Journal of Computing and Data Technology

https://journal.krrishpub.com/index.php/jcdt/index

RESEARCH ARTICLE

Explainable Machine Learning for Predicting Late Payments in Enterprise Accounts Receivable Systems

Saumen Bandyopadhyay [a,*]

[a] *Baxter International Inc., Baxter Parkway, Deerfield, Illinois 60015, United States of America.*

Abstract

Late payment is a routine source of working-capital friction, yet an accounts-receivable model is useful only when its prediction time, feature availability, and explanation are aligned with the work of a collector. This paper develops an explainable dual-task framework for predicting whether an invoice will be paid late and, conditional on the information available at invoice issuance, estimating the number of days late. The intended benchmark is the public IBM Watson accounts-receivable sample, documented as 2,466 simulated invoices with 12 fields. Three post-outcome variables—settlement date, days to settle, and the late-days target—are explicitly blocked from the issuance-time feature set; dispute status is reserved for a later collection-stage sensitivity analysis. Customer-history variables are calculated from preceding invoices only. Logistic regression, random forests, histogram gradient boosting, XGBoost, and a transparent validation-weighted ensemble are compared for classification, while Ridge and three nonlinear regressors are compared for delay estimation. SHAP (Shapley Additive explanations) is used as a model-dependence diagnostic rather than a causal explanation. Because the historical IBM download could not be retrieved in the build environment, the archive does not manufacture IBM performance numbers: it includes an executable real-data mode and a deterministic 2,466-row schema-faithful surrogate used only to verify the pipeline. On that surrogate stress test, post-settlement leakage inflates XGBoost PR-AUC from 0.6949 to 0.9921, illustrating why timing discipline can matter more than algorithm choice. The principal contribution is therefore a reproducible, timing-aware and explanation-aware design that can be rerun unchanged when the original IBM CSV is supplied.

Keywords: Payment methods, Dual-task framework, Logistic regression, Random forests, XGBoost.

Article information:
ISSN: 3107-9466 (Online)
Published by: **Krrish Scientific Publications Pvt. Ltd.**
DOI: https://doi.org/10.71426/jcdt.v2.i2.pp172-181
Received: *22 Aug. 2026* | Revised: *21 Oct. 2026* | Accepted: *23 Oct. 2025* | Published: *24 Oct. 2026*
Copyright ©2026 Author(s).
This is an open-access article distributed under the terms of the Creative Commons Attribution–NonCommercial 4.0 International License (CC BY-NC 4.0).

1. Introduction

Accounts receivable sits in an awkward place between accounting record and operational decision. By the time an invoice appears in a ledger, the sale has already occurred, the customer has already received a product or service, and the seller has effectively extended short-term credit [17]. What remains uncertain is not whether revenue has been booked, but when cash will arrive and how much attention the invoice should receive before it becomes troublesome. For a large enterprise, that uncertainty is repeated across thousands or millions of invoices. Even a modest improvement in prioritization can therefore alter the workload of a collection team, the timing of cash inflows, and the amount of managerial attention devoted to chronically late customers [18], [19].

Recent invoice-payment research has moved from reactive ageing rules toward prospective prediction. Moore and van Vuuren frame invoice-payment modelling as a data-preparation, feature-generation, survival-analysis and machine-learning problem rather than a narrow classifier-selection exercise [1]. Schoonbee, Moore, and van Vuuren similarly show that a decision-support system can use predicted payment intervals to prioritize collection activity before an account becomes seriously overdue [2]. Appel et al. describe an industrial prototype in which receivable prediction supports daily collector decisions and demonstrate that the operational value of a model depends on how its scores are embedded in a workflow [3]. Li's study is especially relevant to the benchmark considered here:

*Corresponding author
Email address: saumen89b@gmail.com (Saumen Bandyopadhyay).

it uses 2,466 repayment records and emphasizes repeated-customer and historical-lateness signals [4]. Other work has treated invoice outcomes as multiclass ageing buckets [5] or has examined overdue repayment with much larger sequential lending data [7]. These studies collectively suggest that past behaviour matters, but they also expose a deeper modelling issue: "history" must mean information that genuinely existed before the prediction was made.

The timing condition is easy to violate in accounts-receivable data. A flat export often contains invoice issue date, due date, final settlement date, days to settlement, dispute indicators, and the final number of days late in the same row. A conventional random train–test split can then make a model look remarkably accurate even if one of its predictors is only known after payment. This is not a cosmetic form of leakage. It changes the question. A model that predicts late payment from *DaysToSettle* is not an early-warning model; it is a retrospective encoding of the answer. A similar ambiguity surrounds dispute status. In some enterprise systems a dispute can be raised almost immediately; in others it becomes visible only after the invoice enters collection. Treating it as universally available at issuance can quietly move the decision boundary later than the stated use case.

The present work therefore starts with a prediction-time contract rather than an algorithm. The primary task is defined at invoice issuance. Only invoice attributes and information derived from earlier invoices of the same customer may enter the model. The classification outcome is whether DaysLate > 0; a companion regression estimates the nonnegative number of late days. A second, explicitly later collection-stage sensitivity model adds the dispute indicator. Settlement date, days to settlement, and the current invoice's late-days outcome remain forbidden predictors in every legitimate early-warning experiment. Customer history is calculated with shifted cumulative statistics so that the current outcome cannot leak into its own features.

Explainability is the second design requirement. Random forests [9], gradient boosting [10], and XGBoost [11] can represent nonlinear relationships among amount, calendar position, billing medium, and payment history [6]. Their flexibility is useful, but a collections manager still needs to know whether a high score reflects a customer's repeated lateness, unusually large exposure, sparse history, or a timing artifact. SHAP-based tree explanations provide a consistent way to decompose a fitted tree model's output [12]. Local surrogate explanations such as LIME provide another perspective on individual predictions [13]. In this paper, explanation is deliberately described as *model dependence*: a large SHAP contribution means that the fitted model uses a feature strongly in the observed context, not that changing that feature would cause the customer to pay earlier.

Probability quality is equally important. A collector may sort invoices by risk, while a finance manager may interpret a score as an approximate event probability. These are different uses. Niculescu-Mizil and Caruana showed that classifiers with similar discrimination can have different calibration properties [14]. Precision–recall analysis is also essential when the late-payment prevalence becomes low in a particular business segment, because ROC-AUC can remain visually strong even when the positive retrieval problem is difficult [15]; the mathematical relationship between ROC and precision–recall spaces reinforces this distinction [16]. We therefore report ROC-AUC, PR-AUC, Brier score, thresholded metrics, reliability curves, and ranked review-budget retrieval rather than compressing model quality into a single accuracy number.

