Information, Computing and Intelligent Systems, 2025, No. 6, 75 – 86

UDC 004.42:004.8 https://doi.org/10.20535/2786-8729.6.2025.333898

DETECTION METHOD OF FRAUDULENT PAYMENT TRANSACTION BASED ON *C*-SCORE METRIC

Dmytro Korynetskyi *
National Technical University of Ukraine
"Igor Sikorsky Kyiv Polytechnic Institute", Kyiv, Ukraine
https://orcid.org/0009-0003-2777-1921

Inna V. Stetsenko
National Technical University of Ukraine
"Igor Sikorsky Kyiv Polytechnic Institute", Kyiv, Ukraine
https://orcid.org/0000-0002-4601-0058

*Corresponding author: d.korynetskyi@kpi.ua

Fraud detection for payment transactions is a cost-sensitive task, as the costs associated with misclassification – such as missing a fraudulent transaction or incorrectly blocking a legitimate one – can vary significantly depending on business priorities. Traditional evaluation metrics, particularly the *F1*-score, ignore this asymmetry, creating a need for more flexible approaches. This research focuses on developing a method for building adaptive, cost-sensitive fraud detection systems. The aim is to develop a method that enables the practical application of the cost-sensitive *C*-score metric to configure a multi-level decision logic. The paper also presents a possible software architecture for its implementation.

The proposed two-phase method (offline calibration and online scoring) uses the *C*-score metric to determine multiple decision thresholds corresponding to different business scenarios. Its validation was conducted on the public "Credit Card Fraud Detection" dataset using the XGBoost algorithm. The Synthetic Minority Over-sampling Technique (SMOTE) was applied to overcome the severe class imbalance in the data, and a comparison was made against the traditional *F1*-score-based approach.

The experimental results showed that the proposed approach allows for the identification of two distinct thresholds from a single classifier. The first threshold ensures high precision, making it suitable for automated blocking of payment transactions with minimal false positives. The second threshold, focused on high recall, enables the selection of suspicious payment transactions for subsequent manual review. It was also confirmed that the SMOTE significantly contributed the model's class separation ability, thereby increasing the reliability of calibrating these thresholds. Based on the method, a practical blueprint for a service-oriented architecture is proposed for creating flexible and configurable anti-fraud systems.

Keywords: fraud detection, C-score, F1-score, cost-sensitive learning, anti-fraud software.

1. Introduction

The rapid growth of e-commerce and online payments has led to a significant increase in fraudulent activities. Financial losses from such activities affect various participants in payment systems, including companies, merchants, card issuers, payment processors and cardholders. For example, in credit card fraud, merchants often face chargebacks and a loss of consumer trust [1]. According to the Nilson Report, global losses from payment card fraud reached substantial figures in recent years, with projections indicating further increases, making this a persistent and costly issue [2]. These substantial and growing losses underscore the need for more effective fraud detection methods. The core of the challenge lies in the fact that, depending on the business specifics, the costs of misclassification can vary greatly. In practice, a false negative – missing a fraudulent transaction – can be significantly more costly than a false positive, which involves flagging a legitimate transaction as fraudulent. Consequently, there is a demand for methods that allow a business to configure the detection system in alignment with its operational priorities and risk strategies. The development of such flexible and cost-aware systems therefore represents a timely and relevant research direction.

2. Literature review and problem statement

Numerous studies have applied machine learning to credit card fraud detection. Common approaches include logistic regression, decision trees, and powerful ensemble methods like Random Forests and XGBoost [3]. A significant and well-documented challenge in this domain is the severe class imbalance, as fraudulent transactions typically represent a very small fraction of the total volume. Techniques like the Synthetic Minority Over-sampling Technique (SMOTE) are often employed to address this issue by creating synthetic examples of the minority class, thus balancing the training data distribution [4].

The evaluation of Fraud Detection System (FDS) performance is crucial for practical deployment. While Precision, Recall, and their harmonic mean, the *F1*-score, are widely reported, they are fundamentally cost-agnostic. The *F1*-score, by its definition, assumes equal importance for these two measures [5]. This is a major limitation in fraud detection, where, for instance, failing to detect a large fraudulent transaction (a false negative) can be orders of magnitude more costly than incorrectly flagging a legitimate one (a false positive), which might only lead to customer inconvenience.

Cost-sensitive learning aims to resolve this by incorporating misclassification costs directly into the model's training or evaluation process [6]. The *C*-score metric is a modern metric specifically designed to evaluate classifiers in a cost-sensitive manner by considering the relative cost ratio (r_c) of false negatives to false positives [7].

A review of the literature shows that while the field has established components for building FDS, including classification algorithms like XGBoost, data balancing techniques like SMOTE, and cost-sensitive evaluation metrics a gap remains in their practical integration. While the *C*-score metric was introduced as a cost-aware alternative to the *F1*-score, the original research focused on the metric itself and its assessment capabilities. It did not, however, propose a specific method for integrating this metric into a configurable software architecture.

Thus, the unresolved problem is the lack of a comprehensive method that bridges the gap between cost-sensitive theory and operational practice. Specifically, what is absent is a method to operationalize a metric like the *C*-score within a software architecture that allows businesses to configure and automate actions based on their specific risk strategies.

3. The aim and objectives of the research

The aim of the research is to develop a method for building an adaptable FDS, centered on a software architecture that operationalizes the cost-sensitive *C*-score metric for configurable, multi-level risk segmentation.

