IARJSET ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Impact Factor 8.311 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 13, Issue 11, November 2026

DetectPay: Payment Fraud Detection Using a Weighted Tree Ensemble and Isolation Forest Risk Scoring

Mohammed Junaith S.P[1], Monishkumar M.K[2], Dhruv S Jain[3], Faheem M[4], S Poornima[5]

Dept. of Computer Science and Engineering (Emerging Technologies), SRM Institute of Science and Technology, Vadapalani Campus Chennai, India[1,2,3,4]

Dept. of Computer Science and Engineering (Emerging Technologies), SRM Institute of Science and Technology, Vadapalani Campus Chennai, India[5]

Abstract: Payment fraud is rare, changes over time and is hidden among large volumes of legitimate transactions, so fixed rules and single classifiers trained on imbalanced data either miss fraud or raise too many false alarms. This paper presents DetectPay, an end-to-end payment-fraud detection system that combines a weighted ensemble of three supervised tree-based learners (a class-weighted Random Forest, XGBoost and LightGBM) with an unsupervised Isolation Forest. A risk engine merges the ensemble fraud probability and the anomaly score into a bounded risk score, assigns a Low, Medium, High or Critical category and is exposed through a FastAPI service and a Streamlit workspace that also accepts screenshots and PDF statements through optical character recognition (OCR) with an editable review step. On a dataset of 15,000 transactions with 3.00% fraud, evaluated on a stratified hold-out set of 3,000 transactions, the ensemble at a decision threshold of 0.40 reached a precision of 0.43, a recall of 0.46, an F1-score of 0.44, an area under the precision–recall curve (AUC-PR) of 0.42 and a ROC-AUC of 0.89, a higher F1-score than any individual model (0.29–0.37). Stratified five-fold cross-validation gave a mean F1-score of 0.40 ± 0.06. The results show that ensembling and threshold selection improve the balance between missed fraud and false alarms, and they quantify how far a small, imbalanced dataset limits recall.

Keywords: Fraud detection, ensemble learning, XGBoost, LightGBM, random forest, isolation forest, imbalanced classification, risk scoring

I. INTRODUCTION

Card payments, online checkout and digital wallets have made payments fast and convenient, and the same convenience attracts fraud through stolen card details, compromised devices, account take-over and social engineering. Fraud-prevention systems traditionally rely on hand-written rules such as amount limits or blocks on foreign transactions. Rules are transparent but static: they miss new fraud patterns and block genuine customers whose behaviour happens to cross a fixed limit. In practice it is the combination of signals—location, device, merchant, card-present status, verification results and recent transaction velocity—that separates a risky transaction from a legitimate one.

Machine learning can learn such combinations from labelled history, but fraud detection is a difficult learning problem. Fraud is rare, so a model that labels every transaction as legitimate already achieves high accuracy while detecting nothing; fraudulent and legitimate transactions overlap; and the patterns drift after training [6]. Analysts also need more than a binary label: they need a graded score and a reason, and in many investigations the evidence arrives as a screenshot or a PDF statement rather than as a structured record.

This paper presents DetectPay, a complete system that addresses these issues. Its contributions are: (1) a weighted ensemble of three diverse gradient-boosted and bagged tree models whose decision threshold is selected from the precision–recall trade-off; (2) a risk engine that merges the supervised fraud probability with an unsupervised Isolation Forest anomaly score into a bounded risk score and four risk categories; (3) a deployable application with a validated REST endpoint, an analyst workspace, OCR-based evidence upload with mandatory human review and a persistent analysis history; and (4) an honest evaluation with class-aware metrics, threshold analysis and cross-validation, which shows both the benefit of ensembling and the limits imposed by a small, imbalanced dataset.

II. RELATED WORK

Tree ensembles are the dominant learners for tabular fraud data. Breiman's Random Forest [1] averages decorrelated trees grown on bootstrap samples and is robust but needs class weighting when the positive class is rare. XGBoost [2] adds regularised, sparsity-aware gradient boosting, and LightGBM [3] accelerates boosting with histogram-based, leaf-wise growth, gradient-based one-side sampling and exclusive feature bundling. Combining diverse learners reduces the variance and bias of individual models [14].