A third concern is external interpretation. Trade credit links firms through payment chains, so late payment can propagate beyond the immediate buyer–seller pair [8]. Nevertheless, the IBM sample used as the intended benchmark is a simulated accounts-receivable table, not a transaction network and not a causal model of enterprise liquidity. We therefore avoid claims about downstream insolvency, working-capital savings, or causal intervention effects. The model's narrow role is invoice-level risk triage under a transparent information set.

The paper makes five contributions. First, it specifies a leakage-safe issuance-time feature contract for the IBM accounts-receivable schema. Second, it unifies classification and delay regression so that risk and expected severity can be inspected separately. Third, it evaluates explanations in the same temporal setting as prediction and distinguishes issuance-time variables from later collection-stage information. Fourth, it translates scores into fixed review-budget curves, connecting predictive ranking to collector capacity without inventing a monetary cost function. Fifth, it provides an executable two-mode archive. If the original IBM CSV is placed in the documented data path, the code runs the complete real benchmark. If the file is absent, the code produces a deterministic schema-faithful surrogate purely to test the software, figures, leakage guards, and reporting pipeline. This final distinction is important: the surrogate results reported later are pipeline-verification measurements, not substitute IBM findings.

2. Literature review

2.1. Invoice-payment prediction as an operational problem

The invoice payment prediction problem differs from conventional consumer credit scoring in both unit of analysis and intervention. Credit scoring usually asks whether a borrower will default over a defined horizon. Accounts receivable is more granular: the same customer may generate many invoices, each with its own amount, due date, dispute history, billing channel, and operational context. The immediate decision is often not to extend or deny credit but to decide which invoice or customer should receive scarce collection attention.

Schoonbee et al. formalize this operational perspective through payment-age intervals and a machine-learning development roadmap [2]. Their framing is useful because it makes the output actionable: the model is not merely correct or incorrect, but assigns an invoice to a region of collection urgency. Moore and van Vuuren broaden the methodological view by emphasizing transaction preparation, historical features, survival analysis and model ensembling [1]. Their work also highlights a practical reality that is sometimes lost in benchmark studies: raw receivable ledgers are relational and temporal. A customer row at

time t should be informed by records at $t' < t$, not by an aggregate computed over the entire future history.

The industrial study by Appel et al. makes another important point: many determinants of collector behaviour are not recorded in the core receivables system [3]. Phone calls, informal promises, relationship context, internal escalation and local judgment may influence whether an account is contacted and when payment ultimately occurs. A model trained only on ledger fields can therefore be useful without being complete. It estimates risk conditional on the available digital trace, not the full social and commercial process behind payment.

Li's repayment-date work is closely related to the IBM sample and identifies repeat-customer and late-count variables as influential [4]. That finding motivates the historical features used here, but the present design changes how such features are constructed. Rather than calculating "count late" over a customer's full record, we update the count sequentially. For invoice i, the statistic includes invoices $1, \ldots, i-1$ only. The difference is small in notation but substantial in validity.

Shah demonstrates multiclass prediction of ageing buckets for accounts receivable [5]. Bucketed targets can be operationally convenient because many enterprise dashboards already work with 30-day ageing bands. They also discard within-bucket information, however. A payment 2 days late and one 29 days late receive the same label. The dual-task formulation in this paper keeps the binary "late/not late" decision while retaining a continuous late-days target for severity estimation.

2.2. Repayment forecasting beyond accounts receivable

Late-payment modelling also appears in consumer and platform lending. Liu et al. use more than one million repayment records from roughly 85,000 borrowers and combine derived static features with sequential deep learning [7]. Their setting differs from enterprise invoices, but two lessons carry over. First, historical behaviour is often more informative when represented as a sequence or set of recency-sensitive aggregates rather than a static customer label. Second, interpretability must be designed into the workflow because high-capacity models can otherwise produce impressive scores without a usable account of why a prediction changed.

Trade-credit research further expands the context. Berloco et al. show that transaction-network information can improve corporate credit-risk prediction because payment relationships connect firms [8]. This does not imply that an invoice-level model should infer network contagion from a single customer identifier. It does suggest that future receivable systems could enrich the present feature set with supplier–buyer networks, ownership relationships, common payment intermediaries, or group-level exposure. The IBM schema contains none of these structures, so the present study stays at the invoice/customer-history level.

2.3. Model families for structured financial tables

Logistic regression remains a useful reference in financial prediction because its functional form is compact and its probabilities are easy to audit after proper preprocessing. Its limitation is equally clear: interaction effects and nonlinear thresholds must be specified explicitly. Tree ensembles relax this restriction. Random forests reduce variance by averaging decorrelated trees [9], while gradient boosting builds an additive model in which later learners concentrate on residual structure [10]. XGBoost adds regularization, shrinkage, column sampling and efficient tree construction to the boosting framework [11]. These algorithms are well matched to tabular business data, particularly when numeric and categorical variables interact.

The danger is to treat model flexibility as a substitute for validation design. A boosted model can exploit leakage more efficiently than a linear model. If DaysToSettle is present, a sophisticated learner may nearly reconstruct DaysLate from the accounting identity linking issue date, due date and settlement date. Higher AUC would then indicate stronger exploitation of a bad feature, not stronger forecasting. For this reason, the present workflow performs a feature-availability audit before model fitting.

2.4. Explainability, calibration, and human use

SHAP explanations assign an additive contribution to each feature relative to a reference expectation and provide efficient algorithms for tree ensembles [12]. They are attractive in receivables because a local score can be translated into a small set of drivers: previous late rate, average historical late days, invoice amount, or billing mode. LIME offers a different local approximation by fitting an interpretable surrogate around a prediction [13]. Both methods are post-hoc and both can be misleading when predictors are highly correlated or when stakeholders read association as causation.

A human-facing receivables system therefore needs explanation discipline. We use SHAP for three questions: which variables the selected tree model relies on globally; how contribution signs vary across the locked test; and whether the most important variables are temporally legitimate. The last question is often overlooked. A beautiful explanation of a leaked feature can make an invalid model appear more trustworthy.