To achieve this aim, the following objectives have been set:

– to develop a fraud detection method that operationalizes the *C*-score metric into a configurable, multi-threshold decision logic,

– to experimentally validate the proposed method by comparing its performance against the traditional *F1*-score-based approach and to evaluate the influence of the SMOTE data balancing technique on its effectiveness.

4. The study materials and methods for a *C*-score-based fraud detection method

This section details the materials and methods used to investigate the process of fraud detection in payment card systems, with a specific focus on developing a method for building a configurable, cost-sensitive system based on the *C*-score metric. The presentation is structured in three subsequent parts. First, the proposed detection method itself is described, focusing on its conceptual two-phase model. Next, a possible software architecture is outlined to illustrate a practical blueprint for the method's implementation. Finally, the setup for the experimental validation is presented, detailing the

dataset, procedures, and metrics used to empirically justify the core principles of the proposed method.

4.1. The proposed method

The core of this research is a novel method for detecting fraudulent transactions, based on a two-phase operational model: offline calibration and online scoring. The novelty of the method lies in its use of the cost-sensitive *C*-score metric to calibrate two separate decision thresholds (T_{low} and T_{high}), which enables a flexible, multi-level approach to risk management.

Offline Calibration and Multi-Threshold Configuration (Phase 1)

This preparatory phase of the method is executed periodically (e.g., daily or weekly) to establish operational decision thresholds. The process includes data preparation, model training, and optimal threshold calibration. The central step is the calibration of two distinct thresholds based on different business-driven cost scenarios:

– high-precision threshold (T_{high}) is determined by minimizing the *C*-score for a cost scenario that heavily penalizes false positives. This is suitable for fully automated actions like blocking a transaction;

– high-recall threshold (T_{low}) is found by minimizing the *C*-score for a cost scenario that heavily penalizes false negatives, thereby prioritizing fraud capture. This is ideal for flagging transactions for manual review.

A validation check is then performed to ensure that $T_{high} > T_{low}$ before the thresholds persisted for online use.

Online Risk-Segmented Transaction Scoring (Phase 2)

This phase of the method executes in real-time for each new transaction. The procedure leverages the pre-calibrated thresholds to segment transactions into three risk categories. It involves generating a probability score (P) for a transaction and then applying the multi-threshold logic. Based on the outcome of comparing P with T_{high} and T_{low}, a specific action is executed: automatic blocking ($P \geq T_{high}$), flagging for manual review ($P \geq T_{low}$), or automatic approval. This tiered logic provides a granular, cost-sensitive way to balance automation with expert oversight.

4.2. Architectural implementation

The two-phase method described in the previous section can be implemented using a service-oriented software architecture. This section outlines one such possible implementation, which serves as a practical blueprint. The proposed method primarily enhances the Data Driven Model (DDM) component in an FDS, which typically operates in near real-time to score transactions as they occur.

In this architectural model, the real-time scoring phase (Phase 2 of the method) involves several interacting components. A central ScoringService would orchestrate the process for each incoming transaction. This service would invoke the pre-trained XGBoostModel to generate a probability score P and then use the multi-threshold logic defined by the method to determine the appropriate action. An ActionDispatcher component would then execute this action, for example, by sending a transaction to a ManualReviewSystem or confirming its approval. The detailed sequence of interactions between these key software components during the online scoring phase is illustrated in Figure 1.

The key advantage of this service-oriented architecture is its modularity. It decouples the core classification logic, encapsulated in the XGBoostModel component, from the business-level decision-making and action-handling logic, which is managed by the ActionDispatcher. This separation of concerns not only makes the system easier to maintain but also provides the flexibility to modify business rules or actions without needing to retrain the underlying model.

