International Review of Financial Analysis 91 (2024) 103005

Contents lists available at ScienceDirect

International Review of Financial Analysis

journal homepage: www.elsevier.com/locate/irfa

Bridging accuracy and interpretability: A rescaled cluster-then-predict approach for enhanced credit scoring

Huei-Wen Teng *, Ming-Hsuan Kang, I-Han Lee, Le-Chi Bai

National Yang Ming Chiao Tung University, No 1001 Daxue Road East District, Hsinchu City, 300093, Taiwan, ROC

ARTICLE INFO

JEL classification:
C38
C53
G21

Keywords:
Credit scoring
Cluster-then-predict
Rescaling
XGBoost
Logistic Regression

ABSTRACT

Credit scoring is pivotal in the financial industry for assessing individuals' creditworthiness and optimizing financial institutions' risk-adjusted returns. While the XGBoost algorithm stands as the state-of-the-art classifier for credit scoring, its intricate nature impedes easy interpretation, a critical aspect for stakeholders' decision-making. This paper introduces a novel approach termed the "Rescaled Cluster-then-Predict Method," aimed at enhancing both the interpretability and predictive performance of credit scoring models. Our method employs a two-step process, initially rescaling the features and subsequently clustering the data into subgroups. Consequently, we employ Logistic Regression within each subgroup to generate predictions. The paper's primary contributions are twofold. Firstly, empirical evaluations on two distinct datasets demonstrate that our proposed method attains a competitive performance compared to XGBoost while substantially improving interpretability. Notably, in some instances, the Logistic Regression outperforms XGBoost. Secondly, we reveal that clustering solely the positive cases, as opposed to the entire dataset, yields comparable results while markedly reducing computational requirements. These insights hold significant practical implications for the financial industry, which consistently seeks credit scoring models that are not only accurate but also interpretable and computationally efficient.

1. Introduction

Credit scoring models statistically predict a borrower's likelihood to default based on factors like financial history, employment, and debt ratio, aiding loan providers in risk assessment (Hand & Henley, 1997; Menard, 2002; Thomas, Crook, & Edelman, 2017). Recently, machine learning algorithms have been juxtaposed with traditional Logistic Regression in this context (Teng & Lee, 2019). Ensemble techniques like boosting, which combine weaker models for stronger predictions, often excel over singular classifiers (Baesens et al., 2003; Lessmann, Baesens, Seow, & Thomas, 2015). Despite its reduced interpretability due to its complex nature (Freund, Schapire, & Abe, 1999; Friedman, 2001; Schapire, 1999), boosting's predictive prowess has made it a leading method in credit scoring (Chang, Chang, & Wu, 2018; Liu, Fan, & Xia, 2022; Qin et al., 2021; Qiu, 2019; Xia, Liu, Li, & Liu, 2017).

The surge in machine learning and AI underscores the importance of regulatory compliance and accuracy in today's context. Both European and US regulations emphasize algorithmic transparency and interpretability, highlighting the timeliness of our study. As of 25 May 2018, the European General Data Protection Regulation (GDPR) stipulates that solely automated decision-making processes should be meaningful and transparent. In the US, the Consumer Financial Protection Bureau (CFPB) has emphasized that, under federal regulations against discrimination, businesses must clearly specify the reasons when rejecting credit applications or taking other negative measures, irrespective of whether they use intricate algorithmic credit models. This reminder was circulated by the CFPB through a Consumer Financial Protection Notification, highlighting the obligations of creditors under the provisions of the Equal Credit Opportunity Act (ECOA) on May 26 2022. Furthermore, Model Fairness and Transparency in Credit Scoring Act (FaTCSA) seeks to solve transparency issues by requiring credit reporting agencies to provide state attorneys generals and the public with detailed explanations of their methodologies. In credit scoring, where decisions significantly affect loan or insurance outcomes, the need for transparent machine learning algorithms is paramount.

In this paper, we juxtapose Logistic Regression and XGBoost. Logistic Regression is prized for its clarity; its coefficients elucidate the variables' interplay, aiding stakeholders' comprehension of decision rationales. Such transparency is sometimes a legal mandate, safeguarding borrowers and ensuring ethical lending (Bussmann, Giudici, Marinelli, & Papenbrock, 2020; Chen et al., 2018; Demajo, Vella, & Dingli, 2020; Goodman & Flaxman, 2017; Misheva, Osterrieder, Hirsa, Kulkarni, & Lin, 2021; Selbst & Powles, 2018). Conversely, XGBoost is a leading

* Corresponding author.
E-mail address: hwteng@nycu.edu.tw (H.-W. Teng).

https://doi.org/10.1016/j.irfa.2023.103005
Received 10 July 2023; Received in revised form 21 August 2023; Accepted 9 October 2023
Available online 20 October 2023

machine learning algorithm, acclaimed for its predictive prowess across domains, including credit scoring. Yet, its ensemble approach with multiple decision trees may compromise interpretability.

This emphasis on explainable AI transcends Logistic Regression, advocating for equitable and ethical financial decisions (Arrieta et al., 2020; Brundage et al., 2018). Several techniques have been devised to enhance the precision of Logistic Regression, including feature scaling, engineering, regularization, hyperparameter tuning, and the cluster-then-predict method (García, Luengo, & Herrera, 2015; Géron, 2022; Hastie, Tibshirani, Friedman, & Friedman, 2009; James, Witten, Hastie, & Tibshirani, 2013). The cluster-then-predict method segments data into homogeneous clusters before model training, and is notable in finance for elevating overall accuracy (Chen & Shyu, 2011; Peikari, Salama, Nofech-Mozes, & Martel, 2018; Radu, Katsikouli, Sarkar, & Marina, 2014; Tsai, 2014). Feature scaling is crucial for clustering because it heavily relies on distance or similarity metrics. However, existing scaling techniques for the cluster-then-predict, such as the min–max and Z-score normalization, do not differentiate feature importance.

In contrast, our proposed rescaled cluster-then-predict approach adjusts features considering their target impact. This promotes a distance measure that mirrors the essential weight of each feature, emphasizing crucial ones while dimming less significant ones. We emphasize the crucial act of rescaling features based on their significance to the target, which has not been a focal point in prior approaches. This is the first primary way this paper diverges from previous studies.