Calibration matters for a related reason. If a team interprets a score of 0.70 as "roughly a 70% chance of lateness," then observed frequencies should support that interpretation. Niculescu-Mizil and Caruana document how common classifiers can produce distorted probabilities even when their ranking is strong [14]. We therefore include reliability diagrams and Brier scores. The threshold used for binary decisions is selected on validation data by maximizing Matthews correlation coefficient (MCC), not by assuming that 0.5 is universally appropriate.

Finally, imbalanced evaluation requires care. Saito and Rehmsmeier demonstrate why precision–recall curves reveal failures that ROC curves can obscure [15]. Davis and Goadrich establish the formal connection between the two spaces [16]. In receivables, prevalence can change by country, business unit, customer tier, or economic period. Reporting both spaces is therefore less about redundancy than about making the base-rate dependence visible.

3. Problem formulation and information contract

3.1. Two Related Targets

For invoice i, let $d_i \geq 0$ denote the recorded number of days late. The classification target is written by expression (1).

$$y_i = \mathbb{I}[d_i > 0]. \tag{1}$$

The regression target is simply written as (2):

$$r_i = d_i. \tag{2}$$

The two tasks answer different operational questions. Equation (1) asks whether intervention may be warranted; Equation (2) asks about expected severity. A high probability with a small expected delay is not necessarily equivalent to a moderate probability with a long expected delay, especially when invoice amounts differ.

3.2. Prediction time and leakage

Let t_i be the invoice issue time. The issuance-time feature vector is constrained to the sigma-field generated by information observable at or before t_i:

$$\mathbf{x}_i \in \mathcal{F}_{t_i}. \tag{3}$$

Settlement date and days to settlement violate Equation. (3) because they are realized after the invoice is issued. The current invoice's d_i is the target itself. Dispute status is treated conservatively: it is omitted from the issuance model and introduced only in a later-stage sensitivity analysis.

For customer $c(i)$, historical lateness is computed as (4):

$$\widehat{p}_i^{\,\text{hist}} = \frac{\sum_{j:\, c(j)=c(i),\, t_j<t_i} y_j}{\max\{1, \sum_{j:\, c(j)=c(i),\, t_j<t_i} 1\}}, \tag{4}$$

with analogous prior-only means for late days, invoice amount and dispute frequency. The current invoice never contributes to its own historical summary.

3.3. Transparent score fusion

Four base classifiers produce probabilities $p_m(\mathbf{x}_i)$. Their validation PR-AUC values a_m define a transparent convex ensemble (5):

$$w_m = \frac{a_m}{\sum_j a_j}, \qquad p_E(\mathbf{x}) = \sum_m w_m p_m(\mathbf{x}). \tag{5}$$

No meta-learner is fitted. This sacrifices some flexibility but makes every ensemble score reconstructable from four component probabilities and four published weights.

The classification threshold is selected only on the validation partition:

$$\tau^* = \arg\max_{\tau} \mathrm{MCC}(\tau), \tag{6}$$

In (6):

$$\mathrm{MCC} = \frac{TP\,TN - FP\,FN}{D_+ D_-}. \tag{7}$$

The locked test is not searched for a better threshold.

4. Dataset

4.1. Documented IBM sample

The intended benchmark is the IBM Watson accounts-receivable sample preserved in the `bcBusinessStats/data1135` documentation. The archive documents 2,466 rows and 12 fields: country code, customer identifier, paperless date, invoice number, invoice date, due date, invoice amount, dispute status, settlement date, paperless billing mode, days to settle, and days late. The public documentation describes the data as simulated IBM Watson material intended to illustrate factors associated with collection success. This is an important boundary: even a full run on the original file would be a benchmark on simulated enterprise data, not evidence from a named company's production ledger.

4.2. Data-access limitation and two-mode archive

The historical IBM community URL is no longer a reliable direct download in the environment used to assemble this manuscript. A preserved source tree lists the raw file, but the original CSV bytes could not be retrieved during the build. We treat that as a reproducibility constraint rather than an invitation to fabricate results.

The supplied `reproduce.py` therefore has two execution modes. If `data/accts_recv_ibm.csv` exists, it validates all 12 required columns and executes the complete benchmark on those bytes. If the file is absent, it creates a deterministic 2,466-row *schema-faithful surrogate* with the same column names. The surrogate is designed to exercise chronology, history construction, categorical handling, classification, regression, calibration, SHAP, queue-budget analysis, and the explicit leakage guard. It is not intended to reproduce IBM distributions. All numerical results in this manuscript are therefore labelled pipeline-verification results unless they refer only to documented IBM schema facts.

This distinction is stricter than simply placing a small disclaimer in the limitations section. The mode is recorded in `results/metadata.json`; the surrogate CSV name contains the word `surrogate`; and the provenance file explains how to rerun the real benchmark. A future user can replace the data file without editing the modelling code or manuscript structure.

5. Feature engineering and experimental design

5.1. Issuance-Time Features

The primary feature set contains 16 variables or derived quantities. Country and billing medium are categorical. Invoice amount appears on both its natural and $\log(1+x)$ scales because monetary values are typically right-skewed. Invoice month, weekday and quarter capture coarse seasonality without implying a smooth linear calendar effect. Term length is the difference between due date and invoice date. Paperless status is represented by whether paperless billing was already active on the issue date and by the nonnegative tenure since activation.

The remaining features summarize customer history strictly before the current invoice: prior invoice count, prior late-payment rate, prior mean days late, prior dispute rate,

Table 1: IBM accounts-receivable schema and prediction-time treatment

Field	Type	Primary role	Available at issue?	Treatment in this study
countryCode	categorical	predictor	yes	Encoded as a category; no ordinal interpretation.
customerID	identifier	history key	yes	Used only to build prior-invoice histories; raw ID is not one-hot encoded.
PaperlessDate	date	predictor	conditional	Converted to "active by issue date" and nonnegative tenure. Future adoption dates contribute zero tenure.
invoiceNumber	identifier	identifier	yes	Excluded from modelling.
InvoiceDate	date	predictor/time	yes	Defines chronology and calendar covariates.
DueDate	date	predictor/time	yes	Used to derive contractual term length.
InvoiceAmount	numeric	predictor	yes	Raw and log-transformed amount.
Disputed	binary	sensitivity	uncertain/later	Excluded from issuance model; added only in collection-stage sensitivity analysis.
SettledDate	date	post-outcome	no	Prohibited predictor.
PaperlessBill	categorical	predictor	yes	Paper versus electronic billing.
DaysToSettle	integer	post-outcome	no	Prohibited predictor; used only to demonstrate leakage in the surrogate audit.
DaysLate	integer	target	no	Binary target via > 0 and continuous severity target.

