2024). Cyberattack Detection and Classification in IIoT systems using XGBoost and Gaussian Naïve Bayes. Engineering, Technology & Applied Science Research.
[2] Brownlee, J. (2020). How to Configure XGBoost for Imbalanced Classification. Machine Learning Mastery.
[3] Chen, T., & Guestrin, C. (2016). XGBoost: A Scalable Tree Boosting System. Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining.
[4] KAFUNDA KATALAY Pierre, Apprentissage artificiel basé sur les machines à vecteur de support pour le traitement des classes déséquilibrées : Application à la détection des intrusions., Anales de la faculté des Sciences, volume 1(2019), UNIKIN, 51 – 61
[5] Sharafaldin, I., et al. (2018). Toward Generating a New Intrusion Detection Dataset and Intrusion Traffic Characterization. *ICISSP*.
[6] Sharafaldin, I., Lashkari, A. H., & Ghorbani, A. A. (2018). Toward generating a new intrusion detection dataset and intrusion traffic characterization. *4th International Conference on Information Systems Security and Privacy (ICISSP)*.
[7] Voros, M. (2023). Handling Imbalanced Datasets with XGBoost. Medium.



Patrick NZUAU NZINGA and Trésor KALONZO MUSENGA, "Optimized XGBoost for Big Data in Cybersecurity: The Impact of the scale_pos_weight Parameter on Stealth Threat Detection," *International Journal of Multidisciplinary Research and Publications (IJMRAP)*, Volume 8, Issue 9, pp. 56-59, 2026.

[8] XGBoost Developers. (2024). XGBoost Parameters Documentation.
[9] XGBoosting.com. (2023). XGBoost Configure "scale_pos_weight" Parameter.



Patrick NZUAU NZINGA and Trésor KALONZO MUSENGA, "Optimized XGBoost for Big Data in Cybersecurity: The Impact of the scale_pos_weight Parameter on Stealth Threat Detection," *International Journal of Multidisciplinary Research and Publications (IJMRAP)*, Volume 8, Issue 9, pp. 56-59, 2026.