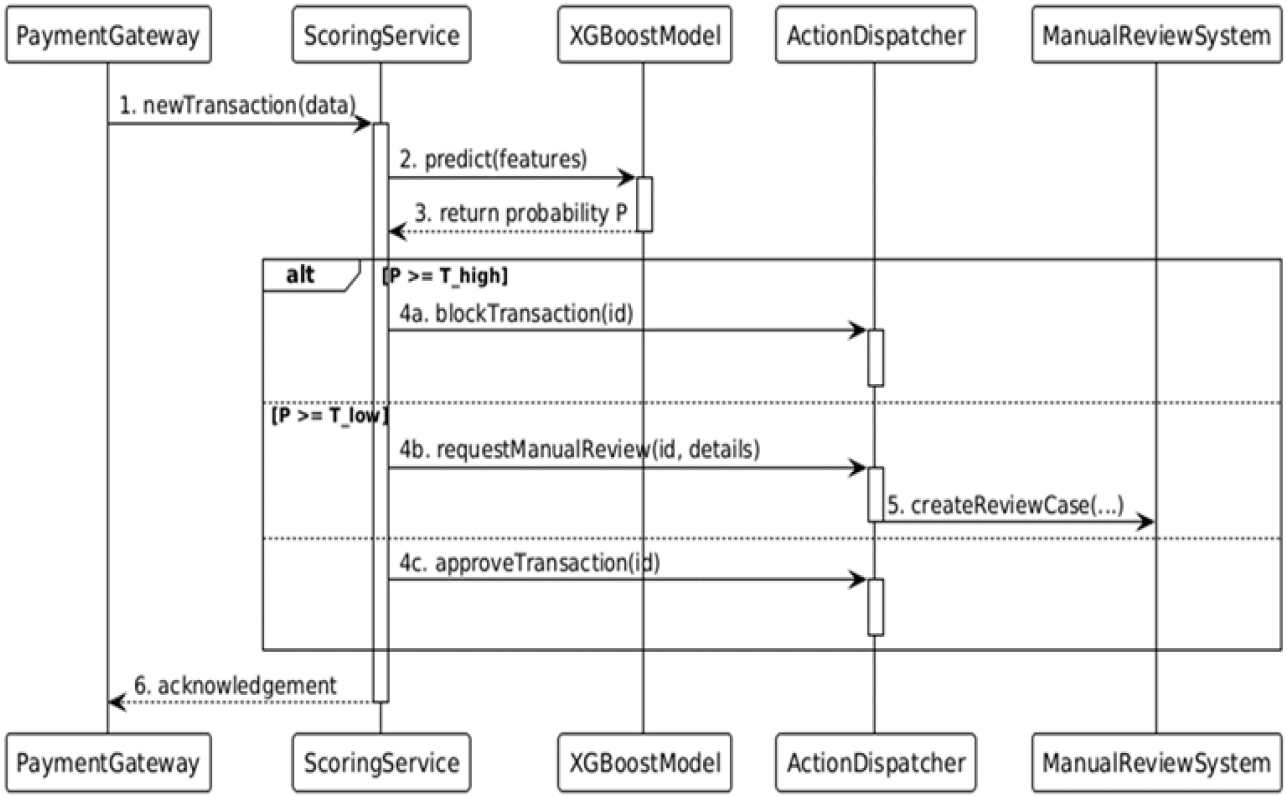


Fig. 1. Sequence diagram for real-time transaction scoring

Having established this practical design, the following section details the experimental setup used to empirically validate the core principle of the method, which serves as the foundation for this architecture.

4.3. Experimental setup for method validation

This section describes the setup of the experiments conducted to validate the core principles of the proposed method. The setup includes a description of the public dataset used for the study, the experimental procedure for model training and threshold selection, the evaluation metrics and cost model applied, and the data balancing technique used to address class imbalance.

Dataset description

The study utilized the public "Credit Card Fraud Detection" dataset from the Kaggle platform [8]. This dataset contains anonymized transactions made by European cardholders. It is characterized by severe class imbalance, with 492 fraudulent transactions (Class 1) out of 284,807 total transactions, which constitutes only 0.172% of the data. The features consist of 28 principal components obtained via PCA (V1–V28) and the transaction amount.

Experimental procedure

To simulate a real-world scenario where a model is trained on past data to predict future events, the dataset was split chronologically. The first 80% of transactions were allocated to the training set, and the subsequent 20% formed the test set. The XGBoost algorithm, implemented via the xgboost Python library, was selected as the classification model due to its state-of-the-art performance in many tabular data tasks [9]. The model was trained using its default hyperparameters to focus the research on the effectiveness of the threshold selection strategy itself, rather than on extensive hyperparameter tuning. After the model generates probability scores for the test set, two distinct strategies for selecting the optimal decision threshold were implemented and compared: *F1*-score-based selection and *C*-score-based selection.

Data balancing technique

The SMOTE was applied to the training data in one of the experimental runs. It addresses class imbalance by generating new, synthetic examples of the minority class (fraudulent transactions). For this research, the sampling strategy was set to 1, meaning synthetic samples were generated until the number of fraudulent transactions in the training set equaled the number of legitimate ones. The test set was left in its original, imbalanced state to ensure a realistic evaluation.

Evaluation metrics and cost model

This research compares two metrics for decision threshold selection: the *F1*-score and the *C*-score. The *C*-score is a cost-sensitive metric calculated using the formula [7]:

$$C_{score} = \left(\frac{1}{Precision} - 1\right) \cdot Recall + r_c \cdot (1 - Recall), \quad (1)$$

where r_c is the cost ratio, defined as $r_c = \frac{Cost_{FN}}{Cost_{FP}}$.

To explore different business scenarios, three cost ratios were investigated:

– $r_c = 0.1$ if a false positive is 10 times more costly than a false negative,

– $r_c = 1$ if false positive and false negative have equal cost,

– $r_c = 10$ if a false negative is 10 times more costly than a false positive.

Each ratio represents a distinct business case. The ratio $r_c = 0.1$ is relevant for fully automated systems where blocking a legitimate user incurs a high reputational and business cost. The scenario with $r_c = 1$ aligns with the implicit assumption of the *F1*-score. Finally, $r_c = 10$ models a common scenario where preventing fraud is the highest priority.

Threshold selection strategy

After training, the XGBoost model outputs a probability score for each transaction on the test set. To convert these probabilities into a binary classification (Fraud/Benign), a decision threshold must be applied. Instead of evaluating a predefined grid of thresholds, a more precise approach was employed. The *precision_recall_curve* function from the scikit-learn library was used to generate all unique probability scores produced by the model on the test set. These scores serve as candidate thresholds. For each candidate threshold, the corresponding Precision and Recall values are calculated.

