algorithms

MDPI

Article

RQS-SVM: Rough Quotient Space Support Vector Machine for Bankruptcy Prediction

Abdelhamid Tamouh [1,†], Mouna Tarik [2,†], Ayoub Mniai [1,†] and Khalid Jebari [1,†,*]

[1] IABL, FSTT, Abdelmalek Essaadi University, Tetouan 93000, Morocco; tamouh.abdelhamid@etu.uae.ac.ma (A.T.); a.mniai@gmail.com (A.M.)
[2] Laboratory of System Control and Decision (LSCD), New Science School of Engineering (ENSI), Tangier 90100 Morocco; mouna.tarik@ensi.ma
* Correspondence: khalid.jebari@gmail.com
† All the authors contributed equally to this work.

Abstract

Bankruptcy prediction is a severely imbalanced classification problem. We propose RQS-SVM, a weighted support vector machine trained on a restructured sample: neighborhood rough approximations separate certain observations from boundary ones, which are compressed into possibilistic prototypes, and both classes receive the same total weight. RQS-Ensemble adds a quantile transformation and averages ten members trained on balanced subsamples. Explanatory, not quantitative, bounds relate compression to decision changes and excess risk. On three bankruptcy datasets (imbalance ratios 14–53), five-fold cross-validation (firm-grouped for the US panel) compares them with 17 competing methods from the literature. Separately, an ablation study of 17 configurations of RQS-SVM and seven of RQS-Ensemble shows that class mass balancing, the quantile transformation and the ensemble drive the performance, while the granulation reduces the training set by 14–37% without significantly changing accuracy. In G-mean (Wilcoxon signed-rank tests over 15 fold–dataset blocks, Holm correction) with default competitor settings, RQS-Ensemble is significantly better than 14 methods, not significantly different from Balanced Random Forest, RUSBoost and Self-Paced Ensemble, and significantly worse than EasyEnsemble; when all the methods tune their threshold, it ranks sixth of ten. Its PR-AUC is lower than that of tree ensembles, largely because of projection-based inference.

Keywords: support vector machine; rough sets; quotient space; possibilistic clustering; class imbalance; bankruptcy prediction; ensemble learning

Check for updates

Academic Editor: Mostafa Abbaszadeh

Received: 6 November 2026
Revised: 22 November 2026
Accepted: 24 November 2026
Published: 29 November 2026

Copyright: © 2026 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license.

1. Introduction

Corporate bankruptcy represents a major economic challenge with detrimental implications that extend first to immediate stakeholders and subsequently to business partners. It undermines investor confidence and introduces instability throughout a country's financial system [1]. Bankruptcy prediction has attracted considerable research interest over the years. Initially, statistical approaches dominated the field. In particular, discriminant analysis-based models such as Altman's Z-score and Springate's model, which was derived from it, emerged and remained influential for several decades [2]. Subsequently, Ohlson's logit-based model and Zmijewski's probit-based model became the subject of extensive investigation [3].

Although these methods contributed significantly to the fundamental understanding of corporate failure mechanisms, they remain constrained by the assumption of a linear

relationship between explanatory variables and bankruptcy risk [4]. Furthermore, they often ignore interactions among financial variables, depend heavily on data quality, and exhibit sensitivity to macroeconomic fluctuations [3].

Machine learning (ML) approaches have emerged as a promising alternative for overcoming many of these limitations. Driven by data, these approaches provide adaptive strategies that flexibly model the complex and nonlinear relationships underlying financial phenomena [5].

Algorithms such as Random Forests, support vector machines (SVMs), and ensemble learning methods, particularly XGBoost, LightGBM, and CatBoost, do not assume a predefined functional form. This enables them to uncover latent patterns and intricate dependencies among variables within large and heterogeneous datasets [6]. Consequently, these methods have gained substantial importance in bankruptcy prediction research [7] owing to their ability to process heterogeneous and incomplete data, adapt to high-dimensional feature spaces, and learn from extensive datasets in order to produce robust and generalizable predictions [8].

Nevertheless, these methods are not without limitations. They often require substantial preprocessing and hyperparameter tuning, raise interpretability challenges—particularly in high-stakes domains—and may involve considerable computational complexity in certain implementations [9].

With the continuous expansion of financial datasets and improvements in computational capabilities, deep learning (DL) has emerged as a powerful extension of ML in the field of bankruptcy prediction. DL leverages deep neural architectures to autonomously extract hierarchical representations from raw inputs, thereby providing a robust mechanism for capturing complex patterns. Architectures such as multilayer perceptrons (MLPs), convolutional neural networks (CNNs), and recurrent neural networks (RNNs), including long short-term memory (LSTM) models, have demonstrated remarkable effectiveness in a wide range of applications, including economic forecasting and risk assessment [10]. Within the context of bankruptcy prediction, DL models enable the modeling of temporal dependencies, the capture of highly nonlinear relationships, and effective generalization across large-scale datasets [11]. For instance, LSTM networks are particularly well-suited for sequential financial data due to their ability to preserve long-term dependencies, whereas CNNs have proven effective in identifying discriminative patterns within structured financial data matrices [12,13].

Despite these advantages, DL approaches are frequently criticized for their high computational requirements and the complexity of their architectural design. Their black-box nature further complicates interpretability, making it difficult to justify predictions, which is an issue of particular concern in regulatory and credit assessment contexts.

A significant challenge for these algorithms lies in class imbalance, which is a prevalent characteristic of bankruptcy datasets in which solvent companies vastly outnumber bankrupt ones. This imbalance biases most ML algorithms toward the majority class. Specifically, classification rules that correctly predict majority class instances contribute disproportionately to overall accuracy, whereas those favoring the detection of minority class instances may be undervalued or even interpreted as noise. Consequently, minority instances tend to be misclassified, thereby reducing the practical utility of predictive models in real-world bankruptcy prediction scenarios [14].

The practical relevance of this problem is also reflected in the statistical characteristics of bankruptcy datasets. In particular, bankrupt firms generally constitute only a small fraction of the observations, resulting in a pronounced class imbalance between financially healthy and distressed companies. This imbalance can lead to misleadingly high overall accuracy when conventional classification models predominantly predict the majority

class while failing to adequately identify financially distressed firms. The three datasets investigated in this study further illustrate this challenge as the proportion of bankrupt firms is substantially lower than that of non-bankrupt firms across the US, Taiwanese, and UK datasets. Such distributions highlight the need for classification approaches that explicitly account for minority class information rather than relying solely on global accuracy. Therefore, addressing class imbalance while preserving the informative structure of minority and boundary observations represents a critical methodological challenge in bankruptcy prediction and provides the primary motivation for the proposed RQS-SVM framework.

To address this gap, the present paper introduces a novel hybrid framework that synergistically combines three powerful paradigms:

- Rough Set Theory: Characterizes data uncertainty through lower approximations and boundary regions, thereby distinguishing reliable instances from ambiguous ones.
- Quotient Space Theory: Provides hierarchical granularity reduction while preserving topological structure, enabling compact yet informative representations.
- Support Vector Machine: Offers robust classification through maximum-margin separation, enhanced by adaptive sample weighting mechanisms.

The key idea is to restructure the training sample before SVM classification: certain observations are kept unchanged, whereas the uncertain boundary region is replaced by a small set of possibilistic prototypes, and the classes are given equal total weight. The individual components are established techniques; the contributions of this study lie in their specific interaction:

1. An instance-level use of neighborhood rough approximations that restricts compression to the boundary region, in contrast to existing rough set/SVM hybrids that use rough sets for attribute reduction or margin modification (Section Hybrid Rough Set, Fuzzy and Granular SVM Models).
2. A class-aware possibilistic granulation of the boundary region of each class, which compresses the training set by 14–37%, combined with a class mass balancing rule that gives the minority class the same total weight as the majority class (Section 3.5). The ablation study shows that the class mass balancing is responsible for the performance of the single model and that this compression is obtained without significantly changing the accuracy of RQS-Ensemble (Section 4.3).
3. A quotient space formalization with a train-only projection together with a bound on the decision disagreement and a bound on the excess risk induced by compression (Theorem 1); the bound is explanatory rather than a quantitative guarantee: on our data it holds but exceeds the observed effects by one to two orders of magnitude (Section 4.9).
4. RQS-Ensemble, an extension for heavy-tailed financial ratios that combines a quantile transformation with an ensemble of RQS-SVM members trained on balanced subsamples drawn from the rough partition (Section 3.8).
5. A component-wise ablation of 17 configurations for RQS-SVM and seven configurations for RQS-Ensemble, each of which changes one component of the proposed method; a comparison with 17 competing methods, including recent ensemble and deep imbalance learning methods, and statistical significance testing on three bankruptcy datasets under a protocol in which all the preprocessing and tuning are fitted on training parts only and the folds of the US panel are grouped by firm (TW and UK, which have no firm identifier, cannot be grouped) (Section 4.2); and a cost-sensitive analysis and a temporal validation with provided code, seeds and fold assignments (Section 4.2 and Data Availability Statement).

The ablation study (Section 4.3) is also used to state precisely which components are responsible for the performance and which ones mainly contribute to compression; we report the components that do not yield a significant improvement as well.

The remainder of this paper is organized as follows. Section 2 reviews the existing literature on bankruptcy prediction with a particular emphasis on machine learning approaches and class imbalance. Section 3 presents the theoretical foundations and detailed methodology of the proposed rough quotient space support vector machine (RQS-SVM) framework. Section 4 describes the datasets, preprocessing procedures, experimental settings, and evaluation metrics followed by a comprehensive performance analysis and discussion. This section also includes a computational efficiency and complexity analysis and a statistical analysis to further assess the effectiveness and robustness of the proposed framework. Section 5 discusses the limitations of the study. Finally, the concluding section summarizes the main findings, contributions, limitations, and directions for future research.

2. Literature Review

Corporate bankruptcy prediction has long represented a major area of interest in quantitative finance and risk management. The identification of early warning signals enables the anticipation of financial losses, the assessment of credit risk, the optimization of investment strategies, and the guidance of economic policy decisions. As the volume and diversity of available data continue to increase, traditional methods suffer from inherent limitations, whereas ML [5,15] and DL techniques [10,16] have emerged as powerful adaptive alternatives.

However, the application of these techniques to bankruptcy prediction faces a persistent structural obstacle: class imbalance within datasets. Bankrupt firms typically constitute only a very small minority (often below 5% and sometimes less than 1%) in real-world databases. This disparity between the majority class (solvent firms) and the minority class (bankrupt firms) poses a fundamental challenge to the development of effective predictive models [17].

To address this imbalance, researchers initially explored data-level rebalancing techniques prior to model training, primarily including:

1. Oversampling: Artificial augmentation of the minority class through random duplication (random oversampling) or synthetic sample generation, such as SMOTE [18], which interpolates new instances from real observations.
2. Undersampling: Reduction of the majority class through random subset selection, although this may lead to information loss regarding healthy firms [19].
3. Advanced Hybrid Methods: Techniques such as ADASYN, SMOTEENN and Borderline-SMOTE [3,20], which combine both oversampling and undersampling strategies. Empirical studies have demonstrated that these techniques significantly improve minority class metrics such as recall, AUC-ROC and weighted precision [21].

Machine learning algorithms have been extensively employed in contemporary bankruptcy research due to their ability to capture nonlinear interdependencies, manage heterogeneous data types, and improve predictive generalization capabilities. The prominent approaches include [6,22]:

- Random Forests: Bootstrap-aggregated decision trees that are robust to noise and feature correlations.
- Support Vector Machines (SVMs): Margin-maximization classifiers operating within transformed feature spaces.
- Decision Trees: Interpretable rule-based predictive models.

- Boosting Algorithms (XGBoost, LightGBM, and CatBoost): Ensemble techniques combining weak learners to enhance predictive accuracy.

To mitigate class dominance effects, several key strategies have been proposed:

- Cost-Sensitive Learning: Algorithms such as XGBoost allow class weighting (scale_pos_weight) to penalize errors on the minority class more heavily [23].
- Integrated Sampling: Techniques such as Balanced Random Forest perform resampling at each iteration [24].
- Alternative Evaluation Metrics: Metrics including the F1-score, AUC-PR, G-mean, and Matthews correlation coefficient (MCC) provide more reliable assessments in imbalanced contexts [25]. Some kernel methods optimize such a criterion directly: Sun et al. [26] propose a fast AUC-maximization learning machine that detects outliers simultaneously, which makes the ranking criterion robust to noisy observations. RQS-SVM addresses the same two issues differently: imbalance through the weighting of the quotient representation and outliers through the possibilistic typicality of the boundary granules.

Ensemble methods that combine resampling and bagging or boosting are now standard baselines for imbalanced data: Balanced Random Forest draws a balanced bootstrap sample for each tree [27]; EasyEnsemble trains AdaBoost learners on several random undersamples of the majority class [28]; RUSBoost inserts random undersampling into boosting [29]; and Self-Paced Ensemble selects majority samples according to their classification hardness [30]. Gradient boosting libraries such as LightGBM [31] support class weights. For neural networks, the focal loss [32], the class-balanced loss based on the effective number of samples [33] and the label-distribution-aware margin loss with deferred reweighting (LDAM-DRW) [34] modify the training objective. These methods are included in our experimental comparison (Section 4.4).

Empirical studies confirm that these approaches, when combined with appropriate data rebalancing techniques, improve overall predictive performance while preserving bankruptcy detection capabilities [35,36]. Recent financial distress studies combine such imbalance handling with richer inputs: Song et al. [37] predict class-imbalanced financial distress with machine learning models built on an index system that adds management, textual and social responsibility features to the financial ones. Others act on the inference model itself: Yu et al. [38] propose a two-stage case-based reasoning classification paradigm for financial distress prediction when the data are both incomplete and imbalanced.

Deep learning methods have rapidly expanded within financial applications due to their automatic feature extraction capabilities. Research has shown that convolutional neural networks [39] perform effectively in bankruptcy prediction tasks [40]. These methodologies also demonstrate substantial effectiveness when integrated with other techniques, including recurrent neural networks (RNN/LSTM) [40] and various artificial neural network (ANN) configurations [9].

However, under severe class imbalance conditions, these models become prone to overfitting the majority class, particularly when bankrupt firms are extremely underrepresented. Moreover, their substantial data requirements further complicate their application in such contexts.

The mitigation strategies include:

- Weighted Loss Functions: Modification of binary cross-entropy using class frequency-based weighting schemes or the adoption of focal loss [41] to emphasize difficult samples.
- Balanced Batch Training: Construction of mini-batches with controlled class proportions [41].

- Temporal Data Augmentation: For LSTM/GRU models, synthetic sequence generation using dynamic time warping or Gaussian perturbation techniques [42].

A critical obstacle to the adoption of complex models—particularly in finance—remains interpretability. Deep neural networks often operate as black boxes, producing accurate predictions without transparent decision-making mechanisms. Post hoc explanation tools such as LIME [43] and SHAP [44] address this limitation by quantifying feature importance for individual predictions, including rare-event cases. Alternative approaches seek to improve interpretability either by integrating domain knowledge and interpretable structures into the model architecture or applying explainability mechanisms to financial prediction models. For example, interpretable models based on accounting information and Bayesian networks have been developed for bankruptcy prediction [45], while fuzzy–neural approaches have also been investigated to obtain interpretable nonlinear bankruptcy models [46]. More recently, explainable artificial intelligence frameworks combining ensemble learning with SHAP, partial dependence, and counterfactual explanations have been proposed for financial distress prediction [47]. CNN-based bankruptcy prediction has likewise been combined with XAI techniques to improve the interpretability of predictions based on financial ratios [48].

Recent innovations also include the integration of textual data (financial reports and news articles) through natural language processing (NLP) models and behavioral indicators. Interpretability and class imbalance have also been addressed together: Sun et al. [49] combined feature selection, resampling methods for imbalanced data and an interpretation module in a decision support system for credit default forecasting.

Hybrid Rough Set, Fuzzy and Granular SVM Models

Several studies have combined rough sets, fuzzy sets or granular computing with SVMs. They can be grouped according to the level at which the uncertainty model acts.

In bankruptcy and financial distress prediction, rough sets have mainly been used for attribute reduction: the reduct obtained from the decision table is used as the input of a standard SVM [50,51]. The training sample and the SVM formulation are unchanged, and class imbalance is not addressed.

Rough-margin SVMs introduce lower and upper margins into the SVM optimization problem [52]. Fuzzy SVMs assign to every training point a membership that scales its misclassification cost [53]; FSVM-CIL adapts these memberships to class imbalance [54], and fuzzy–rough SVMs derive them from fuzzy–rough lower approximations [55]. These methods reweight individual points but still train on the full sample.

Granular SVMs with repetitive undersampling (GSVM-RU) build granules by iteratively extracting support vectors from the majority class [56], while SMOTE-RSB* uses rough set theory to filter synthetic minority samples [57]. These methods modify the class distribution by removing majority samples or adding synthetic ones. Granular structures also continue to be built into SVM formulations: Lang et al. [58] combine granular-ball computing with a fuzzy twin SVM for general pattern classification.

RQS-SVM uses rough sets at the instance level to separate certain from uncertain observations, keeps all certain observations unchanged, and replaces only the boundary region of each class with possibilistic prototypes; the class imbalance is then corrected by giving both classes the same total weight. Following quotient space theory [59], the training sample is thus replaced by a coarser weighted representation onto which test observations are projected. Table 1 summarizes these differences.

Table 1. Positioning of RQS-SVM with respect to hybrid rough set, fuzzy and granular SVM models.

Approach	Level	Uncertainty Model	Training Sample	Guarantee
RST reduction + SVM [50,51]	Features	Rough set reduct	Full, unweighted	–
Rough-margin SVM [52]	Optimization	Rough margins	Full	–
FSVM [53], FSVM-CIL [54]	Optimization	Fuzzy membership	Full, point-weighted	–
FRSVM [55]	Optimization	Fuzzy–rough lower approx.	Full, point-weighted	–
GSVM-RU [56]	Sample	Majority SV granules	Majority undersampled	–
SMOTE-RSB * [57]	Sample	Rough set filtering	Minority oversampled	–
RQS-SVM	Sample (quotient)	Rough partition + PCM	Lower approx. kept; boundary → prototypes; class mass balancing (weights constant within each class)	Theorem 1 (explanatory)

Despite the diversity of the existing solutions, most approaches address class imbalance either by modifying the data distribution or by modifying the learning objective. Resampling methods such as SMOTE, ADASYN, and undersampling alter the original sample distribution and may introduce synthetic observations or discard potentially informative majority class instances. Cost-sensitive and class-weighted methods preserve the original data but mainly modify the penalty associated with classification errors without explicitly modeling the structural uncertainty of financial observations. Ensemble and boosting methods can improve predictive performance, but their effectiveness may depend on model configuration and parameter tuning. Deep learning approaches provide powerful nonlinear representation capabilities, yet they generally require larger amounts of data and may involve substantial computational and interpretability challenges. These limitations suggest that an alternative strategy should not only compensate for class imbalance but also distinguish reliable observations from ambiguous boundary instances while preserving their informative structure. In this context, rough set theory and quotient space theory provide complementary mechanisms for characterizing uncertainty and constructing hierarchical granular representations. Their integration with clustering and a class-weighted SVM, as proposed in this study, offers a structural alternative to conventional resampling-based solutions by reducing majority class dominance while retaining informative boundary information.

ML/DL-based bankruptcy prediction constitutes a rapidly evolving research domain that is constrained by the inherent problem of class imbalance. This structural characteristic biases conventional predictive models, thereby necessitating specialized data treatments and algorithmic adaptations. Although oversampling techniques, cost-sensitive learning, weighted loss functions, and hybrid approaches have demonstrated significant effectiveness, the field must still reconcile predictive performance with stability, generalizability, and interpretability—particularly within highly regulated domains such as finance, banking, and auditing.

3. Rough Quotient Space Support Vector Machines: Proposed Framework

The performance of support vector machines (SVMs) tends to deteriorate considerably in the presence of severe class imbalance. This challenge is especially evident in bankruptcy prediction, where the proportion of bankrupt firms is substantially lower than that of healthy firms. Under such conditions, the optimization process is dominated by the majority class, resulting in a decision boundary that is biased toward non-bankrupt companies. Consequently, the classifier exhibits a strong preference for the majority class, often at the

expense of accurately detecting bankrupt firms. This limitation can significantly reduce the effectiveness of SVM-based bankruptcy prediction models, particularly in terms of minority class recognition and overall discriminatory power.

In response to these limitations, a structured framework is developed to effectively handle class imbalance and enhance minority class discrimination through a sequence of interconnected processing stages. The proposed framework operates through seven procedures (P1 to P7), as illustrated in Figure 1. Each procedure transforms the data while preserving essential information and progressively addressing the challenges of imbalanced classification.

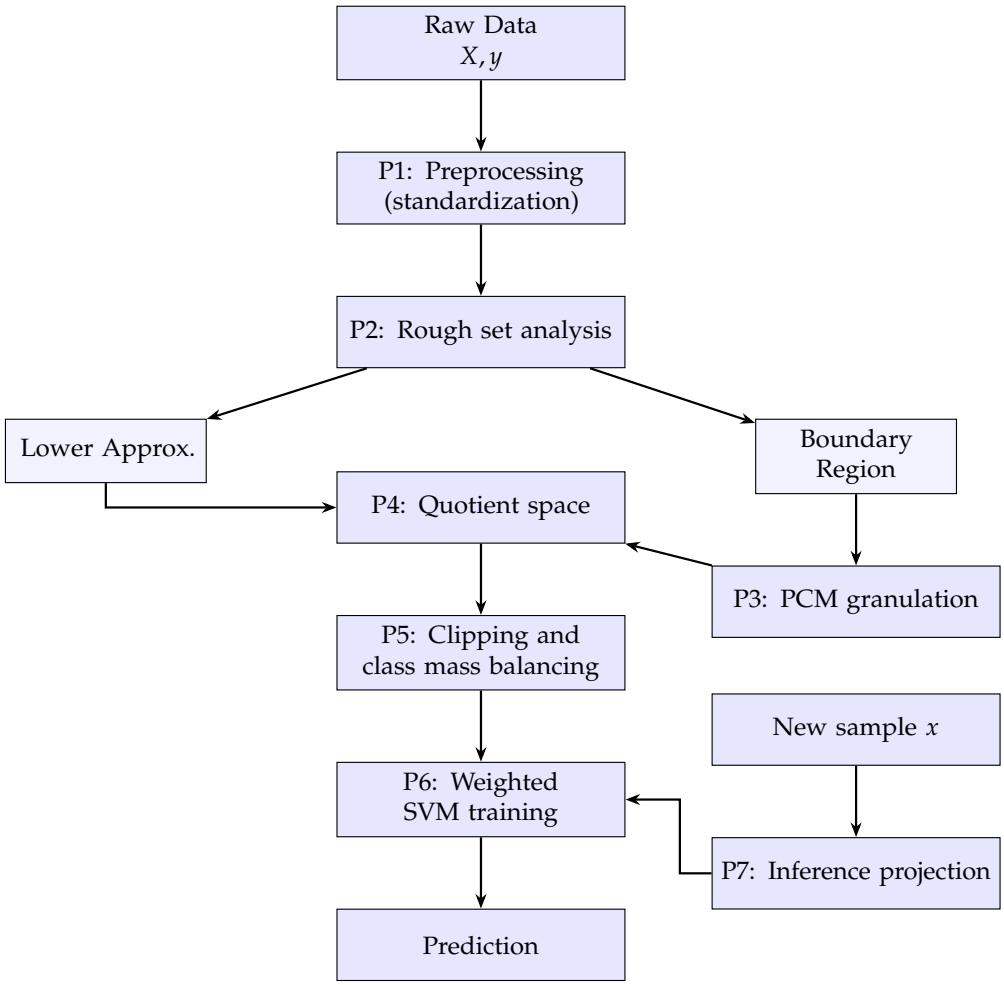


Figure 1. Architecture of RQS-SVM with the seven procedures, P1–P7, of Sections 3.1–3.7. Compared with a standard weighted SVM, which is trained directly on the standardized data (P1 then P6), RQS-SVM inserts four procedures before training: rough set analysis (P2, Section 3.2), PCM granulation of the boundary regions (P3, Section 3.3), construction of the quotient space (P4, Section 3.4), and weight clipping and class mass balancing (P5, Section 3.5); at inference, each new observation is projected onto the quotient space before classification (P7, Section 3.7). P1: preprocessing (Section 3.1); P6: weighted SVM training (Section 3.6).

The methodological contribution of the proposed RQS-SVM framework lies not in introducing rough set theory, quotient space theory, possibilistic C-means, sample weighting, or support vector machines as new individual techniques. These methodologies are well-established in the literature. Rather, the contribution lies in their structured integration into a unified learning architecture in which each component addresses a complementary aspect of imbalanced classification. Rough approximations identify reliable and uncertain regions of the feature space; quotient space construction provides a granular representation

that reduces redundant information; PCM transforms the uncertain boundary region into a compact set of representative prototypes; class mass balancing gives the two classes the same total weight (the weights derived from the granules are canceled by the clipping, Section 3.5); and the weighted SVM performs the final discriminative classification. Thus, the proposed framework establishes a functional interaction between uncertainty modeling, granular compression, prototype representation, imbalance adaptation, and cost-sensitive classification.

This section explicitly defines all the algorithmic parameters, control flags, and processing steps used throughout the RQS-SVM pipeline. In particular, the binary flags `use_pcm` and `use_weights` are formally defined as follows: `use_pcm` $\in \{0, 1\}$ determines whether the boundary region is compressed using possibilistic C-means (PCM), whereas `use_weights` $\in \{0, 1\}$ determines whether the weighting procedure (clipping and class mass balancing) is applied to the resulting quotient representation. When `use_pcm`=1, PCM compression is applied to boundary regions satisfying $|BN_c| > \tau$ with $\tau = 3$; otherwise, boundary samples are retained without clustering. When `use_weights`=1, the initial weights are subsequently adjusted using the IQR clipping followed by class mass balancing (Section 3.5); when `use_weights`=0, all quotient elements receive the weight 1. These definitions ensure that the mathematical formulation, implementation, and pseudocode describe the same computational procedure.

The proposed RQS-SVM framework is organized into seven procedures (Algorithm 1), each addressing a specific phase of the hybrid approach; RQS-Ensemble (Section 3.8) uses RQS-SVM as its base learner.

The main symbols of the procedures are defined in Table 2 with the values used in the experiments.

Table 2. Notation. "Data-driven": set on the training part of each fold by the rules of Section 3.8.

Symbol	Meaning	Value
α	Neighborhood scaling factor: $\epsilon = \alpha\,\epsilon_0$, where ϵ_0 is the median distance to the k-th nearest neighbor (Procedure 2)	Data-driven, in $[0.05, 0.45]$
ϵ	Neighborhood radius of the rough approximation (Procedure 2)	$\alpha\,\epsilon_0$, enlarged until at least 5% of the observations are boundary observations
τ	Minimal size of a boundary region BN_c for PCM to be applied (Procedures 3 and 4)	3
m	PCM fuzzifier (Procedure 3)	Data-driven, in $[1.1, 3]$
K_c	Number of granules (PCM prototypes) of class c, $K_c = \max(2, \min(k_{\max}, \lfloor \lvert BN_c \rvert / 2 \rfloor))$ (Procedure 3)	$k_{\max} = 5$
β	Initial weight of a boundary observation and scale of the prototype weights (Procedure 4)	Data-driven, in $[0.3, 0.9]$
λ	Minority amplification factor $(1 + \lambda)$, canceled by class mass balancing (Procedure 5)	0.5
$\mathbf{w} = (w_i)$	Weight of element i of the quotient representation $\mathcal{Q}$ (Procedures 4 and 5)	Class mass-balanced
C	Regularization parameter of the weighted SVM; the penalty of element i is $C\,w_i$ (Procedure 6)	$\{0.1, 1, 10, 100\}$ (inner CV); 1 in the ablation study
$K(\cdot, \cdot)$	Kernel of the SVM: RBF with parameter $\gamma = g/d$ (Procedures 6 and 7)	$g \in \{0.25, 1, 4\}$ (inner CV); γ = scale in the ablation study

3.1. Procedure 1: Data Preprocessing

The first phase applies standardization to ensure all features contribute equally to distance computations:

$$\tilde{\mathbf{x}}_i = \frac{\mathbf{x}_i - \boldsymbol{\mu}}{\boldsymbol{\sigma}}, \qquad \boldsymbol{\mu} = \frac{1}{n}\sum_{i=1}^{n}\mathbf{x}_i, \qquad \boldsymbol{\sigma} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(\mathbf{x}_i - \boldsymbol{\mu})^2}, \tag{1}$$

here, $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}$ denote the mean and population standard deviation computed from the training data, respectively. The denominator n is therefore used rather than $n-1$ for three reasons. First, standardization is a scaling step, not the estimation of a population variance, so the unbiasedness of the sample variance is not needed. Second, this is the estimator implemented by the `StandardScaler` of scikit-learn used in our code so that Equation (1) describes exactly the computation performed. Third, the choice has no practical effect: the two estimates differ by the factor $\sqrt{n/(n-1)}$, which is below 1.0002 for our training sets ($n \geq 4000$), and this factor is the same for all features so that it rescales all distances by the same amount. Since the radius ϵ is defined relative to the nearest neighbor distances (Equation (2)), the rough partition and the SVM are affected only through relative changes below 0.02%, which are negligible.

This transformation is critical for the neighborhood-based computations that follow as features with larger scales would otherwise dominate distance calculations.