Liu, Ting and Zhou proposed Isolation Forest [4], which isolates observations by random partitioning and needs no labels; Carcillo et al. [7] showed that unsupervised outlier scores add information to supervised credit-card fraud classifiers, although the gain depends on the data. Dal Pozzolo et al. showed that undersampling distorts predicted probabilities and must be corrected [5], and described a realistic fraud-detection setting with concept drift and delayed labels [6]. Oversampling techniques such as SMOTE [12] are an alternative way to handle imbalance. Bahnsen et al. [8] showed that aggregated and time-of-day features improve fraud detection. Saito and Rehmsmeier [10] demonstrated that precision–recall analysis is more informative than ROC analysis when the positive class is rare. For document evidence, the Tesseract engine [9] performs well on clean text but degrades on noisy scans. DetectPay builds on these findings by keeping supervised and unsupervised signals separate until a transparent risk engine, by preferring class weighting to resampling, by reporting precision–recall-based metrics, and by placing a human review step after OCR.

III. Dataset and Feature Engineering

A. Dataset

DetectPay uses a labelled payment-transaction dataset of 15,000 transactions recorded between May and July 2026, of which 450 (3.00%) are fraudulent. The records involve 2,490 users and 600 merchants in 14 merchant categories, three channels (online, in-store and ATM) and four device types, and all amounts are in euro. Each record contains identity fields, the timestamp, amount and currency, home and transaction country, a foreign-transaction flag, distance from home, channel and device, card-present and CVV-match flags, new-device and new-merchant flags, failed PIN attempts, account age, the user's average amount and the number of transactions in the last hour and 24 hours. Fraudulent transactions differ markedly from legitimate ones: 59.6% of fraud is foreign compared with 9.3% of legitimate transactions, the mean distance from home is 2,079 km compared with 255 km, 51.6% occurs at night compared with 24.6%, and the mean amount is 118.7 EUR compared with 63.4 EUR.

B. Engineered Features

A shared feature-engineering module is used both during training and at prediction time, so that the served model receives exactly the inputs it was trained on. From the timestamp it derives the hour, day of week and night and weekend flags. It computes the amount deviation d = amount / average user amount, a high-velocity flag v that is 1 when three or more transactions occurred in the last hour, and a behavioural-risk score

$$b = 0.35\tilde{d} + 0.25n_1 + 0.20n_2 + 0.10(z + v) \qquad (1)$$

where $\tilde{d} = \min(d, 20)/20$, n_1 and n_2 are the new-device and new-merchant flags and z is the night flag. The transaction identifier and raw timestamp are removed, and the six categorical fields are one-hot encoded, giving 59 numeric model inputs whose order is stored with the models and re-imposed on every incoming request.

C. Data Split and Class Imbalance

The data are divided by a stratified 80:20 split with a fixed seed of 42 into 12,000 training transactions (360 fraud) and 3,000 hold-out transactions (90 fraud). The hold-out set is used only for the final evaluation. To counter the 32:1 class imbalance without distorting predicted probabilities [5], class weighting is preferred to resampling: the Random Forest and LightGBM use balanced class weights, while XGBoost is trained with its default objective so that the ensemble combines a conservative and a recall-oriented learner.

IV. Methodology

A. Base Models

Four models are trained on the training set (Table I). The three supervised models estimate the probability of fraud; the Isolation Forest is fitted on the same features without the label. Gradient boosting adds regression trees h_m that fit the negative gradient of the loss, scaled by the learning rate η:

$$F_m(x) = F_{m-1}(x) + \eta\, h_m(x) \quad (2)$$

Isolation Forest scores an observation by its expected path length $E[h(x)]$ in random isolation trees; anomalies are isolated in fewer splits [4]:

$$s(x, n) = 2^{-E[h(x)] / c(n)} \qquad (3)$$

$$c(n) = 2H(n - 1) - 2(n - 1)/n \qquad (4)$$

where n is the subsample size and H is the harmonic number. In DetectPay the anomaly score is the negated decision function of the fitted forest clipped to [0, 1], so that ordinary transactions score 0.

TABLE I. BASE MODELS AND SETTINGS

Model	Key settings
Weighted Random Forest	200 trees, class_weight = balanced
XGBoost	200 trees, max_depth = 6, η = 0.05, subsample = 0.8, colsample = 0.8, no class weighting
LightGBM	200 trees, max_depth = 6, η = 0.05, class_weight = balanced
Isolation Forest	200 trees, contamination = 0.03, unsupervised

B. *Weighted Ensemble and Risk Engine*

The ensemble fraud probability is a weighted average of the three supervised probabilities, with the largest weight on XGBoost and the smallest on the Random Forest:

$$p = 0.25p_{RF} + 0.40p_{XGB} + 0.35p_{LGB} \quad (5)$$

A transaction is labelled as predicted fraud when $p \geq t$, with the threshold t selected in Section VI-C. The risk engine then combines the probability with the anomaly score a (Fig. 1):

$$r = \min(1, 0.70p + 0.30a) \quad (6)$$

and maps it to four categories: Low ($r < 0.25$), Medium ($0.25 \leq r < 0.50$), High ($0.50 \leq r < 0.75$) and Critical ($r \geq 0.75$). Keeping the two signals separate until this step lets an analyst see whether a high score comes from learned fraud patterns, from unusual behaviour or from both.