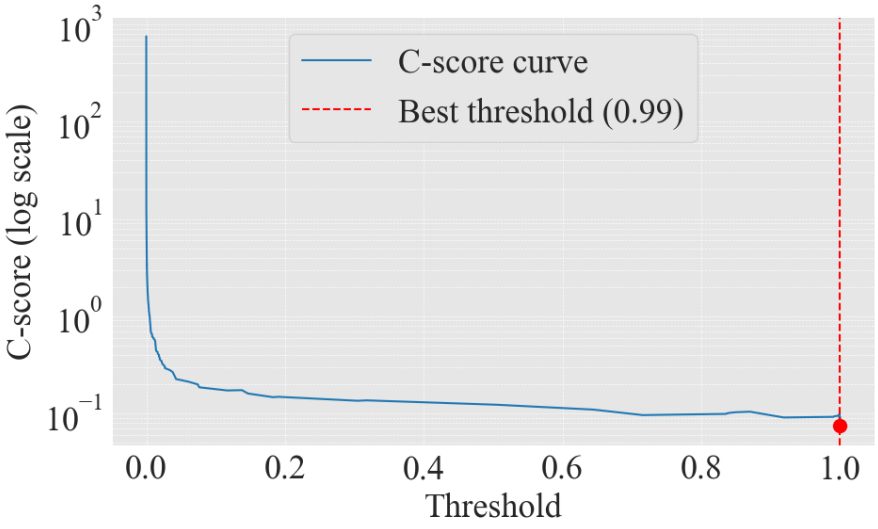
Based on these values, two distinct strategies for selecting the final threshold are implemented and compared:

1) *F1*-score-based selection,

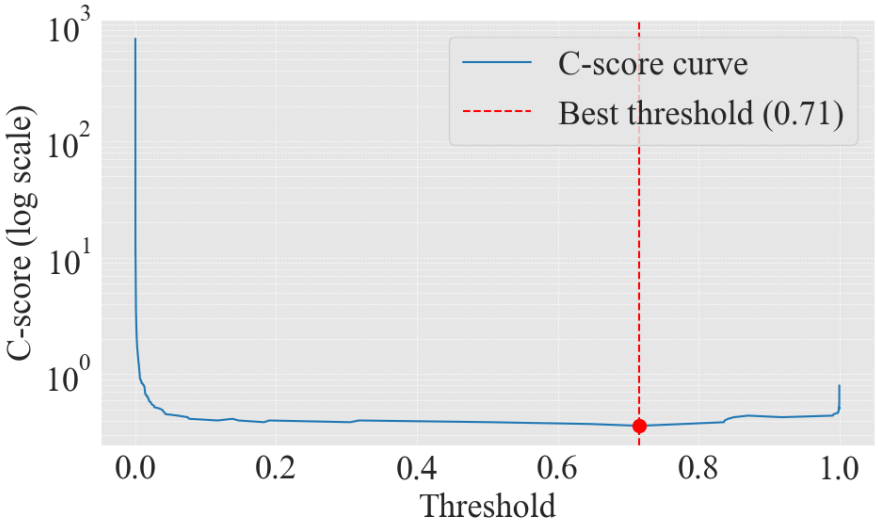
2) *C*-score-based selection.

In the first case, the *F1*-score was calculated for each candidate threshold, and the one that yielded the maximum *F1*-score was chosen as optimal for this strategy. For the second case, the *C*-score was calculated for each candidate threshold and for each defined cost ratio (r_c). The threshold that resulted in the minimum *C*-score was selected as the optimal one for that specific cost scenario. This process was repeated for $r_c = 0.1$,1 and 10.

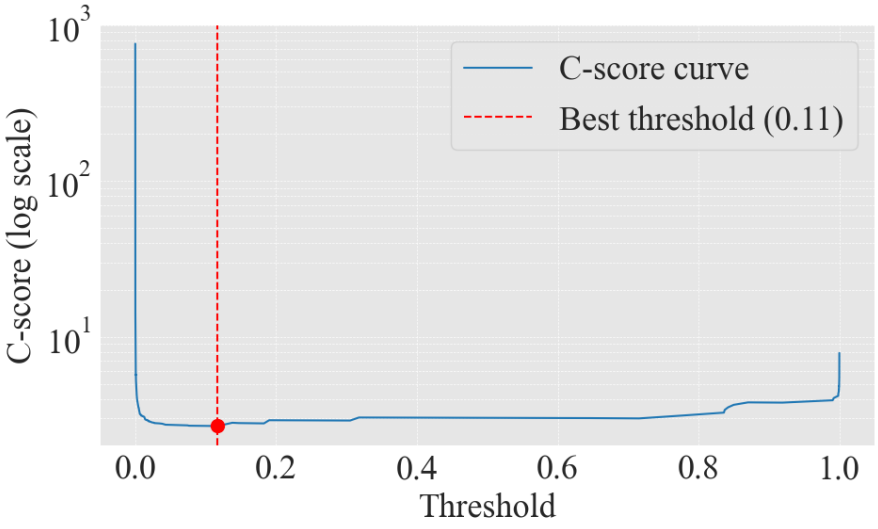
The process of finding the minimum *C*-score for different cost ratios is illustrated in Figure 2. The optimal threshold is identified at the lowest point of each curve. Unlike the *F1*-score, the *C*-score has no upper bound. It returns a value greater than or equal to 0 proportional to the total cost of misclassification, where 0 represents a perfect outcome and higher values indicate worse, more costly performance. This property explains the curve behavior. At very low threshold values (the left part of the graph), the model precision becomes extremely low, causing the *C*-score to rise to very high values. Conversely, in the region near the optimal threshold, the metric value approaches its minimum. To clearly visualize these changes, a logarithmic scale was used for the Y-axis. This approach allows for a detailed examination of the curve behavior near its minimum point.



a



b



c

Fig. 2. *C*-score vs. threshold for different cost ratios, demonstrating the selection of the optimal threshold (minimum point) for each scenario: *a* – cost ratio is 0.1 ($r_c = 0.1$), *b* – cost ratio is 1 ($r_c = 1$), *c* – cost ratio is 10 ($r_c = 10$).