Secondly, our findings highlight that clustering solely the positive cases yields comparable results to clustering all cases, offering a considerable computational advantage. When devising credit scoring models, imbalanced data frequently emerges, with default (positive) cases being outnumbered by non-default (negative) cases. Addressing this challenge involves strategies like adjusting default thresholds, employing pertinent evaluation metrics, leveraging machine learning algorithms adept at handling imbalanced data, adopting cost-sensitive classifications, and sampling techniques (Aggarwal, 2015; Fernández et al., 2018; He & Garcia, 2009; Krawczyk, 2016; Sun, Wong, & Kamel, 2009). Our commitment to precisely identifying positive instances led us to explore the implications of clustering based solely on default cases versus the entire dataset.

In our exploration of clustering strategies for imbalanced data, we unveil four rescaling techniques, applied to preprocessed data, culminating in the formation of distinct subgroups; each subgroup has a unique model. Our "rescaled cluster-then-predict" approach, powered by the sophisticated XGBoost and the transparent Logistic Regression, seeks to elevate prediction quality. Upon analyzing two distinct datasets, we discern that while XGBoost's performance remains unaffected or even dips in terms of the Area Under the Curve (AUC) with increased cluster counts and the inclusion of polynomial transformed features, Logistic Regression thrives, especially when a specific rescaling technique and polynomial features are employed. Intriguingly, clustering just the positive instances mirrors the efficacy of clustering the complete dataset, promising marked computational savings.

The remainder of this paper unfolds as follows: Section 2 delves into the motivations and elucidates our "rescaled cluster-then-predict" methodology. Sections 3 and 4 individually spotlight data analysis on two datasets, underscoring our strategy's prowess in bridging interpretability and accuracy for credit scoring models and addressing imbalanced data. The concluding section encapsulates our findings and suggests avenues for future exploration in credit scoring and explainable AI.

2. Motivations, preliminaries, and features rescaling

In this section, we explore the theoretical foundations and rationale behind our innovative approach, which encompasses feature rescaling during clustering, interpretability considerations, polynomial feature transformations, and our comprehensive data preprocessing strategy. To illustrate the effectiveness of our methodology, we present two distinct datasets that serve both as our inspiration and empirical evidence.

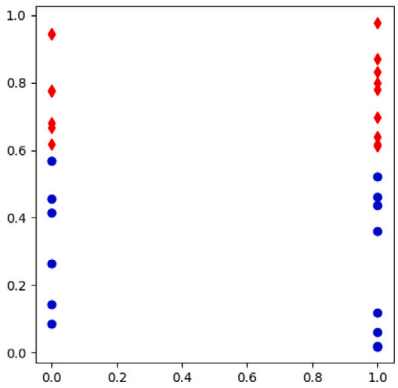

Fig. 1. Data points.

2.1. Feature rescaling in clustering

In predictive modeling, clustering techniques often improve prediction accuracy. Since clustering is sensitive to feature scale, normalizing features, like through z-scoring or minimax standardization (Berkhin, 2006; James et al., 2013), is advised to prevent any one feature from dominating. However, normalization can sometimes skew clustering results by giving undue weight to irrelevant features. Here's a simple example to illustrate this point.

Consider a data generation process with two features: x_1 and x_2. For our 30 data points, x_1 is dichotomous, taking values 0 or 1, while x_2 is continuous, uniformly distributed between 0 and 1. Thus, x_1 and x_2 are considered as traditionally normalized features because they have the same range. The response variable y depends solely on x_2, with $y = 1$ if $x_2 > 0.6$ and 0 otherwise. This results in a dataset where x_1 does not predict y, but x_2 does. The data points are shown in Fig. 1. This scenario reflects real-world data analysis where some predictors may be irrelevant to the outcome.

We use K-medoids clustering with $K = 2$ on the original features. The resulted decision boundary, heavily influenced by x_1, are shown in the left plot in Fig. 2. This underscores the significance of feature scaling and the risks of clustering when irrelevant and relevant predictors share the same scale.

We now rescale the features using logistic regression, obtaining coefficients of 0.039 for x_1 and 2.281 for x_2. As expected, the irrelevant x_1 has a near-zero coefficient, while x_2's is much larger. The rescaled features are $\tilde{x}_1 = 0.039x_1$ and $\tilde{x}_2 = 2.281x_2$. The right plot in Fig. 2 shows these features, with clusters now mainly aligned with x_2. Comparing traditionally normalized and rescaled features illustrates the impact of rescaling. This prompts us to explore if clustering with rescaled features enhances the accuracy of machine learning algorithms.

2.2. Interpretability

Interpreting machine learning models is crucial to understand their decision-making process, especially in domains where the consequences of predictions are significant. In Logistic Regression, the response variable y is either 0 or 1, with 0 for non-default and 1 for default cases. The probability of the dependent event occurring given the predictors x, $P(y = 1|x)$, is defined through a logistic (or sigmoid) function,

$$P(y = 1|x) = \frac{1}{1 + e^{-(\beta_0+\beta_1 x_1+\beta_2 x_2+\cdots+\beta_p x_p)}}, \tag{1}$$

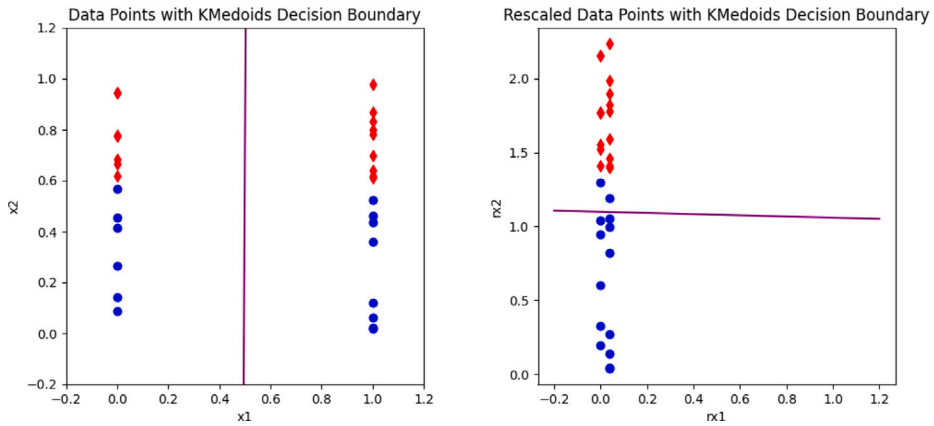

Fig. 2. Visualization of data points and decision boundaries.

where $x = (x_1, x_2, \ldots, x_p)$ is a vector of the predictors and $\beta = (\beta_0, \ldots, \beta_p)$ a vector of coefficients. Let $Bernoulli(\pi)$ denote the Bernoulli distribution with probability π. Equivalently, Eq. (1) can be expressed as

$$y|x \sim Bernoulli\left(\frac{1}{1+e^{-(\beta_0+\beta_1 x_1+\beta_2 x_2 \cdots+\beta_p x_p)}}\right),$$

where the maximum likelihood estimation estimates the coefficients.