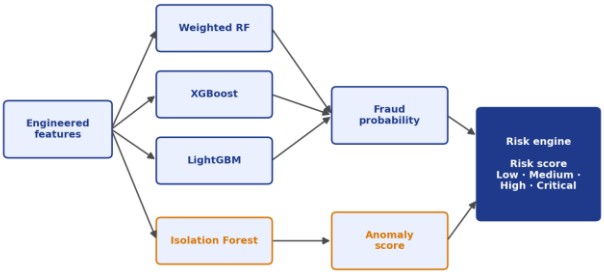


Fig. 1. Ensemble prediction and risk engine of DetectPay.

C. *Evaluation Metrics*

Because accuracy is misleading at a 3% fraud rate, the evaluation reports precision, recall and the F1-score of the fraud class, where TP, FP and FN are true positives, false positives and false negatives:

$$\text{Precision} = TP / (TP + FP) \quad (7)$$

$$\text{Recall} = TP / (TP + FN) \quad (8)$$

$$F1 = 2 \cdot \text{Precision} \cdot \text{Recall} / (\text{Precision} + \text{Recall}) \quad (9)$$

The threshold-free ranking quality is summarised by the ROC-AUC and by the AUC-PR, computed as average precision, whose chance level equals the fraud rate of 0.03 [10].

V. SYSTEM IMPLEMENTATION

DetectPay follows a layered architecture (Fig. 2). A FastAPI service exposes POST /predict, validates every request against a typed Pydantic schema of 23 fields, applies the shared feature engineering, scores the four models and returns the fraud probability, anomaly score, risk score, risk category and predicted-fraud flag. A Streamlit workspace provides four pages: Analyze, for manual entry of a transaction with a risk card and readable risk factors such as a CVV mismatch or a card-not-present channel; Overview, with dataset summaries; Upload Evidence; and History. The evidence path

extracts text from selectable PDFs with pypdf, from scanned PDFs through PyMuPDF rendering and Tesseract, and from PNG, JPG and WEBP images through Tesseract. A rule-based parser splits the text into transaction records in document order and normalises fields such as amount, merchant, date and reference number; the analyst corrects the extracted values in an editable form before each record is scored. Every manual or OCR result is appended to a CSV history with duplicate protection. The models are stored as joblib files with the feature-column list, and the project includes 14 automated pytest tests that cover the API, feature engineering, risk boundaries, saved-model scoring, OCR parsing and the history store.

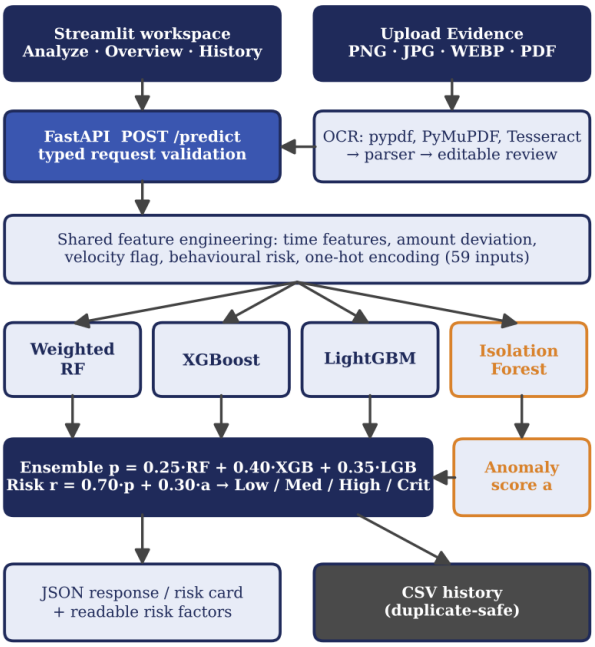


Fig. 2. Layered architecture of DetectPay with the OCR evidence path.