As shown in Figure 2, the optimal decision threshold is highly dependent on the specified cost ratio r_c. For instance, when false positives are heavily penalized ($r_c = 0.1$), the optimal threshold shifts to a very high value (0.99) to ensure high precision. Conversely, when false negatives are more costly ($r_c = 10$), the optimal threshold moves to a much lower value (0.11) to prioritize recall. This empirical evidence is fundamental to the proposed method, as it confirms the feasibility of calibrating distinct T_{high} and T_{low} thresholds.

5. Results of investigating the effectiveness of the C-score-based detection method

This section presents the experimental results that provide the empirical foundation for the multi-threshold detection method proposed in Section 4.1. The primary purpose of these experiments is not to merely re-validate the *C*-score metric itself, but to demonstrate the feasibility of our proposed method. This involves showing that by applying the *C*-score with different cost ratios, it is possible to identify two distinct and operationally useful thresholds (T_{high} and T_{low}) from a single trained classifier. This capability is the prerequisite for the risk-segmentation logic central to our method.

Furthermore, the experiments validate this multi-threshold approach using the XGBoost algorithm, extending its applicability beyond the RandomForest model used in the original *C*-score research. The influence of the SMOTE data balancing technique on the quality and separation of these thresholds is also assessed.

5.1. Experiment 1: model trained on imbalanced data

The first experiment involved training the XGBoost classifier on the original, highly imbalanced training data. The goal was to compare the C-score values of two threshold selection strategies: one based on maximizing the *F1*-score and the other on minimizing the *C*-score.

As the *C*-score is a cost-based metric, its objective is to be minimized; a lower *C*-score value signifies a better, more cost-effective outcome as it represents a lower total cost from misclassifications.

Table 1 summarizes this comparison. It presents the key classification metrics (Precision, Recall) and the final calculated *C*-score for the optimal thresholds that were identified using each of the two strategies [10].

Table 1. Comparison of results using thresholds found from *F1*-score and *C*-score for XGBoost trained on imbalanced data

Cost ratio	The threshold value is selected using *F1*-score				The threshold value is selected using *C*-score				*C*-score savings (%)
	Optimal threshold value	Precision	Recall	*C*-score	Optimal threshold value	Precision	Recall	*C*-score	
0.1	0.715	0.913	0.706	0.096	0.999	0.951	0.520	0.074	22.2%
1				0.360	0.715	0.913	0.706	0.360	0%
10				3.000	0.116	0.835	0.746	2.680	10.67%

The results presented in Table 1 provide the initial validation for the proposed method. Even when trained on imbalanced data, the *C*-score-based strategy successfully identifies different optimal thresholds for unequal error costs ($r_c = 0.1$ or $r_c = 10$). This finding confirms the method's core principle: that it is possible to derive distinct, business-aligned operational points from a single model. The resulting cost savings demonstrate the immediate practical benefit of this approach compared to the cost-agnostic *F1*-score.

5.2. Experiment 2: model trained on SMOTE-balanced data

In line with the second research objective, this experiment evaluates the influence of the SMOTE data balancing technique on the proposed method's effectiveness. To achieve this, the

SMOTE technique was applied to the training set to create a balanced 50/50 distribution of classes, after which the model was retrained and evaluated on the same original, imbalanced test set. The hypothesis is that by training the model on a balanced dataset, the subsequent calibration of cost-sensitive thresholds will become more reliable and distinct, thereby demonstrating the advantages of the *C*-score-based selection strategy. The results of this experiment are presented in Table 2.

Table 2. Comparison of results for XGBoost trained on SMOTE-balanced data

Cost ratio	The threshold value is selected using *F1*-score				The threshold value is selected using *C*-score				*C*-score savings (%)
	Optimal threshold value	Precision	Recall	*C*-score	Optimal threshold value	Precision	Recall	*C*-score	
0.1	0.992	0.964	0.720	0.054	0.999	1.000	0.653	0.034	36.59%
1				0.306	0.992	0.964	0.720	0.306	0%
10				2.826	0.869	0.814	0.760	2.573	8.96%

The application of SMOTE improved the results, which in turn enhances the robustness of the proposed method. By training on a balanced dataset, the model achieved better class separation, allowing for a more defined and reliable calibration of the distinct T_{high} and T_{low} thresholds. As shown in Table 2, this led to a more pronounced cost-saving effect, particularly in the $r_c = 0.1$ scenario, reaching 36.59%.

5.3. Visual analysis of model behavior

Applying SMOTE noticeably altered the class-probability distribution: predictions shifted toward values near 0 and 1. This is reflected in the *F1*-score vs. threshold graph (Fig. 3), where the *F1*-score consistently increases until a very high threshold, after which it drops sharply. This indicates that the model assigns very high probabilities to true fraud cases and very low probabilities to legitimate ones, with few predictions in the middle "gray zone".