3.2. Procedure 2: Rough Set Analysis

The rough set analysis identifies certain and uncertain regions through neighborhood-based approximations. The neighborhood radius is consistently defined throughout the mathematical formulation and pseudocode as

$$\epsilon = \alpha \cdot \text{median}\left(\{d_i^{(k)}\}_{i=1}^{n}\right), \quad k = \max(3, \min(20, \lfloor\sqrt{n}\rfloor)) \tag{2}$$

where $d_i^{(k)}$ denotes the distance from observation $\mathbf{x}_i$ to its k-th nearest neighbor, n denotes the number of samples and $\alpha > 0$ is a predefined scaling parameter. Equation (2) ties the radius to the local scale of the data. The distance to the k-th nearest neighbor measures the typical spacing between observations so that ϵ is expressed as a fraction α of this spacing and does not depend on the units of the ratios or the dimension of the space. The median makes this scale insensitive to the few firms with extreme ratios, and $k = \max(3, \min(20, \lfloor\sqrt{n}\rfloor))$ follows the usual $\sqrt{n}$ rule of nearest neighbor methods, bounded so that k remains small on large samples. Since ϵ is determined by α and the calibration parameter ρ introduced below, its sensitivity is measured through these two parameters in Section 4.7.

We write $\epsilon_0 = \text{median}(\{d_i^{(k)}\})$ so that $\epsilon = \alpha\,\epsilon_0$. The use of the median makes ϵ_0 robust to outliers. If ϵ is too small, every neighborhood contains only its center, the boundary region is empty and RQS-SVM reduces to a standard SVM. We therefore apply a calibration step: while fewer than $\rho = 5\%$ of the training points belong to a boundary region, ϵ is multiplied by 1.4, up to at most $8\,\epsilon_0$. Since a point belongs to a boundary region exactly when its nearest neighbor from another class lies within distance ϵ, the proportion of boundary points is computed from the nearest enemy distances $\delta_i = \min_{j:\,y_j \neq y_i} \|\tilde{\mathbf{x}}_i - \tilde{\mathbf{x}}_j\|_2$ without recomputing the neighborhoods. The value of α (default 0.2) is set in all the experiments for the single RQS-SVM as for the members of RQS-Ensemble by the data-driven rule of Section 3.8, computed on the training part of each fold. This definition, including the calibration, is used identically in the formulation in Algorithm 2 and the code.

Algorithm 1 RQS-SVM: Complete Pipeline

Require: Training data $\mathcal{D}_{\text{train}} = \{(\mathbf{x}_i, y_i)\}_{i=1}^{n}$, test data $\mathcal{D}_{\text{test}}$
Ensure: Predictions $\hat{\mathbf{y}}_{\text{test}}$

// Procedure 1: Preprocessing
$\tilde{\mathcal{D}}_{\text{train}}, \boldsymbol{\mu}, \boldsymbol{\sigma} \leftarrow \text{Preprocess}(\mathcal{D}_{\text{train}})$ ▷ Equation (1)
// Procedure 2: Rough Set Analysis
$\{\underline{C}_c\}_{c\in\mathcal{Y}}, \{BN_c\}_{c\in\mathcal{Y}}, \epsilon \leftarrow \text{RoughSet}(\tilde{\mathcal{D}}_{\text{train}}, \alpha)$ ▷ Algorithm 2
// Procedure 3: Boundary Compression via PCM
for each class $c \in \mathcal{Y}$ **do**
  if $|BN_c| > \tau$ **then**
    $\{v_{cj}\}_{j=1}^{K_c}, \{U_{cj}\} \leftarrow \text{PCM}(BN_c, m)$
    $\mathcal{B}_c \leftarrow \{v_{cj}\}_{j=1}^{K_c}$ ▷ Cluster representatives ▷ Algorithm 3
  else
    $\mathcal{B}_c \leftarrow BN_c$ ▷ No compression
  end if
end for
// Procedure 4: Quotient Space Construction
$\mathcal{Q}, \mathbf{y}_{\mathcal{Q}}, \mathbf{w} \leftarrow \text{QuotientSpace}(\{\underline{C}_c\}, \{BN_c\}, \beta, m)$ ▷ Algorithm 4
// Procedure 5: Weight clipping and class mass balancing
if use_weights **then**
  $\mathbf{w} \leftarrow \text{ClipAndBalance}(\mathbf{w}, \mathbf{y}_{\mathcal{Q}}, \lambda)$ ▷ Algorithm 5
end if
// Procedure 6: Weighted SVM Training
$\mathbf{SV}, \mathbf{a}, b \leftarrow \text{WeightedSVM}(\mathcal{Q}, \mathbf{y}_{\mathcal{Q}}, \mathbf{w}, C, K)$ ▷ Section 3.6
// Procedure 7: Inference
for each test sample $\mathbf{x}_{\text{test}} \in \mathcal{D}_{\text{test}}$ **do**
  $\hat{y} \leftarrow \text{Inference}(\mathbf{x}_{\text{test}}, \mathcal{Q}, \boldsymbol{\mu}, \boldsymbol{\sigma}, \mathbf{SV}, \mathbf{a}, b, K)$ ▷ Algorithm 6
  $\hat{\mathbf{y}}_{\text{test}} \leftarrow \hat{\mathbf{y}}_{\text{test}} \cup \{\hat{y}\}$
end for
  return $\hat{\mathbf{y}}_{\text{test}}$

For each instance $\mathbf{x}_i$, we compute its ϵ-neighborhood:

$$N_\epsilon(\mathbf{x}_i) = \{\mathbf{x}_j \in U : \|\tilde{\mathbf{x}}_i - \tilde{\mathbf{x}}_j\|_2 \leq \epsilon\} \tag{3}$$

The classification of each instance is then:

$$\text{Region}(\mathbf{x}_i) = \begin{cases} \underline{C}_{y_i}, & \text{if } \forall \mathbf{x}_j \in N_\epsilon(\mathbf{x}_i), y_j = y_i \\ BN_{y_i}, & \text{otherwise} \end{cases} \tag{4}$$

Proposition 1 (Properties of Rough Set Regions)**.** *The lower approximation $\underline{C}$ and boundary region $BN(C)$ satisfy:*

1. $\underline{C} \subseteq C \subseteq \underline{C} \cup BN(C)$;
2. $\underline{C} \cap BN(C) = \emptyset$;
3. *Points in $\underline{C}$ have homogeneous neighborhoods, indicating high confidence;*
4. *Points in $BN(C)$ lie near decision boundaries, indicating uncertainty.*

For each class c, two regions are defined:

- **Lower approximation**:

$$\underline{R}(c) = \{x_i \mid N_\epsilon(x_i) \subseteq c\} \tag{5}$$

- **Boundary region**:

$$BND(c) = \{x_i \mid N_\epsilon(x_i) \cap c \neq \emptyset \text{ and } N_\epsilon(x_i) \cap \neg c \neq \emptyset\} \tag{6}$$

A sample is assigned to the lower approximation if all its neighbors belong to the same class. Otherwise, it is assigned to the boundary region.

Algorithm 2 Rough Set Analysis

Require: Standardized data $\tilde{\mathcal{D}} = \{(\tilde{\mathbf{x}}_i, y_i)\}_{i=1}^n$, parameter α
Ensure: Lower approximations $\{\underline{C}_c\}_{c\in\mathcal{Y}}$ and boundary regions $\{BN_c\}_{c\in\mathcal{Y}}$
$n \leftarrow |\tilde{\mathcal{D}}|$
$k \leftarrow \max(3, \min(20, \lfloor\sqrt{n}\rfloor))$
Compute k-NN distances: $\{d_i^{(k)}\}_{i=1}^n \leftarrow$ knn_distances(k)
$\epsilon_0 \leftarrow \text{median}(\{d_i^{(k)}\}_{i=1}^n)$; $\quad \epsilon \leftarrow \alpha\,\epsilon_0$
Compute nearest enemy distances δ_i
while $\frac{1}{n}|\{i : \delta_i \leq \epsilon\}| < \rho$ **and** $\epsilon < 8\epsilon_0$ **do**
    $\epsilon \leftarrow 1.4\,\epsilon$ ▷ calibration, $\rho = 0.05$
end while
for each $i \in \{1, \ldots, n\}$ **do**
    $N_\epsilon(\tilde{\mathbf{x}}_i) \leftarrow$ radius_query$(\tilde{\mathbf{x}}_i, \epsilon)$
    if $\forall \tilde{\mathbf{x}}_j \in N_\epsilon(\tilde{\mathbf{x}}_i) : y_j = y_i$ **then**
        $\underline{C}_{y_i} \leftarrow \underline{C}_{y_i} \cup \{\tilde{\mathbf{x}}_i\}$
    else
        $BN_{y_i} \leftarrow BN_{y_i} \cup \{\tilde{\mathbf{x}}_i\}$
    end if
end forreturn $\{\underline{C}_c\}_{c\in\mathcal{Y}}, \{BN_c\}_{c\in\mathcal{Y}}, \epsilon$

This formulation captures uncertainty in class membership, which is essential in imbalanced and noisy classification problems.
The proposed implementation provides several key advantages:

- **Scalability**: The k nearest neighbor distances and the nearest enemy distances are computed by an exact search. Tree-based indexes have an average cost of $O(n \log n)$ only in low dimension; for $d = 18$ to 95 features, an exact parallel brute-force search was faster in our experiments with a cost of $O(n^2 d)$ operations that is split across processor cores.
- **Efficiency**: Mini-batch sampling limits the cost of inter-class distance estimation.
- **Vectorization**: Distance computations are vectorized using NumPy (version 2.5.3), eliminating costly Python (version 3.14.4) loops.
- **Robustness**: The use of median-based estimation and sampling mitigates the influence of outliers.

Overall, the proposed approach forms an efficient pipeline for:

1. Adaptive estimation of neighborhood radius ϵ;
2. Approximation of inter-class separability;
3. Construction of rough set regions (lower and boundary).

3.3. Procedure 3: Robust Possibilistic C-Means Clustering

This section presents an enhanced implementation of the possibilistic C-means (PCM) clustering algorithm, designed to improve robustness, convergence stability, and clustering quality. The proposed method integrates multiple initialization strategies, adaptive stopping criteria, and several validity indices for model selection.

The use of PCM at this specific stage is an essential aspect of the proposed integration. PCM is not applied as a generic clustering preprocessing step to the entire dataset. Instead, it is selectively applied to the rough boundary regions identified by the preceding RST analysis. This creates a direct methodological dependency between the rough set representation and the clustering stage: the uncertain samples identified by RST become the input to PCM, while the resulting prototypes subsequently constitute representative granules in the quotient space. Consequently, PCM serves as a bridge between uncertainty identification and granular compression rather than functioning as an independent preprocessing technique.

3.3.1. Theoretical Motivation for Boundary Region Compression

The integration of quotient space theory (QST) with rough set theory (RST) is particularly appropriate for handling the ambiguous regions identified by rough approximations. In the proposed framework, the boundary region BN_c contains observations for which the local neighborhood is not homogeneous with respect to the decision classes. These observations therefore provide important information about class overlap and the location of the decision frontier, but they may simultaneously contain redundant, noisy, or weakly informative observations. Treating all boundary observations as independent training instances may consequently result in an unnecessarily complex representation of the decision space. From the point of view of QST, the boundary region of class c is partitioned into K_c granules, which form the quotient set

$$BN_c/\sim = \{[x_1], [x_2], \ldots, [x_{K_c}]\}, \tag{7}$$

where each equivalence class $[x_j]$ represents a group of sufficiently similar observations. In the proposed implementation, PCM provides a data-driven mechanism for constructing these granules, while the PCM centroids serve as representative elements of the corresponding equivalence classes.

This relationship is particularly relevant to imbalanced bankruptcy prediction. The minority bankruptcy class generally contains substantially fewer observations than the healthy class, and many bankruptcy observations may occur in regions of strong overlap with non-bankrupt firms. Consequently, the minority boundary region contains observations that are potentially highly informative for learning the decision frontier. Compressing this region indiscriminately or removing it completely could eliminate precisely the information that is required to recognize difficult bankruptcy cases. The proposed RST–PCM–QST integration therefore does not discard the boundary region. Instead, it transforms it into a compact set of representative prototypes while retaining the class identity and the local geometric structure of the original observations.

Let B_c denote the boundary observations associated with class c, and let $\mathcal{P}_c = \{v_{c1}, \ldots, v_{cK_c}\}$ denote their PCM-derived prototypes. The compression can be represented as

$$B_c \longrightarrow \mathcal{P}_c, \qquad K_c \ll |B_c|. \tag{8}$$

The objective is not merely to reduce the number of observations but to obtain a lower-dimensional representation that preserves the dominant local structure of the uncertain region. Since the prototypes are generated directly from the observations belonging to B_c, each prototype remains associated with class c. Thus, the compression is class-aware rather than class-agnostic. Moreover, each prototype v_{cj} receives the weight

$$w_{cj} = \beta \frac{|C_{cj}|}{|B_c|} \bar{u}_{cj}, \tag{9}$$

where C_{cj} is the set of boundary observations assigned to v_{cj} and $\bar{u}_{cj}$ their mean typicality. This weight preserves information concerning both the size and the coherence of the represented boundary region. This information reaches the SVM only if the weights are not clipped: with the IQR clipping used in all our experiments, the prototype weights are reset to the common value of their class (Section 3.5) so that the granules are represented by their position in $\mathcal{Q}$, not by their size or typicality. From a classification perspective, this construction provides a direct connection between boundary region compression and minority class recognition. The lower approximation preserves highly certain observations, whereas the quotient representation preserves the principal structures of the uncertain region. The resulting representation therefore contains both reliable class-specific information and representative information about class overlap. For bankruptcy prediction, this is important because difficult bankrupt firms are expected to occur disproportionately near regions where the characteristics of bankrupt and healthy firms overlap. Preserving these regions through representative prototypes allows the subsequent weighted SVM to retain access to relevant information to the minority decision boundary while avoiding excessive redundancy.

PCM formulation. Let $X = \{x_1, \ldots, x_n\}$ be the boundary observations of one class and $K_c = \max(2, \min(k_{\max}, \lfloor n/2 \rfloor))$ the number of granules with $k_{\max} = 5$. Unlike fuzzy c-means, PCM does not impose $\sum_j u_{ij} = 1$ so that outliers receive low typicality in every cluster. The centroids, scale parameters and typicalities are updated alternately,

$$v_j = \frac{\sum_i u_{ij}^m x_i}{\sum_i u_{ij}^m}, \qquad \eta_j = \frac{\sum_i u_{ij}^m \|x_i - v_j\|^2}{\sum_i u_{ij}^m}, \qquad u_{ij} = \left[1 + \left(\|x_i - v_j\|^2/\eta_j\right)^{1/(m-1)}\right]^{-1}, \quad (10)$$

until the change in U falls below a tolerance or the objective $J = \sum_{i,j} u_{ij}^m \|x_i - v_j\|^2$ stagnates during three iterations. The centroids are initialized by k-means++ seeding; the algorithm is run $n_{\text{init}} = 3$ times and the run with the highest partition coefficient $\text{PC} = \frac{1}{n}\sum_{i,j} u_{ij}^2$ is retained (the partition entropy and the Xie–Beni index are also reported). Algorithm 3 summarizes the procedure.

3.3.2. Role of Possibilistic C-Means in Rough Quotient Learning

The quotient representation is formed by all observations of the lower approximations and the PCM prototypes of the boundary regions,

$$\mathcal{Q} = \bigcup_c (\underline{C}_c \cup \{v_{c1}, \ldots, v_{cK_c}\}), \quad (11)$$

each prototype carrying the weight of Equation (9). PCM thus acts as a granulation operator: it replaces the uncertain observations with a few representative granules. The following results quantify the effect of this replacement on the decision.

Theorem 1 (Classification stability and Bayes consistency under quotient compression)**.** *Let $X \in \mathbb{R}^d$ be a random observation, and let $Q : \mathbb{R}^d \to \mathcal{V}$ be the quotient map used at inference, which assigns each observation to its representative in the quotient space (for an observation in the boundary region B_c, the PCM prototype $v_{cj^*(x)}$). Define the quantization distortion*

$$D_Q = \mathbb{E}\left[\|X - Q(X)\|^2\right].$$

Let $f : \mathbb{R}^d \to \mathbb{R}$ be an L-Lipschitz decision function, $|f(x) - f(x')| \le L\|x - x'\|$, and define $h(x) = \operatorname{sign}(f(x))$ and $h_Q(x) = \operatorname{sign}(f(Q(x)))$. Assume the margin condition: there exist $c_0 > 0$ and $\nu > 0$ such that

$$\mathbb{P}\big(|f(X)| \le t\big) \le c_0\, t^{\nu} \qquad \text{for all } t > 0.$$

Then:

(i) (Decision stability)

$$\mathbb{P}\big(h(X) \ne h_Q(X)\big) \;\le\; \inf_{t>0}\Big\{c_0 t^{\nu} + \frac{L^2 D_Q}{t^2}\Big\} = C_1\, D_Q^{\frac{\nu}{\nu+2}}, \qquad C_1 = \frac{(\nu+2)\,c_0}{2}\Big(\frac{2L^2}{\nu c_0}\Big)^{\frac{\nu}{\nu+2}}.$$

(ii) (Preservation of the Bayes decision boundary) If $f = 2\eta - 1$, where $\eta(x) = \mathbb{P}(Y = 1 \mid X = x)$ so that $h = h^*$ is the Bayes classifier, then the excess risk of the compressed classifier satisfies

$$R(h_Q) - R(h^*) \;\le\; C_2\, D_Q^{\frac{\nu+1}{\nu+2}}, \qquad C_2 = (\nu+2)\,c_0\Big(\frac{L^2}{(\nu+1)c_0}\Big)^{\frac{\nu+1}{\nu+2}}.$$

In particular, $h_Q \to h^*$ in excess risk whenever $D_Q \to 0$.

Proof. *(i)* If $h(X) \ne h_Q(X)$, then $f(X)$ and $f(Q(X))$ do not have the same strict sign, so $|f(X)| \le |f(X) - f(Q(X))| \le L\|X - Q(X)\|$. Therefore, for any $t > 0$,

$$\{h(X) \ne h_Q(X)\} \subseteq \{|f(X)| \le t\} \cup \{L\|X - Q(X)\| > t\}.$$

The margin condition bounds the probability of the first event by $c_0 t^{\nu}$, and Markov's inequality applied to $\|X - Q(X)\|^2$ bounds the second by $L^2 D_Q / t^2$. Minimizing $g(t) = c_0 t^{\nu} + L^2 D_Q t^{-2}$ gives $t^* = \big(2L^2 D_Q/(\nu c_0)\big)^{1/(\nu+2)}$, at which $L^2 D_Q / t^{*2} = \frac{\nu c_0}{2} t^{*\nu}$ so that $g(t^*) = \frac{(\nu+2)c_0}{2} t^{*\nu} = C_1 D_Q^{\nu/(\nu+2)}$.

(ii) For any classifier g, $R(g) - R(h^*) = \mathbb{E}\big[|2\eta(X) - 1| \mathbb{1}\{g(X) \ne h^*(X)\}\big]$ [60]. On the disagreement event, $|f(X)| \le L\|X - Q(X)\|$, as shown in (i). Splitting according to whether $|f(X)| \le t$,

$$R(h_Q) - R(h^*) \le t\,\mathbb{P}(|f(X)| \le t) + \mathbb{E}\big[L\|X - Q(X)\| \mathbb{1}\{L\|X - Q(X)\| > t\}\big] \le c_0 t^{\nu+1} + \frac{L^2 D_Q}{t},$$

where the last step uses $u \mathbb{1}\{u > t\} \le u^2/t$. Minimizing over t gives $t^* = \big(L^2 D_Q/((\nu+1)c_0)\big)^{1/(\nu+2)}$ and the stated bound. □

Corollary 1 (Dependence on the compression level)**.** *Write $D_Q = \mathbb{P}(X \notin B)\, D_{\text{low}} + \mathbb{P}(X \in B)\, D_B$, where $B = \bigcup_c B_c$, D_{low} is the nearest representative distortion in the lower approximations (whose points are retained without compression), and D_B is the distortion of the $K_B = \sum_c K_c$ PCM prototypes on the boundary region. If the prototypes achieve the quantization rate $D_B \le \kappa K_B^{-2/d_B}$, where d_B is the intrinsic dimension of B (the rate of optimal K_B-point quantizers on a compact support, [61]), then*

$$\mathbb{P}\big(h(X) \ne h_Q(X)\big) \le C_1\Big(\mathbb{P}(X \notin B)\, D_{\text{low}} + \mathbb{P}(X \in B)\,\kappa\, K_B^{-2/d_B}\Big)^{\frac{\nu}{\nu+2}}.$$

With $K_B = \rho_B\,|B|$, where ρ_B is the ratio of the number of prototypes to the number of boundary observations, the bound decreases as $\rho_B^{-2\nu/(d_B(\nu+2))}$.

Remark 1. *(a) The rough partition restricts compression to the boundary region: the term* $\mathbb{P}(X \in B)$ *multiplies the only distortion that depends on* K_B*, so, the smaller the boundary region, the less compression can affect the decision. (b) For an RBF kernel SVM,* $k(x, x') = \exp(-\gamma\|x - x'\|^2)$*, and the decision function is Lipschitz with* $L \leq \sqrt{2\gamma}\,\|f\|_{\mathcal{H}}$*, where* $\|f\|_{\mathcal{H}}^2 = \sum_{i,j} a_i a_j y_i y_j k(\mathbf{q}_i, \mathbf{q}_j)$ *is computed from the dual coefficients. This holds because* $\|\phi(x) - \phi(x')\|^2 = 2 - 2e^{-\gamma\|x-x'\|^2} \leq 2\gamma\|x - x'\|^2$*. (c) The theorem concerns the approximation effect of the quotient map for a given decision function. The estimation error of an SVM trained on the compressed sample is not covered; it is controlled by standard SVM generalization bounds with sample size equal to the size of the quotient space.*

Algorithm 3 Robust Possibilistic C-Means Clustering

Require: Boundary data $X = \{\mathbf{x}_i\}_{i=1}^{n_b}$, parameter m, max clusters $k_{\max} = 5$
Ensure: Centroids $\{\mathbf{v}_j\}_{j=1}^{K_c}$, membership matrix U, cluster assignments

$n \leftarrow |X|$
$K_c \leftarrow \max(2, \min(k_{\max}, \lfloor n/2 \rfloor))$ ▷ Adaptive cluster count
Initialize centroids using lightweight k-means++
Initialize U randomly: $u_{ij} \in (0, 1)$, $\sum_j u_{ij} = 1$
Initialize $\eta_j \leftarrow 1$ for $j = 1, \ldots, K_c$
$prev_obj \leftarrow \infty$, $stagnation \leftarrow 0$
for $iter = 1$ **to** max_iter **do**
    $U_{old} \leftarrow U$
    for $j = 1$ **to** K_c **do**
        $\eta_j \leftarrow \frac{\sum_{i=1}^{n} u_{ij}^m \|\mathbf{x}_i - \mathbf{v}_j\|^2}{\sum_{i=1}^{n} u_{ij}^m}$
    end for
    for $i = 1$ **to** n **do**
        for $j = 1$ **to** K_c **do**
            $u_{ij} \leftarrow \frac{1}{1 + \left(\frac{\|\mathbf{x}_i - \mathbf{v}_j\|^2}{\eta_j}\right)^{\frac{1}{m-1}}}$
        end for
    end for
    for $j = 1$ **to** K_c **do**
        $\mathbf{v}_j \leftarrow \frac{\sum_{i=1}^{n} u_{ij}^m \mathbf{x}_i}{\sum_{i=1}^{n} u_{ij}^m}$
    end for
    $obj \leftarrow \sum_{i=1}^{n} \sum_{j=1}^{K_c} u_{ij}^m \|\mathbf{x}_i - \mathbf{v}_j\|^2$
    if $\|\mathbf{U} - \mathbf{U}_{old}\|_F < tol$ **then**
        break
    end if
    if $|prev_obj - obj| < tol \cdot 0.1$ **then**
        $stagnation \leftarrow stagnation + 1$
        if $stagnation \geq 3$ **then**
            break
        end if
    else
        $stagnation \leftarrow 0$
    end if
    $prev_obj \leftarrow obj$
end for
Compute three clustering validity indices for each initialization: PC, PE and XB
Select the best clustering solution among the three initializations based on the clustering validity indices
labels $\leftarrow \arg\max_j u_{ij}$ **return** $\{\mathbf{v}_j\}_{j=1}^{K_c}$, U, labels

3.3.3. Formal Link Between Rough Sets and Quotient Spaces

Rough set theory and quotient space theory are usually presented separately. The three propositions below make explicit how they are connected in RQS-SVM. Let U be a finite universe, R an equivalence relation on U, U/R the corresponding quotient set and $\pi_R : U \to U/R$, $\pi_R(x) = [x]_R$ the quotient map. In the quotient space theory of Zhang and Zhang [59], a problem (U, f, T) is studied at the coarser granularity $([U], [f], [T])$ induced by π_R, and a subset $A \subseteq U$ is said to be *saturated* (or R-definable) if $A = \pi_R^{-1}(\pi_R(A))$.

Proposition 2 (Pawlak approximations are quotient operators)**.** *For every $X \subseteq U$, the Pawlak lower and upper approximations [62] satisfy*

$$\underline{R}(X) = \pi_R^{-1}\big(\{q \in U/R : \ \pi_R^{-1}(q) \subseteq X\}\big), \qquad \overline{R}(X) = \pi_R^{-1}\big(\pi_R(X)\big).$$

Hence $\underline{R}(X)$ is the largest saturated subset of X, $\overline{R}(X)$ is the smallest saturated superset of X, and the boundary $BN_R(X) = \overline{R}(X) \setminus \underline{R}(X)$ is the union of the quotient elements that meet both X and $U \setminus X$.

Proof. By definition, $\underline{R}(X) = \bigcup\{[x]_R : [x]_R \subseteq X\}$, which is the preimage of the set of quotient elements contained in X. Likewise $\overline{R}(X) = \bigcup\{[x]_R : [x]_R \cap X \neq \emptyset\} = \pi_R^{-1}(\pi_R(X))$. A saturated set is a union of quotient elements; a union of quotient elements is contained in X if and only if each of them is, which gives maximality of $\underline{R}(X)$, and it contains X if and only if it contains every quotient element meeting X, which gives minimality of $\overline{R}(X)$. □

Proposition 2 shows that rough approximation is approximation *in a quotient space*: a concept is described through the elements of U/R. The neighborhood rough sets used in Procedure 2 replace R with the tolerance relation $\|x - x'\| \leq \epsilon$, which is not transitive and therefore does not define a quotient set directly. RQS-SVM restores a genuine equivalence relation on the training sample $S = \{(\mathbf{x}_i, y_i)\}_{i=1}^n$ as follows.

Proposition 3 (The RQS equivalence relation)**.** *Let $j^*(i)$ be the index of the prototype to which a boundary observation $\mathbf{x}_i \in BN_{y_i}$ is assigned. Define $\mathbf{x}_i \sim \mathbf{x}_k$ if and only if either $i = k$ or $\mathbf{x}_i, \mathbf{x}_k \in BN_c$ for the same class c and $j^*(i) = j^*(k)$. Then:*

(a) $\sim$ is an equivalence relation; its classes are the singletons $\{\mathbf{x}_i\}$ of the lower approximations and the granules C_{cj} of the boundary regions, and the map Q that sends an observation to its representative (itself or v_{cj}) is the quotient map $\pi_\sim$ followed by the choice of a representative;

(b) Every decision class $D_c = \{\mathbf{x}_i : y_i = c\}$ is $\sim$-saturated so that $\underline{\sim}(D_c) = \overline{\sim}(D_c) = D_c$: the quotient representation is consistent with the labels;

(c) $\sim$ is finer than the rough partition $\{\underline{C}_c, BN_c\}_c$: every $\sim$-class is contained in a single lower approximation or a single boundary region.

Proof. (a) Reflexivity and symmetry are immediate; transitivity follows from the transitivity of equality of the pairs (c, j^*). The listed sets are disjoint and cover S, and each is a $\sim$-class. (b) and (c) follow because a $\sim$-class is either a singleton or consists of boundary observations of one class assigned to one prototype. □

By Proposition 3, the passage from S to the quotient representation $\mathcal{Q}$ is a change in granularity in the sense of QST that never merges observations of different classes or different rough regions. The cost of this change in granularity on the training criterion is controlled by the within-granule distortion.

Proposition 4 (Empirical risk approximation)**.** *Let f be an L-Lipschitz decision function and ℓ a loss that is L_ℓ-Lipschitz in the decision value (the hinge loss has $L_\ell = 1$). Let $\widehat{R}_S(f) = \frac{1}{n}\sum_{i=1}^{n} \ell(y_i f(\mathbf{x}_i))$ and let $\widehat{R}_{\mathcal{Q}}(f) = \frac{1}{n}\left[\sum_{\mathbf{x}_i \notin B} \ell(y_i f(\mathbf{x}_i)) + \sum_{c,j} |C_{cj}|\, \ell(c\, f(v_{cj}))\right]$ be the risk on the quotient representation in which each prototype is weighted by the size of its granule, where $B = \bigcup_c BN_c$. Then*

$$\left|\widehat{R}_S(f) - \widehat{R}_{\mathcal{Q}}(f)\right| \;\le\; L_\ell\, L\, \frac{|B|}{n}\, \widehat{D}_B^{1/2}, \qquad \widehat{D}_B = \frac{1}{|B|} \sum_{\mathbf{x}_i \in B} \|\mathbf{x}_i - Q(\mathbf{x}_i)\|^2.$$

Proof. The terms of the lower approximations cancel. Each boundary observation $\mathbf{x}_i \in C_{cj}$ appears in $\widehat{R}_S$ through $\ell(cf(\mathbf{x}_i))$ and is counted in $\widehat{R}_{\mathcal{Q}}$ through $\ell(cf(v_{cj}))$; their difference is at most $L_\ell L \|\mathbf{x}_i - v_{cj}\|$. Summing over B and applying the Cauchy–Schwarz inequality, $\sum_B \|\mathbf{x}_i - Q(\mathbf{x}_i)\| \le |B|\, \widehat{D}_B^{1/2}$, gives the result. □

Proposition 4 is the empirical counterpart of Theorem 1: the rough partition limits the error to the fraction $|B|/n$ of boundary observations, and the granulation controls it through $\widehat{D}_B$. The weights actually used by RQS-SVM (Equation (9) and Section 3.5) deliberately depart from the granule sizes in order to correct the class imbalance; the proposition therefore describes the compression error before this cost-sensitive reweighting.