VI. EXPERIMENTAL RESULTS

A. *Experimental Setup*

The models were trained with the settings of Table I in the project's training notebook. For this paper the saved model files were re-loaded and scored on the same stratified hold-out split in a Linux environment with Python 3.13, scikit-learn 1.9, XGBoost 3.4 and LightGBM 4.7; the re-computed metrics reproduce the values recorded during training exactly. All 14 automated tests passed, and the API answered the documented sample request with a fraud probability of 0.0174 and a Low risk score of 0.0122.

B. *Model Comparison*

Table II compares the models on the hold-out set at the default threshold of 0.5 and the ensemble at the selected threshold. The three supervised models reach almost the same ranking quality (ROC-AUC 0.877–0.890) but behave very differently at the default threshold. The unweighted XGBoost is precise but cautious (precision 0.76, recall 0.24), the class-weighted LightGBM finds the most fraud at the cost of 123 false alarms (precision 0.28, recall 0.54), and the Weighted Random Forest has the lowest AUC-PR (0.310). The Isolation Forest alone, which never sees the label, reaches a ROC-AUC of only 0.759. The weighted ensemble at t = 0.40 achieves the best F1-score (0.44), ahead of LightGBM (0.37) and XGBoost (0.37).

TABLE II. HOLD-OUT RESULTS (3,000 TRANSACTIONS, 90 FRAUD)

Model	P	R	F1	AUC-PR	ROC-AUC
Weighted RF	0.47	0.21	0.29	0.310	0.886
XGBoost	0.76	0.24	0.37	0.441	0.890
LightGBM	0.28	0.54	0.37	0.407	0.877
Isolation Forest	0.24	0.24	0.24	0.195	0.759
Ensemble, t = 0.50	0.54	0.33	0.41	0.417	0.890
Ensemble, t = 0.40	0.43	0.46	0.44	0.417	0.890

IARJSET ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Impact Factor 8.311 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 13, Issue 11, November 2026

[a] P = precision, R = recall. Single models and the ensemble at t = 0.50 use the default threshold; the Isolation Forest flags the 3% most anomalous transactions.

C. *Threshold Selection*

Lowering the threshold trades precision for recall (Fig. 3, Table III). Between 0.30 and 0.50 the F1-score varies only from 0.41 to 0.44 while recall falls from 0.53 to 0.33; 0.40 was selected because it gives the highest recall among the thresholds with the best F1-score. At this threshold the ensemble raises 95 alerts per 3,000 transactions, of which 41 are fraud.

TABLE III. EFFECT OF THE DECISION THRESHOLD ON THE ENSEMBLE

t	Precision	Recall	F1	Alerts	TP
0.30	0.338	0.533	0.414	142	48
0.35	0.389	0.489	0.433	113	44
0.40	0.432	0.456	0.443	95	41
0.45	0.500	0.400	0.444	72	36
0.50	0.536	0.333	0.411	56	30

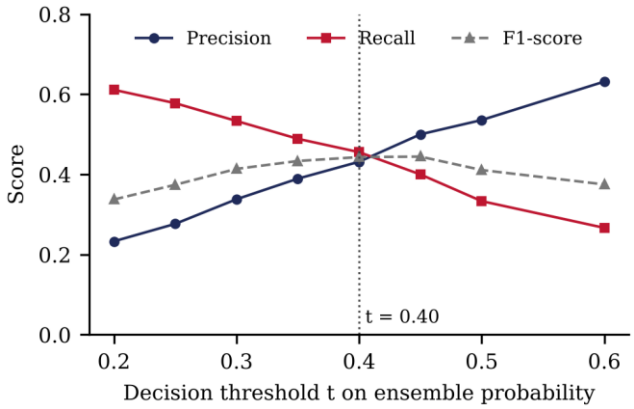


Fig. 3. Precision, recall and F1-score of the ensemble against the decision threshold.

D. *Error Analysis and Precision–Recall Behaviour*

Fig. 4 shows the confusion matrix of the ensemble at $t = 0.40$: 41 of 90 fraud transactions were detected, 49 were missed and 54 of 2,910 legitimate transactions were flagged, giving 96.6% accuracy—below the 97.0% of a model that never predicts fraud, which illustrates why accuracy is not a useful headline metric here. The precision–recall curves in Fig. 5 separate the models more clearly than their nearly identical ROC curves; the ensemble curve stays at or near the top over most of the recall range, while the Random Forest curve is markedly lower at low recall.