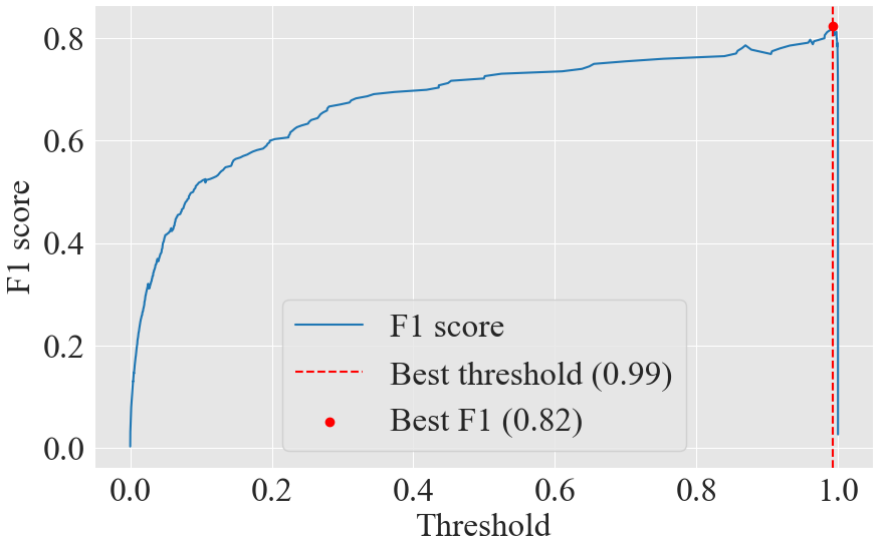


Fig. 3. *F1*-score as a function of Classifier Threshold for the model trained on SMOTE data.

To visualize the trade-offs, confusion matrices were plotted for the four optimal thresholds identified in the SMOTE experiment (one for *F1*-score, and three for *C*-score). The balance of errors shifts depending on the selection criteria, as illustrated in Figure 4.

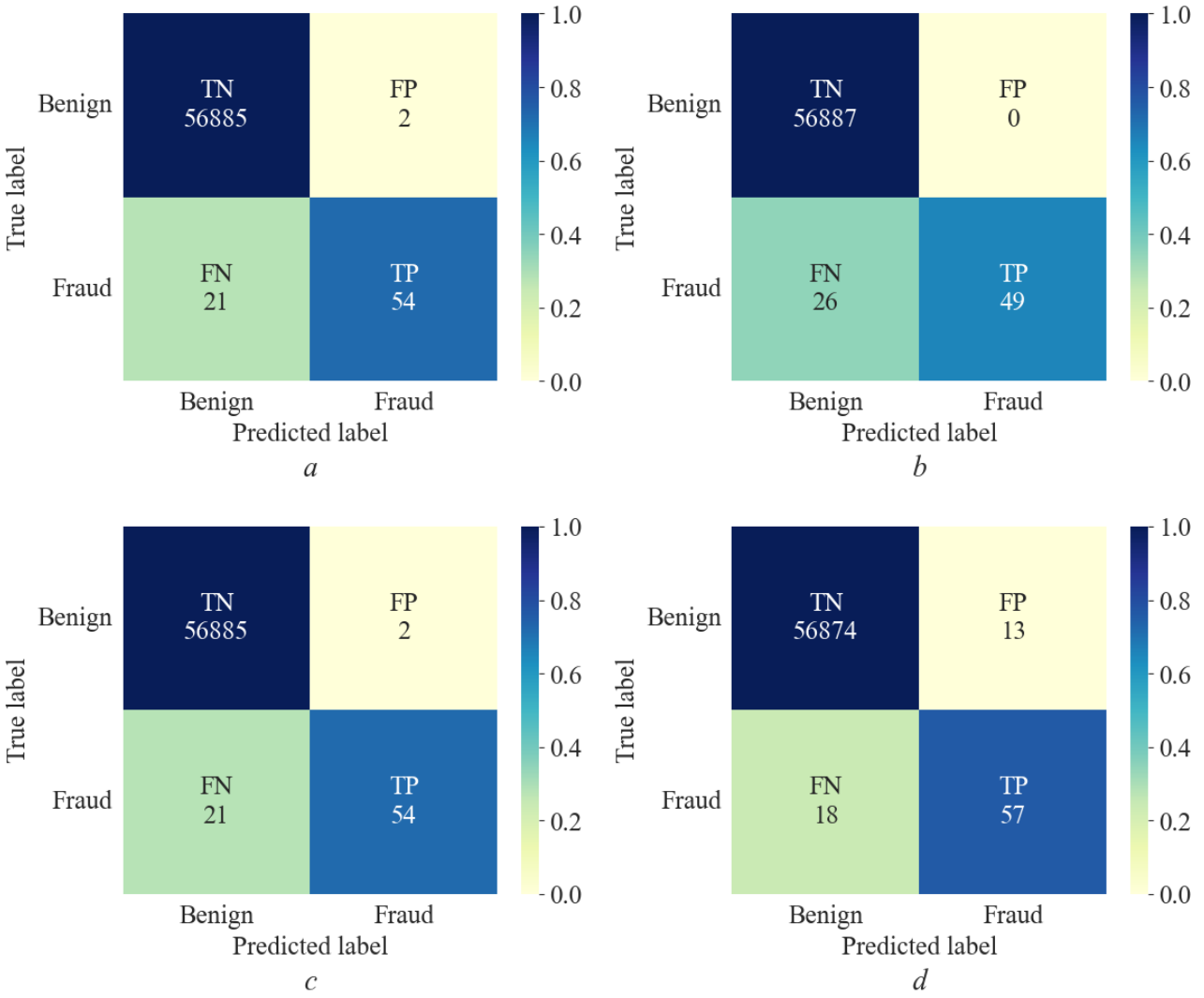


Fig. 4. Normalized confusion matrices for different optimal thresholds (SMOTE experiment): *a* – *F1*-score optimal, *b* – *C*-score ($r_c = 0.1$), *c* – *C*-score ($r_c = 1$), *d* – *C*-score ($r_c = 10$).