The computational benefit of the proposed RQS-SVM framework is primarily associated with the reduction in the effective training representation. PCM is selectively applied to the rough boundary regions and replaces redundant boundary observations with a smaller number of representative prototypes. Consequently, the final quotient representation contains the complete lower approximations and only K_c representatives for each compressed boundary region, where $K_c \ll |B_c|$. If n denotes the original number of training observations and $n_{\mathcal{Q}} = |\mathcal{Q}|$ denotes the size of the resulting quotient representation, the reduction in the SVM training set is quantified by $1 - n_{\mathcal{Q}}/n$.

The computational cost of the complete framework should nevertheless be distinguished from the cost of the final SVM optimization. The construction of the representation introduces additional costs associated with neighborhood computation and PCM clustering. For neighborhood construction, the exact nearest neighbor and nearest enemy searches cost $O(n^2 d)$ operations, which are parallelized across processor cores (Section 4.6). PCM is applied only to the boundary observations rather than applied to the complete dataset, and its computational cost depends on the boundary size, number of prototypes, number of iterations, and feature dimensionality. After representation construction, however, the weighted SVM is trained on m representatives instead of n original observations. Therefore, the proposed framework is expected to provide its principal computational advantage when the boundary region contains substantial redundancy and $m \ll n$. This computational effect is evaluated empirically in terms of training set reduction, training time, memory consumption, and number of support vectors.

3.4. Procedure 4: Quotient Space Construction

The construction of the quotient space constitutes a fundamental step in the proposed framework, aiming to reduce data complexity while preserving the essential structural and class-discriminative information. This process relies on the principles of granular computing and rough set theory, where the universe of discourse is partitioned into lower approximations and boundary regions for each class.

Let $\{\underline{C}_c\}_{c\in\mathcal{Y}}$ denote the set of lower approximations and $\{BN_c\}_{c\in\mathcal{Y}}$ the corresponding boundary regions for the classes $c \in \mathcal{Y}$. The algorithm generates a quotient space $\mathcal{Q}$ along with its associated label vector $\mathbf{y}_{\mathcal{Q}}$ and weight vector $\mathbf{w}$.

Initially, the quotient space $\mathcal{Q}$ is set to the empty set. For each class c, all elements belonging to the lower approximation $\underline{C}_c$ are directly transferred into $\mathcal{Q}$. These elements represent the most reliable samples as they can be unambiguously assigned to class c. Consequently, they are assigned the maximum weight equal to 1.0, reflecting their high certainty and importance in the learning process.

The treatment of boundary region samples BN_c depends on the selected configuration. When the flag `use_pcm` is activated and the cardinality of the boundary region is sufficiently large (i.e., $|BN_c| > \tau$ with $\tau = 3$), a clustering-based compression strategy is applied using the possibilistic C-means (PCM) algorithm. In this case, the boundary samples are grouped into K_c clusters, each represented by a prototype $\mathbf{v}_j$. These prototypes are then incorporated into the quotient space $\mathcal{Q}$ as representative elements of the boundary region. The threshold τ is the minimal size of a boundary region for PCM to be applied, and its default value $\tau = 3$ is the smallest size for which PCM can reduce the region: since PCM uses at least $K_c = 2$ granules, a region of at most three observations would be replaced by two or three prototypes, which saves at most one point, while the scale parameters η_j would be estimated from one or two observations. Such regions are therefore kept as they are. In our experiments, the threshold is never active: in all 15 folds, both classes have more than three boundary observations and PCM is applied to both (the per-fold numbers of PCM prototypes are given in the Zenodo repository). Any value of τ smaller than the smallest boundary region therefore gives the same model, and the opposite extreme, in which no boundary region is compressed, is the configuration "raw boundary (no PCM)" of the ablation study (Section 4.3).

Each prototype $\mathbf{v}_j$ is assigned the label c, and its associated weight is computed based on three factors: (i) the relative size of the cluster, (ii) the average membership degree $\bar{u}_j$ of samples within the cluster, and (iii) a scaling parameter $\beta \in (0, 1)$. Formally, the weight assigned to prototype $\mathbf{v}_j$ is defined as

$$w_j = \beta \cdot \frac{|\text{cluster}_j|}{|BN_c|} \cdot \bar{u}_j,$$

where $\bar{u}_j = \frac{1}{|\text{cluster}_j|} \sum_{i \in \text{cluster}_j} u_{ij}$. This formulation ensures that more representative and denser clusters receive higher importance.

Alternatively, if the PCM-based compression is not activated, each boundary sample $\mathbf{x} \in BN_c$ is directly included in the quotient space with a uniform weight equal to β, reflecting its uncertain nature compared to lower approximation samples.

After processing all classes, Algorithm 4 returns the initial weights $\mathbf{w}$. If the flag `use_weights` is enabled, these weights are then adjusted by Procedure 5 (clipping and class mass balancing, function ClipAndBalance$(\cdot)$ of Algorithm 5), which is a separate step of Algorithm 1.

Finally, the algorithm returns the constructed quotient space $\mathcal{Q}$, the corresponding label vector $\mathbf{y}_\mathcal{Q}$, and the initial weight vector $\mathbf{w}$. This reduced representation serves as an efficient and informative input for subsequent learning models, particularly in imbalanced classification scenarios.

Algorithm 4 Quotient Space Construction

Require: Lower approximations $\{\underline{C}_c\}_{c\in\mathcal{Y}}$, boundary regions $\{BN_c\}_{c\in\mathcal{Y}}$, parameters β, m, flag use_pcm
Ensure: Quotient space $\mathcal{Q}$, labels $\mathbf{y}_\mathcal{Q}$, initial weights $\mathbf{w}$

$\mathcal{Q} \leftarrow \emptyset$, $\mathbf{y}_\mathcal{Q} \leftarrow []$, $\mathbf{w} \leftarrow []$
for each class $c \in \mathcal{Y}$ **do**
  for each $\mathbf{x} \in \underline{C}_c$ **do**
    $\mathcal{Q} \leftarrow \mathcal{Q} \cup \{\mathbf{x}\}$
    $\mathbf{y}_\mathcal{Q} \leftarrow \mathbf{y}_\mathcal{Q} \cup \{c\}$
    $\mathbf{w} \leftarrow \mathbf{w} \cup \{1.0\}$ ▷ Maximum weight for certain points
  end for
  if use_pcm **and** $|BN_c| > \tau$ **then**
    $\{\mathbf{v}_j\}_{j=1}^{K_c}, U, \text{labels} \leftarrow \text{PCM}(BN_c, m)$ ▷ Algorithm 3
    for each cluster j **do**
      $\mathcal{Q} \leftarrow \mathcal{Q} \cup \{\mathbf{v}_j\}$
      $\mathbf{y}_\mathcal{Q} \leftarrow \mathbf{y}_\mathcal{Q} \cup \{c\}$
      $\bar{u}_j \leftarrow \frac{1}{|\text{cluster}_j|} \sum_{i\in\text{cluster}_j} u_{ij}$
      $w \leftarrow \beta \cdot \frac{|\text{cluster}_j|}{|BN_c|} \cdot \bar{u}_j$
      $\mathbf{w} \leftarrow \mathbf{w} \cup \{w\}$
    end for
  else
    for each $\mathbf{x} \in BN_c$ **do**
      $\mathcal{Q} \leftarrow \mathcal{Q} \cup \{\mathbf{x}\}$
      $\mathbf{y}_\mathcal{Q} \leftarrow \mathbf{y}_\mathcal{Q} \cup \{c\}$
      $\mathbf{w} \leftarrow \mathbf{w} \cup \{\beta\}$
    end for
  end if
end for
  return $\mathcal{Q}$, $\mathbf{y}_\mathcal{Q}$, $\mathbf{w}$

3.5. Procedure 5: Weight Clipping and Class Mass Balancing

The weighting procedure was designed to refine the importance assigned to samples in the quotient space by addressing two issues: the presence of outliers in the weight distribution and the imbalance between classes. As shown below, the clipping step removes all within-class differences on our data so that only the class balance remains in the final weights. Let $\mathbf{w}$ denote the initial weight vector associated with the quotient space samples, and let $\mathbf{y}_\mathcal{Q}$ be the corresponding label vector. The procedure begins by applying a statistical normalization technique based on the interquartile range (IQR) to mitigate the influence of extreme weight values. Specifically, the first and third quartiles, denoted by Q_1 and Q_3, are computed as the 25th and 75th percentiles of the weight distribution, respectively. The interquartile range is then defined as:

$$IQR = Q_3 - Q_1.$$

To reduce the impact of outliers, the weights are clipped within the interval:

$$[Q_1 - 1.5 \cdot IQR,\ Q_3 + 1.5 \cdot IQR].$$

This operation ensures that excessively large or small weights do not dominate the learning process, thereby improving numerical stability and generalization performance. Following this normalization step, the algorithm addresses class imbalance by identifying the minority class. Let $c_{\min}$ denote the class with the smallest number of samples in $\mathbf{y}_\mathcal{Q}$. To compensate for this imbalance, the weights of all samples belonging to the minority class are amplified

using a boosting factor controlled by the parameter $\lambda > 0$. Formally, the updated weight for each sample i such that $\mathbf{y}_{\mathcal{Q}}[i] = c_{\min}$ is given by:

$$w_i \leftarrow w_i \cdot (1 + \lambda).$$

This operation increases the contribution of minority class samples during model training, thereby improving classification performance in imbalanced scenarios. Finally, a normalization step is applied to ensure that all weights lie within a standardized range. The weight vector is scaled by its maximum value as follows:

$$\mathbf{w}_{\text{final}} = \frac{\mathbf{w}}{\max(\mathbf{w})}.$$

This normalization guarantees that the final weights are bounded within the interval $[0, 1]$, facilitating stable optimization in subsequent learning algorithms. The resulting vector $\mathbf{w}_{\text{final}}$ reflects a balanced and robust weighting scheme that accounts for both statistical dispersion and class imbalance, making it particularly suitable for imbalanced classification tasks.

Class mass balancing. The scheme above does not control the *total* weight of each class. In bankruptcy data the minority class has a few dozen to a few hundred observations, whereas the majority lower approximation contains thousands of observations of weight 1; amplifying each minority weight by $(1 + \lambda)$ with $\lambda = 0.5$ leaves the minority class with a few percent of the total weight. Moreover, when more than 75% of the weights are equal to 1 (which is the case when the lower approximation dominates), $Q_1 = Q_3 = 1$, the interquartile range is zero and the clipping interval reduces to $\{1\}$ so that the prototype weights of Equation (9) are lost. The ablation study confirms that the model trained with these weights alone predicts almost only the majority class (Section 4.3). The procedure therefore adds a class mass balancing step after the minority amplification,

$$w_i \leftarrow w_i \frac{\sum_{k:\, y_k \neq c_{\min}} w_k}{\sum_{k:\, y_k = c_{\min}} w_k} \qquad \text{for all } i \text{ with } y_i = c_{\min}, \tag{12}$$

so that both classes have the same total weight in the weighted SVM problem. This step multiplies all the weights of a class by the same factor, so it keeps whatever within-class differences remain after the clipping; in our data none remain because the IQR clipping has already made the weights constant within each class (see the weight diagnostics below and Section 4.3). In the full model the weighting therefore reduces to class mass balancing: every element of $\mathcal{Q}$ of a given class has the same weight, and the typicality and size of the granules are not passed on to the SVM. The weights are then divided by their maximum as before.

Remark 2 (Redundancy of λ)**.** *The balancing step* (12) *is invariant to any rescaling of the minority weights. Therefore the factor* $(1 + \lambda)$ *cancels and the final weights, hence the trained SVM, do not depend on* λ*. This is confirmed numerically: the variant with* $\lambda = 0$ *gives exactly the same predictions as the full model in all* 15 *folds (Section 4.3). We keep* λ *in Algorithm 5 only to show the effect of the weighting without balancing; it can be removed.*

Weight diagnostics. Table 3 reports, for each dataset, the composition of the quotient representation and the final weights, computed on the whole dataset (on a stratified sample of 20,000 firm–years for US) with the default parameters of the single model.

Table 3. Composition of the quotient representation and final weights of the single RQS-SVM. Without balancing: IQR clipping and minority amplification only; with balancing: after class mass balancing (Equation (12)). Raw IQR: interquartile range of the initial weights. Within each class, the minimum, median and maximum of the final weights are equal, so a single value is reported.

Dataset	Class	n	Lower Approx.	Boundary	Prototypes	$\lvert\mathcal{Q}\rvert$	Raw IQR	Final Weight/Share of Total Weight	
								Without Balancing	With Balancing
TW	non-failed	6599	5715	884	5	5720	0	0.667/96.8%	0.022/50%
	failed	220	124	96	1	125	0	1.000/3.2%	1.000/50%
US	non-failed	18,673	11,814	6859	2	11,816	0	0.667/91.0%	0.066/50%
	failed	1327	777	550	2	779	0	1.000/9.0%	1.000/50%
UK	non-failed	4908	3901	1007	3	3904	0	0.667/97.5%	0.017/50%
	failed	92	63	29	4	67	0	1.000/2.5%	1.000/50%

Three observations follow. First, the interquartile range of the initial weights is zero on the three datasets because the observations of the lower approximations, all of weight 1, are the large majority of $\mathcal{Q}$. The IQR clipping therefore sets every weight to 1, the weights of Equation (9) (and hence β, the typicality and the granule sizes) have no effect, and without class mass balancing the only correction is the factor $(1+\lambda)$, which leaves the minority class with 2% to 9% of the total weight. This explains why the model without balancing predicts almost only the majority class. Second, with class mass balancing each class has exactly half of the total weight; the final weights are constant within each class, so the single RQS-SVM is, in practice, a class-weighted SVM trained on the quotient representation. Third, the granulation is very strong: on TW, the 96 boundary observations of the failed class are replaced by a single prototype and on UK the 29 boundary failures by four prototypes. Although the total weight of the minority class is restored by the balancing, the geometric information carried by the minority boundary is reduced to a few points, which is consistent with the better results obtained with the raw boundary in the ablation study (Section 4.3). Limiting the compression of the minority boundary or replacing the IQR clipping with a rule that preserves the granule weights are therefore natural improvements of the method.

3.6. Procedure 6: Weighted SVM Training

The final stage of the proposed framework consists of training a weighted support vector machine (SVM) on the constructed quotient space. This step integrates the structural reduction achieved previously with a cost-sensitive learning mechanism, where the contribution of each sample is modulated through the class-balanced weights of Procedure 5. Let $\mathcal{Q} = \{\mathbf{q}_i\}_{i=1}^{n_\mathcal{Q}}$ denote the quotient space, $\mathbf{y}_\mathcal{Q} = \{y_i\}_{i=1}^{n_\mathcal{Q}}$ the associated class labels with $y_i \in \{-1, +1\}$ (or extended to multi-class settings), and $\mathbf{w} = \{w_i\}_{i=1}^{n_\mathcal{Q}}$ the corresponding weight vector. The objective is to learn a decision function of the form

$$f(\mathbf{q}) = \sum_{i=1}^{n_\mathcal{Q}} a_i y_i K(\mathbf{q}_i, \mathbf{q}) + b,$$

where $K(\cdot,\cdot)$ is a kernel function and $\mathbf{a} = \{a_i\}$ are the Lagrange multipliers. The training process is formulated as a weighted dual optimization problem, where the classical SVM constraints are modified to incorporate sample-specific importance. The optimization problem is defined as

$$\max_{\mathbf{a}} \sum_{i=1}^{n_\mathcal{Q}} a_i - \frac{1}{2} \sum_{i=1}^{n_\mathcal{Q}} \sum_{j=1}^{n_\mathcal{Q}} a_i a_j y_i y_j K(\mathbf{q}_i, \mathbf{q}_j),$$

subject to

$$\sum_{i=1}^{n_{\mathcal{Q}}} a_i y_i = 0, \quad 0 \le a_i \le C \cdot w_i,$$

where $C > 0$ is the regularization parameter controlling the trade-off between margin maximization and classification error. The sample-dependent upper bound Cw_i controls the contribution of each observation to the optimization problem. Consequently, larger values of w_i impose a stronger penalty on classification errors for the corresponding observations, whereas smaller values reduce their influence (Algorithm 5). This formulation allows the SVM to incorporate the reliability and importance information encoded by the quotient space representation.

The problem is solved by the sequential minimal optimization algorithm of LIBSVM through the `SVC` class of scikit-learn with per-sample weights; the support vectors are the elements of $\mathcal{Q}$ with $a_i > 0$.

Algorithm 5 Weight Clipping and Class Mass Balancing

Require: Weights $\mathbf{w}$, labels $\mathbf{y}_{\mathcal{Q}}$, minority boost λ
Ensure: Normalized weights $\mathbf{w}_{\text{final}}$
$Q_1 \leftarrow \text{percentile}(\mathbf{w}, 25)$
$Q_3 \leftarrow \text{percentile}(\mathbf{w}, 75)$
$IQR \leftarrow Q_3 - Q_1$
$\mathbf{w} \leftarrow \text{clip}(\mathbf{w}, Q_1 - 1.5 \cdot IQR, Q_3 + 1.5 \cdot IQR)$
$c_{\min} \leftarrow \arg\min_c |\{\mathbf{y}_{\mathcal{Q}} = c\}|$ ▷ Minority class identification
for each i **where** $\mathbf{y}_{\mathcal{Q}}[i] = c_{\min}$ **do**
    $\mathbf{w}[i] \leftarrow \mathbf{w}[i] \cdot (1 + \lambda)$
end for
$s \leftarrow \sum_{i: y_i \neq c_{\min}} \mathbf{w}[i] \;/\; \sum_{i: y_i = c_{\min}} \mathbf{w}[i]$
for each i **where** $\mathbf{y}_{\mathcal{Q}}[i] = c_{\min}$ **do**
    $\mathbf{w}[i] \leftarrow s\,\mathbf{w}[i]$ ▷ Class mass balancing, Equation (12)
end for
$\mathbf{w}_{\text{final}} \leftarrow \mathbf{w} / \max(\mathbf{w})$ ▷ Normalization **return** $\mathbf{w}_{\text{final}}$

3.7. Procedure 7: Inference Projection

The inference phase aims to classify an unseen sample by leveraging the reduced representation provided by the quotient space and the trained weighted SVM model. This process involves three main steps: data standardization, projection onto the quotient space, and decision function evaluation (Algorithm 6).

Importantly, the inference procedure operates on a quotient space that is constructed exclusively from the training data. No test observation or test label is used to construct, modify, or update the quotient space, its representative elements, the associated weights, or the trained SVM model. Thus, the quotient space remains fixed throughout the inference phase.

Let $\mathbf{x}_{\text{test}}$ denote an unseen test sample in the original feature space. Since the SVM model is trained on standardized data, the test sample must first undergo the same transformation. Given the mean vector $\boldsymbol{\mu}$ and standard deviation vector $\boldsymbol{\sigma}$ computed exclusively from the training set, the standardized sample $\tilde{\mathbf{x}}_{\text{test}}$ is obtained as:

$$\tilde{\mathbf{x}}_{\text{test}} = \frac{\mathbf{x}_{\text{test}} - \boldsymbol{\mu}}{\boldsymbol{\sigma}}.$$

The parameters $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}$ are estimated once from the training set and are kept fixed during inference. In particular, the test observations are not used to recompute these parameters. This normalization ensures consistency between training and inference phases, preventing scale-related distortions.

Next, the standardized sample is projected onto the training-derived quotient space

$$\mathcal{Q}_{\text{train}} = \{\mathbf{q}_i\},$$

through a nearest neighbor mapping. Specifically, the representative element $\hat{\mathbf{q}}$ is determined by minimizing the Euclidean distance:

$$\hat{\mathbf{q}} = \arg\min_{\mathbf{q}\in\mathcal{Q}_{\text{train}}} \|\tilde{\mathbf{x}}_{\text{test}} - \mathbf{q}\|_2.$$

This nearest neighbor operation is performed only against representatives that have been generated during the training phase. The test observation is therefore mapped to an already established representative and does not participate in the formation of that representative. In particular, no test observation is included in the rough set analysis, boundary region identification, PCM clustering, quotient space construction, or weighting procedures. Consequently, the projection step does not modify $\mathcal{Q}_{\text{train}}$ and does not introduce information from the test set into the learned representation.

This step reflects the core idea of quotient space theory, where each original sample is associated with its closest granule or representative point, thereby reducing the complexity of the decision process while preserving the underlying structure.

Once the projection is performed, the classification decision is obtained using the trained weighted SVM model. The decision function is evaluated as

$$f(\hat{\mathbf{q}}) = \sum_{i:a_i>0} a_i y_i K(\mathbf{q}_i, \hat{\mathbf{q}}) + b,$$

where $\mathbf{SV} = \{\mathbf{q}_i : a_i > 0\}$ denotes the set of support vectors obtained exclusively from the training-derived quotient representation, $\mathbf{a}$ represents the learned coefficients, b is the bias term, and $K(\cdot,\cdot)$ is the kernel function.

Finally, the predicted class label $\hat{y}$ is determined based on the sign of the decision function:

$$\hat{y} = \text{sign}(f(\hat{\mathbf{q}})).$$

This rule assigns the sample to one of the two classes in the binary setting and can be extended to multi-class scenarios using standard strategies such as one-vs-one or one-vs-rest.

Therefore, the complete inference procedure follows a strict train–test separation: all data-dependent operations are fitted using the training partition, whereas the test partition is used only after the representation and the SVM model have been fixed. The test sample is used solely as an input to the deterministic projection and prediction operations.

Overall, this inference mechanism combines the efficiency of quotient space projection with the robustness of weighted SVM classification, enabling accurate and computationally efficient predictions, particularly in high-dimensional and imbalanced-data contexts.

Projection versus direct evaluation. Because the decision function f is defined on the whole input space, an alternative is to evaluate it directly at $\tilde{\mathbf{x}}_{\text{test}}$. Projection makes the classifier piecewise constant on the cells of the quotient representation: by Theorem 1 it changes the decision only for observations close to the decision boundary, but it also replaces the continuous score $f(\tilde{\mathbf{x}}_{\text{test}})$ with the score of the nearest representative, which reduces the resolution of the ranking (AUC-ROC and PR-AUC). For the lower approximation, whose observations are kept unchanged, the nearest representative of a test observation is a training observation, and the difference between the two rules is small. Both rules are compared below (Table 4) and in the ablation of RQS-Ensemble (Section 4.3). We report the main results with projection, which is the rule fixed before the experiments, and provide direct evaluation as an option in the code. AUC-ROC and PR-AUC are computed from

the continuous value $f(\hat{\mathbf{q}})$ (for RQS-Ensemble, the average of the member values); test observations projected onto the same representative receive tied scores, which are handled by the trapezoidal rule of the ROC curve (ties count one half) and the step-wise definition of average precision. Table 4 compares the two rules for the single RQS-SVM on the same folds.

Table 4. Projection versus direct evaluation for the single RQS-SVM (mean over the five folds). Distinct scores: percentage of distinct score values among the test observations. Projection distance: distance between a test observation and its representative; NN distance: median distance between an element of $\mathcal{Q}$ and its nearest neighbor in $\mathcal{Q}$.

Dataset	Inference	G-Mean	MCC	AUC-ROC	PR-AUC	Distinct Scores (%)	Projection Distance		NN Distance Median
							Median	90th Pct.	
TW	projection	0.772	0.314	0.896	0.305	80.9	3.03	5.15	3.12
	direct	0.832	0.336	0.911	0.328	99.7	–	–	–
US	projection	0.373	0.066	0.622	0.113	33.9	0.027	0.475	0.064
	direct	0.378	0.066	0.634	0.122	99.9	–	–	–
UK	projection	0.631	0.113	0.785	0.075	79.6	1.71	5.76	1.84
	direct	0.703	0.141	0.815	0.107	100.0	–	–	–

The median distance between a test observation and its representative is of the same order as the median distance between neighboring elements of $\mathcal{Q}$, so most test observations are close to a representative. The distribution has a long tail, however: the 90th percentile is about 1.7, 3 and 7 times the median nearest neighbor distance on TW, UK and US, so one test observation in ten is far from any representative. Projection also produces many ties: only 81%, 78% and 34% of the test scores are distinct on TW, UK and US. Direct evaluation improves the G-mean, AUC-ROC and PR-AUC in all 15 fold$\times$dataset blocks (mean gains $+0.045$, $+0.018$ and $+0.022$; Wilcoxon $p < 0.001$ for each metric), mainly through a higher recall ($+0.086$) at the price of a slightly lower specificity (-0.016). For the single model, the projection step is therefore not justified by the results: it loses information. For RQS-Ensemble the gain of direct evaluation is smaller and not significant after correction (Section 4.3) because averaging ten members already reduces the ties.

Algorithm 6 Inference Projection and Prediction

Require: Unseen test sample $\mathbf{x}_{\text{test}}$, training-derived quotient space $\mathcal{Q}_{\text{train}}$, training scaling parameters $\boldsymbol{\mu}, \boldsymbol{\sigma}$, trained SVM model $\mathbf{SV}$, $\mathbf{a}$, b, kernel K
Ensure: Predicted class $\hat{y}$

$\tilde{\mathbf{x}}_{\text{test}} \leftarrow \frac{\mathbf{x}_{\text{test}} - \boldsymbol{\mu}}{\boldsymbol{\sigma}}$ ▷ Standardization using training parameters
$\hat{\mathbf{q}} \leftarrow \arg\min_{\mathbf{q} \in \mathcal{Q}_{\text{train}}} \|\tilde{\mathbf{x}}_{\text{test}} - \mathbf{q}\|_2$ ▷ Projection onto fixed training representation
$f \leftarrow \sum_{i:a_i>0} a_i y_i K(\mathbf{q}_i, \hat{\mathbf{q}}) + b$ ▷ Decision function
$\hat{y} \leftarrow \text{sign}(f)$
return $\hat{y}$

3.8. RQS-Ensemble: Extension for Heavy-Tailed Financial Ratios

A single RQS-SVM has two practical limitations on bankruptcy data. First, standardization does not correct the heavy tails of financial ratios (Section 4.3 shows that the single model reaches a G-mean of only 0.37 on the US data). Second, the lower approximation of the majority class still contains thousands of observations, so the training set remains large and the SVM decision function is dominated by its geometry. RQS-Ensemble addresses both limitations while keeping RQS-SVM as its base learner.

Step 1: Quantile transformation.

The single RQS-SVM, i.e., Algorithm 1 and all its configurations in the ablation study (Section 4.3) and the comparison (RQS-SVM (tuned) and its variants), uses only the stan-

dardization of Equation (1). The quantile transformation described below is a component of RQS-Ensemble only,

where it replaces Equation (1); in the code it is controlled by the option `transform` $\in$ {`quantile`, `standard`} of RQS-Ensemble, whose second value gives the ablation variant "w/o quantile transform" (Section 4.3). Financial ratios are often heavy-tailed: a few firms have values several orders of magnitude larger than the median so that standardization leaves most observations concentrated in a small region of the space, which distorts the ϵ-neighborhoods and the RBF kernel. RQS-Ensemble therefore replaces standardization with a quantile transformation: each feature is mapped through its empirical distribution function, estimated on the training data with $\min(1000, n)$ quantiles, and then through the inverse standard normal distribution function; the result is clipped to $[-10, 10]$. The transformation is monotone for each feature, fitted on the training folds only and then applied unchanged to the test observations. Its effect is measured in the ablation study (Section 4.3).

Step 2: Data-driven parameters.

The parameters of the base learners are set from the transformed training data by three rules: the neighborhood factor $\alpha = \text{clip}\left(n^{-1/(d+4)}\,\bar{\Delta},\ 0.05,\ 0.45\right)$, where $n^{-1/(d+4)}$ is Scott's bandwidth factor and $\bar{\Delta}$ is the mean inter-class distance estimated on at most 150 observations per class; the boundary weight $\beta = \text{clip}\left(1/(1 + 0.1\log(1 + \text{IR})),\ 0.3,\ 0.9\right)$, which decreases with the imbalance ratio IR; and the PCM fuzzifier $m = \text{clip}(1 + 2/d_{\text{int}},\ 1.1,\ 3)$, where d_{int} is the number of principal components that explain 95% of the variance.

Step 3: Rough partition of the majority class.

The nearest enemy distances δ_i are computed on the full training set and the radius is set to $\epsilon = \max(\alpha\,\epsilon_0,\ q_{0.05}(\delta))$, where $q_{0.05}$ is the 5% quantile, which guarantees that at least 5% of the observations are boundary observations. The majority class is split into its lower approximation $\underline{N}$ and its boundary N_B.

Step 4: Rough-guided balanced subsamples.

For $b = 1, \ldots, B$, a subsample S_b is formed with all minority observations and the same number of majority observations drawn without replacement from $\underline{N}$ (completed from N_B if $\underline{N}$ is too small). The motivation is that, inside each member, the boundary of the majority class is then rebuilt from a sample in which the classes are balanced.

Step 5: Members and aggregation.

An RQS-SVM (Procedures 2 to 6 with PCM granulation and class mass balancing) is trained on each S_b with an RBF kernel of parameter $\gamma = g/d$. The score of a test observation is the average of the member decision values, $F(\mathbf{x}) = \frac{1}{B}\sum_b f_b(\mathbf{x})$, and the prediction is $\text{sign}(F(\mathbf{x}) - \theta)$.

Step 6: Hyperparameter selection.

The regularization parameter $C \in \{0.1, 1, 10, 100\}$, the kernel factor $g \in \{0.25, 1, 4\}$ and the threshold θ are chosen by an inner three-fold stratified cross-validation on the training part of each outer fold only: for each (C, g), out-of-fold scores are computed, the threshold maximizing the G-mean on these scores is retained, and the pair with the highest out-of-fold G-mean (ties broken by AUC-ROC) is refitted on the whole training part. The test fold is never used. Since Steps 1–4 do not depend on (C, g), they are computed once per inner fold.