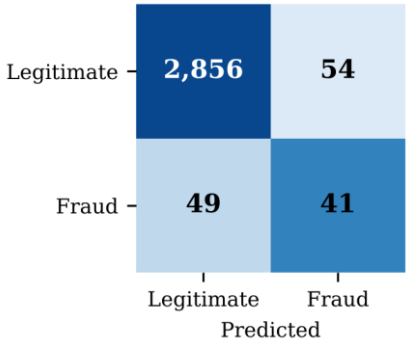


Fig. 4. Confusion matrix of the ensemble on the hold-out set (t = 0.40).

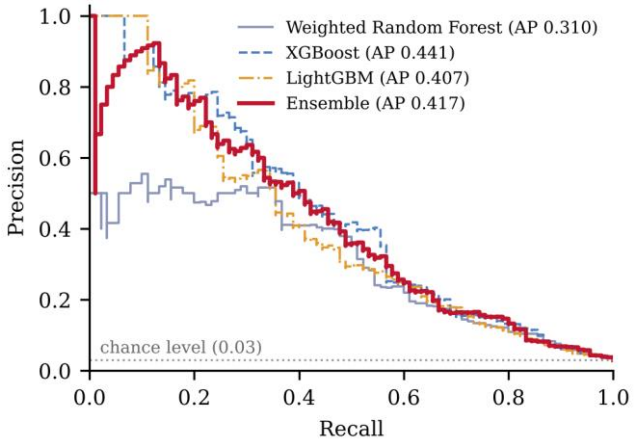


Fig. 5. Precision–recall curves of the supervised models and the ensemble.

E. Cross-Validation

To check that the hold-out result does not depend on one split, the three supervised models were re-trained with the same settings in stratified five-fold cross-validation on the training set (about 72 fraud cases per fold, $t = 0.40$). The mean precision was 0.44 ± 0.05, recall 0.37 ± 0.07, F1-score 0.40 ± 0.06, AUC-PR 0.36 ± 0.07 and ROC-AUC 0.856 ± 0.016. These values are slightly below but consistent with the hold-out results, and the fold-to-fold spread of the F1-score (0.32 to 0.46) shows how noisy the estimates are with so few fraud cases.

F. Risk Categories and Feature Importance

Table IV shows how the risk engine distributes the hold-out transactions. The Low category holds 2,891 transactions with a fraud share of only 1.6%, while fraud makes up 33% of the Medium and 86% of the High category. An analyst reviewing the 109 Medium and High transactions (3.6% of traffic) would see 43 of the 90 fraud cases. No hold-out transaction reached the Critical band; the highest risk score was 60.4%, because the cautious ensemble probabilities and the 0.30 weight of the anomaly score keep the combined score below 0.75 on this data. Averaged over the three supervised models, the most important features were the distance from home (0.111), the foreign-transaction flag (0.099), the account age (0.063), the engineered behavioural-risk score (0.062) and the amount (0.042), which matches the differences between the classes described in Section III-A.

TABLE IV. RISK CATEGORIES OF THE HOLD-OUT TRANSACTIONS

Category	Legitimate	Fraud	Fraud share
Low	2,844	47	1.6%
Medium	64	31	32.6%
High	2	12	85.7%
Critical	0	0	—

G. Application Output

Through the API, the documented sample transaction (13.56 EUR, online retail, domestic) received a fraud probability of 1.7% and a Low risk score of 1.2%, whereas a synthetic high-risk request (foreign, 6,200 km from home, new device and merchant, CVV mismatch, three failed PIN attempts and 82 times the user's average amount) received a fraud probability of 86.3%, an anomaly score of 15.3% and a High risk score of 65.0%. In the evidence workflow, two OCR records with the same amount of 48,750 produced a Low (7.2%) and a Critical (94.0%) result; in the first, the date and merchant had not been read correctly, which confirms the need for the editable review step. In the second, the 94.0% score was the risk score stated in the uploaded document, which the evidence page reports when it exceeds the model score.

VII. DISCUSSION AND LIMITATIONS

The ensemble improves the balance between missed fraud and false alarms, but the absolute detection quality is moderate: at the selected threshold more than half of the hold-out fraud is missed. The dataset is small, with only 90 fraud cases in the hold-out set and about 72 per cross-validation fold, single-currency and drawn from one three-month period, so the results may not transfer to other institutions or to later periods with concept drift [6]. XGBoost was trained without class weighting, and the ensemble weights and the risk-engine coefficients were set by design rather than optimised. The displayed risk score is not calibrated against real fraud rates, and the anomaly component contributes little because the clipped Isolation Forest score is zero for most transactions. During testing the API accepted out-of-

range binary values (for example a CVV-match value of 5) because the schema declares these fields as integers without a range; the dashboard prevents such input. Finally, OCR quality depends on the clarity of the evidence, so human review remains mandatory.