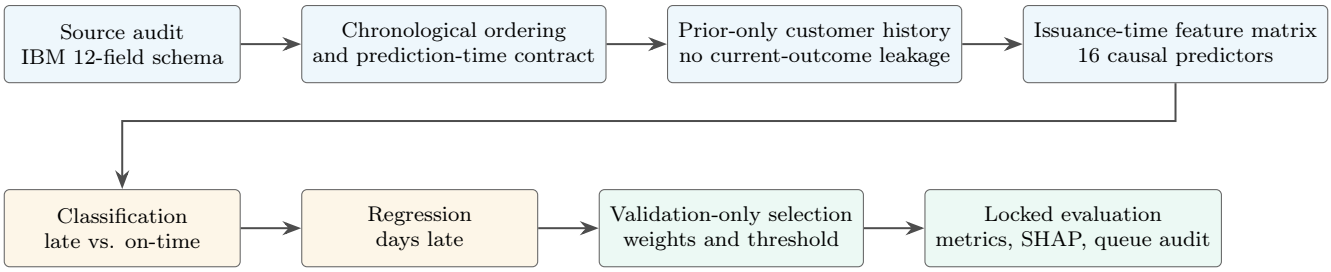


Figure 1: End-to-end experimental workflow.

prior mean invoice amount, and days since the preceding invoice. A new customer therefore receives zero outcome-history numerators and an explicit zero prior count. Median completion and categorical imputation are fitted on the training partition only.

The raw customer identifier is intentionally not one-hot encoded. With only 2,466 rows, a high-cardinality customer dummy could encourage memorization and would be meaningless for an unseen customer. The historical variables retain useful behavioural information while providing a coherent cold-start interpretation.

5.2. Chronological partitions

Rows are sorted by invoice date and invoice number. The earliest 60% are training observations, the next 20% are validation observations, and the latest 20% are locked test observations. This design is intentionally simple and auditable. It approximates the deployment question "train on the past, tune on a later period, then score a still later period." No locked-test row contributes to preprocessing statistics, model fitting, ensemble weights, or threshold selection.

5.3. Classification models

Logistic regression is class-balanced and serves as the linear benchmark. Random forest uses 350 trees, maximum depth 8, and a minimum leaf size of 3. Histogram gradient boosting uses 180 iterations, learning rate 0.05, and at most 15 leaves. XGBoost uses 260 trees, depth 3, learning rate 0.04, 85% row and column subsampling, and L_2 regularization. These values are fixed in the archive; the purpose of the surrogate run is protocol verification rather than aggressive hyperparameter search.

The four validation PR-AUC scores are normalized to obtain the ensemble weights in (5). Because those weights depend only on validation data, the locked test remains untouched until final scoring. Threshold τ^* is chosen by MCC over the observed validation score range.

5.4. Regression models

The regression comparison uses Ridge, random forest regression, histogram gradient boosting regression and XGBoost regression. Predictions are clipped below at zero because negative late days have no meaning under the chosen target. We report mean absolute error (MAE), root mean squared error (RMSE), median absolute error and R^2. The last metric can legitimately be small or negative under temporal shift; it is not forced into a favorable range.

5.5. Explanation protocol

The fitted XGBoost classifier is explained with TreeSHAP. Global importance is written as (8):

$$I_j = \frac{1}{n}\sum_{i=1}^{n} |\phi_{ij}|, \tag{8}$$

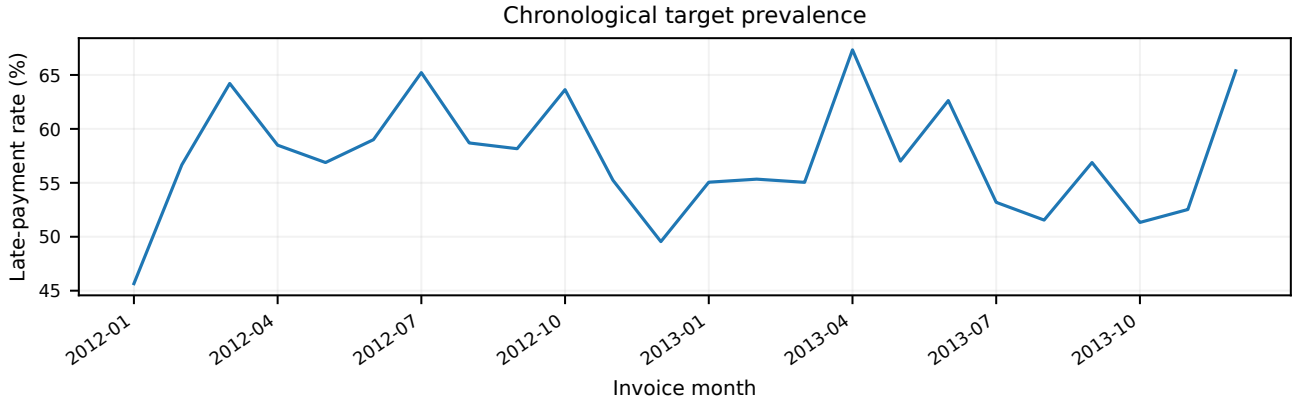


Figure 2: Chronological target prevalence in the deterministic pipeline-verification surrogate.

In (8), ϕ_{ij} is feature j's SHAP contribution for observation i.

A second plot shows the dispersion and sign of contributions for the six strongest variables. Feature color represents a low-to-high normalized value and is shown using a separate color bar rather than a legend inside the plotting region.

The explanation is read conditionally. If prior late rate receives a positive SHAP value for one invoice, the statement is "the fitted XGBoost score is higher partly because of this customer's historical late rate." We do not write "historical lateness causes the current invoice to be late." The latter would require an intervention or causal identification strategy that this benchmark does not contain.