We use $B = 10$ members. Each member is trained on $2n_{\min}$ observations instead of n, so the cost of the SVM optimization is reduced, and the members are trained in parallel. Algorithm 7 summarizes the method.

The contribution of each step is measured in Section 4.3 by replacing it with a simpler alternative (standardization, a single member, random undersampling, raw boundary, or direct evaluation).

Algorithm 7 RQS-Ensemble

Require: Training data $\{(\mathbf{x}_i, y_i)\}_{i=1}^{n}$, number of members B, grids for C and g
Ensure: Score function F and threshold θ
1: Fit the quantile transformation T on the training data; $\mathbf{z}_i \leftarrow \mathrm{clip}(T(\mathbf{x}_i), -10, 10)$
2: Compute α, β, m by the data-driven rules; compute ϵ_0 and the nearest enemy distances δ_i
3: $\epsilon \leftarrow \max(\alpha\epsilon_0, q_{0.05}(\delta))$; split the majority class into $\underline{N} = \{i : \delta_i > \epsilon\}$ and N_B
4: **for** $b = 1$ **to** B **do**
5: $\quad S_b \leftarrow P \cup$ a random sample of $|P|$ indices of $\underline{N}$ ▷ P: minority indices
6: **end for**
7: Select (C, g, θ) by inner cross-validation (Step 6)
8: **for** $b = 1$ **to** B **in parallel do**
9: $\quad f_b \leftarrow$ RQS-SVM$(\{(\mathbf{z}_i, y_i)\}_{i \in S_b};\ \alpha, \beta, m, C, \gamma = g/d)$ ▷ Algorithm 1
10: **end for**
11: **return** $F(\mathbf{x}) = \frac{1}{B}\sum_b f_b\big(\mathrm{clip}(T(\mathbf{x}), -10, 10)\big)$ and θ

3.9. Summary of Procedures

The proposed integration is further examined experimentally through component-wise configurations. In addition to the complete RQS-SVM framework, the experimental study considers RQS-SVM only and RQS-SVM k-means as reduced or modified configurations. These configurations make it possible to examine the contribution of the integrated rough–quotient–PCM–weighting architecture rather than evaluating the complete framework only as a black-box combination of existing techniques. The framework is also compared with DEC, SVM-SMOTE, SVM-ADASYN, SVM-Undersampling, Random Forest, KNN, logistic regression, and XGBoost and the more recent imbalance learning methods described in Section 2: weighted SVM, Balanced Random Forest, EasyEnsemble, RUSBoost, Self-Paced Ensemble, LightGBM and multilayer perceptrons trained with the focal, class-balanced focal and LDAM-DRW losses. These methods represent different methodological families, including clustering-based learning, resampling-based imbalance handling, kernel-based classification, and conventional machine learning classifiers. This experimental design provides both an internal component-wise comparison and an external comparison with established alternatives for imbalanced classification.

Table 5 summarizes the seven procedures that constitute the RQS-SVM framework.

Table 5. Summary of RQS-SVM procedures.

Procedure	Algorithm	Output
1. Preprocessing	Equation (1)	Standardized data
2. Rough set analysis	Algorithm 2	Lower approximations, boundary regions
3. PCM clustering	Algorithm 3	Centroids, membership matrix
4. Quotient space	Algorithm 4	Quotient space $\mathcal{Q}$, weights
5. Weight clipping and class mass balancing	Algorithm 5	Class-balanced weights
6. Weighted SVM	Section 3.6	SVM model
7. Inference	Algorithm 6	Predictions
Ensemble extension	Algorithm 7	Averaged scores, threshold

4. Experimental Studies

The selection of datasets plays a fundamental role in the experimental evaluation as it enables the assessment of the robustness and comparative effectiveness of different techniques.

4.1. Datasets

1. The first dataset (data1) corresponds to the Bankruptcy Data provided by the Taiwan Economic Journal covering the period from 1999 to 2009, publicly available on Kaggle: https://www.kaggle.com/datasets/fedesoriano/company-bankruptcy-prediction/data (accessed on 18 November 2026).
2. The second dataset (data2) is the US Company Bankruptcy Prediction dataset, also obtained from Kaggle: https://www.kaggle.com/datasets/utkarshx27/american-companies-bankruptcy-prediction-dataset (accessed on 18 November 2026).
   It is a panel: each row is a firm–year, and the same firm appears in several years. The firm identifier (`company_name`) and the fiscal year (`year`) are not financial ratios; they are removed from the features, and the firm identifier is used only to build the cross-validation folds (Section 4.2). The remaining 18 columns are financial variables.
3. The third dataset (data3) refers to the UK Bankruptcy dataset, which contains 5000 UK companies described by 68 financial ratios. It is proprietary and was kindly provided to the authors by Professor Jamal Ouenniche of the University of Edinburgh Business School. (Email: `jamal.ouenniche@ed.ac.uk`. The dataset cannot be redistributed under the terms of the data-sharing agreement.) The file contains neither a firm identifier nor a year; the rows are described by the provider as distinct companies, but this cannot be verified from the data, which also do not allow a temporal ordering. The file also contains a column that numbers the rows from 1 to 5000. The file is not in random order: the failed firms tend to have higher row numbers, and this index alone gives an AUC-ROC of 0.60. The column is therefore excluded from the variables.

It can be observed from Figure 2 that the datasets are severely imbalanced.

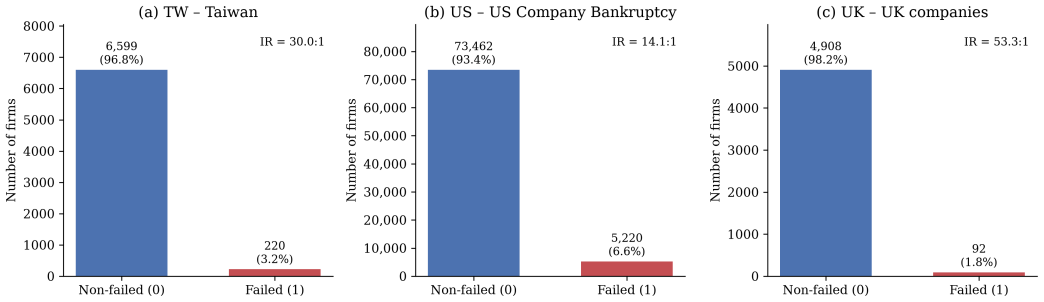


Figure 2. The visualizations of the datasets.

Table 6 summarizes the different characteristics of the three datasets.

Table 6. Characteristics of the 3 datasets. Imbalance: ratio of the number of non-failed to failed observations. For US, the observations are firm–years.

Code	Market	Access	Firms	Ratios	Failures (%)	Imbalance
TW	Taiwan	public	6819	95	3.23	30.0
US	United States	public	78,682	18	6.63	14.1
UK	United Kingdom	proprietary	5000	68	1.84	53.3

4.2. Experimental Protocol

All the methods are evaluated with the same outer five-fold stratified cross-validation (random seed 42) so that every comparison is paired. For the US panel data, the folds are built with stratified group cross-validation using the firm as the group: all the years of a firm are in the same fold, so no firm appears both in a training part and the corresponding test fold. Without this grouping, a classifier could recognize a firm from its other years, which would overestimate its performance. The indices of the test folds are saved with the code so that the splits can be reproduced exactly. The TW and UK datasets contain no firm identifier and no date, so neither grouping by firm nor a temporal split is possible for them; we checked that no row of the three datasets has identical features (Table 7). This check is not equivalent to grouping by firm: if a firm contributes observations from several years, these observations differ and may fall in both a training part and the corresponding test fold. For TW and UK, the cross-validation is therefore not protected against this form of leakage, and their results may be optimistic for all the methods (Section 5). For US, a temporal validation is reported in addition to the grouped cross-validation (Section 4.8).

Table 7. Leakage diagnostics. Rows with identical features: rows whose feature values appear in at least one other row.

	Rows	Firm Identifier	Year	Rows with Identical Features
TW	6819	no	no	0
US	78,682	yes	yes (1999–2018)	0
UK	5000	no	no	0

For US, the training part of each fold (about 63,000 firm–years) is reduced to 20,000 observations by stratified random sampling for computational reasons; the same reduced training set is used for every method and the test fold is always complete. For TW and UK the complete training parts are used.

All the data-dependent operations (standardization or quantile transformation, rough approximation, granulation, weighting, and hyperparameter and threshold selection) are fitted on the training part of each fold only. The test fold is used once to compute the metrics.

The positive class is the bankrupt (minority) class. For RQS-Ensemble, C, γ and the decision threshold are selected by the inner three-fold cross-validation described in Section 3.8. This selection is nested: it uses only the training part of each outer fold, and the outer test fold is never seen during tuning. The tuning budget is fixed: $4 \times 3 = 12$ pairs (C, g), each evaluated by three inner fits of the ten members, followed by one refit on the whole training part, i.e., 37×10 member fits per outer fold; the threshold is the value that maximizes the G-mean on the out-of-fold scores of the selected pair; the single RQS-SVM reported as "RQS-SVM (tuned)" uses the same inner procedure. Hyperparameter and threshold selection is part of the proposed framework. All the competing methods are used with the default settings of their libraries (scikit-learn, imbalanced-learn, XGBoost, and LightGBM) or those of their original publications and the default decision rule. This choice is discussed in Section 5. The configurations of the competing methods are listed in Table A1 (Appendix A).

With TP, FN, FP, TN denoting the entries of the confusion matrix, we report precision $= TP/(TP + FP)$, recall (sensitivity) $= TP/(TP + FN)$, specificity $= TN/(TN + FP)$, F1-score, G-mean $= \sqrt{\text{Recall} \times \text{Specificity}}$ and the Matthews correlation coefficient,

$$\text{MCC} = \frac{TP \cdot TN - FP \cdot FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}},$$

which are computed from the predicted labels, and two threshold-free measures computed from the continuous scores: the area under the ROC curve (AUC-ROC) and the area under the precision–recall curve (PR-AUC, computed as average precision). Under strong imbalance, PR-AUC is more informative than AUC-ROC because its baseline value equals the proportion of failures (0.032, 0.066 and 0.018 for TW, US and UK) instead of 0.5. Accuracy is reported for completeness but is not used to compare methods. The G-mean is the main criterion. All the metrics are averaged over the five test folds and reported with their standard deviation; the confusion matrices summed over the folds are given in Table A5 (Appendix B).

The comparison of k methods uses $N = 15$ blocks (3 datasets $\times$ 5 folds). We apply the Friedman test and its Iman–Davenport correction, $F_F = (N-1)\chi_F^2 / \left(N(k-1) - \chi_F^2\right)$, followed by the Nemenyi post hoc test with its critical difference diagram [63] and Wilcoxon signed-rank tests between the proposed method and each other method with Holm correction over the $k-1$ comparisons. Unless stated otherwise, "significantly better" and "significantly worse" refer to these Wilcoxon signed-rank tests over the 15 blocks (two-sided, zero differences split between the two signs) with Holm-adjusted p-value $p_{\text{Holm}} < 0.05$ and "not significantly different" to $p_{\text{Holm}} \geq 0.05$, which does not establish equivalence. The same tests with Holm correction over the configurations of each table are used in the ablation study. Because the five folds of one dataset share training data, they are not independent; we therefore also report, for each dataset separately, the corrected resampled t-test of Nadeau and Bengio [64], which inflates the variance of the fold differences by the factor $1/k + n_{\text{test}}/n_{\text{train}}$ with Holm correction.

All the methods are implemented in Python 3.14.4 with NumPy 2.5.3, SciPy 1.18.1, pandas 3.0.6, scikit-learn 1.9.1, imbalanced-learn 0.14.2, XGBoost 3.4.1, LightGBM 4.7.0, PyTorch 2.14.0, joblib 1.6.0 and Matplotlib 3.11.2 and run under Linux (kernel 7.0) on a 20-core x86-64 processor with 32 GB of memory. Fitting times are wall-clock times on this machine. All the random operations use the seed 42 (outer folds, models, samplers, PCM and member subsamples); the stratified reduction in the US training part of fold k uses the seed $42 + k$. The complete code (one Jupyter notebook), the indices of the test folds, and the fold-level results of every method are deposited at Zenodo, https://doi.org/10.5281/zenodo.24355667.

4.3. Ablation Study

The ablation study answers two questions: which components of RQS-SVM are responsible for its performance and whether the chosen integration in this work is better than simple alternatives. Each variant changes exactly one component of the full model and is evaluated on the same folds as the full model. Table 8 reports the 17 configurations of the single RQS-SVM (these are variants of the proposed method, not the 17 competing methods of Section 4.4), grouped into four families: (A) weighting, (B) characterization of uncertainty, (C) granulation of the boundary region and (D) prototypes and reference SVMs.

(A) Weighting:

The class mass balancing step is decisive. Without it (IQR clipping and minority amplification only), the G-mean falls from 0.771 to 0.290 on TW and to 0 on US and UK: the model predicts almost only the majority class, like the standard SVM. Uniform weights give the same result. Both differences are significant ($p_{\text{Holm}} < 0.001$, the full model is better in the 15 blocks, $r_{\text{rb}} = 1$). As explained in Remark 2, the variant with $\lambda = 0$ is identical to the full model.

Table 8. Ablation of the single RQS-SVM: mean ± standard deviation over the five folds ($C = 1$, γ = scale, threshold 0; α, β and m set by the data-driven rules of Section 3.8 on each training part). Δ: mean paired difference in G-mean over the 15 fold×dataset blocks, full model minus variant (positive: the full model is better); r_{rb}: matched-pairs rank biserial correlation (effect size from −1 to 1); p_{Holm}: Wilcoxon signed-rank test with Holm correction over the 16 comparisons; Rank: mean Friedman rank among the 17 configurations. Friedman $\chi^2_F = 154.45$, Iman–Davenport $F_F = 25.28$, $df = (16, 224)$, $p = 1.8 \times 10^{-41}$.

Configuration	G-Mean (Mean ± Sd)			PR-AUC (Mean ± Sd)			G-Mean, 15 Blocks			
	TW	US	UK	TW	US	UK	Δ	r_{rb}	p_{Holm}	Rank
RQS-SVM (full)	0.771 ± 0.028	0.373 ± 0.031	0.631 ± 0.095	0.305 ± 0.058	0.113 ± 0.017	0.075 ± 0.023	–	–	–	7.70
w/o class mass balancing (clipping and λ only)	0.290 ± 0.053	0.000 ± 0.000	0.000 ± 0.000	0.302 ± 0.029	0.094 ± 0.016	0.058 ± 0.033	+0.495	+1.00	<0.001	15.37
w/o weighting (uniform weights)	0.275 ± 0.039	0.000 ± 0.000	0.000 ± 0.000	0.290 ± 0.035	0.092 ± 0.012	0.057 ± 0.035	+0.500	+1.00	<0.001	15.60
w/o minority amplification (λ=0)	0.771 ± 0.028	0.373 ± 0.031	0.631 ± 0.095	0.305 ± 0.058	0.113 ± 0.017	0.075 ± 0.023	+0.000	+0.00	1.000	7.70
w/o weight clipping	0.768 ± 0.031	0.375 ± 0.033	0.623 ± 0.105	0.307 ± 0.061	0.113 ± 0.017	0.073 ± 0.022	+0.004	−0.10	1.000	7.70
quotient weights → balanced class weights	0.681 ± 0.053	0.433 ± 0.034	0.415 ± 0.090	0.281 ± 0.043	0.119 ± 0.017	0.065 ± 0.018	+0.082	+0.67	0.215	9.00
rough → fuzzy–rough (OWA)	0.826 ± 0.040	0.379 ± 0.035	0.623 ± 0.081	0.317 ± 0.062	0.112 ± 0.017	0.077 ± 0.022	−0.017	−0.73	0.113	4.53
rough → k-NN entropy	0.797 ± 0.020	0.340 ± 0.013	0.711 ± 0.075	0.233 ± 0.051	0.091 ± 0.004	0.106 ± 0.039	−0.024	−0.48	0.856	5.47
no partition: PCM on whole dataset	0.813 ± 0.017	0.000 ± 0.000	0.619 ± 0.049	0.093 ± 0.008	0.076 ± 0.010	0.026 ± 0.005	+0.115	+0.33	1.000	8.70
PCM → none (raw boundary)	0.832 ± 0.042	0.398 ± 0.022	0.669 ± 0.090	0.313 ± 0.060	0.111 ± 0.015	0.086 ± 0.033	−0.041	−0.97	0.002	3.07
PCM → k-means	0.772 ± 0.046	0.373 ± 0.031	0.638 ± 0.099	0.302 ± 0.052	0.112 ± 0.018	0.079 ± 0.025	−0.002	+0.07	1.000	7.20
PCM → fuzzy C-means	0.773 ± 0.048	0.373 ± 0.031	0.645 ± 0.088	0.315 ± 0.053	0.113 ± 0.017	0.079 ± 0.027	−0.005	+0.01	1.000	7.17
PCM → GMM	0.782 ± 0.038	0.373 ± 0.031	0.636 ± 0.097	0.302 ± 0.055	0.112 ± 0.017	0.081 ± 0.028	−0.005	−0.13	1.000	7.40
SVM → k-NN on RQS prototypes	0.412 ± 0.058	0.183 ± 0.047	0.000 ± 0.000	0.295 ± 0.050	0.075 ± 0.006	0.035 ± 0.012	+0.394	+1.00	<0.001	14.20
ClusterCentroids + SVM	0.805 ± 0.017	0.336 ± 0.009	0.359 ± 0.039	0.339 ± 0.040	0.072 ± 0.004	0.066 ± 0.033	+0.092	+0.55	0.573	10.00
Standard SVM	0.113 ± 0.112	0.000 ± 0.000	0.000 ± 0.000	0.302 ± 0.033	0.086 ± 0.014	0.101 ± 0.057	+0.555	+1.00	<0.001	16.00
Weighted SVM (balanced)	0.775 ± 0.053	0.450 ± 0.021	0.558 ± 0.045	0.328 ± 0.040	0.126 ± 0.018	0.120 ± 0.050	−0.002	−0.17	1.000	6.20

In the full model the IQR clipping sets all weights of a class to the same value (Tables 3 and 9), so the full model uses class mass balancing *alone*. The variant "w/o weight clipping" keeps the class mass balancing but restores the weights derived from the size and typicality of the granules (Equation (9)); it therefore isolates the contribution of the granule weighting. The two configurations are not significantly different ($\Delta = +0.004$, $r_{\mathrm{rb}} = -0.10$, $p_{\mathrm{Holm}} = 1$): the granule weights bring no measurable gain once the classes are balanced. Replacing the quotient-based weights with scikit-learn's balanced class weights on the same quotient set, which also equalizes the class masses but changes the effective regularization, decreases the mean G-mean by 0.082 without significant difference ($p_{\mathrm{Holm}} = 0.215$). Finally, a class-weighted SVM trained on the complete training set (group D) is not significantly different from RQS-SVM over the 15 blocks ($\Delta = -0.002$, $r_{\mathrm{rb}} = -0.17$, $p_{\mathrm{Holm}} = 1$). The performance of the single model is thus due to the balancing step, not the quotient or PCM construction. Table 9 gives, for every dataset and fold, the minority share of the total weight, the within-class dispersion of the weights and the G-mean and PR-AUC of the three configurations side by side. It confirms that the clipping reduces the within-class dispersion to zero in every fold and that the full model and the variant with granule weights differ by at most 0.012 in G-mean in 14 of the 15 folds; the only larger difference (UK, fold 5, 0.042) favors the full model. The absence of a significant difference with the class-weighted SVM, by contrast, is an aggregate result and should not be read as an equivalence: the class-weighted SVM has a higher G-mean in all five folds of US (+0.058 to +0.087), a lower G-mean in four of the five folds of UK (−0.018 to −0.163), and a higher PR-AUC in 12 of the 15 folds. The differences between the two models therefore depend on the dataset and largely cancel out in the overall test.

Before balancing, the minority class holds between 1.5% (UK) and 6.3% (US) of the total weight and between 2.3% and 9.2% after the clipping and the factor λ without balancing. Class mass balancing raises it to exactly 50% in every fold. The interquartile range of the weights is zero in every fold because the observations of the lower approximations, all of weight 1, are more than 99% of $\mathcal{Q}$; the clipping interval therefore reduces to a single value and the clipping removes all within-class dispersion. Without clipping, the granule weights vary by 1.3–3.5% of their mean in the majority class and by 5–17% in the minority class. These variations are too small to change the trained SVM significantly, which is

consistent with the absence of a significant difference between the two configurations. The granulation is also strong (Table 10): the boundary region, which contains 12–16% of the training observations on TW, 5–23% on UK and 37% on US, is replaced by one to five prototypes per class.

Table 9. Weights and performance per dataset and fold for the single RQS-SVM with the data-driven parameters of the ablation study (same models as Table 8). Minority share: share of the total weight held by the minority class before clipping (raw) and in the full model (after IQR clipping and class mass balancing). CV: largest coefficient of variation of the weights within a class (maximum over the two classes); in the full model it is zero up to rounding ($< 10^{-15}$) in every fold, whereas without clipping the granule weights vary within each class. G-mean and PR-AUC for the full model of the variant without clipping (granule weights active, same class mass balancing) and the class-weighted SVM trained on the whole training part.

Data	Fold	Minority Share		Max. Within-Class CV		G-Mean			PR-AUC		
		Raw	Full	Full	No Clip.	Full	No Clip.	W-SVM	Full	No Clip.	W-SVM
TW	1	2.1%	50.0%	0	0.140	0.738	0.739	0.758	0.215	0.213	0.302
	2	2.1%	50.0%	0	0.099	0.796	0.797	0.797	0.292	0.293	0.303
	3	2.1%	50.0%	0	0.101	0.803	0.804	0.757	0.369	0.371	0.388
	4	2.3%	50.0%	0	0.134	0.749	0.737	0.710	0.341	0.351	0.296
	5	2.2%	50.0%	0	0.141	0.771	0.761	0.851	0.307	0.305	0.352
	mean					0.771	0.768	0.775	0.305	0.307	0.328
US	1	5.9%	50.0%	0	0.051	0.358	0.359	0.446	0.091	0.091	0.103
	2	5.7%	50.0%	0	0.053	0.356	0.355	0.439	0.135	0.136	0.138
	3	6.3%	50.0%	0	0.050	0.370	0.371	0.450	0.124	0.124	0.139
	4	6.2%	50.0%	0	0.080	0.427	0.433	0.485	0.108	0.109	0.139
	5	6.2%	50.0%	0	0.051	0.355	0.355	0.428	0.105	0.105	0.111
	mean					0.373	0.375	0.450	0.113	0.113	0.126
UK	1	1.6%	50.0%	0	0.122	0.754	0.754	0.603	0.096	0.090	0.090
	2	1.6%	50.0%	0	0.140	0.717	0.717	0.553	0.098	0.099	0.190
	3	1.7%	50.0%	0	0.131	0.565	0.566	0.597	0.043	0.043	0.149
	4	1.6%	50.0%	0	0.168	0.561	0.561	0.543	0.075	0.073	0.062
	5	1.5%	50.0%	0	0.137	0.562	0.520	0.494	0.061	0.061	0.107
	mean					0.631	0.623	0.558	0.075	0.073	0.120

Table 10. Quantities of Theorem 1, Remark 1(b) and Proposition 4 computed on the trained single RQS-SVM (same models as Table 8): median and range over the five folds. $\widehat{D}_B$ is computed on the training boundary; the hinge risks use the granule size weights of Proposition 4; the projection replaces each test observation with its nearest representative in $\mathcal{Q}$.

Quantity	TW	US	UK
Boundary fraction $\lvert B\rvert/n$	0.137 (0.119–0.157)	0.372 (0.370–0.375)	0.172 (0.051–0.231)
Number of prototypes K	3 (2–4)	5 (4–8)	3 (2–5)
Distortion on B, $\widehat{D}_B$	25.6 (24.6–31.8)	0.00075 (0.00063–0.001)	6.9 (4.2–9.2)
Median $\lVert\mathbf{x}-Q(\mathbf{x})\rVert^2$ (test)	9.5 (8.6–9.7)	0.0007 (0.00058–0.00087)	2.9 (2.5–3.4)
$\lVert w\rVert_{\mathcal{H}}$	5.65 (5.55–5.68)	9.88 (9.31–10.31)	5.41 (5.26–5.58)
$L=\sqrt{2\gamma}\lVert w\rVert_{\mathcal{H}}$	0.78 (0.76–0.79)	2.64 (2.48–2.76)	0.84 (0.80–0.93)
Bound of Proposition 4, $L\,(\lvert B\rvert/n)\,\widehat{D}_B^{1/2}$	0.552 (0.527–0.599)	0.027 (0.023–0.030)	0.425 (0.102–0.536)
Observed $\lvert\widehat{R}_S-\widehat{R}_{\mathcal{Q}}\rvert$ (hinge)	0.0474 (0.0428–0.0507)	0.0015 (0.0009–0.0018)	0.0076 (0.0013–0.0172)
Ratio bound/observed	12 (11–13)	19 (13–30)	31 (16–427)
Test decisions changed by the projection	5.1% (4.7–5.4%)	2.0% (1.8–2.8%)	4.9% (4.5–5.8%)
Test decisions certified, $\lvert f(\mathbf{x})\rvert > L\lVert\mathbf{x}-Q(\mathbf{x})\rVert$	14.7% (13.7–18.1%)	79.8% (73.8–82.6%)	25.2% (22.6–27.3%)

(B) Uncertainty:

Replacing the crisp neighborhood rough partition with a fuzzy–rough partition (OWA lower approximation) gives a slightly better G-mean, not significant after correction ($\Delta = -0.017$, $p_{\text{Holm}} = 0.113$), and the k-NN entropy criterion gives a result that is not significantly different. Applying PCM to the whole dataset without any partition is unstable: it works on TW and UK but collapses on US (G-mean 0) because the few resulting prototypes cannot represent the majority class. The partition into certain and uncertain observations is therefore useful, but the crisp neighborhood rough set is not the best way to compute it.

(C) Granulation:

Keeping the raw boundary observations instead of PCM prototypes gives the best configuration (mean rank 3.07, $\Delta = -0.041$, $p_{\text{Holm}} = 0.002$). Replacing PCM with k-means, fuzzy c-means or a Gaussian mixture does not change the results ($p_{\text{Holm}} = 1$). The granulation of the boundary thus acts as a compression step, whose cost–benefit balance is given in Tables 11 and 12: it reduces the training set by 14% on TW (4691 instead of 5455 observations), 14% on UK (3441 instead of 4000) and 37% on US (12,557 instead of 20,000) and reduces the fitting time on US from 24.6 s to 14.5 s at the price of a loss of 0.041 in mean G-mean. The sensitivity analysis (Section 4.7) shows the same trade-off on TW: the larger the compressed region (ρ, α), the lower the G-mean, whereas the number of granules and the PCM parameters ($k_{\max}$, m, n_{init}) have almost no effect.

Table 11. Cost side of the ablation of the single RQS-SVM: size of the training set given to the classifier ($|\mathcal{Q}|$, or n for the reference SVMs) and fitting time per fold (seconds), mean over the five folds. Training parts contain about 5455 (TW), 20,000 (US) and 4000 (UK) observations. Peak memory is reported in Table 12.

Configuration	Training Size			Fit Time (s)		
	TW	US	UK	TW	US	UK
RQS-SVM (full)	4691	12,557	3441	1.8	14.5	1.0
w/o class mass balancing (clipping and λ only)	4691	12,557	3441	1.4	7.0	0.7
w/o weighting (uniform weights)	4691	12,557	3441	1.4	6.7	0.6
w/o minority amplification (λ = 0)	4691	12,557	3441	1.8	14.6	1.0
w/o weight clipping	4691	12,557	3441	1.5	12.1	1.0
quotient weights → balanced class weights	4691	12,557	3441	0.9	10.7	0.6
rough → fuzzy–rough (OWA)	4692	12,557	3442	8.1	17.1	2.9
rough → k-NN entropy	4691	12,557	3441	2.2	12.9	1.1
no partition: PCM on whole dataset	2	4	4	2.8	6.3	1.7
PCM → none (raw boundary)	5455	20,000	4000	1.7	24.6	1.0
PCM → k-means	4698	12,562	3447	1.2	9.9	0.7
PCM → fuzzy C-means	4698	12,562	3447	1.4	10.4	0.8
PCM → GMM	4698	12,562	3447	1.1	9.9	0.7
SVM → k-NN on RQS prototypes	4691	12,557	3441	0.6	2.3	0.4
ClusterCentroids + SVM	352	2654	147	0.5	2.8	0.2
Standard SVM	5455	20,000	4000	0.5	3.5	0.2
Weighted SVM (balanced)	5455	20,000	4000	0.9	16.9	0.3

Table 12. Cost–benefit of the main configurations (same folds, data-driven parameters, $C = 1$, γ = scale, threshold 0; mean over the five folds). Fit: time of the whole pipeline (rough approximation, granulation, weighting and SVM training). Predict: time to score the test fold, including the projection onto $\mathcal{Q}$ for RQS-SVM. Memory: peak memory allocated by Python and NumPy during fitting (`tracemalloc`; the internal buffers of LIBSVM are not included). SV: number of support vectors. Times measured on a laptop (Intel Core Ultra 7 165U, 32 GB, wall-clock). The G-mean and PR-AUC values are identical, fold by fold, to those of Tables 8 and 13. For RQS-Ensemble, $|\mathcal{Q}|$ and SV are not reported because each of the ten members has its own representation.