To interpret the logistic regression, we use the log-odds, which is the logarithm of the odds ratio:

$$\text{Log-odds} = \log\left(\frac{P(y=1|x)}{1-P(y=1|x)}\right) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_p x_p.$$

The coefficients in logistic regression offer insights into the relationship between predictors and the log-odds of the outcome. Specifically, for a one-unit increase in a predictor x_i, the log-odds of the outcome (i.e., $p(y = 1)$) versus non-occurrence increase by β_i units, while holding other predictors constant. As a consequence, Logistic Regression is easy to interpret through log-odds.

On the other hand XGBoost, short for eXtreme Gradient Boosting, is a refined gradient boosting implementation designed for enhanced speed and performance. It has gained prominence in machine learning competitions and practical applications due to its accuracy and efficiency. Essentially, XGBoost constructs multiple decision trees sequentially, with subsequent trees addressing the errors of the previous ones. This "boosting" denotes the additive manner in which trees are formed.

Being an ensemble of numerous decision trees, XGBoost is inherently more intricate than simpler models like linear regression or a singular decision tree. Its complexity arises from building many trees (often hundreds or thousands). Directly interpreting each tree is impractical. The trees in XGBoost are assembled in sequence, where later trees refine earlier iterations. This cascading relationship complicates identifying the impact of an individual tree. Even though single trees can detect non-linear feature interactions, an ensemble deepens this understanding, making extracting intuitive insights challenging. Unlike linear regression, which offers clear coefficients for each feature, XGBoost lacks such straightforward interpretability.

2.3. Why polynomial features enhance the prediction of Logistic Regression?

To illustrate how polynomial-transformed features and interactions can enhance Logistic Regression's predictions, consider these two examples. For the first, with a single feature ($p = 1$), we generate 100 data points for x between -1 and 1. y is set to 1 for x values less than -0.2 or greater than 0.5, and 0 otherwise. Refer to Fig. 3 for this data representation.

We fit Logistic Regression using the original feature, x, and its quadratic polynomial, x^2. This quadratic inclusion equips the Logistic Regression to detect and adapt to data curves. Fig. 3 displays the predicted probability with two overlaying lines. The red solid line represents predictions using just x, which might not effectively capture the data's nuances. In contrast, the green dashed line depicts predictions with polynomial features, showing more adaptability to our data's patterns. The visualization clearly favors the green dashed line for accurately reflecting the data's relationship. Adding x^2 grants the model enhanced flexibility to capture intricate non-linear relationships, leading to better predictions.

In our subsequent illustration with two predictors, x_1 and x_2, the coordinates of points are stored in X, while y denotes a binary label identifying each point's respective circle. Fig. 4 displays two concentric circles: the inner red diamond points represent $y = 1$, and the outer blue circle points signify $y = 1$.

We trained two Logistic Regression models on this synthetic data: one using only the original features, x_1, x_2, and another enhanced with quadratic features (x_1, x_2, x_1^2, x_2^2, and $x_1 x_2$). This inclusion captures non-linearities. The decision boundaries of both models are visualized in Fig. 5. The left plot, based on original features, has a linear boundary that does not distinguish the two classes effectively. The right plot, using polynomial features, has a flexible boundary that accurately separates the classes. This comparison highlights that polynomial features can bolster Logistic Regression's ability to detect complex data patterns that a linear model might overlook.

2.4. Data preprocessing

Data pre-processing is key to successful modeling (Heaton, 2016). For a fair comparison between XGBoost and Logistic Regression, we undertake essential preprocessing steps, optimizing the data for their application and evaluation.

(1) Log transformation: To counteract the sensitivity of Logistic Regression to outliers, features with kurtosis exceeding 10 undergo a log transformation. For features with negative values, we adjust them for log transformation by adding the absolute of their minimum value, plus a slight increment of 0.00001.
(2) Z-score normalization: To circumvent potential round-off errors and numerical instabilities during the estimation of Logistic Regression parameters, we normalize non-binary features using Z-score scaling. Z-score scaling normalizes a feature by subtracting its sample mean and then dividing by the sample standard deviation.
(3) Handling missing values: Rather than omitting observations with absent values or substituting them with measures like the mean or median, we employ a missing indicator. The missing indicator method for features with absent values involves two steps:

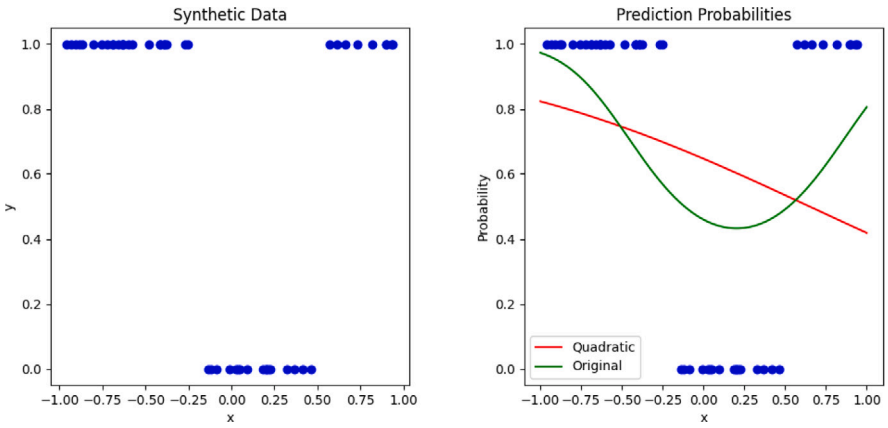

Fig. 3. Data points and predicted probabilities in Illustration 1.

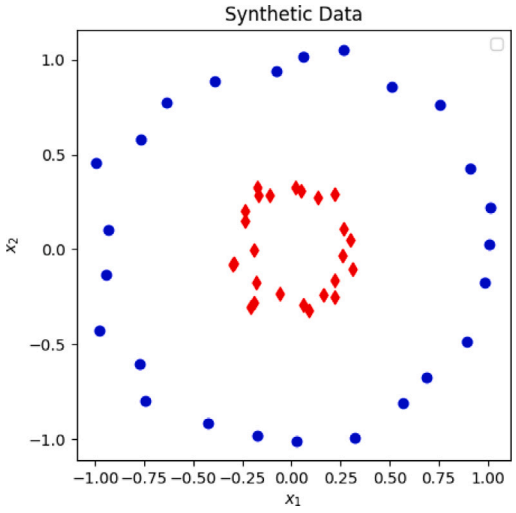

Fig. 4. Data points in Illustration 2.