The visual analysis of the confusion matrices in Figure 4 provides clear insights into the model behavior under different threshold selection criteria. The *F1*-score optimal threshold (Fig. 4*a*) results in a balanced outcome with 21 false Negatives and 2 false positives. When the cost ratio is set to $r_c = 0.1$ (Fig. 4*b*), where false positives are highly penalized, the model becomes extremely conservative, achieving zero false positives at the cost of an increased number of false negatives (26). Conversely, for $r_c = 10$ (Fig. 4*d*), where missing a fraudulent transaction is the most critical error, the model adjusts to minimize false negatives, reducing their count to 18, but this is achieved at the expense of a significant increase in false positives to 13. As expected, the result for $r_c = 1$ (Fig. 4*c*) is nearly identical to the *F1*-score optimal, confirming that both metrics converge when error costs are equal. Ultimately, this visual analysis provides clear evidence of the proposed method's practical utility: it empowers a business to consciously choose its operational strategy, whether that is the conservative, high-precision approach ideal for T_{high}, or the high-recall approach needed to define T_{low} for manual review.

6. Discussion of the obtained results

The experimental results confirm the key hypothesis of this research: the proposed method, based on a multi-threshold logic guided by the *C*-score metric, allows for the construction of a more flexible and efficient FDS compared to the traditional *F1*-score-based approach.

The primary advantage of the method lies in its capacity for cost-sensitive configuration. The *F1*-score, by balancing precision and recall, is inherently cost-agnostic and is only optimal from a cost perspective when misclassification costs are equal ($r_c = 1$). In all other, more realistic scenarios, it yields a suboptimal solution in terms of business costs.

In contrast, the *C*-score guides the threshold selection towards a point that minimizes the total misclassification cost according to a business-defined cost function. This principle is clearly demonstrated by the experimental data. For automated blocking (high-precision), the scenario where a false positive is ten times more costly than a false negative ($r_c = 0.1$) guides the *C*-score optimization to a very high threshold (e.g., 0.99 in the SMOTE experiment). As shown in the confusion matrix (Fig. 4*b*), this successfully eliminated false positives (FP = 0), validating this threshold (T_{high}) as ideal for automated actions that minimize friction for legitimate customers. Conversely, for manual review (high-recall), where missing fraud is the most critical error ($r_c = 10$), the *C*-score selects a much lower threshold. This approach maximizes fraud capture by reducing the count of false negatives, justifying its use as the T_{low} threshold to flag transactions for manual analysis where preventing fraud is the highest priority.

The application of the SMOTE technique contributed significantly to these results. By balancing the training dataset, the XGBoost model learned to separate the classes more effectively, which in turn made the calibration of the T_{high} and T_{low} thresholds more distinct and reliable. Furthermore, the successful application of this method with the XGBoost algorithm extends the findings of the original *C*-score research, which used a RandomForest model, demonstrating its compatibility with other, high-performance algorithms. These findings are therefore not merely a theoretical comparison of metrics but the empirical foundation that proves the practical feasibility of the proposed two-phase software architecture.

While the proposed method has demonstrated its effectiveness, several promising directions exist for its future extension and refinement. Firstly, the robustness of the performance estimates could be enhanced using more advanced validation strategies like Prequential validation, while systematic hyperparameter tuning could unlock further performance gains [11]. A particularly valuable extension would be the development of dynamic cost models, where the cost of a misclassification is weighted by the transaction amount, allowing the system to react more flexibly to higher-risk scenarios. Finally, the proposed architecture can be evolved towards a highly individualized system by incorporating customer-specific features, such as transaction history or spending patterns, to create personalized, adaptive thresholds. The logical next step involves the full-scale implementation and deployment of this software architecture to evaluate its real-world performance and scalability.

Conclusion

This research introduced and validated a method for developing an adaptable fraud detection system based on a multi-threshold software architecture.

First, a fraud detection method was developed, realized through a specific software architecture. The proposed method operationalizes the cost-sensitive *C*-score metric to establish a configurable, multi-threshold decision logic. This allows for the segmentation of transactions into different risk levels, enabling the system's behavior to be aligned with specific business requirements through adjustable parameters.

Second, the method was experimentally validated and compared against a traditional *F1*-score-based approach. The results demonstrated that selecting decision thresholds based on the *C*-score metric leads to a more favorable balance of classification errors according to predefined cost scenarios. The study also confirmed that applying the SMOTE data balancing technique improves the model's ability to distinguish between classes, thereby enhancing the overall effectiveness of the proposed method.

References

[1] R. J. Bolton and D. J. Hand, "Statistical Fraud Detection: A Review," Statistical Science, vol. 17, no. 3, pp. 235–255, Aug. 2002, https://doi.org/10.1214/ss/1042727940.

[2] "Card and Mobile Payment Fraud Worldwide" The Nilson Report, no. 1276, Dec. 2024. [Online]. Available: https://nilsonreport.com/newsletters/1276/.