Configuration	Fit (s)	Predict (s)	Memory (MiB)	$\lvert\mathcal{Q}\rvert$	SV	G-Mean	PR-AUC
TW (n = 5455)							
RQS-SVM (full)	0.89	0.49	14.5	4691	1660	0.771	0.305
RQS-SVM w/o clipping (granule weights)	0.58	0.48	14.4	4691	1635	0.768	0.307
RQS-SVM, raw boundary (no PCM)	0.53	0.56	16.0	5455	2433	0.832	0.313
RQS-SVM, PCM → k-means	0.37	0.48	14.4	4698	1705	0.772	0.302
Weighted SVM (balanced)	0.29	0.10	8.0	5455	1610	0.775	0.328
Standard SVM	0.14	0.04	8.0	5455	715	0.113	0.302
RQS-Ensemble ($C = 1$, $g = 1$, 10 members)	1.21	0.24	37.5	–	–	0.848	0.284
US (n = 20,000)							
RQS-SVM (full)	5.22	5.12	9.3	12,557	11,456	0.373	0.113
RQS-SVM w/o clipping (granule weights)	4.81	5.12	9.3	12,557	11,459	0.375	0.113
RQS-SVM, raw boundary (no PCM)	7.49	8.25	11.8	20,000	18,660	0.398	0.111
RQS-SVM, PCM → k-means	3.02	5.09	9.3	12,562	11,455	0.373	0.112
Weighted SVM (balanced)	7.31	7.34	5.7	20,000	17,835	0.450	0.126
Standard SVM	1.25	1.26	5.7	20,000	3025	0.000	0.086
RQS-Ensemble ($C = 1$, $g = 1$, 10 members)	1.31	8.86	16.0	–	–	0.639	0.136
UK (n = 4000)							
RQS-SVM (full)	0.61	0.26	7.7	3441	2311	0.631	0.075
RQS-SVM w/o clipping (granule weights)	0.54	0.26	7.6	3441	2272	0.623	0.073
RQS-SVM, raw boundary (no PCM)	0.47	0.35	8.5	4000	2787	0.669	0.086
RQS-SVM, PCM → k-means	0.39	0.29	7.6	3447	2318	0.638	0.079
Weighted SVM (balanced)	0.19	0.07	4.2	4000	1317	0.558	0.120
Standard SVM	0.09	0.03	4.2	4000	552	0.000	0.101
RQS-Ensemble ($C = 1$, $g = 1$, 10 members)	1.33	0.15	14.2	–	–	0.753	0.112

Table 13. Ablation of RQS-Ensemble ($B = 10$, $C = 1$, $g = 1$, threshold 0; no tuning so that only the modified step changes). Same notation as Table 8; Holm correction over the 6 comparisons. Rank among the 7 configurations.

Configuration	G-Mean (Mean ± Sd)			PR-AUC (Mean ± Sd)			G-Mean, 15 Blocks			
	TW	US	UK	TW	US	UK	Δ	r_{rb}	p_{Holm}	Rank
RQS-Ensemble (full)	0.848 ± 0.040	0.639 ± 0.011	0.753 ± 0.040	0.284 ± 0.033	0.136 ± 0.006	0.112 ± 0.024	–	–	–	2.67
w/o quantile transform (standardization)	0.829 ± 0.026	0.410 ± 0.023	0.711 ± 0.087	0.289 ± 0.042	0.104 ± 0.013	0.098 ± 0.058	+0.097	+0.85	0.010	5.53
w/o ensemble (single member, $B = 1$)	0.840 ± 0.047	0.626 ± 0.009	0.732 ± 0.060	0.207 ± 0.031	0.128 ± 0.004	0.074 ± 0.013	+0.014	+0.98	<0.001	5.10
rough-guided → random undersampling	0.836 ± 0.038	0.639 ± 0.010	0.746 ± 0.047	0.293 ± 0.019	0.134 ± 0.006	0.109 ± 0.030	+0.006	+0.52	0.250	3.73
rough-guided → half boundary/half lower	0.804 ± 0.051	0.632 ± 0.007	0.745 ± 0.055	0.303 ± 0.084	0.133 ± 0.008	0.102 ± 0.031	+0.020	+0.60	0.165	4.27
PCM → none (raw boundary) in members	0.847 ± 0.035	0.639 ± 0.009	0.726 ± 0.048	0.277 ± 0.073	0.135 ± 0.006	0.106 ± 0.018	+0.010	+0.47	0.250	4.20
projection → direct inference	0.858 ± 0.032	0.650 ± 0.013	0.738 ± 0.035	0.381 ± 0.056	0.147 ± 0.008	0.127 ± 0.043	−0.002	−0.13	0.679	2.50

Table 12, measured on a laptop with the same folds, completes this picture. On TW and UK the compression does not reduce the fitting time of the single model: the rough approximation and the granulation cost more than they save in the SVM (0.89 s against 0.53 s with the raw boundary on TW). On US, where the boundary is the largest, the full model is 30% faster than the raw-boundary variant (5.2 s against 7.5 s) and also faster than the class-weighted SVM on the whole training part (7.3 s). The compression reduces the number of support vectors by 32% (TW), 39% (US) and 17% (UK) with respect to the raw-boundary variant, but the class-weighted SVM has fewer support vectors than RQS-SVM on TW and UK. Prediction is slower than for the weighted SVM because each test observation is first projected onto $\mathcal{Q}$. The peak memory allocated by Python remains

below 40 MiB for every configuration. The same quantities for all 17 configurations of the ablation study are given in Appendix C (Table A8): the peak memory stays between 4 and 21 MiB in every configuration, the largest value being that of the fuzzy–rough partition on US; the configurations without class mass balancing have few support vectors (about 460 on TW) only because they nearly ignore the minority class, as their G-mean shows. At these sample sizes, the compression of the single model therefore has a measurable cost in G-mean and a computational benefit only on the largest dataset. RQS-Ensemble, whose members are trained on small balanced subsamples, has the best G-mean of the Table 13 on the three datasets for a fitting time of 1.2–1.3 s per fold.

(D) Prototypes and reference SVMs:

The weighted SVM remains necessary: a k-NN classifier trained on the same quotient representation fails on UK and US. ClusterCentroids undersampling followed by an SVM is not significantly different from RQS-SVM, and the class-weighted SVM obtains a slightly better mean rank (6.20 against 7.70) without significant difference ($p_{\text{Holm}} = 1$). The single RQS-SVM therefore does not improve on a well-weighted SVM in terms of G-mean; its advantage is a smaller training set. On US, all SVMs trained on standardized data reach a G-mean below 0.45. This observation motivated RQS-Ensemble.

On the same folds, RQS-Ensemble without tuning reaches G-mean values of 0.848, 0.639 and 0.753 on TW, US and UK against 0.771, 0.373 and 0.631 for the single RQS-SVM. Table 13 identifies the source of this gain. The quantile transformation is the most important step: with standardization, the G-mean on US falls from 0.639 to 0.410 ($\Delta = +0.097$, $p_{\text{Holm}} = 0.010$). Aggregating ten members instead of one improves the G-mean in 14 of the 15 blocks ($\Delta = +0.014$, $p_{\text{Holm}} < 0.001$) and the PR-AUC on TW from 0.207 to 0.284. By contrast, drawing the majority observations from the rough lower approximation is not significantly different from random undersampling ($\Delta = +0.006$, $p_{\text{Holm}} = 0.250$), drawing half of them from the majority boundary tends to reduce the G-mean ($\Delta = +0.020$, $p_{\text{Holm}} = 0.165$), and PCM granulation inside the members has no significant effect ($p_{\text{Holm}} = 0.250$). Finally, evaluating the decision function directly at the test observation instead of its projection gives the best mean rank (2.50) and a higher PR-AUC on the three datasets with no significant difference in G-mean ($p_{\text{Holm}} = 0.679$).

The components that significantly contribute to the performance are the class mass-balanced weighting, the partition into certain and uncertain observations, the quantile transformation and the ensemble. The rough guidance of the subsamples and the choice of the granulation algorithm do not significantly change the accuracy, and the PCM granulation, which costs 0.041 in mean G-mean in the single model, has no significant effect within the ensemble ($p_{\text{Holm}} = 0.250$); their contribution is a reduction in the training set and a structured interpretable representation of the uncertain region. We report these negative results because they delimit precisely what the rough–quotient construction brings.

4.4. Performance Analysis and Discussion

In the proposed RQS-SVM framework, we use the Radial Basis Function (RBF) kernel, which is implemented through the `SVC` classifier of the Scikit-learn library. The RBF kernel is defined as

$$K(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(-\gamma \|\mathbf{x}_i - \mathbf{x}_j\|^2\right),$$

where C is the regularization parameter and γ controls the width/locality of the RBF kernel. Equivalently, when the kernel width is expressed using σ, the relationship is

$$\gamma = \frac{1}{2\sigma^2}.$$

C, γ and the decision threshold of RQS-Ensemble and that of RQS-SVM (tuned) are selected by the inner cross-validation of Section 3.8 using the training part of each outer fold only; the competing methods keep their default settings (Section 4.2).

Table 14 reports the G-mean, MCC, AUC-ROC and PR-AUC of the 22 methods on the three datasets: the 17 competing methods and five methods of the RQS family (RQS-Ensemble and four RQS-SVM variants). These 22 methods should not be confused with the 17 configurations of the ablation study (Section 4.3), which are variants of RQS-SVM evaluated separately; the eight metrics with their standard deviations, the confusion matrices, and 95% confidence intervals with effect sizes are given in Appendix B (Tables A2–A7). The methods are grouped into four families: the proposed method and its variants (RQS-Ensemble, RQS-SVM (tuned), RQS-SVM only, RQS-SVM k-means and RQS-SVM with raw boundary), classical methods (DEC, SVM-SMOTE, SVM-ADASYN, SVM-Undersampling, Random Forest, KNN, logistic regression, and XGBoost), recent ensemble methods (weighted SVM, Balanced Random Forest, EasyEnsemble, RUSBoost, Self-Paced Ensemble, and LightGBM) and deep models (MLP with focal, class-balanced focal and LDAM-DRW losses).

Table 14. Mean performance over the five test folds (best value of each column in bold). RQS-Ensemble and RQS-SVM (tuned) select C, γ and the threshold by inner cross-validation; the other methods use default settings. All eight metrics with standard deviations are given in Tables A2–A4 (Appendix B).

Method	TW				US				UK			
	G-Mean	MCC	AUC	PR-AUC	G-Mean	MCC	AUC	PR-AUC	G-Mean	MCC	AUC	PR-AUC
RQS-Ensemble (proposed)	0.840	0.307	0.914	0.300	0.607	0.117	0.661	0.115	0.700	0.162	0.834	0.102
RQS-SVM (tuned)	0.821	0.277	0.907	0.323	0.587	0.099	0.651	0.122	0.631	0.105	0.760	0.056
RQS-SVM only	0.275	0.197	0.842	0.302	0.000	−0.003	0.557	0.085	0.000	−0.003	0.622	0.056
RQS-SVM k-means	0.772	0.303	0.900	0.302	0.373	0.066	0.621	0.112	0.638	0.116	0.789	0.079
RQS-SVM (raw boundary)	0.832	0.306	0.909	0.313	0.398	0.068	0.611	0.111	0.669	0.124	0.795	0.086
Weighted SVM	0.775	0.315	0.914	0.328	0.450	0.077	0.632	0.126	0.558	0.131	0.786	0.120
Balanced Random Forest	**0.865**	0.388	0.940	0.431	0.580	**0.156**	**0.700**	**0.142**	**0.774**	**0.205**	**0.860**	0.131
EasyEnsemble	0.863	0.344	0.939	0.441	**0.615**	0.118	0.658	0.130	0.765	0.160	0.856	**0.163**
RUSBoost	0.844	0.369	0.937	0.398	0.583	0.117	0.625	0.100	0.761	0.172	0.854	0.100
Self-Paced Ensemble	0.827	0.364	0.918	0.317	0.602	0.118	0.653	0.109	0.739	0.161	0.809	0.074
LightGBM (balanced)	0.577	**0.452**	**0.943**	**0.513**	0.473	0.128	0.671	0.138	0.000	−0.003	0.826	0.106
MLP + Focal loss	0.838	0.317	0.922	0.394	0.536	0.125	0.681	0.138	0.680	0.156	0.767	0.108
MLP + CB-Focal loss	0.762	0.407	0.923	0.405	0.053	0.023	0.680	0.141	0.462	0.124	0.761	0.103
MLP + LDAM-DRW	0.771	0.386	0.909	0.381	0.033	0.013	0.677	0.137	0.340	0.100	0.738	0.102
DEC	0.738	0.367	0.923	0.349	0.187	0.051	0.641	0.109	0.093	0.027	0.828	0.106
SVM-SMOTE	0.770	0.307	0.912	0.321	0.478	0.077	0.631	0.125	0.519	0.106	0.753	0.114
SVM-ADASYN	0.765	0.308	0.907	0.295	0.475	0.075	0.621	0.121	0.537	0.117	0.754	0.116
SVM-Undersampling	0.856	0.329	0.924	0.350	0.417	0.070	0.625	0.120	0.744	0.155	0.802	0.109
Random Forest	0.599	0.404	0.932	0.411	0.191	0.070	0.684	0.133	0.227	0.113	0.817	0.104
KNN	0.444	0.319	0.784	0.253	0.167	0.037	0.582	0.086	0.000	-0.003	0.582	0.033
Logistic Regression	0.837	0.335	0.893	0.354	0.530	0.090	0.661	0.132	0.681	0.135	0.737	0.068
XGBoost	0.583	0.379	0.918	0.383	0.410	0.102	0.647	0.120	0.193	0.094	0.798	0.113

Random Forest and XGBoost, although trained with class weights, and KNN reach the highest accuracies (above 0.96 on TW and UK) but detect few failures with the default threshold: their G-mean is between 0.44 and 0.60 on TW, below 0.23 on UK and below 0.42 on US, and KNN never predicts a failure on UK. Logistic regression, which is used with balanced class weights, is much better balanced (G-mean 0.837, 0.530 and 0.681 on TW, US and UK). The SVM-based resampling methods (SVM-SMOTE and SVM-ADASYN) reach a G-mean of about 0.77 on TW but only about 0.53 on UK and 0.48 on US, where they predict most firms as bankrupt (specificity about 0.26). SVM-Undersampling is better on TW and UK (0.856 and 0.744) but has the same problem on US (0.417). DEC detects few failures on UK and US.

The ensembles that combine undersampling with bagging or boosting are the strongest competitors. EasyEnsemble has the best G-mean on US (0.615) and Balanced Random Forest the best G-mean on TW and UK (0.865 and 0.774); RUSBoost and Self-Paced Ensemble follow closely. For the threshold-free metrics, Balanced Random Forest has the best AUC-ROC on US and UK and the best PR-AUC on US, EasyEnsemble the best PR-AUC on UK, and LightGBM the best AUC-ROC and PR-AUC on TW (0.943 and 0.513); Balanced Random Forest has the best MCC on US and UK and LightGBM on TW. However, LightGBM with balanced class weights and the default threshold detects only a third of the failures on TW, none on UK and a quarter on US so that its G-mean is low. The MLPs trained with the class-balanced focal loss and LDAM-DRW give competitive results on TW but almost never predict a failure on US (recall below 0.01), whereas the focal loss gives more stable results.

RQS-Ensemble reaches G-mean of 0.840, 0.607 and 0.700 on TW, US and UK with recalls of 0.855, 0.548 and 0.588 and specificities of 0.830, 0.674 and 0.847. It is the second best method on US, after EasyEnsemble (0.615), and ranks fifth on TW (behind Balanced Random Forest, EasyEnsemble, SVM-Undersampling and RUSBoost) and sixth on UK (behind Balanced Random Forest, EasyEnsemble, RUSBoost, SVM-Undersampling and Self-Paced Ensemble). Compared with the single RQS-SVM tuned with the same inner procedure, the ensemble improves the G-mean on the three datasets (by 0.019, 0.020 and 0.069) and the specificity substantially. The configurations without class mass balancing (RQS-SVM only) predict almost only the majority class, which confirms the ablation study. The weakness of RQS-Ensemble is its ranking quality: its PR-AUC (0.300, 0.115 and 0.102) is lower than that of Balanced Random Forest (0.431, 0.142 and 0.131) and of LightGBM on TW and US, and its AUC-ROC is lower than that of Balanced Random Forest on all datasets.

The ablation study of Table 13 was run on the same folds with fixed parameters ($C = 1$, $g = 1$, threshold 0). Comparing the two runs shows that the inner selection of C, γ and the threshold did not help: the fixed configuration has higher G-mean (0.848, 0.639 and 0.753) than the tuned one (0.840, 0.607 and 0.700) in 14 of the 15 blocks (mean difference 0.031, Wilcoxon $p < 0.001$) because tuning increases the recall but reduces the specificity more. With about 176, 1330 and 74 failures in the training parts (the US training part being reduced to 20,000 firm–years), the inner validation folds contain only about 59, 440 and 25 failures, and the threshold that maximizes the G-mean on them is unstable. We keep the tuned model as the main result because this protocol was fixed before the experiments; as a supplementary analysis, when the fixed configuration replaces it in the comparison, its average G-mean rank becomes 3.47 (second, after EasyEnsemble with 2.80, and before Balanced Random Forest with 3.80); its differences with EasyEnsemble, Balanced Random Forest and RUSBoost are not significant (Wilcoxon $p = 1.00$, 0.72 and 0.09 without correction), and it is better than Self-Paced Ensemble ($+0.024$, $p = 0.01$ without correction). In this configuration all the methods, including ours, use default parameters.

The rankings by G-mean and the threshold-free metrics differ markedly (Table 15). Methods that balance the classes during training and predict with the default threshold (EasyEnsemble, Balanced Random Forest, RUSBoost, Self-Paced Ensemble, and RQS-Ensemble) obtain the best G-mean ranks. Methods that only reweight the classes and keep the default probability threshold of 0.5 (Random Forest, XGBoost, and LightGBM) can have a high AUC-ROC or PR-AUC, which shows that their scores separate the classes well, but their default threshold is too high for the minority class, which lowers their G-mean. Conversely, RQS-Ensemble has a good G-mean but an intermediate AUC-ROC rank (9.13) and a low PR-AUC rank (13.53). Part of this gap is due to the projection step, which gives tied scores to all test observations that share a representative: evaluating the members directly at the test observations increased the PR-AUC on TW from 0.284 to 0.381 in the ablation study (Table 13).

Table 15. Average Friedman ranks over the 15 fold×dataset blocks (22 methods; lower is better), sorted by G-mean.

Method	G-Mean	MCC	F1-Score	AUC-ROC	PR-AUC
EasyEnsemble	2.47	7.20	10.93	5.27	5.80
Balanced Random Forest	3.73	2.47	3.87	2.13	4.40
RUSBoost	4.07	6.20	7.27	7.53	11.53
Self-Paced Ensemble	4.93	6.60	7.40	10.67	16.13
RQS-Ensemble (proposed)	5.07	9.80	10.53	9.13	13.53
SVM-Undersampling	6.93	12.13	13.47	11.53	10.33
Logistic Regression	7.33	10.93	11.20	14.07	12.27
MLP + Focal loss	7.33	8.87	9.20	9.07	7.47
RQS-SVM (tuned)	7.60	14.13	15.07	13.53	15.33
RQS-SVM (raw boundary)	10.53	14.53	15.30	15.20	14.73
SVM-ADASYN	12.07	14.13	11.57	16.60	13.40
SVM-SMOTE	12.33	14.47	12.00	14.47	11.67
Weighted SVM	12.60	12.53	10.53	13.00	10.07
RQS-SVM k-means	13.47	15.40	14.13	15.33	16.07
MLP + LDAM-DRW	15.93	12.00	12.07	12.27	8.27
MLP + CB-Focal loss	16.07	11.13	10.00	8.80	8.00
LightGBM (balanced)	16.63	8.47	8.23	5.33	4.87
XGBoost	17.00	9.53	7.93	11.27	9.33
Random Forest	17.87	10.13	9.73	5.80	8.07
DEC	17.87	14.27	14.07	10.00	12.40
KNN	19.90	16.97	17.23	21.40	21.33
RQS-SVM only	21.27	21.10	21.27	20.60	18.00

Effect of the projection on the main comparison. The main results use projection-based inference because it was the inference rule fixed before the experiments and changing it after seeing the test results would bias the comparison in favor of the proposed method. Its cost can nevertheless be quantified with the runs already made. We replaced, in the comparison of the 22 methods, the results of RQS-Ensemble with those of the fixed configuration ($C = 1$, $g = 1$, threshold 0) of Table 13, evaluated on the same folds either with projection or direct evaluation. With projection, the mean G-mean rank of RQS-Ensemble is 3.47 (second of 22 methods) and its mean PR-AUC rank 10.6 (tenth); with direct evaluation it becomes 3.40 (second) and 6.07 (third), and the mean AUC-ROC rank improves from 7.27 to 4.07. Direct evaluation hardly changes the G-mean (higher in 8 of the 15 blocks, mean $+0.002$, lower on UK; not significant, Table 13), but it increases the AUC-ROC in 14 blocks ($+0.014$) and the PR-AUC in 13 blocks ($+0.042$, $+0.098$ on TW). The projection step is therefore the main reason for the low PR-AUC rank of RQS-Ensemble in Table 14: with direct evaluation the method would be among the best three for G-mean, AUC-ROC and PR-AUC. We recommend direct evaluation for applications, and we keep the projection in Table 14 only so that the reported comparison follows the protocol fixed in advance (Section 5).

In bankruptcy prediction a missed failure usually costs much more than a false alarm. Table 16 reports the normalized expected misclassification cost for cost ratios c from 5 to 50, computed from the confusion matrices of Table A5; the value 1 corresponds to the best trivial classifier. Three conclusions follow. First, the methods trained without rebalancing and used with their default threshold (Random Forest, XGBoost, and LightGBM) are among the cheapest methods when $c = 5$ but become worse than the trivial classifier on TW and US when $c = 50$. Second, on TW and UK, RQS-Ensemble belongs to the group of low-cost methods for c between 20 and 50, i.e., for cost ratios close to the imbalance ratios: it ranks fourth among the 22 methods on TW for $c = 50$ (0.413 against 0.357 for EasyEnsemble) and third on UK for $c = 20$ (0.822 against 0.693 for Balanced Random

Forest), but it is never the cheapest method. Third, on US, no method reduces the cost much below that of the trivial classifier (the lowest value is 0.83), which reflects the weak discriminating power of the 18 US variables (AUC-ROC at most 0.70 for all the methods). The practical benefit of RQS-Ensemble is therefore comparable to, not larger than, that of the undersampling ensembles.

Table 16. Normalized expected misclassification cost, $(c\,\mathrm{FN} + \mathrm{FP}) / \min(c\,P, N)$, computed from the confusion matrices summed over the five folds for cost ratios c of a missed failure to a false alarm. A value of 1 corresponds to the best trivial classifier (all firms non-failed or all failed); lower is better; best value of each column in bold. The G-mean-optimal threshold corresponds to c close to the imbalance ratio (30, 14 and 53 for TW, US and UK).

Method	TW				US				UK			
	$c = 5$	$c = 10$	$c = 20$	$c = 50$	$c = 5$	$c = 10$	$c = 20$	$c = 50$	$c = 5$	$c = 10$	$c = 20$	$c = 50$
RQS-Ensemble (proposed)	1.166	0.656	0.401	0.413	1.369	0.911	0.968	1.932	2.048	1.230	0.822	0.577
RQS-SVM (tuned)	1.347	0.751	0.453	0.456	1.730	1.010	0.923	1.540	4.752	2.490	1.359	0.681
EasyEnsemble	0.999	0.563	0.345	**0.357**	1.493	0.929	0.920	1.700	3.150	1.667	0.926	0.481
Balanced Random Forest	0.791	**0.475**	**0.317**	0.371	1.056	**0.827**	1.012	2.285	1.891	1.092	**0.693**	**0.453**
RUSBoost	0.826	0.513	0.357	0.438	1.296	0.905	1.009	2.106	2.691	1.465	0.852	0.484
Self-Paced Ensemble	0.821	0.531	0.386	0.498	1.315	0.900	0.985	2.019	2.452	1.384	0.849	0.529
SVM-Undersampling	1.065	0.599	0.365	0.375	2.359	1.221	0.926	1.102	2.765	1.518	0.895	0.521
Logistic Regression	0.965	0.580	0.388	0.454	2.031	1.100	**0.901**	1.261	2.376	1.400	0.912	0.619
Weighted SVM	0.915	0.624	0.478	0.650	2.280	1.188	0.913	1.118	1.457	1.060	0.861	0.742
LightGBM (balanced)	0.705	0.685	0.674	1.113	1.011	0.882	1.162	2.768	1.011	1.005	1.003	1.001
Random Forest	0.717	0.677	0.657	1.074	**0.987**	0.975	1.377	3.431	**0.972**	**0.942**	0.928	0.919
XGBoost	0.740	0.697	0.676	1.105	1.019	0.919	1.234	2.978	0.985	0.960	0.947	0.940
MLP + Focal loss	1.075	0.624	0.398	0.438	1.181	0.893	1.065	2.355	1.898	1.177	0.817	0.601

The decision threshold of RQS-Ensemble is the value that maximizes the G-mean on out-of-fold scores of the training part. Maximizing the G-mean amounts to weighting errors on the two classes inversely to their frequencies, i.e., a cost ratio close to the imbalance ratio; when the actual cost ratio of the user is known, the same out-of-fold scores can be used to choose the threshold that minimizes the expected cost instead. The scores of RQS-SVM are SVM margins averaged over the members, not probabilities. They are adequate for ranking (AUC-ROC and PR-AUC) and thresholding, but they are ranking scores, not probabilities of default. To check what a probability would add, we recalibrated the scores of RQS-Ensemble and those of the competitors by Platt scaling fitted on the out-of-fold scores of the inner cross-validation, i.e., the training part only, and computed the Brier score and the expected calibration error (ECE) on the test folds (Table 17). The raw probabilities of the undersampling ensembles are strongly miscalibrated (ECE between 0.15 and 0.42) because they are estimated on balanced samples; after recalibration, all the methods, RQS-Ensemble included, have an ECE below 0.025 (the Platt fit of logistic regression degenerates on TW). Their Brier scores, however, are below that of a constant prediction equal to the failure rate only on TW (0.022–0.027 against 0.031); on US and UK they are equal to it within ± 0.002. The recalibrated scores are therefore well-calibrated but carry almost no information on the probability of default beyond the base rate on US and UK. For these reasons, the scores of RQS-Ensemble should be used for ranking and thresholding only, as in this article, and not read as probabilities.

Table 17. Calibration of the predicted probabilities (mean over the five test folds; lower is better). Raw: probabilities returned by the method with default parameters; Platt: logistic recalibration of the scores of the same model, fitted on the out-of-fold scores of the inner 3-fold cross-validation of the training part only. ECE: expected calibration error with ten equal-width bins. Reference: constant prediction equal to the failure rate of the test fold (Brier score $\pi(1-\pi)$, ECE 0). The SVM-based methods and RQS-Ensemble output decision scores, not probabilities, and have no raw value.

Method	TW				US				UK			
	Brier		ECE		Brier		ECE		Brier		ECE	
	Raw	Platt	Raw	Platt	Raw	Platt	Raw	Platt	Raw	Platt	Raw	Platt
RQS-Ensemble ($C = 1$, $g = 1$)	–	0.027	–	0.008	–	0.060	–	0.006	–	0.017	–	0.002
RQS-Ensemble (tuned)	–	0.026	–	0.011	–	0.062	–	0.019	–	0.017	–	0.007
Balanced Random Forest	0.077	0.024	0.150	0.011	0.147	0.061	0.267	0.017	0.112	0.017	0.237	0.006
EasyEnsemble	0.166	0.023	0.372	0.010	0.232	0.061	0.409	0.004	0.196	0.017	0.417	0.004
RUSBoost	0.086	0.024	0.214	0.008	0.233	0.061	0.288	0.009	0.153	0.017	0.301	0.007
Self-Paced Ensemble	0.088	0.025	0.170	0.009	0.224	0.061	0.347	0.015	0.160	0.018	0.284	0.003
LightGBM	0.024	0.022	0.023	0.011	0.104	0.062	0.142	0.019	0.019	0.018	0.019	0.005
Random Forest	0.026	0.025	0.023	0.011	0.069	0.062	0.062	0.021	0.021	0.018	0.019	0.005
XGBoost	0.029	0.027	0.025	0.009	0.094	0.063	0.103	0.019	0.020	0.018	0.019	0.003
Weighted SVM	–	0.026	–	0.007	–	0.061	–	0.004	–	0.017	–	0.004
Logistic Regression	0.094	fail †	0.152	fail †	0.238	0.062	0.407	0.002	0.142	0.018	0.245	0.001
Reference (constant failure rate)	0.031		0		0.062		0		0.018		0	

† Platt fit degenerate (constant prediction 0.5) in four of the five folds because of the extreme scores produced on the heavy-tailed ratios.

Since RQS-Ensemble is not more accurate than the best undersampling ensembles, its interest lies elsewhere. (i) It is the only SVM-based method of the leading group: it is significantly better than all the SVM-based alternatives (weighted SVM, SVM-SMOTE, SVM-ADASYN, DEC and RQS-SVM without class mass balancing) with their default settings, which makes it relevant where kernel methods are preferred; when the weighted SVM selects C, γ and its threshold by the same inner cross-validation, the difference is no longer significant (Section 4.5). (ii) Its training sample is an explicit analyzable object: certain observations, prototypes of the uncertain region and class-balanced weights, for which Theorem 1 and Propositions 2–4 provide explanatory bounds (Section 4.9). (iii) Its ablation identifies precisely which ingredients matter for imbalanced financial data (class mass balancing, quantile transformation, and ensembling) and which do not (granule weighting and rough guidance), which is useful for the design of other SVM-based methods. We do not claim that it should replace the undersampling ensembles in practice.

4.5. Sensitivity to the Tuning Protocol

In Table 14, only RQS-Ensemble selects its hyperparameters and threshold by inner cross-validation; the competitors use the library defaults of Table A1. To measure the effect of this asymmetry, we applied the same inner 3-fold cross-validation with the same selection criterion to RQS-Ensemble as well as the nine strongest or most common competitors on the same outer folds (Table 18). Two levels of tuning were applied: the threshold only and a small grid of parameters together with the threshold. With default settings, the results reproduce exactly, fold by fold, those of Table 14.