Step 1. Assign a value of 1 to the missing indicator for missing entries and 0 for the rest.
Step 2. Substitute missing values with zero.

To elucidate the advantages of the missing indicator, consider a simple linear regression model:

$$y = \beta_0 + \beta_1 x + \varepsilon,$$

where some instances of x are missing. Denoting the ith observation with subscript i, the missing indicator d for Step 1 is:

$$d_i = \begin{cases} 1, & \text{if } x_i \text{ is missing} \\ 0, & \text{if } x_i \text{ is complete} \end{cases}$$

For Step 2, the imputed feature $\tilde{x}$ is defined by replacing missing values with zero:

$$\hat{x}_i = \begin{cases} 0, & \text{if } x_i \text{ is missing} \\ x, & \text{if } x_i \text{ is complete} \end{cases}$$

Upon applying the two steps, our modified model becomes:

$$y = \beta_0 + \beta_1 \hat{x} + \gamma d + \varepsilon.$$

This can be further expressed as:

$$y = \begin{cases} \beta_0 + \gamma + \varepsilon, & \text{if } x_i \text{ is missing} \\ \beta_0 + \beta_1 x + \varepsilon, & \text{if } x_i \text{ is complete} \end{cases}$$

Effectively, the modified model bifurcates into two scenarios: one where x is missing, rendering y independent of x, and another where x is complete, reverting y to the basic linear regression relationship.

In the context of Logistic Regression, incorporating a missing indicator effectively splits the dataset into two groups: one with complete observations and another with missing values. Importantly, while both groups use the same model parameters, they differ in the intercept, optimizing the use of the available data.

2.5. Features rescaling

The proposed rescaled cluster-then-predict approach rescales features based on their impact on the target to produce a similarity or distance measure that better reflects the significance of each feature. This enhances data representation by emphasizing and increasing variation for important features and reducing variation for less significant ones.

Suppose we have n observations in our dataset, each with p features as explanatory variables. For the ith observation, its feature vector is denoted as $x^{(i)} = (x_1^{(i)}, x_2^{(i)}, \ldots, x_p^{(i)})$ and its target value is denoted as $y^{(i)}$ for $i = 1, \ldots, n$. The goal of rescaling is to adjust the influence of the features on the target by finding the rescalor $w = (w_1, \ldots, w_p)$. Once w is determined, we rescale each feature by multiplying it by the corresponding weight:

$$\tilde{x}_j^{(i)} = w_j x_j^{(i)},$$

for $j = 1, \ldots, p$ and $i = 1, \ldots, n$. We propose four methods for determining the rescalor w.

2.5.1. The equal weight rescalor

The Equal Weight (EW) rescalor is simply a vector of ones, $w = (1, 1, \ldots, 1)$. The EW rescalor is simply serves as a benchmark, so that we can compare if other rescalors lead to a clustering scenario that helps to improve the prediction.

2.5.2. The linear regression rescalor

For the Linear Regression (REG) rescalor, we use the coefficients from multiple linear regression to determine the rescalor. The multiple linear regression model is

$$y^{(i)} = \beta_0 + \beta_1 x_1^{(i)} + \beta_2 x_2^{(i)} + \cdots + \beta_p x_p^{(i)} + \varepsilon_i,$$

where the error ε_i is normally distributed with mean zero and variance σ^2. The coefficients $\beta_0, \beta_1, \ldots, \beta_p$ are estimated by the least-squared estimation. The rescalor w_j equals the estimated value of β_j for $j = 1, \ldots, p$.

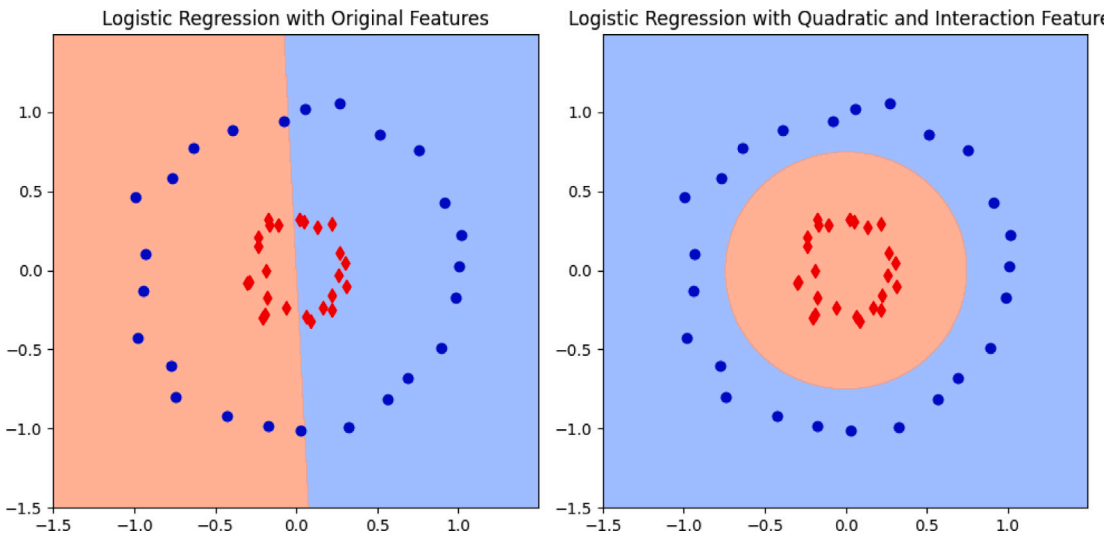

Fig. 5. Decision boundaries in Illustration 2.

2.5.3. The logistic regression rescalor

For the Logistic Regression (LR) rescalor, we consider the logistic regression because the target variable is binary in the credit scoring model. Let $Bernoulli(\pi)$ denote the Bernoulli distribution with probability π. In Logistic Regression, recall that we assume

$$y^{(i)}|x^{(i)} \sim Bernoulli\left(\frac{1}{1+e^{-\left(\beta_0+\beta_1 x_1^{(i)}+\beta_2 x_2^{(i)}\cdots+\beta_p x_p^{(i)}\right)}}\right),$$

where the maximum likelihood estimation estimates the coefficients. The rescalor w_j equals the estimated value of β_j for $j = 1, \ldots, p$.