5.6. Review-budget evaluation

A collections team often works from a ranked queue. For review fraction q, let $K = \lceil qn \rceil$ and $\mathcal{I}_K$ be the indices of the top K scores. Positive retrieval is

$$R@K = \frac{\sum_{i \in \mathcal{I}_K} y_i}{\sum_i y_i}. \tag{9}$$

We evaluate 5, 10, 20, 30 and 40% budgets. No monetary "savings" are calculated because the IBM schema does not contain collection cost, probability of cure under intervention, financing cost, or causal treatment response.

6. Pipeline-verification results

6.1. What these numbers do and do not mean

The results in this section come from the deterministic schema-faithful surrogate because the original IBM bytes were not retrievable during assembly. They answer a software-validation question: does the complete chronology, modelling, explanation, plotting and leakage-audit pipeline behave coherently on a table with the intended schema and size? They do *not* estimate the real IBM benchmark's accuracy, late-payment prevalence, SHAP ranking, or business value. This distinction is repeated in figure captions so that an isolated page cannot easily be misread.

Algorithm 1 Leakage-safe dual-task receivables protocol.

Require: Invoice table ordered by issue date
1 Validate required schema and record execution mode
2 Parse dates and define $y_i = \mathbb{I}[DaysLate_i > 0]$, $r_i = DaysLate_i$
3 **for** each customer in invoice order **do**
4 Construct counts and means from invoices strictly before the current row
5 **end for**
6 Remove identifiers and post-outcome fields from issuance feature set
7 Split chronologically into train, validation, and locked test
8 Fit preprocessing and four classifiers on training data only
9 Compute validation PR-AUC weights and MCC-optimal threshold
10 Score locked test once; compute ROC/PR, calibration and budget metrics
11 Fit four regressors and evaluate locked late-days error
12 Explain locked XGBoost predictions with TreeSHAP
13 Add `Disputed` only in collection-stage sensitivity analysis
14 Add `DaysToSettle` only in an explicitly labelled leakage stress test

The generated surrogate contains 2,466 rows, mirroring the documented IBM row count, with 1,479 training, 493 validation and 494 locked test rows. In the locked test, 275 surrogate invoices are late (55.67%). The numerical generator intentionally combines customer-specific risk, prior behaviour, amount, billing channel, seasonality and dispute status with stochastic outcome variation. The task is therefore learnable without being deterministic.

6.2. Late-payment classification

Table 2 shows the issuance-time results. Random forest has the highest surrogate PR-AUC (0.7062) and ROC-AUC (0.6978). Logistic regression has lower ranking performance but the highest recall at its validation-selected threshold (0.7745). The validation-weighted ensemble does not dominate: its PR-AUC is 0.7015 and F1 is 0.6189. That is a useful pipeline check because it demonstrates that the ensemble rule is not mechanically designed to win every metric.

Figure 3 makes the same point visually. The curves are relatively close, and none supports a claim of overwhelming algorithmic superiority. More importantly for the user's production requirement, the common legend is deliberately

Table 2: Surrogate pipeline verification: Locked classification metrics.

Model	ROC-AUC	PR-AUC	Precision	Recall	F1	Bal. Acc.	MCC	Brier	Threshold
Logistic	0.6708	0.7003	0.6377	0.7745	0.6995	0.6110	0.2357	0.2374	0.3657
Random Forest	0.6978	0.7062	0.7198	0.5418	0.6183	0.6385	0.2789	0.2238	0.5352
HistGB	0.6821	0.6957	0.7232	0.2945	0.4186	0.5765	0.1815	0.2325	0.7066
XGBoost	0.6804	0.6949	0.7014	0.3673	0.4821	0.5855	0.1868	0.2245	0.6258
PR-weighted ensemble	0.6939	0.7015	0.7089	0.5491	0.6189	0.6330	0.2668	0.2245	0.5155

Table 3: Surrogate pipeline verification: Late-days regression

Model	MAE	RMSE	Median AE	R^2
Ridge	7.6415	10.3636	6.0529	0.0792
Random Forest	8.1235	10.4078	6.8173	0.0713
HistGB	7.9419	10.5723	6.0516	0.0417
XGBoost	7.6776	10.4172	6.0501	0.0696

placed *below* both panels, entirely outside the data rectangles. The curves, reference lines and corner regions remain unobstructed.

The threshold-dependent ranking differs from the threshold-independent ranking because the cutoff is selected on validation data. This is not a contradiction. ROC-AUC and PR-AUC assess score ordering across all thresholds; F1 and MCC assess one operating point. In a deployed system, threshold stability can matter as much as a few thousandths of AUC.

6.3. Late-days regression

Table 3 summarizes the surrogate regression task. Ridge records the lowest RMSE (10.3636 days) and MAE (7.6415 days), while the three nonlinear models are similar and do not improve the stress test. The low R^2 values are intentional evidence against overclaiming: the surrogate includes substantial stochastic variation after the binary late/not-late decision, so estimating exact delay length is harder than ranking lateness risk.

Figure 4 shows why R^2 is modest. Many invoices with similar predicted delay realize quite different late-day outcomes. A production system should therefore avoid presenting the point forecast as a promised settlement date. It is better interpreted as a severity estimate to accompany the binary risk score. Prediction intervals or quantile regression would be a natural extension once the original data are available.

6.4. Calibration and probability use

Figure 5 compares reliability for three representative classifiers. The legend is placed below the axes and outside the plotting rectangle. The purpose of the figure is not to declare one surrogate probability model "correct"; it is to ensure that the reporting package exposes calibration rather than relying on discrimination alone.

The Brier scores in Table 2 range from 0.2238 to 0.2374. Because the surrogate test prevalence exceeds one half, these numbers cannot be compared casually with a rare-event benchmark. In the real IBM run, the same table should be interpreted jointly with the observed late-payment prevalence and the reliability plot.

6.5. What the XGBoost Model uses

The global SHAP ranking in Figure 6 is generated from the issuance-time XGBoost model only. No post-settlement variable can appear because the explainer receives the same feature matrix used at prediction time. That detail matters: explainability is downstream of the leakage audit, not a substitute for it.

The companion dispersion view in Figure 7 adds direction. Points to the right of zero raise the model's log-odds relative to its baseline; points to the left reduce them. The low-to-high scale is shown as a separate color bar rather than a legend box over the observations. This choice is visual but substantive: dense explanation plots are especially easy to obscure with default legend placement.