VIII. Conclusion and Future Work

DetectPay combines a weighted ensemble of Random Forest, XGBoost and LightGBM, an Isolation Forest anomaly score and a transparent risk engine in a usable application with a validated API, an analyst workspace and OCR-based evidence upload. On a stratified hold-out set the ensemble at a threshold of 0.40 reached precision 0.43, recall 0.46, F1-score 0.44, AUC-PR 0.42 and ROC-AUC 0.89, outperforming every single model on F1-score, and cross-validation confirmed this level. Future work includes class-weighted XGBoost and cross-validated tuning of the ensemble weights and threshold, probability calibration, range constraints in the API schema, per-prediction explanations with SHAP [13], drift monitoring with scheduled retraining, a database back-end with authentication and audit logging, and evaluation on larger public fraud datasets.

Acknowledgment

The authors thank G. Karthiha, Assistant Professor (O.G), and Dr. P. Chitra, Professor and Head, Department of Computer Science and Engineering (Emerging Technologies), SRM Institute of Science and Technology, Vadapalani Campus, for their guidance and support.

References

[1] L. Breiman, “Random forests,” Machine Learning, vol. 45, no. 1, pp. 5–32, 2001.

[2] T. Chen and C. Guestrin, “XGBoost: A scalable tree boosting system,” in Proc. 22nd ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining (KDD), San Francisco, CA, USA, 2016, pp. 785–794.

[3] 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 30 (NeurIPS), 2017, pp. 3146–3154.

[4] F. T. Liu, K. M. Ting, and Z.-H. Zhou, “Isolation forest,” in Proc. 8th IEEE Int. Conf. Data Mining (ICDM), Pisa, Italy, 2008, pp. 413–422.

[5] A. Dal Pozzolo, O. Caelen, R. A. Johnson, and G. Bontempi, “Calibrating probability with undersampling for unbalanced classification,” in Proc. IEEE Symp. Series on Computational Intelligence (SSCI), Cape Town, South Africa, 2015, pp. 159–166.

[6] A. Dal Pozzolo, G. Boracchi, O. Caelen, C. Alippi, and G. Bontempi, “Credit card fraud detection: A realistic modeling and a novel learning strategy,” IEEE Trans. Neural Netw. Learn. Syst., vol. 29, no. 8, pp. 3784–3797, Aug. 2018.

[7] F. Carcillo, Y.-A. Le Borgne, O. Caelen, Y. Kessaci, F. Oblé, and G. Bontempi, “Combining unsupervised and supervised learning in credit card fraud detection,” Information Sciences, vol. 557, pp. 317–331, 2021.

[8] A. Correa Bahnsen, D. Aouada, A. Stojanovic, and B. Ottersten, “Feature engineering strategies for credit card fraud detection,” Expert Systems with Applications, vol. 51, pp. 134–142, 2016.

[9] R. Smith, “An overview of the Tesseract OCR engine,” in Proc. 9th Int. Conf. Document Analysis and Recognition (ICDAR), Curitiba, Brazil, 2007, pp. 629–633.

[10] T. Saito and M. Rehmsmeier, “The precision-recall plot is more informative than the ROC plot when evaluating binary classifiers on imbalanced datasets,” PLoS ONE, vol. 10, no. 3, e0118432, 2015.

[11] F. Pedregosa et al., “Scikit-learn: Machine learning in Python,” J. Machine Learning Research, vol. 12, pp. 2825–2830, 2011.

[12] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, “SMOTE: Synthetic minority over-sampling technique,” J. Artificial Intelligence Research, vol. 16, pp. 321–357, 2002.

[13] S. M. Lundberg and S.-I. Lee, “A unified approach to interpreting model predictions,” in Advances in Neural Information Processing Systems 30 (NeurIPS), 2017, pp. 4765–4774.

[14] T. G. Dietterich, “Ensemble methods in machine learning,” in Multiple Classifier Systems (MCS 2000), LNCS, vol. 1857. Berlin, Germany: Springer, 2000, pp. 1–15.

 This work is licensed under a Creative Commons Attribution 4.0 International License