Three results stand out. First, tuning the threshold matters greatly for the methods whose default threshold ignores the imbalance: it raises the mean G-mean by 0.39 for Random Forest, 0.38 for LightGBM, 0.34 for XGBoost and 0.10 for the weighted SVM but by at most 0.02 for the undersampling ensembles, which already train on balanced samples. Second, the parameter grids add little to the threshold (-0.009 to $+0.046$), and for RQS-Ensemble the inner tuning lowers the G-mean (-0.016 with respect to its fixed

configuration with a tuned threshold), as already observed in Table 13. Third, the position of RQS-Ensemble depends on the protocol: among these ten methods, its mean G-mean rank is 4.00 (fifth) with the protocol of Table 14, 2.80 (second) when all the methods use their defaults, 5.40 (sixth) when all the methods tune their threshold, and 8.13 (tenth, last) when all the methods tune their parameters and threshold. When all thresholds are tuned, the fixed RQS-Ensemble is significantly worse than Balanced Random Forest ($\Delta = -0.028$, lower in all 15 blocks, $p_{\text{Holm}} = 0.001$) and not significantly different from the eight other competitors; when everything is tuned, the tuned RQS-Ensemble is significantly worse than Balanced Random Forest, EasyEnsemble, Self-Paced Ensemble and Random Forest ($p_{\text{Holm}} \leq 0.018$).

Table 18. Sensitivity of the comparison to the tuning protocol for RQS-Ensemble and the nine strongest or most common competitors (same outer folds as Table 14). Def.: default parameters and default threshold (0 for the SVM decision function and RQS-Ensemble, 0.5 for probabilities); Thr.: default parameters, threshold maximizing the G-mean on the out-of-fold scores of the inner 3-fold cross-validation; Tuned: parameters chosen in the grid of the last column and threshold, both by the same inner cross-validation as RQS-Ensemble (G-mean at the best threshold, then AUC). For RQS-Ensemble, Def. and Thr. use the fixed configuration $C = 1$, $g = 1$ and Tuned is the configuration of Table 14. Mean G-mean over the five folds and mean rank of the G-mean among the ten methods over the 15 dataset–fold blocks under four protocols: (a) protocol of Table 14 (RQS-Ensemble tuned, competitors Def.), (b) all Def., (c) all Thr., and (d) all Tuned. Grids (the first value is the default): Balanced Random Forest, max. features $\in \{\sqrt{d}, 0.3\}\times$ min. samples per leaf $\in \{1, 5\}$; EasyEnsemble, $\{10, 30\}$ members; RUSBoost, $\{200, 500\}$ rounds $\times$ learning rate $\{0.1, 0.5\}$; Self-Paced Ensemble, $\{10, 50\}$ members $\times$ $\{10, 5\}$ bins; LightGBM, 31 leaves and learning rate 0.05, 15 or 63 leaves or learning rate 0.1; weighted SVM, $C \in \{0.1, 1, 10, 100\} \times g \in \{0.25, 1, 4\}$ with $\gamma = g/d$. Random Forest, XGBoost and logistic regression: threshold only.

Method	TW			US			UK			Mean G-Mean Rank				Grid
	Def.	Thr.	Tuned	Def.	Thr.	Tuned	Def.	Thr.	Tuned	(a)	(b)	(c)	(d)	
RQS-Ensemble	0.848	0.840	0.840	0.639	0.637	0.607	0.753	0.718	0.700	4.00	2.80	5.40	8.13	$C \times g$ (12)
Balanced Random Forest	0.865	0.862	0.854	0.580	0.646	0.642	0.774	0.771	0.758	3.00	3.07	2.33	3.40	4
EasyEnsemble	0.863	0.857	0.856	0.615	0.618	0.616	0.765	0.768	0.756	2.13	2.47	4.13	4.80	2
RUSBoost	0.844	0.857	0.857	0.583	0.602	0.602	0.761	0.780	0.780	3.20	3.47	4.20	4.40	4
Self-Paced Ensemble	0.827	0.840	0.868	0.602	0.602	0.616	0.739	0.700	0.768	3.60	4.00	7.47	3.53	4
LightGBM	0.577	0.850	0.864	0.473	0.621	0.616	0.000	0.728	0.710	8.80	8.80	4.80	5.47	4
Random Forest	0.599	0.851	0.851	0.191	0.621	0.621	0.227	0.726	0.726	9.03	9.03	5.33	5.53	–
XGBoost	0.583	0.839	0.839	0.410	0.603	0.603	0.193	0.748	0.748	8.83	8.83	6.00	6.73	–
Weighted SVM	0.774	0.837	0.859	0.450	0.571	0.601	0.558	0.680	0.766	7.33	7.33	8.20	5.33	$C \times g$ (12)
Logistic Regression	0.837	0.833	0.833	0.530	0.599	0.599	0.681	0.698	0.698	5.07	5.20	7.13	7.67	–

Consequently, the significant advantages of RQS-Ensemble over Random Forest, XGBoost, LightGBM and the weighted SVM in Table 19 come from the default thresholds of these methods and disappear when they select their threshold in the same way. The conclusion that holds under every protocol is more modest: RQS-Ensemble obtains G-mean values of the same order as the balanced ensembles without outperforming them, and Balanced Random Forest is the most robust method, among the three best under every protocol and the best when thresholds are tuned. Some of the competitors (RUSBoost, Random Forest, XGBoost and logistic regression) always selected their default parameters; the selected configurations are given in the fold-level results deposited with the code.

Table 19. Pairwise comparison of RQS-Ensemble with each method over the 15 blocks. Δ: mean difference RQS-Ensemble minus method (positive: RQS-Ensemble better); W: number of blocks won by RQS-Ensemble; p_H: Wilcoxon signed-rank test with Holm correction over the 21 comparisons; p_N: Nemenyi test.

Method	G-Mean				AUC-ROC				PR-AUC			
	Δ	W	p_H	p_N	Δ	W	p_H	p_N	Δ	W	p_H	p_N
EasyEnsemble	−0.032	2	0.018	1.000	−0.015	3	0.181	0.992	−0.073	1	0.009	0.136
Balanced Random Forest	−0.024	5	0.606	1.000	−0.031	1	0.002	0.292	−0.063	2	0.022	0.020
RUSBoost	−0.014	6	1.000	1.000	−0.002	7	1.000	1.000	−0.027	8	1.000	1.000
Self-Paced Ensemble	−0.007	6	1.000	1.000	+0.009	8	1.000	1.000	+0.005	10	1.000	1.000
SVM-Undersampling	+0.044	7	1.000	1.000	+0.019	8	0.749	1.000	−0.021	3	0.833	0.999
Logistic Regression	+0.034	11	0.248	1.000	+0.039	11	0.181	0.898	−0.012	8	1.000	1.000
MLP + Focal loss	+0.031	11	0.535	1.000	+0.013	7	1.000	1.000	−0.041	3	0.226	0.586
RQS-SVM (tuned)	+0.036	14	0.038	1.000	+0.030	13	0.162	0.966	+0.005	9	1.000	1.000
RQS-SVM (raw boundary)	+0.083	13	0.034	0.775	+0.032	13	0.010	0.586	+0.002	8	1.000	1.000
SVM-ADASYN	+0.124	15	0.001	0.292	+0.042	13	0.005	0.184	−0.005	8	1.000	1.000
SVM-SMOTE	+0.127	15	0.001	0.226	+0.038	13	0.064	0.811	−0.014	5	1.000	1.000
Weighted SVM	+0.122	15	0.001	0.171	+0.026	13	0.094	0.992	−0.019	4	0.454	0.998
RQS-SVM k-means	+0.121	15	0.001	0.059	+0.033	13	0.005	0.541	+0.008	9	1.000	1.000
MLP + LDAM-DRW	+0.334	14	0.002	<0.001	+0.028	8	1.000	1.000	−0.034	3	0.212	0.827
MLP + CB-Focal loss	+0.290	15	0.001	<0.001	+0.015	6	1.000	1.000	−0.044	3	0.302	0.756
LightGBM (balanced)	+0.366	15	0.001	<0.001	−0.010	4	0.479	0.994	−0.080	3	0.185	0.041
XGBoost	+0.320	15	0.001	<0.001	+0.015	10	0.479	1.000	−0.033	4	0.332	0.979
Random Forest	+0.377	15	0.001	<0.001	−0.008	4	0.666	0.999	-0.044	4	0.212	0.775
DEC	+0.377	15	0.001	<0.001	+0.006	8	1.000	1.000	−0.016	7	1.000	1.000
KNN	+0.512	15	0.001	<0.001	+0.153	15	0.001	<0.001	+0.049	15	0.001	0.126
RQS-SVM only	+0.624	15	0.001	<0.001	+0.130	15	0.001	<0.001	+0.025	10	0.424	0.960

4.6. Computational Efficiency and Complexity Analysis

The cost of the single RQS-SVM consists of three parts: the exact nearest neighbor and nearest enemy searches of the rough approximation, $O(n^2 d)$ operations parallelized across cores; the PCM granulation of the boundary regions, $O(r\,T\,|B|\,c\,d)$ for r initializations, T iterations and c granules; and the SVM optimization on $|\mathcal{Q}|$ instead of n observations between $O(|\mathcal{Q}|^2 d)$ and $O(|\mathcal{Q}|^3)$. In the ablation study, the granulation reduces the training set by 14% to 37% (Section 4.3); on US it reduces the fitting time from 24.6 s (raw boundary) to 14.5 s, whereas the class-weighted SVM on the full training set takes 16.9 s.

RQS-Ensemble trains each member on $2n_{\min}$ observations (352 on TW, about 147 on UK and 2654 on US) in parallel. Without tuning, fitting takes on average 1.8 s, 13.4 s and 1.1 s per fold on TW, US and UK. The inner tuning multiplies this cost by the number of hyperparameter pairs and inner folds (12×3 plus the final fit) excluding the preparation steps that are shared: the tuned model takes 40.6 s, 299.7 s and 18.2 s per fold against 3.0 s, 5.7 s and 2.2 s for Balanced Random Forest and 4.6 s, 4.8 s and 2.3 s for EasyEnsemble. Since the fixed configuration gave better results than the tuned one (Section 4.4), the tuning, which accounts for most of this cost, can be omitted in practice. This cost remains the main practical limitation of the proposed method (Section 5).

4.7. Sensitivity to the Parameters

The parameters of RQS-SVM are either fixed or set by the data-driven rules of Section 3.8. To assess their influence, each parameter was varied over a grid while the others were kept at their data-driven values on the same five folds (US training parts reduced to 10,000 observations): for the single RQS-SVM, the neighborhood factor $\alpha \in \{0.05, 0.1, 0.2, 0.4, 0.8\}$, the boundary weight $\beta \in \{0.3, \ldots, 0.9\}$, the PCM fuzzifier $m \in \{1.5, \ldots, 3\}$, the minority factor $\lambda \in \{0, \ldots, 1\}$, the maximal number of granules $k_{\max} \in \{2, \ldots, 12\}$, the number

of PCM initializations $n_{\text{init}} \in \{1, 3, 5\}$ and the minimal proportion of boundary points $\rho \in \{0.01, 0.05, 0.1, 0.2\}$ of the calibration of Procedure 2; for RQS-Ensemble, the number of members $B \in \{5, 10, 20, 40\}$ and the ratio $r \in \{0.5, 1, 2, 3\}$ of majority to minority observations in each subsample. Figure 3 shows the mean G-mean for each value and Table 20 the range of the mean G-mean over each grid.

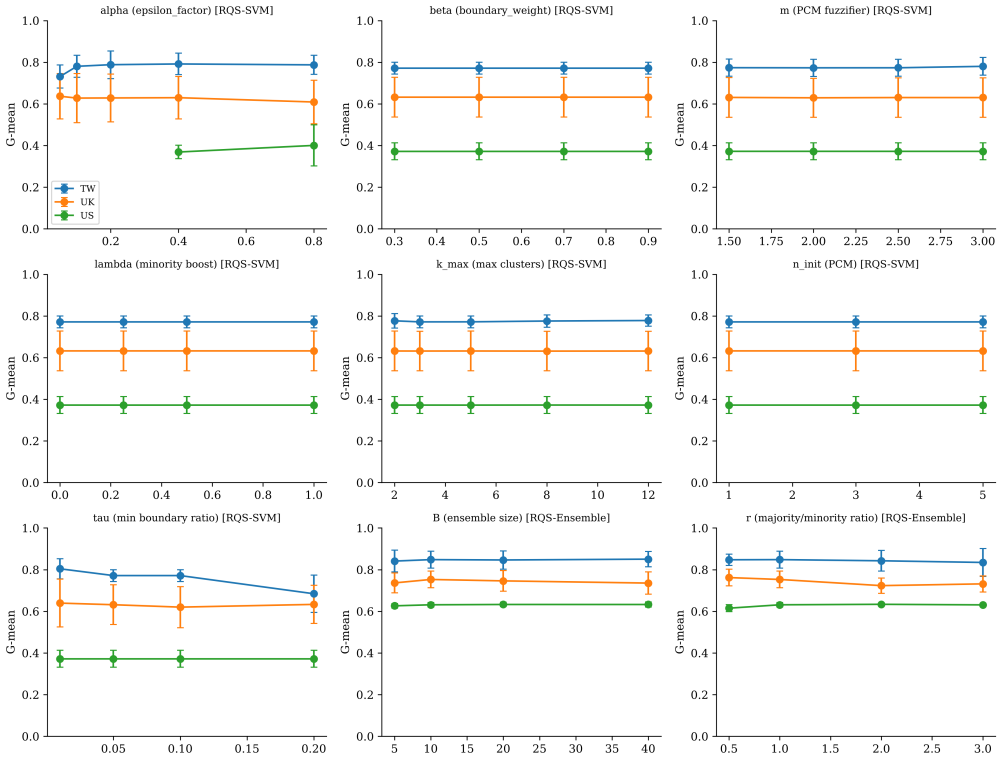


Figure 3. Sensitivity of the G-mean to each parameter (mean over the five folds; one panel per parameter, one curve per dataset).

Table 20. Range (maximum minus minimum) of the mean G-mean when one parameter varies over its grid, the others being fixed. * On US, $\alpha \in \{0.05, 0.1, 0.2\}$ could not be evaluated with the data-driven parameters (failure of the PCM initialization), so the range covers $\alpha \in \{0.4, 0.8\}$ only.

Model	Parameter	TW	US	UK
RQS-SVM	ρ (minimal proportion of boundary points)	0.120	0.000	0.019
	α (neighborhood factor)	0.061	0.032 *	0.028
	m (PCM fuzzifier)	0.007	0.000	0.002
	$k_{\max}$ (maximal number of granules)	0.006	0.000	0.000
	n_{init} (PCM initializations)	0.000	0.000	0.000
	β (boundary weight)	0.000	0.000	0.000
	λ (minority factor)	0.000	0.000	0.000
RQS-Ensemble	r (majority/minority ratio)	0.014	0.019	0.039
	B (number of members)	0.009	0.007	0.017

The method is insensitive to most of its parameters. As predicted by Remark 2 and the weight diagnostics of Table 3, λ and β have no effect at all: the class mass balancing cancels λ and the IQR clipping cancels β. The PCM parameters (m, $k_{\max}$, n_{init}) change the mean G-mean by less than 0.015, which is far below the standard deviation across folds (about 0.03 on TW and 0.10 on UK). RQS-Ensemble is stable with respect to B and r (ranges below 0.02 except r on UK, 0.039): five members are almost as good as forty, and $B = 10$, $r = 1$ are reasonable defaults.

The only influential parameters are those that determine the size of the boundary region. Requiring at least 20% of boundary points lowers the G-mean from 0.772 to 0.684 on TW, whereas $\rho = 0.01$ raises it to 0.804; on UK the effect of ρ is small and not monotone (G-mean between 0.620 and 0.639), and on US the calibration is inactive because the boundary already contains more than 20% of the points. A very small neighborhood factor ($\alpha = 0.05$) also lowers the G-mean on TW (0.731 against 0.788 for $\alpha = 0.2$). On TW, both results agree with the ablation study: the larger the part of the data that is replaced by a few prototypes, the more information is lost. A smaller boundary region, or a weaker compression of it, is therefore preferable, and the default $\rho = 0.05$ is a compromise between compression and accuracy. These results also give the sensitivity to the neighborhood radius ϵ, which is not a free parameter but the product of α and the median k-th nearest neighbor distance, possibly enlarged by the calibration with ρ: varying α and ρ over their grids varies ϵ and the size of the boundary region, and the G-mean changes by at most 0.061 (α) and 0.120 (ρ) on TW and by at most 0.032 on US (for $\alpha \in \{0.4, 0.8\}$ only) and 0.028 on UK (Table 20). The threshold τ was not varied because it is not active on our data (Section 3.4).

4.8. Leakage Checks and Temporal Validation

Cross-validation with random folds can overestimate performance when the same firm appears several times or the model is trained on observations that are later than the test observations. Table 7 summarizes the checks performed on the three datasets.

For US, which has a firm identifier and a fiscal year, we added a temporal validation: the models are trained on the years up to 2014 (stratified sample of 20,000 firm–years) and tested on all later firm–years (2015–2018, 12,282 observations, and 287 failures) and separately on the firm–years of firms that do not appear in the training years (1212 observations; 10 failures). The proposed method is tuned by the same nested procedure on the training years only. Table 21 reports the results.

Table 21. Temporal validation on US: training on 1999–2014 (stratified sample of 20,000 firm–years), test on 2015–2018. All firms: 12,282 firm–years, 287 failures. New firms only: 1212 firm–years of firms absent from the training years, 10 failures (results on this set are very uncertain). Methods sorted by G-mean on all firms; best value of each column in bold.

Method	All Firms (287 Failures)						New Firms Only (10 Failures)		
	G-Mean	Recall	Specificity	MCC	AUC-ROC	PR-AUC	G-Mean	AUC-ROC	PR-AUC
Balanced Random Forest	**0.787**	0.749	0.828	**0.224**	**0.853**	0.186	0.690	0.780	0.030
RUSBoost	0.733	0.714	0.753	0.161	0.774	0.064	**0.752**	0.815	0.023
Self-Paced Ensemble	0.718	0.735	0.701	0.142	0.782	0.075	0.667	0.719	0.019
EasyEnsemble	0.715	0.847	0.604	0.139	0.821	**0.194**	0.645	**0.901**	0.172
RQS-Ensemble (fixed C=1, g=1)	0.711	0.787	0.641	0.134	0.805	0.153	0.667	0.855	0.140
RQS-Ensemble (proposed)	0.708	0.808	0.621	0.133	0.800	0.115	0.663	0.837	**0.192**
RQS-SVM (tuned)	0.676	0.540	0.846	0.158	0.756	0.134	0.646	0.666	0.026
Logistic Regression	0.618	**0.927**	0.413	0.104	0.837	0.122	0.494	0.811	0.074
XGBoost	0.580	0.366	**0.921**	0.155	0.751	0.107	0.426	0.682	0.018
Weighted SVM	0.533	0.902	0.315	0.071	0.803	0.169	0.321	0.485	0.118

Table 7 also shows an important property of the US data: for all 8971 firms, the status is the same in every year of the firm (100%); i.e., the label is a firm-level attribute repeated over 8.8 years on average. Random folds would therefore let a model recognize a failed firm from its other years. This is why the firm-grouped cross-validation is the reference protocol of this study and why the first temporal test set, which contains firms already seen in training, probably gives optimistic values: all the methods obtain a higher G-mean on it (up to 0.787) than in the grouped cross-validation (at most 0.615). The test set restricted to

new firms is free of this effect, but its 10 failures make it very uncertain: one failure changes the recall by 0.1.

The ranking of the methods is nevertheless consistent with the grouped cross-validation. On all firms, Balanced Random Forest is the best method for G-mean, MCC and AUC-ROC; RUSBoost, Self-Paced Ensemble, EasyEnsemble and RQS-Ensemble form the next group (G-mean between 0.708 and 0.733); and the methods that do not rebalance the training sample (XGBoost) or rely on class weights alone (weighted SVM and logistic regression) have the lowest G-mean. RQS-Ensemble ranks sixth of the ten methods in G-mean with tuning and fifth with the fixed configuration, which again performs slightly better than the tuned one. On new firms, RQS-Ensemble ranks fifth in G-mean but has the highest PR-AUC and the third highest AUC-ROC; given the 10 failures, we do not draw conclusions from these differences.

4.9. Empirical Check of the Theoretical Results

Theorem 1 and Proposition 4 state that the effect of the compression is controlled by the fraction of boundary observations $|B|/n$, by the distortion D of the quotient map and by the Lipschitz constant $L \leq \sqrt{2\gamma}\|w\|_{\mathcal{H}}$ of the decision function. All these quantities can be computed from a trained model. Table 10 reports them for the single RQS-SVM on each dataset (mean over the five folds) with the observed difference of hinge risks between the original sample and the quotient representation, which Proposition 4 bounds, and the proportion of test decisions changed by the projection, which Theorem 1 bounds. It also reports the proportion of test observations whose decision is *certified* to be unchanged, i.e., for which $|f(\mathbf{x})| > L\,\|\mathbf{x} - Q(\mathbf{x})\|$.

Four observations follow from Table 10. First, the bound of Proposition 4 holds in all 15 folds, but it is loose: it exceeds the observed difference of hinge risks by a factor of 11 to 13 on TW, 13 to 30 on US and 16 to 427 on UK. The bound uses the worst case $L\|\mathbf{x} - Q(\mathbf{x})\|$ for every boundary observation, whereas the changes of loss of different observations are smaller and partly compensate. Second, the quantity that drives the effect of the compression is the distortion, not the size of the boundary alone. US has the largest boundary fraction (0.37) and the largest Lipschitz constant, but its boundary observations are concentrated and the distortion is tiny ($\widehat{D}_B \approx 7 \times 10^{-4}$); the projection changes only 2.0% of the test decisions and 80% of them are certified. TW and UK have smaller boundaries but much larger distortions because one to five prototypes replace several hundred boundary observations; the projection changes 4.9–5.1% of the decisions and only 15–25% are certified. This is the behavior described by Corollary 1 in which the boundary probability multiplies the boundary distortion. Third, these quantities are ordered as the losses observed in the ablation study. The observed risk difference is the largest on TW (0.047), then UK (0.008) and US (0.0015), and so is the disagreement caused by the projection; the loss of G-mean caused by replacing the raw boundary with the PCM prototypes follows the same order: 0.061 on TW, 0.038 on UK and 0.025 on US (Table 12). The theory thus explains why the compression is not free and on which datasets it costs the most, but its constants are too large to predict the size of the loss. Fourth, Theorem 1 is stated with the mean distortion $D_Q = \mathbb{E}\|X - Q(X)\|^2$. On the standardized TW ratios, a few extreme values make the empirical mean of $\|\mathbf{x} - Q(\mathbf{x})\|^2$ on the test folds range from 19 to 10^{19}, while its median is about 9.5: with heavy-tailed ratios the mean distortion bound is uninformative. This is one more reason for the quantile transformation used in RQS-Ensemble, which maps every ratio to a bounded range. Finally, the proportion of certified decisions is a conservative lower bound on the proportion of unchanged decisions: on TW, 15% of the decisions are certified but 95% are unchanged. In summary, Theorem 1 and Proposition 4 should be read as explanatory results, not quantitative guarantees: they

identify the quantities that govern the cost of the compression (the boundary fraction and the distortion) and predict the direction and the ordering of the effects across datasets, but their numerical values are one to two orders of magnitude above the observed effects and cannot be used to certify the accuracy of a trained model. The abstract and the contributions are worded accordingly.

4.10. Statistical Analysis

This section tests whether the differences observed in Table 14 are statistically significant using the procedures described in Section 4.2. The analysis is performed on the G-mean (main criterion) and repeated for MCC, AUC-ROC and PR-AUC; it is also repeated on the subset of the proposed method and the recent methods (weighted SVM, Balanced Random Forest, EasyEnsemble, RUSBoost, Self-Paced Ensemble, LightGBM and the three MLPs) so that the conclusions do not depend on the presence of weak baselines.

Table 22 reports the Friedman and Iman–Davenport tests and Table 15 the average ranks.

Table 22. Friedman and Iman–Davenport tests over the 15 fold×dataset blocks for the 22 methods.

Metric	χ_F^2	p	F_F	(df_1, df_2)	p
G-mean	251.45	2.1×10^{-41}	55.39	(21, 294)	3.7×10^{-89}
MCC	126.65	4.3×10^{-17}	9.41	(21, 294)	1.4×10^{-22}
F1-score	108.18	1.0×10^{-13}	7.32	(21, 294)	2.9×10^{-17}
AUC-ROC	175.10	2.7×10^{-26}	17.52	(21, 294)	2.9×10^{-40}
PR-AUC	142.78	4.1×10^{-20}	11.61	(21, 294)	8.0×10^{-28}

The Friedman and Iman–Davenport tests reject the hypothesis that all the methods are equivalent for the five metrics ($p < 10^{-12}$ in all cases; for the G-mean, $\chi_F^2 = 251.45$ and $F_F = 55.39$ with $(21, 294)$ degrees of freedom). For the G-mean, RQS-Ensemble has the fifth best average rank (5.07), after EasyEnsemble (2.47), Balanced Random Forest (3.73), RUSBoost (4.07) and Self-Paced Ensemble (4.93). Restricted to the proposed method and the ten recent methods, the conclusions are the same ($\chi_F^2 = 109.70$, $F_F = 38.11$ with $(10, 140)$ degrees of freedom, $p < 10^{-17}$; RQS-Ensemble ranks fifth, average rank 4.13, after EasyEnsemble, Balanced Random Forest, RUSBoost and Self-Paced Ensemble).

Figure 4 shows the Nemenyi critical difference diagram: methods joined by a bar are not significantly different at the 5% level.

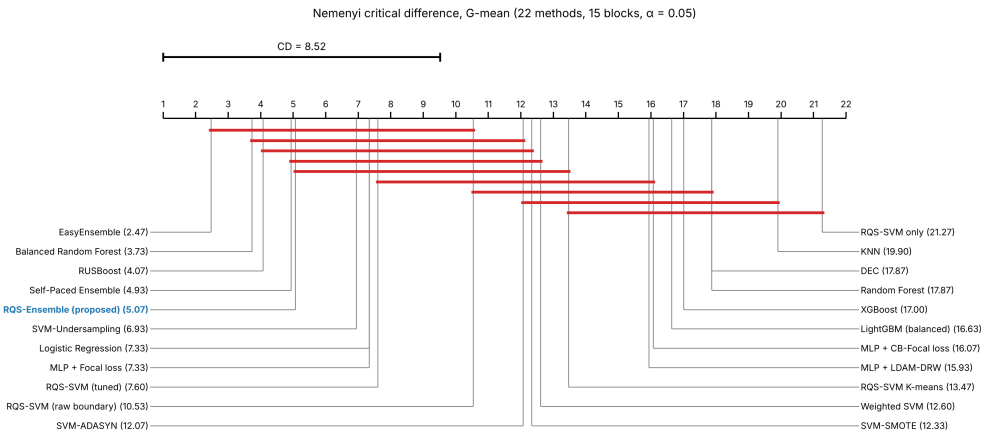


Figure 4. Nemenyi critical difference diagram for the G-mean (15 fold×dataset blocks, $\alpha = 0.05$).

With the Nemenyi test (critical difference 8.52 ranks), RQS-Ensemble is significantly better than RQS-SVM only, DEC, Random Forest, KNN, XGBoost, LightGBM and the MLPs with the class-balanced focal and LDAM-DRW losses, and no method is significantly better than it. The Wilcoxon tests with Holm correction (Table 19) are more powerful: for the G-mean, RQS-Ensemble is significantly better than 14 methods (the eight methods above, weighted SVM, SVM-SMOTE, SVM-ADASYN, RQS-SVM k-means, RQS-SVM with raw boundary and the tuned single RQS-SVM) with their default settings; several of these advantages disappear when the competitors tune their threshold (Section 4.5) and significantly worse than EasyEnsemble ($\Delta = -0.032$, $p_{\mathrm{H}} = 0.018$); the differences with Balanced Random Forest, RUSBoost, Self-Paced Ensemble, SVM-Undersampling, logistic regression and the MLP with focal loss are not significant. For AUC-ROC, RQS-Ensemble is significantly worse than Balanced Random Forest, and for PR-AUC it is significantly worse than Balanced Random Forest and EasyEnsemble.

The corrected resampled t-tests, which take into account the dependence between folds of the same dataset, are reported in Table A6 (Appendix B).

With only five folds and the variance correction, these tests are conservative. On each dataset, no method is significantly better than RQS-Ensemble. RQS-Ensemble is significantly better than Random Forest, XGBoost, LightGBM, KNN and RQS-SVM only on TW; 13 methods on US, including SVM-SMOTE, SVM-ADASYN, weighted SVM, LightGBM, XGBoost and the two single-model RQS variants; and LightGBM, DEC, KNN and RQS-SVM only on UK. The differences with the undersampling ensembles are not significant on any dataset.

The statistical analysis does not support a claim that RQS-Ensemble is better than the best recent imbalance learning ensembles: it is significantly worse than EasyEnsemble in G-mean over the 15 blocks, and it is not significantly different from Balanced Random Forest, RUSBoost and Self-Paced Ensemble (Wilcoxon signed-rank tests with Holm correction over the 21 comparisons, Table 19; a non-significant difference does not establish equivalence). It does support the conclusion that RQS-Ensemble belongs to this leading group in G-mean, that it significantly improves on the SVM-based methods (weighted SVM, SVM-SMOTE, SVM-ADASYN, and DEC) and the single RQS-SVM without class mass balancing, and that its weak point is the ranking quality of its scores.

5. Limitations

The results of this study must be interpreted with the following limitations in mind.