For a real deployment, the global SHAP ranking should be audited by time period and business segment. A feature can be globally influential because it is a proxy for country, customer cohort, or process change. Stability checks are therefore more useful than a single timeless "top ten" list.

6.6. Collector Capacity Rather Than Arbitrary Accuracy

Figure 8 translates ranking into review capacity. At each budget, the team is assumed to inspect the highest-risk invoices first. The figure does not assume that inspection prevents late payment. It simply reports how much of the positive class is concentrated near the top of the score list.

This distinction avoids a common leap from prediction to economics. A top-ranked invoice may already be destined to pay late regardless of collection effort, while another may be highly responsive to a reminder. Risk ranking is therefore a prerequisite for intervention design, not evidence of intervention effect. A genuine value model would require randomized or quasi-experimental treatment data, contact cost, invoice exposure, and the marginal effect of a collection action.

6.7. The leakage stress test

The strongest numerical result in the entire surrogate experiment is also the result that should *not* be celebrated. When `DaysToSettle` is deliberately added to the XGBoost predictor set, PR-AUC jumps from 0.6949 to 0.9921 and ROC-AUC from 0.6804 to 0.9869 (Fig. 9). Because days to settlement is realized only after payment, this spectacular gain is invalid for issuance-time prediction.

The leakage experiment is useful because it provides a concrete failure mode. If a future reproduction on the original IBM file shows unexpectedly extreme performance, the first diagnostic should be feature availability rather than model sophistication. Settlement-related fields should be inspected immediately.

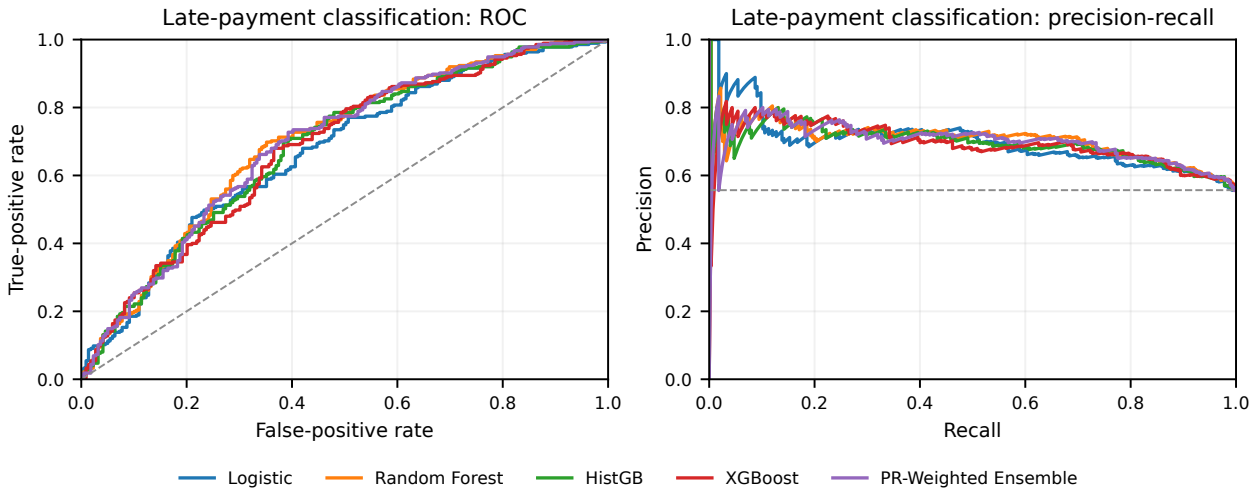


Figure 3: Surrogate pipeline-verification ROC and precision–recall curves.

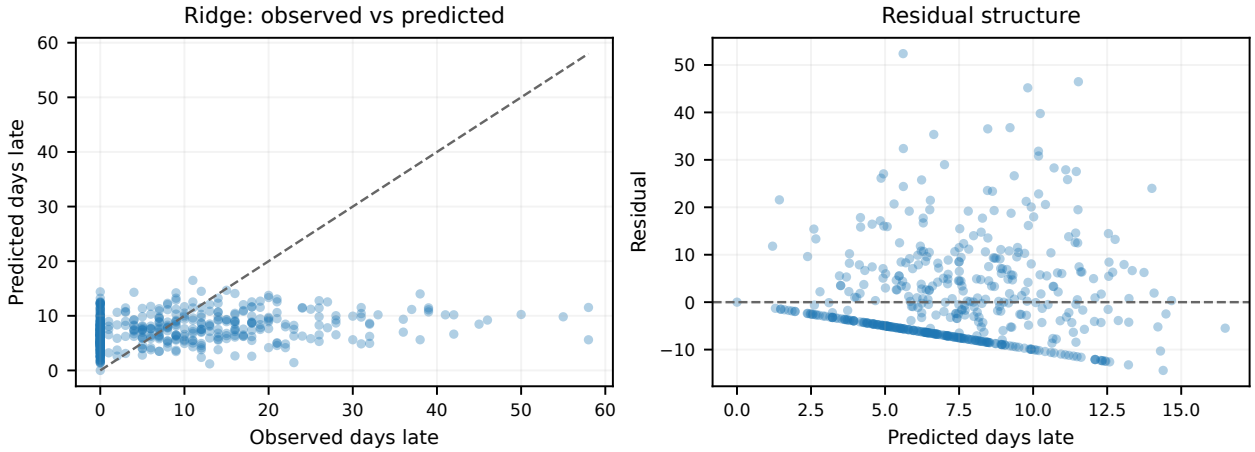


Figure 4: Surrogate pipeline-verification regression diagnostics for the lowest-RMSE model.

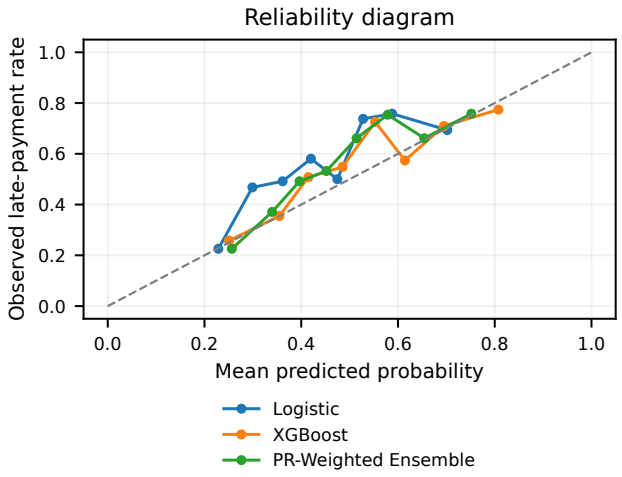


Figure 5: Surrogate reliability diagram.