2.5.4. The mutual information rescalor

The Mutual Information (MI) rescalor is inspired by the mutual information concept. Mutual information, also known as information gain, is a powerful tool rooted in information theory that quantifies the relationship between two variables. In the context of feature selection, MI can be invaluable in determining how relevant a given feature x_j is to the target variable y. Intuitively, MI measures how much knowing the value of one variable tells you about the other. If two variables are independent, their MI is zero.

Given the target variable y and the jth feature x_j, we denote $P_{x_j}(a)$ as the sample marginal probability of x_j taking on the value a. Similarly, $P_y(b)$ and $P_{x_j,y}(a,b)$ are defined. For discrete features, the MI is computed as:

$$MI(x_j, y) = \sum_{(a,b)} P_{x_j,y}(a,b) \log \frac{P_{x_j,y}(a,b)}{P_{x_j}(a)P_y(b)}, \tag{2}$$

where the summation over (a, b) spans all pairs of $(x_j^{(i)}, y^{(i)})$ for every i.

When x_j is a continuous feature with numerical values, we estimate the mutual information using the k-nearest neighbor method with $k = 3$ proposed by Ross (2014). Detailed calculations of MI for continuous features are omitted for the sake of brevity. Interested readers can find detailed treatments of this topic in Ross (2014).

2.6. Example 1: The PAK dataset

The PAK dataset is a private dataset from a Brazilian credit company and its partner shops, which is available at https://pakdd.org/archive/pakdd2010/PAKDDCompetition.html It contains 50,000 cases, with a target variable indicating default and 53 features, of which 40 are discrete, and 13 are continuous. The dataset also has an imbalanced data problem with a default rate of 26.1%. For detailed information regarding the PAK dataset, please refer to Appendix A. The final variables considered for subsequent analysis consist of the binary target variable y and the original features x_1 to x_{12}.

With these features, we employ Principal Component Analysis (PCA) as a dimensionality reduction technique to efficiently visualize the multidimensional data. In Fig. 6, the data is projected onto a two-dimensional plane defined by the first and second principal components. This visualization reveals a natural segregation of the data into four distinct clusters. Motivated by this observation, we hypothesize that developing separate predictive models for each of these clusters could enhance the overall predictive accuracy. The rationale behind this hypothesis is that data points within each cluster may share certain underlying characteristics, which could be exploited by a specialized model to make more accurate predictions.

Nonetheless, our preliminary analysis reveals that predictions based on clusters formed using these features do not result in a significant improvement in prediction accuracy. It is posited that this lack of improvement might be attributed to the application of Z-score normalization, which could be concealing the intrinsic importance of individual features by standardizing their scale. To address this issue, we propose alternative rescaling methods for the features in the subsequent sections, aimed at preserving their inherent significance and enhancing the predictive accuracy of the models.

Utilizing the PAK dataset, we embark on visualizing the features rescaled through the REG, LG, and MI methods using PCA. The visualizations are depicted in Fig. 6. Interestingly, a comparative analysis of these visualizations with those of the non-rescaled features (EW-rescaled) reveals markedly distinct patterns. Specifically, REG- and LR-rescaled features manifest six discernible clusters, whereas the MI-rescaled features exhibit a bifurcated structure with only two clusters. These differences underscore the impact of rescaling methods on the data distribution and clustering patterns.

2.7. Example 2: The GMC dataset

In our second experiment, we employ the 'Give Me Some Credit' (GMC) dataset, sourced from Kaggle, which comprises ten features alongside a target variable denoting credit defaults. The dataset is accessible at https://www.kaggle.com/competitions/GiveMeSomeCredit/data. It is noteworthy that the dataset is imbalanced, containing 150,000 observations, of which only 10,026 instances are credit defaults, translating to a default rate of approximately 6.7%. Summary statistics for the data after data preprocessing can be found in Appendix B. The processed dataset encompasses a target variable, y, and 12 features, denoted as $x_1, \ldots, x_{12}$.

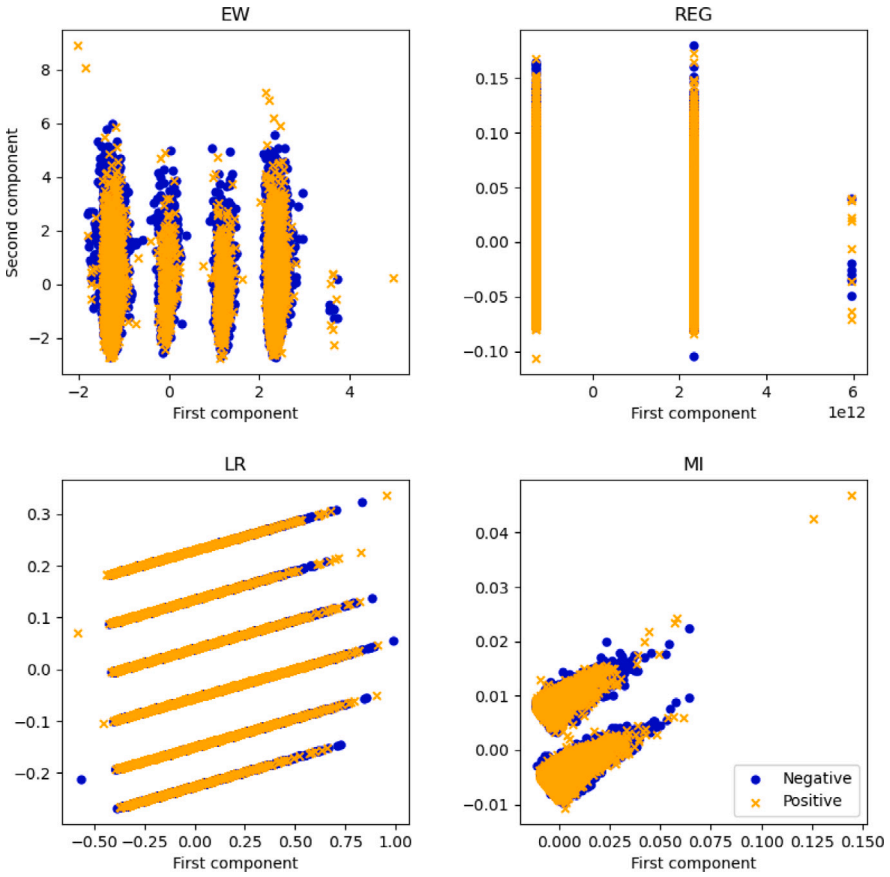

Fig. 6. PCA visualizations of the PAK dataset with various rescaling methods. Each subfigure represents the data projected onto the first and second principal components after applying a specific rescaling method.