- **Projection-based inference.** The main results use the projection of test observations onto the representatives, which was fixed before the experiments. This choice lowers the reported performance: for the single model, direct evaluation is significantly better for G-mean, AUC-ROC and PR-AUC (in 15, 15 and 14 of the 15 blocks, Table 4); for RQS-Ensemble, it raises the PR-AUC by 0.042 on average and would move its fixed configuration from the tenth PR-AUC rank to the third PR-AUC rank of the 22 methods (Section 4.4). The PR-AUC and AUC-ROC of RQS-Ensemble in Table 14 therefore underestimate what the same models achieve with direct evaluation.
- **Asymmetric tuning.** In the main comparison (Table 14), the hyperparameters and the threshold of the proposed method are selected by inner cross-validation, whereas the competitors use the default settings. Section 4.5 shows that this asymmetry matters: with the same threshold selection for all the methods, RQS-Ensemble is sixth of ten in G-mean and significantly worse than Balanced Random Forest, and with full tuning for all the methods it is last. The ranks and significance tests of Tables 14 and 19 must be read with this protocol in mind; they do not show that RQS-Ensemble is better than tuned competitors.

- **Weights.** The IQR clipping cancels the granule weights, so the single model is, in practice, a class-weighted SVM on the quotient representation, and the strong granulation of the small minority boundary (one or two prototypes per fold on TW and UK) discards geometric information.
- **Components without significant gain.** The ablation study shows that the PCM granulation and the rough guidance of the subsamples do not significantly improve the predictive performance and that a fuzzy–rough partition or the raw boundary can be slightly better. Their contribution is the compression of the training set and the quotient representation.
- **Computational cost.** The rough approximation requires exact nearest neighbor searches, and RQS-Ensemble with inner tuning trains B members for each of the 12 hyperparameter pairs and three inner folds. The method is therefore slower than tree ensembles such as Balanced Random Forest or LightGBM (Section 4.6), and the US training sets were limited to 20,000 observations. It is designed for periodic batch scoring rather than real-time use.
- **Possible repeated firms in TW and UK.** TW and UK have no firm identifier and no date. If they contain several yearly observations of the same firm, these observations may be split between training and test parts, and our cross-validation cannot prevent it; the absence of identical rows (Table 7) does not exclude near-identical observations of the same firm in different years. On US, where this can be checked, the failure status is constant over the years of every firm, and random folds would let a model recognize a failed firm from its other years. The TW and UK results are therefore not protected against firm-level leakage and may be optimistic for all the methods; only the US results come from firm-grouped folds and a temporal validation.
- **Data.** The evaluation uses three datasets, one of which (UK) is proprietary; only US could be validated over time (Section 4.8). The US variables have weak discriminating power for all the methods.
- **Theory and calibration.** Theorem 1 and Proposition 4 bound the effect of the compression for a given decision function; they do not bound the estimation error of the SVM trained on the quotient representation, nor the effect of the cost-sensitive weights, and their constants are evaluated empirically in Section 4.9: the bounds hold in every fold but exceed the observed effects by a factor of 11 to 427, and the mean distortion form of Theorem 1 is uninformative on heavy-tailed ratios. They explain the direction of the effects, not their size. The scores are not calibrated probabilities: after Platt recalibration on the training part they are well-calibrated, but, on US and UK, their Brier score does not improve on the base rate (Table 17).

6. Conclusions

This study introduced the rough quotient space support vector machine (RQS-SVM), a unified and modular framework designed to address bankruptcy prediction under severe class imbalance. Rather than proposing new standalone algorithms, the framework integrates established concepts from rough set theory, quotient space theory, possibilistic clustering, class mass balancing, and support vector machines within a problem-specific classification architecture. The main methodological contribution lies in coordinating these components to characterize uncertainty, compress redundant majority class information, and improve the representation of minority and boundary samples before classification.

On the theoretical side, we showed that Pawlak approximations are quotient operators and that RQS-SVM defines a genuine equivalence relation on the training sample that is finer than both the decision classes and the rough partition (Propositions 2–3); the effect of

the compression on the decision and empirical risk is controlled by the quotient distortion, which is restricted to the boundary region (Theorem 1 and Proposition 4).

On the experimental side, the evaluation (five-fold cross-validation, firm-grouped folds for the US panel, 22 methods, and non-parametric tests) leads to the following conclusions. First, the class mass balancing step is what enables the single model to detect failures. Second, the partition into certain and uncertain observations avoids the collapse that is observed on US when PCM is applied to the whole dataset, and the PCM granulation compresses the training set by 14% to 37% without significantly changing the accuracy of RQS-Ensemble. Third, the quantile transformation and the ensemble of RQS-SVM members trained on balanced subsamples significantly improve the results on the three datasets, particularly for the heavy-tailed US data. Among the 22 methods, RQS-Ensemble belongs to the leading group in G-mean with EasyEnsemble, Balanced Random Forest, RUSBoost and Self-Paced Ensemble; it is significantly better than the SVM-based methods and gradient boosting with the default threshold but significantly worse than EasyEnsemble, and its advantages over those competitors whose threshold ignores the imbalance disappear when every method tunes its threshold (Section 4.5), and its PR-AUC is lower than that of tree ensembles; with direct evaluation instead of projection, the PR-AUC rank of its fixed configuration would rise from tenth to third. Finally, the inner tuning of the threshold was counterproductive with so few failures, and the fixed configuration performed better.

The main limitations are the higher computational cost of RQS-Ensemble compared with tree ensembles, the fact that only the proposed method was tuned, and the evaluation on three datasets, of which only US could be validated over time (Section 5). Future work will (i) replace the crisp neighborhood rough partition with a fuzzy–rough partition, which gave slightly better, although not significant, results in the ablation; (ii) use the quotient representation to accelerate tuning since the rough partition does not depend on the SVM hyperparameters; (iii) evaluate the method with temporal splits and on additional datasets; and (iv) use the granules of the boundary region to explain individual predictions.

From a practical perspective, the proposed RQS-SVM framework can support financial institutions and investors in identifying financially distressed firms at an early stage. Reducing false negatives may help to mitigate inappropriate lending and investment decisions, while controlling false positives can limit unnecessary credit restrictions and monitoring. Therefore, RQS-SVM can serve as an effective early warning and decision support tool, while final financial decisions should remain subject to professional judgment and complementary financial information. Therefore, although RQS-SVM can support timely early warning decisions, its current implementation is primarily designed for batch analysis rather than strict real-time deployment.

Author Contributions: Conceptualization, A.T. and A.M.; methodology, K.J.; software, A.T.; validation, A.T., M.T., A.M. and K.J.; formal analysis, A.T., M.T.; investigation, A.T., M.T., A.M. and K.J.; resources, A.T.; data curation, A.T.; writing—original draft preparation, A.T., M.T., A.M. and K.J.; writing—review and editing, A.T., M.T., A.M. and K.J.; visualization, A.T., M.T., A.M. and K.J.; supervision, K.J.; project administration, K.J.; funding acquisition, A.T. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding

Data Availability Statement: The TW and US datasets are publicly available at the Kaggle addresses given in Section 4.1. The UK dataset is proprietary and cannot be shared. The code, which fixes all random seeds (Section 4.2), the fold indices and the outputs needed to reproduce all tables and figures are available at Zenodo, https://doi.org/10.5281/zenodo.24355667.

Conflicts of Interest: The authors declare no conflicts of interest.

Appendix A. Configurations of the Compared Methods

Table A1. Configurations of the compared methods. All methods receive the same training parts. Unless stated otherwise, library defaults are used (scikit-learn 1.9.1, imbalanced-learn 0.14.2, XGBoost 3.4.1, and LightGBM 4.7.0) with the random seed 42. Except for RQS-Ensemble and RQS-SVM (tuned), the decision threshold is the default one: 0.5 on the predicted probability or 0 on the SVM decision function. The grids used in the tuning sensitivity analysis are given in Table 18.

Method	Configuration
RQS-Ensemble (proposed)	Quantile transformation; $B = 10$; data-driven α, β, m; PCM, class mass balancing; $C \in \{0.1, 1, 10, 100\}$, $\gamma = g/d$ with $g \in \{0.25, 1, 4\}$ and threshold selected by inner 3-fold CV (G-mean)
RQS-SVM (tuned)	Single RQS-SVM; same inner selection of C, γ and threshold
RQS-SVM only	Quotient representation without PCM and without quotient weights; RBF SVM, $C = 1$, γ = scale
RQS-SVM k-means	k-means granulation, class mass balancing; $C = 1$, γ = scale
RQS-SVM (raw boundary)	No granulation, class mass balancing; $C = 1$, γ = scale
Weighted SVM	RBF SVM, class weights inversely proportional to class frequencies
Balanced Random Forest	300 trees, balanced bootstrap per tree, all classes resampled with replacement, $\sqrt{d}$ features per split, one sample per leaf
EasyEnsemble	10 AdaBoost learners on random undersamples (imbalanced-learn default AdaBoost learner)
RUSBoost	200 rounds, depth-3 trees, learning rate 0.1
Self-Paced Ensemble	10 members, 10 hardness bins, fully grown decision trees
LightGBM	400 trees, learning rate 0.05, 31 leaves, class weights balanced, subsampling 0.8
MLP + Focal/CB-Focal/LDAM-DRW	Two hidden layers (128 and 64) with batch normalization, ReLU and dropout 0.2; Adam (learning rate 10^{-3}, weight decay 10^{-4}), batch size 512, at most 100 epochs with early stopping on the AUC of a stratified 15% validation split (patience 10); focal $\gamma = 2$, class-balanced $\beta = 0.999$; LDAM margins $\propto n_j^{-1/4}$ with reweighting from epoch 50
DEC	RBF SVM ($C = 1$, γ = scale) with class weights 1 (failed) and 0.1 (non-failed)
SVM-SMOTE/-ADASYN/-Undersampling	RBF SVM ($C = 1$, γ = scale) after SMOTE, ADASYN ($k = 5$) or random undersampling of the training part
Random Forest	100 trees, balanced class weights
KNN	$k = 5$
Logistic Regression	Balanced class weights, at most 1000 iterations
XGBoost	Default parameters, `scale_pos_weight` = ratio of non-failed to failed firms

Appendix B. Detailed Results of the Comparison

Tables A2–A4 give the eight metrics of the 22 methods on each dataset (mean $\pm$ standard deviation over the five test folds). Table A5 gives the confusion matrices summed over the five test folds with the bankrupt class as the positive class; all metrics were recomputed from the saved predictions. Table A6 gives the per-dataset corrected resampled t-tests, and Table A7 presents the 95% confidence intervals and effect sizes.

Table A2. Performance on the Taiwan (TW) dataset: mean ± standard deviation over the five test folds. Best value of each column (except accuracy) in bold. RQS-Ensemble and RQS-SVM (tuned) select C, γ and the threshold by inner cross-validation; the other methods use default settings.

Method	Accuracy	Precision	Recall	Specificity	F1-Score	G-Mean	MCC	AUC-ROC	PR-AUC
RQS-Ensemble (proposed)	0.831 ± 0.019	0.144 ± 0.009	0.855 ± 0.110	0.830 ± 0.023	0.246 ± 0.015	0.840 ± 0.046	0.307 ± 0.030	0.914 ± 0.029	0.300 ± 0.015
RQS-SVM (tuned)	0.803 ± 0.032	0.126 ± 0.013	0.845 ± 0.090	0.801 ± 0.035	0.218 ± 0.018	0.821 ± 0.033	0.277 ± 0.021	0.907 ± 0.019	0.323 ± 0.042
RQS-SVM only	0.968 ± 0.001	0.549 ± 0.126	0.077 ± 0.020	**0.998** ± 0.001	0.135 ± 0.035	0.275 ± 0.039	0.197 ± 0.049	0.842 ± 0.044	0.302 ± 0.041
RQS-SVM k-means	0.888 ± 0.010	0.176 ± 0.020	0.668 ± 0.081	0.895 ± 0.010	0.278 ± 0.030	0.772 ± 0.046	0.303 ± 0.040	0.900 ± 0.026	0.302 ± 0.052
RQS-SVM (raw boundary)	0.842 ± 0.007	0.148 ± 0.013	0.823 ± 0.081	0.842 ± 0.007	0.251 ± 0.023	0.832 ± 0.042	0.306 ± 0.036	0.909 ± 0.022	0.313 ± 0.060
Weighted SVM	0.895 ± 0.010	0.186 ± 0.009	0.668 ± 0.101	0.903 ± 0.014	0.291 ± 0.016	0.775 ± 0.053	0.315 ± 0.032	0.914 ± 0.016	0.328 ± 0.040
Balanced Random Forest	0.893 ± 0.012	0.211 ± 0.015	0.841 ± 0.119	0.895 ± 0.016	0.336 ± 0.023	**0.865** ± 0.056	0.388 ± 0.039	0.940 ± 0.023	0.431 ± 0.051
EasyEnsemble	0.855 ± 0.015	0.167 ± 0.009	**0.873** ± 0.076	0.855 ± 0.018	0.281 ± 0.010	0.863 ± 0.030	0.344 ± 0.016	0.939 ± 0.019	0.441 ± 0.065
RUSBoost	0.893 ± 0.011	0.203 ± 0.016	0.800 ± 0.123	0.896 ± 0.014	0.324 ± 0.028	0.844 ± 0.062	0.369 ± 0.046	0.937 ± 0.020	0.398 ± 0.064
Self-Paced Ensemble	0.899 ± 0.014	0.209 ± 0.022	0.759 ± 0.089	0.903 ± 0.016	0.327 ± 0.027	0.827 ± 0.044	0.364 ± 0.034	0.918 ± 0.030	0.317 ± 0.069
LightGBM (balanced)	0.972 ± 0.005	**0.655** ± 0.158	0.336 ± 0.049	0.993 ± 0.005	**0.436** ± 0.054	0.577 ± 0.042	**0.452** ± 0.063	**0.943** ± 0.020	**0.513** ± 0.075
MLP + Focal loss	0.849 ± 0.010	0.155 ± 0.002	0.827 ± 0.073	0.850 ± 0.013	0.261 ± 0.006	0.838 ± 0.031	0.317 ± 0.018	0.922 ± 0.017	0.394 ± 0.049
MLP + CB-Focal loss	0.943 ± 0.007	0.308 ± 0.032	0.609 ± 0.019	0.954 ± 0.008	0.408 ± 0.027	0.762 ± 0.010	0.407 ± 0.023	0.923 ± 0.014	0.405 ± 0.049
MLP + LDAM-DRW	0.927 ± 0.024	0.284 ± 0.099	0.650 ± 0.192	0.936 ± 0.032	0.369 ± 0.031	0.771 ± 0.107	0.386 ± 0.033	0.909 ± 0.025	0.381 ± 0.036
DEC	0.936 ± 0.007	0.272 ± 0.024	0.577 ± 0.092	0.948 ± 0.009	0.368 ± 0.032	0.738 ± 0.056	0.367 ± 0.040	0.923 ± 0.025	0.349 ± 0.048
SVM-SMOTE	0.892 ± 0.006	0.180 ± 0.011	0.664 ± 0.114	0.900 ± 0.010	0.282 ± 0.024	0.770 ± 0.062	0.307 ± 0.042	0.912 ± 0.016	0.321 ± 0.053
SVM-ADASYN	0.896 ± 0.010	0.184 ± 0.006	0.650 ± 0.089	0.904 ± 0.013	0.287 ± 0.011	0.765 ± 0.048	0.308 ± 0.025	0.907 ± 0.014	0.295 ± 0.040
SVM-Undersampling	0.845 ± 0.010	0.157 ± 0.014	0.868 ± 0.057	0.844 ± 0.010	0.266 ± 0.022	0.856 ± 0.031	0.329 ± 0.031	0.924 ± 0.021	0.350 ± 0.048
Random Forest	0.966 ± 0.005	0.492 ± 0.106	0.364 ± 0.016	0.987 ± 0.006	0.414 ± 0.037	0.599 ± 0.013	0.404 ± 0.047	0.932 ± 0.023	0.411 ± 0.060
KNN	0.969 ± 0.002	0.553 ± 0.090	0.200 ± 0.047	0.995 ± 0.002	0.292 ± 0.056	0.444 ± 0.051	0.319 ± 0.056	0.784 ± 0.040	0.253 ± 0.046
Logistic Regression	0.870 ± 0.013	0.174 ± 0.011	0.805 ± 0.067	0.872 ± 0.015	0.285 ± 0.015	0.837 ± 0.030	0.335 ± 0.020	0.893 ± 0.037	0.354 ± 0.075
XGBoost	0.965 ± 0.004	0.458 ± 0.082	0.345 ± 0.037	0.986 ± 0.005	0.391 ± 0.040	0.583 ± 0.031	0.379 ± 0.045	0.918 ± 0.021	0.383 ± 0.046

Table A3. Performance on the US Company Bankruptcy (US) dataset: mean ± standard deviation over the five test folds. Best value of each column (except accuracy) in bold. RQS-Ensemble and RQS-SVM (tuned) select C, γ and the threshold by inner cross-validation; the other methods use default settings.

Method	Accuracy	Precision	Recall	Specificity	F1-Score	G-Mean	MCC	AUC-ROC	PR-AUC
RQS-Ensemble (proposed)	0.666 ± 0.025	0.107 ± 0.005	0.548 ± 0.032	0.674 ± 0.028	0.179 ± 0.007	0.607 ± 0.012	0.117 ± 0.011	0.661 ± 0.012	0.115 ± 0.009
RQS-SVM (tuned)	0.503 ± 0.036	0.090 ± 0.005	0.711 ± 0.068	0.488 ± 0.042	0.159 ± 0.008	0.587 ± 0.017	0.099 ± 0.021	0.651 ± 0.034	0.122 ± 0.014
RQS-SVM only	0.933 ± 0.000	0.000 ± 0.000	0.000 ± 0.000	**1.000** ± 0.000	0.000 ± 0.000	0.000 ± 0.000	-0.003 ± 0.003	0.557 ± 0.026	0.085 ± 0.007
RQS-SVM k-means	0.201 ± 0.024	0.073 ± 0.003	**0.944** ± 0.029	0.149 ± 0.026	0.136 ± 0.005	0.373 ± 0.031	0.066 ± 0.024	0.621 ± 0.031	0.112 ± 0.018
RQS-SVM (raw boundary)	0.221 ± 0.017	0.074 ± 0.003	0.931 ± 0.026	0.170 ± 0.018	0.137 ± 0.005	0.398 ± 0.022	0.068 ± 0.022	0.611 ± 0.026	0.111 ± 0.015
Weighted SVM	0.269 ± 0.019	0.076 ± 0.003	0.904 ± 0.031	0.224 ± 0.020	0.141 ± 0.006	0.450 ± 0.021	0.077 ± 0.022	0.632 ± 0.030	0.126 ± 0.018
Balanced Random Forest	0.808 ± 0.011	0.150 ± 0.007	0.403 ± 0.024	0.837 ± 0.013	**0.218** ± 0.009	0.580 ± 0.015	**0.156** ± 0.011	**0.700** ± 0.013	**0.142** ± 0.011
EasyEnsemble	0.602 ± 0.034	0.101 ± 0.003	0.634 ± 0.050	0.600 ± 0.040	0.175 ± 0.005	**0.615** ± 0.009	0.118 ± 0.009	0.658 ± 0.016	0.130 ± 0.006
RUSBoost	0.707 ± 0.095	0.114 ± 0.011	0.485 ± 0.119	0.722 ± 0.110	0.182 ± 0.010	0.583 ± 0.015	0.117 ± 0.008	0.625 ± 0.026	0.100 ± 0.019
Self-Paced Ensemble	0.693 ± 0.012	0.110 ± 0.004	0.515 ± 0.022	0.705 ± 0.013	0.182 ± 0.006	0.602 ± 0.011	0.118 ± 0.010	0.653 ± 0.015	0.109 ± 0.006
LightGBM (balanced)	0.865 ± 0.005	0.161 ± 0.014	0.247 ± 0.026	0.909 ± 0.005	0.195 ± 0.018	0.473 ± 0.025	0.128 ± 0.020	0.671 ± 0.012	0.138 ± 0.009
MLP + Focal loss	0.769 ± 0.137	0.134 ± 0.024	0.395 ± 0.185	0.795 ± 0.160	0.190 ± 0.020	0.536 ± 0.060	0.125 ± 0.018	0.681 ± 0.023	0.138 ± 0.011
MLP + CB-Focal loss	0.933 ± 0.001	**0.301** ± 0.336	0.004 ± 0.003	0.999 ± 0.001	0.007 ± 0.005	0.053 ± 0.033	0.023 ± 0.027	0.680 ± 0.011	0.141 ± 0.018
MLP + LDAM-DRW	0.933 ± 0.001	0.185 ± 0.250	0.002 ± 0.002	0.999 ± 0.001	0.004 ± 0.004	0.033 ± 0.032	0.013 ± 0.017	0.677 ± 0.022	0.137 ± 0.010
DEC	0.924 ± 0.003	0.173 ± 0.076	0.038 ± 0.022	0.987 ± 0.004	0.061 ± 0.033	0.187 ± 0.051	0.051 ± 0.039	0.641 ± 0.017	0.109 ± 0.007
SVM-SMOTE	0.303 ± 0.018	0.078 ± 0.003	0.872 ± 0.029	0.263 ± 0.019	0.142 ± 0.006	0.478 ± 0.019	0.077 ± 0.020	0.631 ± 0.030	0.125 ± 0.018
SVM-ADASYN	0.300 ± 0.013	0.077 ± 0.003	0.872 ± 0.030	0.259 ± 0.013	0.142 ± 0.006	0.475 ± 0.016	0.075 ± 0.021	0.621 ± 0.030	0.121 ± 0.020
SVM-Undersampling	0.239 ± 0.039	0.075 ± 0.004	0.918 ± 0.029	0.191 ± 0.042	0.138 ± 0.007	0.417 ± 0.042	0.070 ± 0.024	0.625 ± 0.030	0.120 ± 0.015
Random Forest	0.928 ± 0.002	0.241 ± 0.059	0.037 ± 0.005	0.991 ± 0.002	0.064 ± 0.008	0.191 ± 0.012	0.070 ± 0.018	0.684 ± 0.014	0.133 ± 0.011
KNN	0.925 ± 0.002	0.148 ± 0.027	0.028 ± 0.004	0.988 ± 0.002	0.047 ± 0.006	0.167 ± 0.011	0.037 ± 0.011	0.582 ± 0.016	0.086 ± 0.005
Logistic Regression	0.371 ± 0.027	0.082 ± 0.003	0.831 ± 0.034	0.338 ± 0.031	0.149 ± 0.005	0.530 ± 0.019	0.090 ± 0.014	0.661 ± 0.029	0.132 ± 0.021
XGBoost	0.879 ± 0.005	0.153 ± 0.017	0.182 ± 0.024	0.928 ± 0.006	0.166 ± 0.019	0.410 ± 0.026	0.102 ± 0.020	0.647 ± 0.013	0.120 ± 0.009

Table A4. Performance on the United Kingdom (UK) dataset: mean ± standard deviation over the five test folds. Best value of each column (except accuracy) in bold. RQS-Ensemble and RQS-SVM (tuned) select C, γ and the threshold by inner cross-validation; the other methods use default settings.

Method	Accuracy	Precision	Recall	Specificity	F1-Score	G-Mean	MCC	AUC-ROC	PR-AUC
RQS-Ensemble (proposed)	0.842 ± 0.053	0.071 ± 0.013	0.588 ± 0.122	0.847 ± 0.056	0.125 ± 0.019	0.700 ± 0.058	0.162 ± 0.020	0.834 ± 0.031	0.102 ± 0.022
RQS-SVM (tuned)	0.580 ± 0.236	0.038 ± 0.012	0.773 ± 0.241	0.576 ± 0.244	0.071 ± 0.022	0.631 ± 0.105	0.105 ± 0.041	0.760 ± 0.075	0.056 ± 0.025
RQS-SVM only	0.981 ± 0.002	0.000 ± 0.000	0.000 ± 0.000	0.999 ± 0.001	0.000 ± 0.000	0.000 ± 0.000	-0.003 ± 0.004	0.622 ± 0.088	0.056 ± 0.033
RQS-SVM k-means	0.824 ± 0.025	0.052 ± 0.012	0.502 ± 0.165	0.830 ± 0.028	0.094 ± 0.022	0.638 ± 0.099	0.116 ± 0.047	0.789 ± 0.071	0.079 ± 0.025
RQS-SVM (raw boundary)	0.800 ± 0.030	0.051 ± 0.010	0.568 ± 0.168	0.805 ± 0.033	0.094 ± 0.019	0.669 ± 0.090	0.124 ± 0.045	0.795 ± 0.065	0.086 ± 0.033
Weighted SVM	0.915 ± 0.008	0.079 ± 0.013	0.338 ± 0.053	0.926 ± 0.008	0.128 ± 0.020	0.558 ± 0.045	0.131 ± 0.026	0.786 ± 0.033	0.120 ± 0.050
Balanced Random Forest	0.848 ± 0.017	0.082 ± 0.008	0.708 ± 0.075	0.850 ± 0.018	**0.146** ± 0.014	**0.774** ± 0.040	**0.205** ± 0.023	**0.860** ± 0.048	0.131 ± 0.053
EasyEnsemble	0.724 ± 0.035	0.052 ± 0.003	**0.818** ± 0.147	0.722 ± 0.039	0.098 ± 0.006	0.765 ± 0.050	0.160 ± 0.026	0.856 ± 0.051	**0.163** ± 0.063
RUSBoost	0.770 ± 0.070	0.062 ± 0.015	0.763 ± 0.143	0.770 ± 0.074	0.113 ± 0.024	0.761 ± 0.049	0.172 ± 0.030	0.854 ± 0.036	0.100 ± 0.017
Self-Paced Ensemble	0.798 ± 0.021	0.060 ± 0.008	0.687 ± 0.121	0.800 ± 0.023	0.111 ± 0.015	0.739 ± 0.062	0.161 ± 0.034	0.809 ± 0.053	0.074 ± 0.028
LightGBM (balanced)	0.981 ± 0.001	0.000 ± 0.000	0.000 ± 0.000	**0.999** ± 0.001	0.000 ± 0.000	0.000 ± 0.000	-0.003 ± 0.003	0.826 ± 0.049	0.106 ± 0.036
MLP + Focal loss	0.859 ± 0.019	0.069 ± 0.013	0.547 ± 0.170	0.865 ± 0.022	0.123 ± 0.023	0.680 ± 0.103	0.156 ± 0.049	0.767 ± 0.060	0.108 ± 0.060
MLP + CB-Focal loss	0.944 ± 0.007	0.093 ± 0.042	0.242 ± 0.130	0.958 ± 0.008	0.133 ± 0.063	0.462 ± 0.148	0.124 ± 0.074	0.761 ± 0.054	0.103 ± 0.059
MLP + LDAM-DRW	0.954 ± 0.025	0.067 ± 0.065	0.208 ± 0.212	0.968 ± 0.029	0.100 ± 0.098	0.340 ± 0.319	0.100 ± 0.106	0.738 ± 0.037	0.102 ± 0.036
DEC	0.976 ± 0.004	0.065 ± 0.093	0.022 ± 0.030	0.994 ± 0.004	0.032 ± 0.044	0.093 ± 0.127	0.027 ± 0.052	0.828 ± 0.061	0.106 ± 0.041
SVM-SMOTE	0.911 ± 0.006	0.066 ± 0.010	0.294 ± 0.052	0.922 ± 0.007	0.108 ± 0.016	0.519 ± 0.047	0.106 ± 0.022	0.753 ± 0.016	0.114 ± 0.049
SVM-ADASYN	0.912 ± 0.007	0.071 ± 0.012	0.316 ± 0.072	0.923 ± 0.008	0.116 ± 0.021	0.537 ± 0.063	0.117 ± 0.030	0.754 ± 0.013	0.116 ± 0.049
SVM-Undersampling	0.766 ± 0.032	0.055 ± 0.006	0.731 ± 0.153	0.766 ± 0.035	0.103 ± 0.012	0.744 ± 0.065	0.155 ± 0.034	0.802 ± 0.057	0.109 ± 0.047
Random Forest	0.978 ± 0.001	**0.171** ± 0.161	0.088 ± 0.084	0.994 ± 0.002	0.116 ± 0.110	0.227 ± 0.210	0.113 ± 0.115	0.817 ± 0.062	0.104 ± 0.034
KNN	0.981 ± 0.002	0.000 ± 0.000	0.000 ± 0.000	0.999 ± 0.001	0.000 ± 0.000	0.000 ± 0.000	-0.003 ± 0.003	0.582 ± 0.044	0.033 ± 0.009
Logistic Regression	0.813 ± 0.014	0.055 ± 0.013	0.580 ± 0.169	0.817 ± 0.016	0.101 ± 0.024	0.681 ± 0.104	0.135 ± 0.051	0.737 ± 0.038	0.068 ± 0.018
XGBoost	0.978 ± 0.001	0.167 ± 0.156	0.065 ± 0.072	0.995 ± 0.001	0.092 ± 0.096	0.193 ± 0.186	0.094 ± 0.103	0.798 ± 0.053	0.113 ± 0.033

Table A5. Confusion matrices summed over the five test folds (positive class: bankrupt). The totals of positives are 220 (TW), 5220 (US) and 92 (UK).