7. Conclusion

Late-payment prediction is useful only when the information set matches the decision time. This study therefore places feature timing ahead of algorithm ranking. Its issuance-time design excludes settlement date, days to settlement and the current late-days outcome, builds customer history from preceding invoices only, and postpones dispute status to a later collection-stage sensitivity analysis. Classification, delay regression, calibration, SHAP diagnostics and fixed-capacity retrieval are then evaluated under one chronological protocol.

The historical IBM CSV could not be retrieved in the build environment, although its preserved documentation verifies a 2,466-row, 12-field simulated accounts-receivable schema. The archive consequently provides a real-data execution mode and a deterministic schema-surrogate mode rather than inventing IBM performance. The surrogate's deliberate leakage audit illustrates the practical reason for this restraint: adding post-settlement `DaysToSettle` raises XGBoost PR-AUC from 0.6949 to 0.9921 and ROC-AUC from 0.6804 to 0.9869. That apparent improvement is

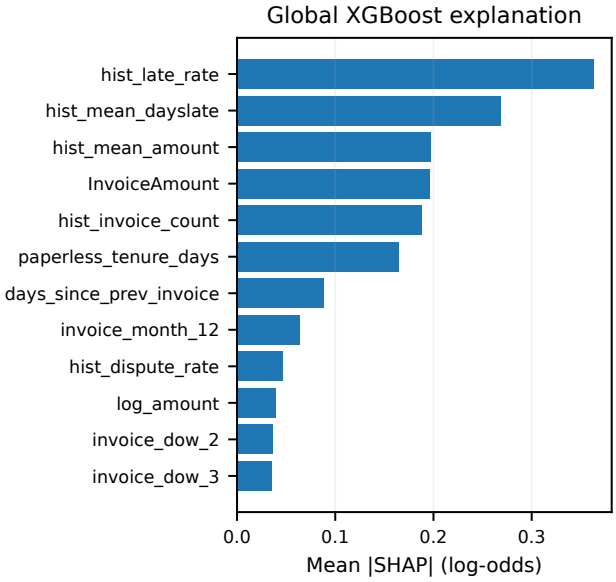


Figure 6: Global mean absolute TreeSHAP values in the deterministic surrogate verification run.

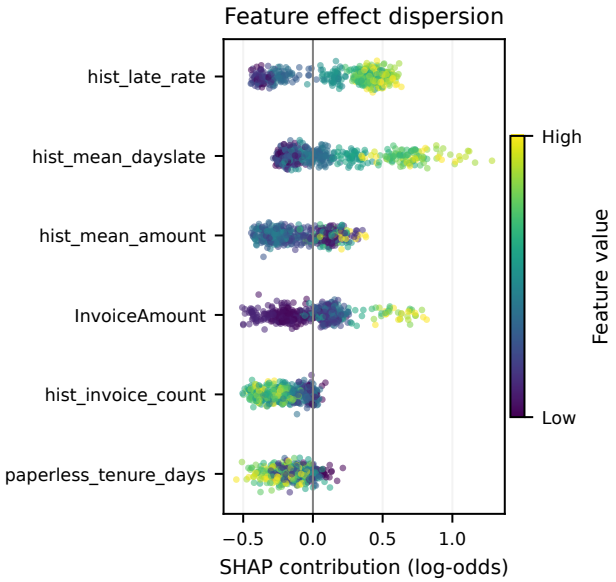


Figure 7: Surrogate SHAP contribution dispersion for the six strongest transformed features.

invalid for early warning.

The immediate empirical next step is to place the preserved IBM CSV in the documented data path and rerun the package unchanged. A stronger production study should additionally retain timestamps for disputes and collection actions, evaluate rolling temporal and unseen-customer tests, estimate delay uncertainty, and test interventions prospectively. Those additions would move the system from describing which invoices look risky toward learning which collection actions improve payment timing under real capacity constraints.

A practical system can separate ledger extraction, as-of feature computation, model scoring, explanation, and collector policy. The user interface should show only what

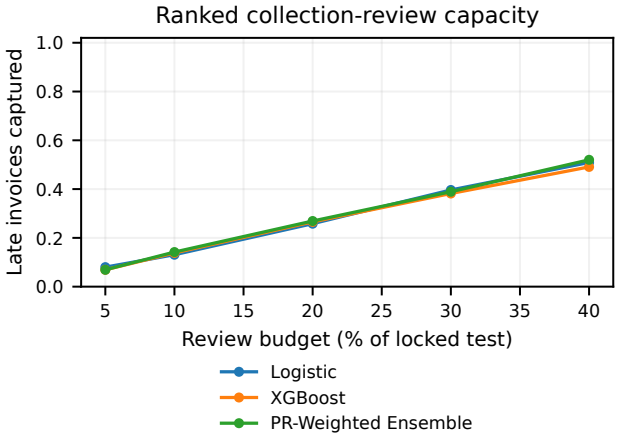


Figure 8: Surrogate late-invoice retrieval under fixed review budgets.

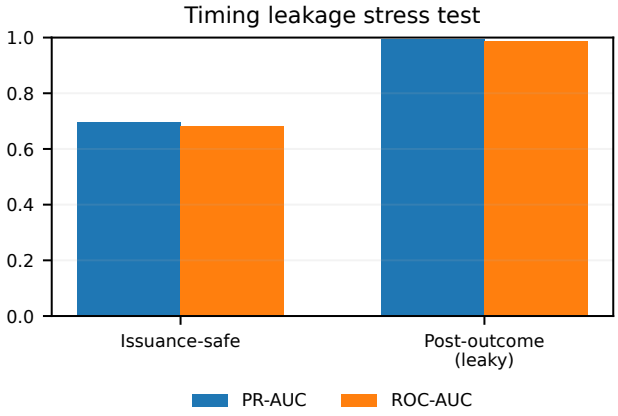


Figure 9: Deliberate timing-leakage stress test on the surrogate.

supports the decision: invoice amount, due date, risk score or band, estimated delay, a few model drivers, and relevant prior history. Managers need calibration and queue performance over time; model-risk reviewers need versioned features and audit logs. Keeping model output separate from collection policy also allows a team to reserve capacity for rule-based, relationship-driven, or exploratory cases rather than letting the score become the sole determinant of attention.