In contrast to the distinct clustering observed in the PAK dataset shown in Fig. 7, the GMC dataset exhibits a different structure. Specifically, we observe four clusters, which are not clearly separated but rather exhibit overlaps. This serves as an alternative scenario and provides an opportunity to evaluate the robustness and versatility of our proposed cluster-then-predict approach in handling datasets with different characteristics. In the ensuing sections, we undertake a comprehensive analysis of these clusters and assess the impact of the observed patterns on predictive modeling".

Fig. 7 also illustrates the PCA visualizations of the REG-, LR-, and MI-rescaled features. A striking divergence in clustering patterns is observed when compared to the visualizations of the unscaled (EW-rescaled) features. In particular, the REG-scaled features give rise to approximately eight clusters, the LR-scaled features manifest close to 16 clusters, while the MI-scaled features yield around 18 clusters. These variations in clustering underscore the profound impact of feature scaling on the underlying data structure, which in turn holds implications for subsequent predictive modeling.

3. Experiment of the PAK data

In the feature engineering phase, quadratic and interaction terms are integrated into the model. Specifically, we incorporate x_i, x_i^2, and $x_i x_j$ terms for $i = 1, \ldots, p$, where $0 \leq i < j \leq p$ and p represents the number of features ($p = 12$ in the PAK dataset). To circumvent collinearity, cubic features are excluded from the analysis. The dataset is subsequently segmented into an 80–20 split for training and testing, respectively.

We proceed by computing four rescaling methods – EW, REG, LR, MI – and applying them to the features. The K-medoids clustering algorithm is adopted owing to its superiority in robustness compared to K-means (Choi & Kwon, 2015). Our preliminary analysis encompasses experiments with clusters ranging from $K = 2$ to $K = 6$, where $K = 1$ signifies no clustering. This stage is vital for assessing how clustering impacts default rates and entropy reduction within clusters.

As algorithms for classification, XGBoost is acknowledged for its advanced performance in credit scoring, while Logistic Regression is employed for its traditional simplicity in binary classification. We examine performance with $K = 1, 2, 3, 4$ clusters, where $K = 1$ denotes no clustering. The AUC metric, reflecting the area under the Receiver Operating Characteristic Curve, is used for evaluating classifier performance.

We outline our methodology as follows:

1. Data Preprocessing and Feature Engineering:
   (a) Preprocess features through logarithmic transformation, z-score normalization, and missing value imputation. The outcome is termed 'original features.'
   (b) Generate quadratic transformations of the original features, referred to as 'quadratic features.'
   (c) Compute EW, REG, LR, and MI rescalers for both original and quadratic features to derive rescaled features.
   (d) Partition the dataset into 80% training and 20% testing subsets.
2. Execute K-medoids Clustering:
   (a) Observe default rates for each cluster.
   (b) Investigate both the entropy of individual clusters and total entropy.
3. Develop K sub-models for Each Cluster using XGBoost and Logistic Regression:

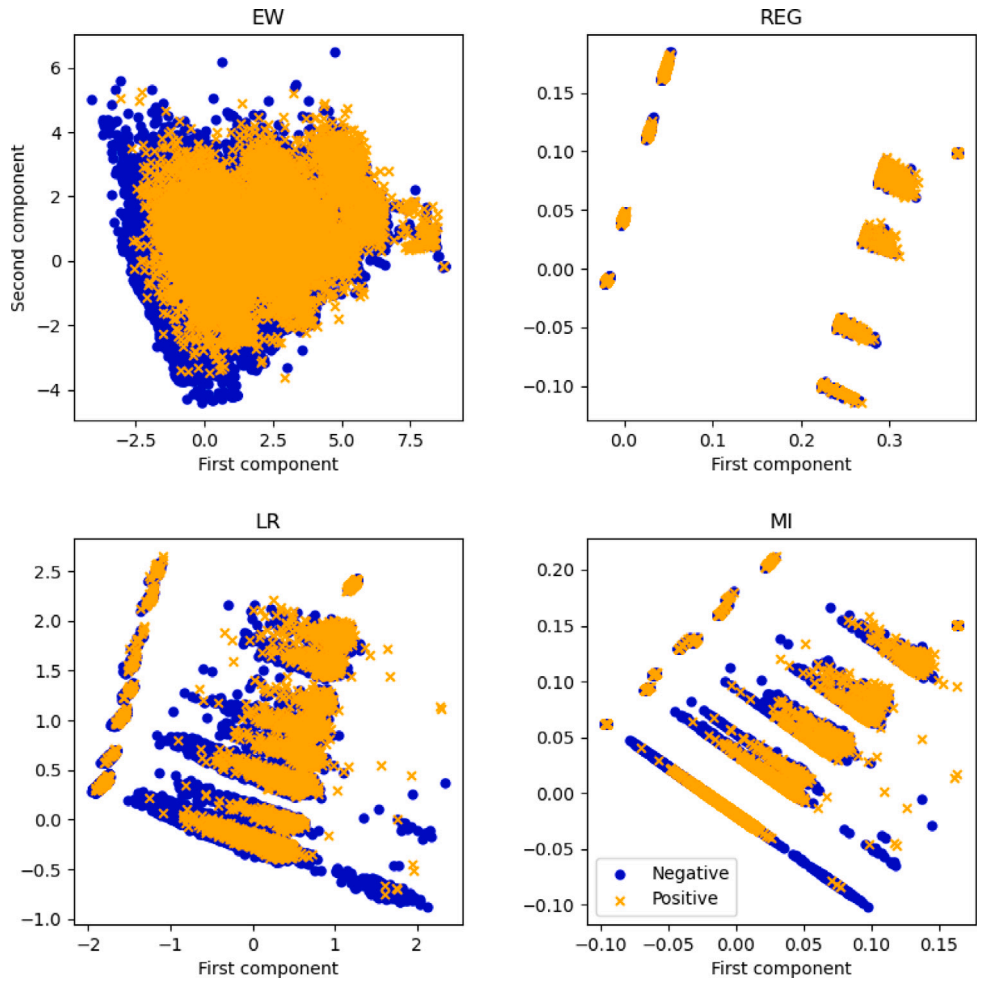

Fig. 7. PCA visualizations of the GMC dataset with various rescaling methods. Each subfigure represents the data projected onto the first and second principal components after applying a specific rescaling method.