Method	TW				US				UK			
	TP	FP	FN	TN	TP	FP	FN	TN	TP	FP	FN	TN
RQS-Ensemble (proposed)	188	1123	32	5476	2860	23,938	2360	49,524	54	752	38	4156
RQS-SVM (tuned)	186	1312	34	5287	3710	37,597	1510	35,865	71	2081	21	2827
RQS-SVM only	17	14	203	6585	0	17	5220	73,445	0	5	92	4903
RQS-SVM k-means	147	692	73	5907	4929	62,549	291	10,913	46	835	46	4073
RQS-SVM (raw boundary)	181	1041	39	5558	4860	60,937	360	12,525	52	958	40	3950
Weighted SVM	147	642	73	5957	4718	56,998	502	16,464	31	365	61	4543
Balanced Random Forest	185	695	35	5904	2103	11,984	3117	61,478	65	735	27	4173
EasyEnsemble	192	959	28	5640	3310	29,412	1910	44,050	75	1364	17	3544
RUSBoost	176	689	44	5910	2534	20,406	2686	53,056	70	1128	22	3780
Self-Paced Ensemble	167	638	53	5961	2686	21,650	2534	51,812	63	983	29	3925
LightGBM (balanced)	74	46	146	6553	1287	6715	3933	66,747	0	5	92	4903
MLP + Focal loss	182	992	38	5607	2060	15,030	3160	58,432	50	663	42	4245
MLP + CB-Focal loss	134	305	86	6294	19	75	5201	73,387	22	208	70	4700
MLP + LDAM-DRW	143	423	77	6176	10	38	5210	73,424	19	157	73	4751
DEC	127	341	93	6258	196	950	5024	72,512	2	31	90	4877
SVM-SMOTE	146	662	74	5937	4552	54,156	668	19,306	27	381	65	4527
SVM-ADASYN	143	633	77	5966	4552	54,409	668	19,053	29	376	63	4532
SVM-Undersampling	191	1027	29	5572	4790	59,422	430	14,040	67	1147	25	3761
Random Forest	80	89	140	6510	192	628	5028	72,834	8	27	84	4881
KNN	44	36	176	6563	147	861	5073	72,601	0	5	92	4903
Logistic Regression	177	846	43	5753	4340	48,605	880	24,857	53	898	39	4010
XGBoost	76	94	144	6505	950	5255	4270	68,207	6	23	86	4885

Table A6. Per-dataset comparison of RQS-Ensemble with each method (G-mean): mean difference over the five folds and p-value of the corrected resampled t-test [64] with Holm correction over the 21 comparisons of each dataset.

Method	TW		US		UK	
	Δ	p_H	**Δ**	p_H	**Δ**	p_H
EasyEnsemble	−0.023	1.000	−0.008	0.940	−0.064	0.600
Balanced Random Forest	−0.025	1.000	+0.027	0.588	−0.074	0.858
RUSBoost	-0.004	1.000	+0.024	0.594	−0.061	0.823
Self-Paced Ensemble	+0.013	1.000	+0.005	0.940	−0.038	1.000
SVM-Undersampling	−0.016	1.000	+0.190	0.062	−0.044	1.000
Logistic Regression	+0.003	1.000	+0.078	0.117	+0.020	1.000
MLP + Focal loss	+0.003	1.000	+0.071	0.940	+0.021	1.000
RQS-SVM (tuned)	+0.019	1.000	+0.020	0.824	+0.069	1.000
RQS-SVM (raw boundary)	+0.008	1.000	+0.209	**0.008**	+0.031	1.000
SVM-ADASYN	+0.075	0.350	+0.132	**0.011**	+0.164	0.823
SVM-SMOTE	+0.070	0.461	+0.129	**0.019**	+0.181	0.453
Weighted SVM	+0.066	0.422	+0.158	**0.018**	+0.142	0.396
RQS-SVM k-means	+0.068	0.423	+0.234	**0.019**	+0.063	0.858
MLP + LDAM-DRW	+0.069	1.000	+0.574	**<0.001**	+0.360	1.000
MLP + CB-Focal loss	+0.078	0.483	+0.554	**<0.001**	+0.239	0.718
LightGBM (balanced)	+0.263	**0.012**	+0.134	**0.049**	+0.700	**0.001**
XGBoost	+0.257	**0.040**	+0.197	**0.031**	+0.507	0.396
Random Forest	+0.241	**0.044**	+0.417	**<0.001**	+0.473	0.485
DEC	+0.102	0.051	+0.420	**0.019**	+0.608	**0.027**
KNN	+0.396	**0.004**	+0.441	**<0.001**	+0.700	**0.001**
RQS-SVM only	+0.565	**0.007**	+0.607	**<0.001**	+0.700	**0.001**

Table A7. G-mean, MCC and PR-AUC of the 22 methods with 95% confidence intervals (t distribution, 4 degrees of freedom, over the five test folds) and effect size of the difference with RQS-Ensemble in G-mean over the 15 fold×dataset blocks (matched-pairs rank biserial correlation r_{rb}; positive: RQS-Ensemble better).

Method	TW			US			UK			r_{rb}
	G-Mean	MCC	PR-AUC	G-Mean	MCC	PR-AUC	G-Mean	MCC	PR-AUC	
RQS-Ensemble (proposed)	0.840 [0.783, 0.897]	0.307 [0.270, 0.344]	0.300 [0.281, 0.320]	0.607 [0.592, 0.622]	0.117 [0.103, 0.131]	0.115 [0.104, 0.126]	0.700 [0.629, 0.772]	0.162 [0.137, 0.187]	0.102 [0.075, 0.128]	–
RQS-SVM (tuned)	0.821 [0.781, 0.862]	0.277 [0.251, 0.304]	0.323 [0.270, 0.376]	0.587 [0.567, 0.608]	0.099 [0.073, 0.126]	0.122 [0.106, 0.139]	0.631 [0.501, 0.761]	0.105 [0.054, 0.156]	0.056 [0.026, 0.087]	+0.78
RQS-SVM only	0.275 [0.227, 0.324]	0.197 [0.136, 0.258]	0.302 [0.251, 0.353]	0.000 [0.000, 0.000]	−0.003 [−0.007, 0.001]	0.085 [0.076, 0.094]	0.000 [0.000, 0.000]	−0.003 [−0.007, 0.002]	0.056 [0.015, 0.098]	+1.00
RQS-SVM K−means	0.772 [0.715, 0.830]	0.303 [0.253, 0.353]	0.302 [0.237, 0.367]	0.373 [0.335, 0.412]	0.066 [0.037, 0.095]	0.112 [0.090, 0.134]	0.638 [0.515, 0.761]	0.116 [0.058, 0.175]	0.079 [0.048, 0.110]	+1.00
RQS-SVM (raw boundary)	0.832 [0.780, 0.883]	0.306 [0.261, 0.351]	0.313 [0.239, 0.387]	0.398 [0.370, 0.426]	0.068 [0.041, 0.095]	0.111 [0.092, 0.129]	0.669 [0.557, 0.781]	0.124 [0.069, 0.179]	0.086 [0.045, 0.128]	+0.80
Weighted SVM	0.775 [0.709, 0.840]	0.315 [0.276, 0.354]	0.328 [0.278, 0.378]	0.450 [0.423, 0.476]	0.077 [0.050, 0.104]	0.126 [0.104, 0.148]	0.558 [0.503, 0.613]	0.131 [0.099, 0.164]	0.120 [0.057, 0.182]	+1.00
Balanced Random Forest	0.865 [0.795, 0.935]	0.388 [0.339, 0.436]	0.431 [0.368, 0.495]	0.580 [0.562, 0.599]	0.156 [0.142, 0.169]	0.142 [0.129, 0.156]	0.774 [0.725, 0.824]	0.205 [0.176, 0.233]	0.131 [0.065, 0.197]	−0.43
EasyEnsemble	0.863 [0.826, 0.900]	0.344 [0.324, 0.363]	0.441 [0.360, 0.522]	0.615 [0.604, 0.627]	0.118 [0.107, 0.130]	0.130 [0.123, 0.138]	0.765 [0.703, 0.826]	0.160 [0.128, 0.192]	0.163 [0.086, 0.241]	−0.85
RUSBoost	0.844 [0.767, 0.921]	0.369 [0.312, 0.425]	0.398 [0.319, 0.477]	0.583 [0.564, 0.602]	0.117 [0.107, 0.127]	0.100 [0.076, 0.123]	0.761 [0.701, 0.822]	0.172 [0.135, 0.208]	0.100 [0.078, 0.121]	−0.22
Self-Paced Ensemble	0.827 [0.772, 0.881]	0.364 [0.322, 0.405]	0.317 [0.232, 0.403]	0.602 [0.588, 0.616]	0.118 [0.106, 0.131]	0.109 [0.102, 0.117]	0.739 [0.661, 0.816]	0.161 [0.119, 0.202]	0.074 [0.039, 0.109]	−0.08
LightGBM (balanced)	0.577 [0.525, 0.629]	0.452 [0.374, 0.531]	0.513 [0.420, 0.607]	0.473 [0.442, 0.503]	0.128 [0.103, 0.152]	0.138 [0.127, 0.150]	0.000 [0.000, 0.000]	−0.003 [−0.007, 0.001]	0.106 [0.061, 0.151]	+1.00
MLP + Focal loss	0.838 [0.799, 0.876]	0.317 [0.294, 0.339]	0.394 [0.334, 0.455]	0.536 [0.462, 0.610]	0.125 [0.102, 0.148]	0.138 [0.125, 0.152]	0.680 [0.552, 0.807]	0.156 [0.096, 0.217]	0.108 [0.034, 0.183]	+0.48
MLP + CB-Focal loss	0.762 [0.749, 0.775]	0.407 [0.379, 0.435]	0.405 [0.343, 0.466]	0.053 [0.012, 0.093]	0.023 [−0.011, 0.057]	0.141 [0.119, 0.163]	0.462 [0.277, 0.646]	0.124 [0.032, 0.216]	0.103 [0.029, 0.176]	+1.00
MLP + LDAM-DRW	0.771 [0.638, 0.904]	0.386 [0.345, 0.427]	0.381 [0.337, 0.426]	0.033 [−0.006, 0.073]	0.013 [−0.008, 0.035]	0.137 [0.125, 0.149]	0.340 [−0.056, 0.736]	0.100 [−0.031, 0.231]	0.102 [0.057, 0.147]	+0.97
DEC	0.738 [0.668, 0.808]	0.367 [0.317, 0.417]	0.349 [0.289, 0.409]	0.187 [0.124, 0.250]	0.051 [0.003, 0.099]	0.109 [0.100, 0.118]	0.093 [−0.065, 0.251]	0.027 [−0.038, 0.092]	0.106 [0.055, 0.156]	+1.00
SVM-SMOTE	0.770 [0.694, 0.847]	0.307 [0.254, 0.360]	0.321 [0.255, 0.386]	0.478 [0.454, 0.503]	0.077 [0.052, 0.102]	0.125 [0.102, 0.148]	0.519 [0.461, 0.577]	0.106 [0.079, 0.133]	0.114 [0.053, 0.176]	+1.00
SVM-ADASYN	0.765 [0.706, 0.824]	0.308 [0.277, 0.339]	0.295 [0.246, 0.344]	0.475 [0.455, 0.496]	0.075 [0.050, 0.101]	0.121 [0.096, 0.145]	0.537 [0.459, 0.615]	0.117 [0.080, 0.154]	0.116 [0.055, 0.177]	+1.00
SVM-Undersampling	0.856 [0.818, 0.894]	0.329 [0.290, 0.368]	0.350 [0.290, 0.409]	0.417 [0.364, 0.469]	0.070 [0.040, 0.099]	0.120 [0.101, 0.138]	0.744 [0.664, 0.824]	0.155 [0.114, 0.197]	0.109 [0.050, 0.168]	+0.17
Random Forest	0.599 [0.583, 0.614]	0.404 [0.345, 0.463]	0.411 [0.336, 0.486]	0.191 [0.175, 0.206]	0.070 [0.048, 0.092]	0.133 [0.119, 0.146]	0.227 [−0.034, 0.489]	0.113 [−0.030, 0.256]	0.104 [0.063, 0.146]	+1.00
KNN	0.444 [0.380, 0.507]	0.319 [0.249, 0.389]	0.253 [0.196, 0.309]	0.167 [0.153, 0.180]	0.037 [0.024, 0.050]	0.086 [0.080, 0.093]	0.000 [0.000, 0.000]	−0.003 [−0.007, 0.001]	0.033 [0.021, 0.044]	+1.00
Logistic Regression	0.837 [0.799, 0.874]	0.335 [0.310, 0.360]	0.354 [0.261, 0.446]	0.530 [0.506, 0.553]	0.090 [0.073, 0.108]	0.132 [0.106, 0.159]	0.681 [0.551, 0.810]	0.135 [0.071, 0.198]	0.068 [0.046, 0.090]	+0.60
XGBoost	0.583 [0.544, 0.622]	0.379 [0.322, 0.435]	0.383 [0.325, 0.440]	0.410 [0.378, 0.443]	0.102 [0.077, 0.127]	0.120 [0.110, 0.131]	0.193 [−0.038, 0.425]	0.094 [−0.033, 0.222]	0.113 [0.072, 0.154]	+1.00

Appendix C. Resources of the Ablation Configurations

Table A8 gives, for the 17 configurations of the ablation study, the training set size, the number of prototypes, the number of support vectors and the peak memory, which complete the fitting times of Table 11 and the seven key configurations of Table 12.

Table A8. Resources of the 17 configurations of the ablation study (same models, folds and data-driven parameters as Table 8; mean over the five folds). $|\mathcal{Q}|$: training set size after the representation step (n = 5,455, 20,000 and 4000 for TW, US and UK); K: number of synthetic prototypes in the training set (PCM, k-means, fuzzy C-means or GMM prototypes of the boundary, or majority class centroids for ClusterCentroids; 0 when the boundary is kept, – when no prototypes are built); SV: number of support vectors (– for the k-NN classifier); Mem.: peak memory allocated by Python and NumPy during fitting (`tracemalloc`; the internal buffers of LIBSVM are not counted) in MiB. Fitting times are given in Table 11. Measured sequentially on an Intel Core Ultra 7 165U laptop with 32 GB of memory.

Configuration	TW				US				UK			
	$\|\mathcal{Q}\|$	K	SV	Mem.	$\|\mathcal{Q}\|$	K	SV	Mem.	$\|\mathcal{Q}\|$	K	SV	Mem.
RQS-SVM (full)	4691	3.0	1660	14.5	12,557	5.2	11,456	9.3	3441	3.6	2311	7.7
w/o class mass balancing (clipping and λ only)	4691	3.0	459	14.4	12,557	5.2	2285	9.3	3441	3.6	464	7.6
w/o weighting (uniform weights)	4691	3.0	457	14.4	12,557	5.2	1836	9.3	3441	3.6	458	7.6
w/o minority amplification (λ=0)	4691	3.0	1660	14.4	12,557	5.2	11,456	9.3	3441	3.6	2311	7.6
w/o weight clipping	4691	3.0	1635	14.4	12,557	5.2	11,459	9.3	3441	3.6	2272	7.6
quotient weights → balanced class weights	4691	3.0	743	14.4	12,557	5.2	10,680	9.3	3441	3.6	874	7.6
rough → fuzzy–rough (OWA)	4692	4.0	2068	15.7	12,557	5.2	11,488	20.8	3442	4.8	2431	9.0
rough → k-NN entropy	4691	3.0	3297	14.5	12,557	5.4	11,780	10.1	3441	4.0	2687	7.6
no partition: PCM on whole dataset	2	2.4	2	13.0	4	4.4	4	12.6	4	3.8	4	7.2
PCM → none (raw boundary)	5455	0.0	2433	16.0	20,000	0.0	18,660	11.8	4000	0.0	2787	8.5
PCM → k-means	4698	10.0	1705	14.4	12,562	10.0	11,455	9.3	3447	10.0	2318	7.6
PCM → fuzzy C-means	4698	10.0	1712	14.4	12,562	10.0	11,456	12.0	3447	10.0	2337	7.6
PCM → GMM	4698	10.0	1702	14.4	12,562	10.0	11,456	9.3	3447	10.0	2330	7.6
SVM → k-NN on RQS prototypes	4691	3.0	–	14.4	12,557	5.2	–	9.3	3441	3.6	–	7.6
ClusterCentroids + SVM	352	176	227	15.5	2654	1327	755	11.1	147	74	121	8.3
Standard SVM	5455	–	715	8.0	20,000	–	3025	5.7	4000	–	552	4.2
Weighted SVM (balanced)	5455	–	1610	8.0	20,000	–	17,835	5.7	4000	–	1317	4.2

References

1. Wu, Z.; Zhu, W.; Zhang, C.; Wu, Y.; Zhang, T.; Li, Z. Recovery or deterioration: A financial distress prognostic model. *Inf. Process. Manag.* **2026**, *63*, 104503.
2. Garcia, J. Bankruptcy prediction using synthetic sampling. *Mach. Learn. Appl.* **2022**, *9*, 100343.
3. Radovanovic, J.; Haas, C. The evaluation of bankruptcy prediction models based on socio-economic costs. *Expert Syst. Appl.* **2023**, *227*, 120275.
4. Lohmann, C.; Möllenhoff, S.; Ohliger, T. Nonlinear relationships in bankruptcy prediction and their effect on the profitability of bankruptcy prediction models. *J. Bus. Econ.* **2023**, *93*, 1661–1690.
5. Dasilas, A.; Rigani, A. Machine learning techniques in bankruptcy prediction: A systematic literature review. *Expert Syst. Appl.* **2024**, *255*, 124761.
6. Park, M.S.; Son, H.; Hyun, C.; Hwang, H.J. Explainability of Machine Learning Models for Bankruptcy Prediction. *IEEE Access* **2021**, *9*, 124887–124899.
7. Wang, Z.; Jiang, C.; Zhao, H. Depicting Risk Profile over Time: A Novel Multiperiod Loan Default Prediction Approach. *Manag. Inf. Syst. Q.* **2023**, *47*, 1455–1486.
8. Chen, Z.; Chen, W.; Shi, Y. Ensemble learning with label proportions for bankruptcy prediction. *Expert Syst. Appl.* **2020**, *146*, 113155.
9. Iparraguirre-Villanueva, O.; Cabanillas-Carbonell, M. Predicting business bankruptcy: A comparative analysis with machine learning models. *J. Open Innov. Technol. Mark. Complex.* **2024**, *10*, 100375.
10. Elhoseny, M.; Metawa, N.; Sztano, G.; El-Hasnony, I.M. Deep learning-based model for financial distress prediction. *Ann. Oper. Res.* **2025**, *345*, 885–907.
11. Gaurav, A.; Gupta, B.B.; Bansal, S.; Psannis, K.E. Bankruptcy forecasting in enterprises and its security using hybrid deep learning models. *Cyber Secur. Appl.* **2025**, *3*, 100070.
12. Fasano, F.; Adornetto, C.; Zahid, I.; La Rocca, M.; Montaleone, L.; Greco, G.; Cariola, A. The dilemma of accuracy in bankruptcy prediction: A new approach using explainable AI techniques to predict corporate crises. *Eur. J. Innov. Manag.* **2025**, *28*, 1–22.
13. Zhang, L. The evaluation on the credit risk of enterprises with the CNN-LSTM-ATT model. *Comput. Intell. Neurosci.* **2022**, *2022*, 6826573.
14. Brenes, R.F.; Johannssen, A.; Chukhrova, N. An intelligent bankruptcy prediction model using a multilayer perceptron. *Intell. Syst. Appl.* **2022**, *16*, 200136–200136.

15. Chen, T.K.; Liao, H.H.; Chen, G.D.; Kang, W.H.; Lin, Y.C. Bankruptcy prediction using machine learning models with the text-based communicative value of annual reports. *Expert Syst. Appl.* **2023**, *233*, 120714.
16. Wang, S.; Chi, G. Cost-sensitive stacking ensemble learning for company financial distress prediction. *Expert Syst. Appl.* **2024**, *255*, 124525–124525.
17. Chen, Y.; Guo, J. LiFoL: An Efficient Framework for Financial Distress Prediction in High-Dimensional Unbalanced Scenario. *IEEE Trans. Comput. Soc. Syst.* **2023**, *11*, 2784–2795.
18. Li, J.; Zhu, Q.; Wu, Q.; Zhang, Z.; Gong, Y.; He, Z.; Zhu, F. SMOTE-NaN-DE: Addressing the noisy and borderline examples problem in imbalanced classification by natural neighbors and differential evolution. *Knowl.-Based Syst.* **2021**, *223*, 107056.
19. Gnip, P.; Kanasz, R.; Zoričak, M.; Drotar, P. An experimental survey of imbalanced learning algorithms for bankruptcy prediction. *Artif. Intell. Rev.* **2025**, *58*, 104.
20. Wang, X.; Zhang, R.; Zhang, Z. A novel hybrid sampling method ESMOTE+ SSLM for handling the problem of class imbalance with overlap in financial distress detection. *Neural Process. Lett.* **2023**, *55*, 3081–3105.
21. Leng, Q.; Guo, J.; Tao, J.; Meng, X.; Wang, C. OBMI: Oversampling borderline minority instances by a two-stage Tomek link-finding procedure for class imbalance problem. *Complex Intell. Syst.* **2024**, *10*, 4775–4792.
22. Perboli, G.; Arabnezhad, E. A Machine Learning-based DSS for mid and long-term company crisis prediction. *Expert Syst. Appl.* **2021**, *174*.
23. Hou, Z.; Tang, J.; Li, Y.; Fu, S.; Tian, Y. MVQS: Robust multi-view instance-level cost-sensitive learning method for imbalanced data classification. *Inf. Sci.* **2024**, *675*, 120467.
24. Patel, V.; Bhavsar, H. Optimizing kernel transformations to handle binary class imbalanced dataset classification. *Appl. Artif. Intell.* **2024**, *38*, 2408933.
25. Xia, S.; Lian, X.; Wang, G.; Gao, X.; Chen, J.; Peng, X. GBSVM: An Efficient and Robust Support Vector Machine Framework via Granular-Ball Computing. *IEEE Trans. Neural Netw. Learn. Syst.* **2024**, *36*, 9253–9267.
26. Sun, Y.; Vong, C.M.; Wang, S. Fast AUC Maximization Learning Machine with Simultaneous Outlier Detection. *IEEE Trans. Cybern.* **2023**, *53*, 6843–6857.
27. Chen, C.; Liaw, A.; Breiman, L. *Using Random Forest to Learn Imbalanced Data*; Technical Report 666; Department of Statistics, University of California, Berkeley: Berkeley, CA, USA, 2004.
28. Liu, X.Y.; Wu, J.; Zhou, Z.H. Exploratory Undersampling for Class-Imbalance Learning. *IEEE Trans. Syst. Man Cybern. Part B (Cybern.)* **2009**, *39*, 539–550.
29. Seiffert, C.; Khoshgoftaar, T.M.; Van Hulse, J.; Napolitano, A. RUSBoost: A Hybrid Approach to Alleviating Class Imbalance. *IEEE Trans. Syst. Man, Cybern. Part A Syst. Humans* **2010**, *40*, 185–197.
30. Liu, Z.; Cao, W.; Gao, Z.; Bian, J.; Chen, H.; Chang, Y.; Liu, T.Y. Self-paced Ensemble for Highly Imbalanced Massive Data Classification. In Proceedings of the IEEE 36th International Conference on Data Engineering (ICDE), Dallas, TX, USA, 20–24 April 2020; pp. 841–852.
31. Ke, G.; Meng, Q.; Finley, T.; Wang, T.; Chen, W.; Ma, W.; Ye, Q.; Liu, T.Y. LightGBM: A Highly Efficient Gradient Boosting Decision Tree. In Proceedings of the Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, Long Beach, CA, USA, 4–9 December 2017; pp. 3146–3154.
32. Lin, T.Y.; Goyal, P.; Girshick, R.; He, K.; Dollár, P. Focal Loss for Dense Object Detection. In Proceedings of the IEEE International Conference on Computer Vision (ICCV), Venice, Italy, 22–29 October 2017; pp. 2980–2988.
33. Cui, Y.; Jia, M.; Lin, T.Y.; Song, Y.; Belongie, S. Class-Balanced Loss Based on Effective Number of Samples. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Long Beach, CA, USA, 15–20 June 2019; pp. 9268–9277.
34. Cao, K.; Wei, C.; Gaidon, A.; Arechiga, N.; Ma, T. Learning Imbalanced Datasets with Label-Distribution-Aware Margin Loss. In Proceedings of the Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, Vancouver, BC, Canada, 8–14 December 2019.
35. Dablain, D.; Krawczyk, B.; Chawla, N.V. DeepSMOTE: Fusing deep learning and SMOTE for imbalanced data. *IEEE Trans. Neural Netw. Learn. Syst.* **2022**, *34*, 6390–6404.
36. Tharwat, A.; Gabel, T. Parameters optimization of support vector machines for imbalanced data using social ski driver algorithm. *Neural Comput. Appl.* **2020**, *32*, 6925–6938.
37. Song, Y.; Jiang, M.; Li, S.; Zhao, S. Class-imbalanced financial distress prediction with machine learning: Incorporating financial, management, textual, and social responsibility features into index system. *J. Forecast.* **2024**, *43*, 593–614.
38. Yu, L.; Li, M.; Liu, X. A two-stage case-based reasoning driven classification paradigm for financial distress prediction with missing and imbalanced data. *Expert Syst. Appl.* **2024**, *249*, 123745.
39. Hosaka, T. Bankruptcy prediction using imaged financial ratios and convolutional neural networks. *Expert Syst. Appl.* **2019**, *117*, 287–299.

40. Jang, Y.; Jeong, I.; Cho, Y.K. Business failure prediction of construction contractors using a LSTM RNN with accounting, construction market, and macroeconomic variables. *J. Manag. Eng.* **2020**, *36*, 04019039.
41. Li, Y.; Baofeng, S.; Yizhe, D. A Credit Risk Evaluation Model for Imbalanced Data Classification Based on Class Balanced Loss Modified Cross Entropy Function. *J. Syst. Manag.* **2022**, *31*, 255–269.
42. Le, T. A comprehensive survey of imbalanced learning methods for bankruptcy prediction. *IET Commun.* **2022**, *16*, 433–441.
43. Velmurugan, M.; Ouyang, C.; Moreira, C.; Sindhgatta, R. Evaluating stability of post-hoc explanations for business process predictions. In Proceedings of the International Conference on Service-Oriented Computing, Dubai, United Arab Emirates, 22–25 November 2021; pp. 49–64.
44. Nguyen, H.H.; Viviani, J.L.; Jabeur, S.B. Bankruptcy prediction using machine learning and Shapley additive explanations. *Rev. Quant. Financ. Account.* **2023**, *65*, 107–148.
45. Cao, Y.; Liu, X.; Zhai, J.; Hua, S. A two-stage Bayesian network model for corporate bankruptcy prediction. *Int. J. Financ. Econ.* **2022**, *27*, 455–472.
46. Purvinis, O.; Virbickaitė, R.; Šukys, P. Interpretable nonlinear model for enterprise bankruptcy prediction. *Nonlinear Anal. Model. Control* **2008**, *13*, 61–70.
47. Zhang, Z.; Wu, C.; Qu, S.; Chen, X. An explainable artificial intelligence approach for financial distress prediction. *Inf. Process. Manag.* **2022**, *59*, 102988.
48. Lin, Y.C.; Padliansyah, R.; Lu, Y.H.; Liu, W.R. Bankruptcy prediction: Integration of convolutional neural networks and explainable artificial intelligence techniques. *Int. J. Account. Inf. Syst.* **2025**, *56*, 100744.
49. Sun, W.; Zhang, X.; Li, M.; Wang, Y. Interpretable high-stakes decision support system for credit default forecasting. *Technol. Forecast. Soc. Change* **2023**, *196*, 122825.
50. Yeh, C.C.; Chi, D.J.; Hsu, M.F. A hybrid approach of DEA, rough set and support vector machines for business failure prediction. *Expert Syst. Appl.* **2010**, *37*, 1535–1541. https://doi.org/10.1016/j.eswa.2009.06.088.
51. Cao, Y.; Wan, G.; Wang, F. Predicting financial distress of Chinese listed companies using rough set theory and support vector machine. *Asia-Pac. J. Oper. Res.* **2011**, *28*, 95–109.
52. Zhang, J.; Wang, Y. A rough margin based support vector machine. *Inf. Sci.* **2008**, *178*, 2204–2214.
53. Lin, C.F.; Wang, S.D. Fuzzy support vector machines. *IEEE Trans. Neural Netw.* **2002**, *13*, 464–471.
54. Batuwita, R.; Palade, V. FSVM-CIL: Fuzzy support vector machines for class imbalance learning. *IEEE Trans. Fuzzy Syst.* **2010**, *18*, 558–571.
55. Chen, D.; He, Q.; Wang, X. FRSVMs: Fuzzy rough set based support vector machines. *Fuzzy Sets Syst.* **2010**, *161*, 596–607.
56. Tang, Y.; Zhang, Y.Q.; Chawla, N.V.; Krasser, S. SVMs modeling for highly imbalanced classification. *IEEE Trans. Syst. Man Cybern. Part B* **2009**, *39*, 281–288.
57. Ramentol, E.; Caballero, Y.; Bello, R.; Herrera, F. SMOTE-RSB*: A hybrid preprocessing approach based on oversampling and undersampling for high imbalanced data-sets using SMOTE and rough sets theory. *Knowl. Inf. Syst.* **2012**, *33*, 245–265.
58. Lang, G.; Zhao, L.; Miao, D.; Ding, W. Granular-ball computing-based fuzzy twin support vector machine for pattern classification. *IEEE Trans. Fuzzy Syst.* **2025**, *33*, 2148–2160.
59. Zhang, L.; Zhang, B. The quotient space theory of problem solving. *Fundam. Inform.* **2004**, *59*, 287–298.
60. Devroye, L.; Györfi, L.; Lugosi, G. *A Probabilistic Theory of Pattern Recognition*; Springer: New York, NY, USA, 1996.
61. Graf, S.; Luschgy, H. *Foundations of Quantization for Probability Distributions*; Lecture Notes in Mathematics; Springer: Berlin, Germany, 2000; Volume 1730.
62. Pawlak, Z. Rough sets. *Int. J. Comput. Inf. Sci.* **1982**, *11*, 341–356.
63. Demšar, J. Statistical Comparisons of Classifiers over Multiple Data Sets. *J. Mach. Learn. Res.* **2006**, *7*, 1–30.
64. Nadeau, C.; Bengio, Y. Inference for the Generalization Error. *Mach. Learn.* **2003**, *52*, 239–281.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.