Declarations and ethical statements

Conflict of interest: The author declares that there is no conflict of interest.

Funding statement: The author declares that no specific funding was received for this research.

Artificial Intelligence usage statement: During the preparation of this manuscript, the author utilized ChatGPT solely for language refinement and grammatical corrections. The author carefully reviewed and revised the generated content and take full responsibility for the accuracy, integrity, and originality of the final manuscript.

Availability of data and materials: The data and/or materials that support the findings of this study are avail-

able from the corresponding author upon reasonable request.

CRediT authorship contribution statement

Saumen Bandyopadhyay: Conceptualization, Investigation & Writing—original draft, review and editing.

Publisher's note

Krrish Scientific Publications Pvt. Ltd. and the ***Journal of Computing and Data Technology*** remain neutral with regard to jurisdictional claims in published maps and institutional affiliations.

References

[1] Moore WR, van Vuuren JH. A framework for modelling customer invoice payment predictions. *Machine Learning with Applications*. 2024 Sep 1;17:100578. Available from: https://doi.org/10.1016/j.mlwa.2024.100578.

[2] Schoonbee L, Moore WR, Van Vuuren JH. AA Machine-Learning Approach towards Solving the Invoice Payment Prediction Problem. *South African Journal of Industrial Engineering*. 2022 Dec 1;33(4):126-46. Available from: https://doi.org/10.7166/33-4-2726.

[3] Appel AP, Malfatti GL, Cunha RL, Lima B, de Paula R. Predicting Account Receivables with Machine Learning. *arXiv preprint* arXiv:2008.07363. 2020 Aug 11. Available from: https://doi.org/10.48550/arXiv.2008.07363.

[4] Li H. Estimation of Customer's Repayment Date Based on Machine Learning Methods. In *Proceedings of the 4th International Conference on Economic Management and Green Development* 2021 Aug 14 (pp. 209-221). Singapore: Springer Singapore. Available from: https://doi.org/10.1007/978-981-16-5359-9_25.

[5] Shah HS. Customer Payment Prediction in Account Receivable . *International Journal of Science and Research (IJSR)*. 2019;8(1):642-4. Available from: 10.21275/ART20194177.

[6] Mogili VB. Empowering Human-AI Collaboration: Enterprise Technology Platforms and Human Expertise Synergy in Healthcare, Finance, and Scientific Research. Journal of Computer Science and Technology Studies. 2025 Jul 10;7(7):513–20. Available from: https://doi.org/10.32996/jcsts.2025.7.7.57

[7] Liu B, Zhang Z, Yan J, Zhang N, Zha H, Li G, Li Y, Yu Q. A Deep Learning Approach with Feature Derivation and Selection for Overdue Repayment Forecasting. *Applied Sciences*. 2020 Nov 27;10(23):8491. Available from: https://doi.org/10.3390/app10238491.

[8] Berloco C, De Francisci Morales G, Frassineti D, Greco G, Kumarasinghe H, Lamieri M, Massaro E, Miola A, Yang S. Predicting corporate credit risk: Network contagion via trade credit. *PLoS One*. 2021 Apr 29;16(4):e0250115. Available from: https://doi.org/10.1371/journal.pone.0250115.

[9] Breiman L. Random Forests. Machine learning. 2001 Oct;45(1):5-32. Available from: https://doi.org/10.1023/A:1010933404324.

[10] Friedman JH. Greedy function approximation: A gradient boosting machine. *Annals of statistics*. 2001 Oct 1:1189-232. Available from: https://doi.org/10.1214/aos/1013203451.

[11] Chen T, Guestrin C. XGBoost: A Scalable Tree Boosting System. In*Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining* 2016 Aug 13 (pp. 785-794). Available from: https://doi.org/10.1145/2939672.2939785.

[12] Lundberg SM, Erion G, Chen H, DeGrave A, Prutkin JM, Nair B, Katz R, Himmelfarb J, Bansal N, Lee SI. From local explanations to global understanding with explainable AI for trees. *Nature Machine Intelligence*. 2020 Jan;2(1):56-67. Available from: https://doi.org/10.1038/s42256-019-0138-9.

[13] Ribeiro MT, Singh S, Guestrin C. "Why Should I Trust You?": Explaining the Predictions of Any Classifier. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* 2016 Aug 13 (pp. 1135-1144). Available from: https://doi.org/10.1145/2939672.2939778.

[14] Niculescu-Mizil A, Caruana R. Predicting good probabilities with supervised learning . In*Proceedings of the 22nd international conference on Machine learning* 2005 Aug 7 (pp. 625-632). Available from: https://doi.org/10.1145/1102351.1102430.

[15] Saito T, Rehmsmeier M. The Precision-Recall Plot Is More Informative than the ROC Plot When Evaluating Binary Classifiers on Imbalanced Datasets. *PloS One*. 2015 Mar 4;10(3):e0118432. Available from: https://doi.org/10.1371/journal.pone.0118432.

[16] Davis J, Goadrich M. The relationship between Precision-Recall and ROC curves. In *Proceedings of the 23rd International conference on Machine Learning* 2006 Jun 25 (pp. 233-240). Available from: https://doi.org/10.1145/1143844.1143874.

[17] Bandyopadhyay S. Explainable Artificial Intelligence for Retail Price Estimation and Pricing-Risk Detection Using Multivariate Product Attributes. *Journal of Computing and Data Technology*. 2025 Oct. 22;1(2):19-27. Available from: https://doi.org/10.71426/jcdt.v1.si1.pp19-27

[18] Taghizadeh A, Arash Behzadi, Nika Fakhri, Seyedeh Atena Naghipour. Comparing Machine Learning Models for Stock Prediction: LSTM Comes Out on Top. *Journal of Modern Technology*. 2026 Feb. 16;3(1):361-6. Available from: https://doi.org/10.71426/jmt.v3.i1.pp361-366

[19] Penaganti R. Graph Neural Network-Based Framework for Real-Time Financial Fraud Detection in Digital Payment Ecosystems. *Journal of Computing and Data Technology*. 2025 Dec. 31;1(2):91-7. Available from: https://doi.org/10.71426/jcdt.v1.i2.pp91-97