(a) Allocate new samples to the nearest cluster and use the relevant sub-model for prediction.
(b) Compute AUC scores for each configuration on both the training and testing datasets.

3.1. Data exploration and clustering

Fig. 8 depicts the default rates for varying numbers of clusters ($K = 2, \ldots, 6$). Solid circles represent the training set, while hollow circles represent the testing set. The overall default rate of 26.1% is denoted by a horizontal black line. Notably, the default rates within individual clusters diverge from the overall default rate, as evidenced in both the training and testing datasets. Moreover, a close alignment is observed between default rates of individual clusters in the training and testing sets.

We further evaluate the efficacy of our clustering methodology by computing the total entropy, a metric rooted in information theory that quantifies the uncertainty or information content of a random variable or message (Gray, 2011). For the ith cluster with a default rate represented as p_i, the entropy is calculated as:

$$h_i = -\left(p_i \log(p_i) + (1 - p_i)\log(1 - p_i)\right).$$

A lower value of entropy within a cluster signifies a more concentrated distribution of default cases, indicative of higher proportions of either positive or negative instances. The total entropy is computed as the weighted sum of the individual entropies across K clusters, with the weights being the respective proportions of the clusters in the dataset:

$$h = \sum_{i=1}^{K} q_i h_i,$$

where q_i denotes the fraction of data points in the ith cluster. A lower total entropy points to more concentrated default cases within clusters, reflecting the efficacy of the clustering approach.

Fig. 9 illustrates the entropy values for different configurations in the PAK dataset with the number of clusters ranging from $K = 2$ to $K = 6$. The figure displays the entropy values for the EW-, REG-, LR-, and MI- rescaled features with solid lines representing the training dataset and dashed lines representing the testing dataset. The horizontal black solid and dashed lines indicate the entropy values for the training and testing datasets, respectively, in the scenario where $K = 1$, i.e., without clustering. The similarity between the entropy values for the training and testing sets substantiates the reliability of the entropy metric and the consistency across clusters. Notably, as the number of clusters increases, the entropy tends to decrease for both clustering methods when utilizing original features. We exclude the results for REG-rescaled quadratic features from further analysis as clustering proved ineffective.

3.2. Evaluating the impact of clustering on prediction accuracy

Fig. 10 contrasts the Area Under the Curve (AUC) obtained through XGBoost and Logistic Regression on training and testing sets when

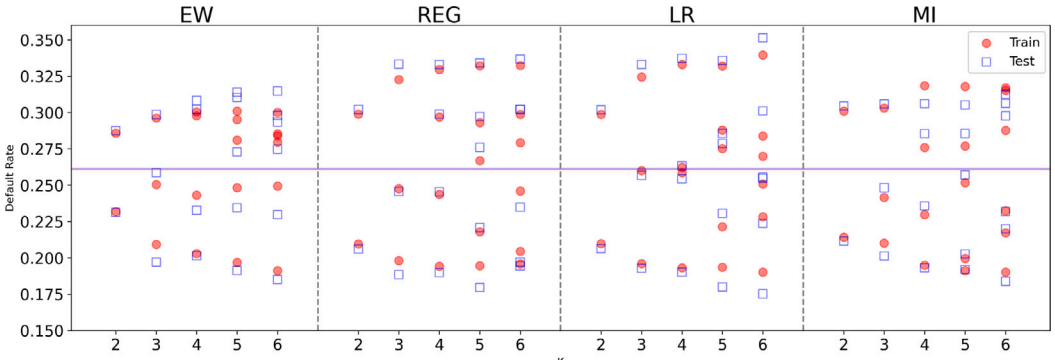

Fig. 8. Default rates comparison for $K = 2, \ldots, 6$ clusters using EW-, REG-, LG-, and MI-rescaled features in the PAK dataset. Solid circles and hollow squares denote training and testing sets, respectively. The horizontal black line represents the overall default rate of 26.1%.

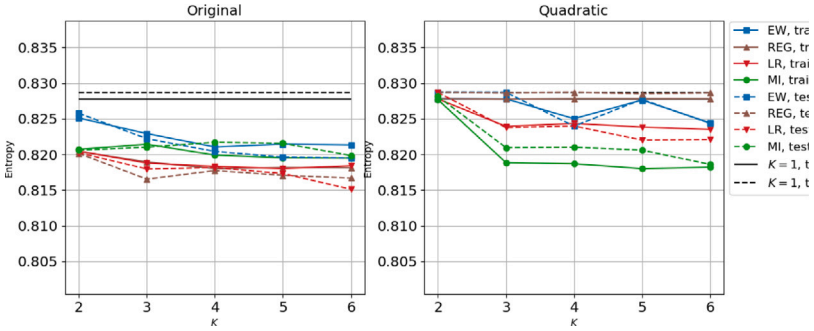

Fig. 9. Comparison of total entropies for $K = 2, \ldots$ clusters, utilizing EW-, REG-, LG-, and MI-rescaled features in the PAK dataset. Solid and dashed lines represent training and testing sets, respectively.