[3] J. O. Awoyemi, A. O. Adetunmbi, and S. A. Oluwadare, "Credit card fraud detection using machine learning techniques: A comparative analysis," in 2017 International Conference on Computing Networking and Informatics (ICCNI), Oct. 2017, pp. 1–9, https://doi.org/10.1109/ICCNI.2017.8123782.

[4] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "SMOTE: Synthetic Minority Over-sampling Technique," jair, vol. 16, pp. 321–357, Jun. 2002, https://doi.org/10.1613/jair.953.

[5] D. M. W. Powers, "Evaluation: from precision, recall and F-measure to ROC, informedness, markedness and correlation," Oct. 11, 2020, arXiv: arXiv:2010.16061. http://doi.org/10.48550/arXiv.2010.16061.

[6] C. Elkan, "The foundations of cost-sensitive learning," in Proc. 17th Int. Joint Conf. on Artificial Intelligence (IJCAI'01), 2001, vol. 17, pp. 973–978. Accessed: Jun. 26, 2025. [Online]. Available: https://cseweb.ucsd.edu/~elkan/rescale.pdf.

[7] M. Marwah, A. Narayanan, S. Jou, M. F. Arlitt, and M. Pospelova, "Is F1 Score Suboptimal for Cybersecurity Models? Introducing C-score, a Cost-Aware Alternative for Model Assessment," in Proc. CAMLIS 2024 Workshop, 2024. [Online]. Available: http://ceur-ws.org/Vol-3920/paper11.pdf.

[8] Machine Learning Group - ULB, "Credit Card Fraud Detection," Kaggle. Accessed: May 26, 2025. [Online]. Available: https://www.kaggle.com/datasets/mlg-ulb/creditcardfraud.

[9] XGBoost Developers, "XGBoost Documentation." Accessed: May 26, 2025. [Online]. Available: https://xgboost.readthedocs.io/en/stable/.

[10] D. S. Korynetskyi and I. V. Stetsenko, "Vykorystannia metryky C-score dlia optymizatsii metodiv vyiavlennia shakhraiskykh platizhnykh operatsii," in Inzheneriia prohramnoho zabezpechennia i peredovi informatsiini tekhnolohii (SoftTech-2025): materialy VIII Mizhnarodnoi naukovo-praktychnoi konferentsii molodykh vchenykh ta studentiv, May 13–15, 2025, Kyiv, FIOT, pp. 41–44. Accessed: Jun. 26, 2025. [Online]. Available: https://drive.google.com/file/d/1XLXmL-vxpT7X10OqdHF2Yg00ztbxbaxK/view.

[11] Y. Le Borgne, G. Bontempi, et al., "Fraud Detection Handbook." Accessed: Jun. 26, 2025. [Online]. Available: https://fraud-detection-handbook.github.io/fraud-detection-handbook/Chapter_5_ModelValidationAndSelection/ValidationStrategies.html.

УДК 004.42:004.8

МЕТОД ВИЯВЛЕННЯ ШАХРАЙСЬКИХ ПЛАТІЖНИХ ОПЕРАЦІЙ З ВИКОРИСТАННЯМ МЕТРИКИ *C*-SCORE

Коринецький Дмитро *
Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського», Київ, Україна
https://orcid.org/0009-0003-2777-1921

Інна Стеценко
Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського», Київ, Україна
https://orcid.org/0000-0002-4601-0058

Виявлення шахрайських платіжних операції є вартісно-чутливою задачею, оскільки вартість помилок, таких як пропуск шахрайських або помилкове блокування легітимних транзакцій, суттєво різниться і залежить від бізнес-пріоритетів. Традиційні методи оцінки, зокрема *F1*-score, ігнорують цю асиметрію, що створює потребу в розробці більш гнучких підходів.

Дослідження сфокусоване на розробці методу для побудови адаптивних, вартісно-чутливих систем виявлення шахрайських платіжних операції. Метою роботи є розробка методу, який дозволяє практично застосувати вартісно-чутливу метрику *C*-score для налаштування багаторівневої логіки прийняття рішень. У роботі також представлена можлива архітектура програмного забезпечення для реалізації цього методу.

Запропонований двофазний метод (офлайн-калібрування та онлайн-скоринг) використовує метрику *C*-score для визначення кількох порогових значень, що відповідають різним бізнес-сценаріям. Його валідація проводилася на публічному наборі даних «Credit Card Fraud Detection» за допомогою алгоритму XGBoost. Для подолання сильного дисбалансу класів у даних було застосовано метод синтетичного доповнення вибірки меншого класу (SMOTE), а результати порівнювалися з традиційним підходом на основі *F1*-score.

Результати експериментів показали, що запропонований підхід дозволяє ідентифікувати два різні пороги з одного класифікатора. Перший поріг забезпечує високу точність, що дозволяє мінімізувати кількість хибних спрацювань і використовувати його для автоматичного блокування платіжних транзакцій. Другий поріг, орієнтований на високу повноту, дає змогу відбирати підозрілі платіжні транзакції для подальшого ручного аналізу. Було також підтверджено, що SMOTE значно покращує здатність моделі до розділення класів, що підвищує надійність калібрування цих порогів. На основі методу запропоновано сервісно-орієнтовану архітектуру для створення гнучкої систем протидії шахрайству.

Ключові слова: виявлення шахрайства, *C*-score, *F1*-score, вартісно-чутливе навчання, програмне забезпечення протидії шахрайству.