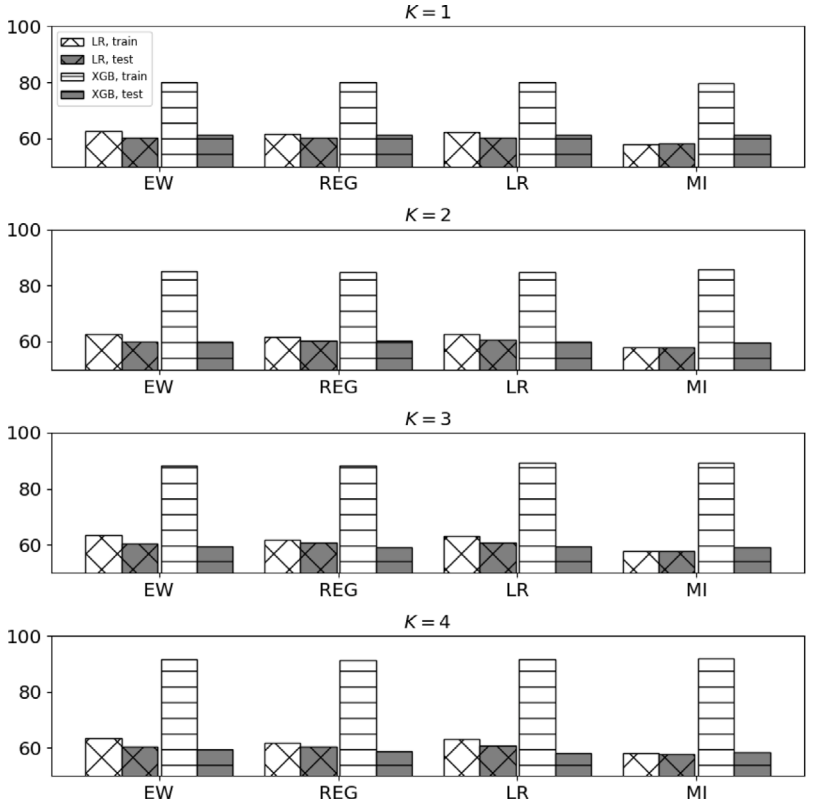

Fig. 10. Comparative analysis of AUC between XGBoost and Logistic Regression across various cluster sizes ($K = 1, 2, 3, 4$) using the EW, REG, LR, and MI rescaling on the PAK dataset.

Table 1
AUC measurements using EW, REG, LR, and MI rescalors applied to the original and quadratic feature sets in the PAK dataset's testing set. The benchmark AUCs for XGBoost and Logistic Regression, derived when $K = 1$ and utilizing the EW rescalor, are underlined. The highest AUC values for both XGBoost and Logistic Regression are highlighted in bold. For Logistic Regression, an asterisk denotes configurations that yield an AUC exceeding the maximum AUC attained by XGBoost.

K	XGBoost				Logistic Regression			
	EW	REG	LR	MI	EW	REG	LR	MI
The original feature set								
1	60.59	61.20	61.31	**61.32**	60.16	58.22	60.14	60.19
2	60.59	59.57	60.07	60.15	60.16	57.90	60.56	60.29
3	60.59	59.16	59.40	59.19	60.16	57.81	60.86	60.75
4	60.59	58.63	57.98	58.76	60.16	57.84	60.96	60.32
The quadratic feature set								
1	59.94	59.94	59.94	60.38	61.45*	**62.00***	61.99*	59.96
2	59.07		59.07	59.25	60.60		61.70*	59.69
3	58.81		58.57	59.03	60.01		61.93*	59.49
4	58.50		58.30	58.43	58.96		61.81*	59.09

applying the EW, REG, LR, and MI rescalors with varying numbers of clusters ($K = 1, 2, 3, 4$) based on the original feature set.

Initially, for the original feature set and $K = 1$ (indicating no clustering), XGBoost exhibits a substantially higher AUC in the training set, whereas a diminished AUC is observed in the testing set, hinting at overfitting. Conversely, Logistic Regression yields consistent AUC values across both sets, indicating stability. Notably, as K increases, the discrepancy in AUC between the training and testing sets for XGBoost widens, further accentuating its overfitting tendency. Logistic Regression, however, maintains its stability with increasing K. Shifting focus to the AUC in the testing set, XGBoost does not demonstrate an increase in AUC with escalating K across any of the rescaled features (EW-, REG-, LR-, MI-).

This is further evidenced when employing quadratic features; XGBoost exhibits a significantly elevated AUC in the training dataset compared to the testing dataset, reaffirming the overfitting issue. Contrastingly, Logistic Regression consistently exhibits comparable AUC values for training and testing datasets across all rescaled features with quadratic transformations, indicating robustness. Figures depicting these consistent findings for Logistic Regression have been omitted for the sake of brevity.

This analysis reveals that clustering does not inherently improve the predictive accuracy of XGBoost, which exhibits a susceptibility to overfitting, particularly as the complexity of features and number of clusters increases. On the other hand, Logistic Regression demonstrates stable performance and robustness across various feature rescaling techniques and cluster sizes.

We rigorously compare the AUC on the testing dataset to discern if clustering enhances performance. Table 1 presents the AUC for the testing set employing both the original and quadratic feature sets. As a values, for the original feature set with the EW rescaling (indicating no scaling), the XGBoost yield an AUC of value 60.59, and the Logistic Regression yields an AUC of value 60.16. Although the XGBoost produces higher AUC, it is simply a 0.71% change compared with the Logistic Regression.

For the original feature set, XGBoost attains a superior AUC compared to Logistic Regression when $K = 1$. The peak AUC for XGBoost, 61.32, is achieved with the MI-rescaled features. XGBoost's AUC exhibits a decline as K ascends. With the quadratic feature set, XGBoost's AUC is diminished compared to the original features and continues to wane as the dataset is partitioned into more clusters (i.e., with increasing K). For the original feature set, XGBoost attains a superior AUC compared to Logistic Regression when $K = 1$. The peak AUC for XGBoost, 61.32, is achieved with the MI-rescaled features. XGBoost's AUC exhibits a decline as K ascends. With quadratic features, XGBoost's AUC is diminished compared to the original features and continues to wane as the dataset is partitioned into more clusters (i.e., with increasing K).

Conversely, Logistic Regression's AUC escalates with increasing K for the original feature set. When employing quadratic features, Logistic Regression surpasses XGBoost at $K = 1$ for the EW-, REG-, and LR-rescaled features, signifying that quadratic transformations are beneficial for Logistic Regression. Generally, Logistic Regression maintains or increases its AUC with the escalation of K, especially with the LR rescaling, and proves superior to XGBoost for $K = 2, 3$. Notably, with the quadratic feature set and LR rescaling, Logistic Regression reaches an AUC of 61.99 at $K = 1$, a 1.11% enhancement. As K increases, Logistic Regression with LR rescaling sustains an AUC exceeding XGBoost's peak AUC. These observations reveal that quadratic transformations bolster Logistic Regression's performance, while clustering has a positive impact on Logistic Regression and a detrimental effect on XGBoost. Moreover, with appropriate rescaling, Logistic Regression can outshine XGBoost in terms of AUC.

In summary, Table 1 establishes that the efficacy of clustering in enhancing predictions is contingent upon (1) the machine learning algorithm, and (2) the rescaling method employed prior to clustering. XGBoost's AUC diminishes with additional clusters and quadratic features, whereas Logistic Regression's AUC thrives, particularly when the LR rescaling is applied.

3.3. Evaluating clustering strategies: All cases versus positive cases

Given that credit scoring models frequently grapple with class imbalance, this section evaluates and compares two distinct clustering approaches: all-case clustering and positive-case clustering. Both employ the K-medoids algorithm, albeit with differing strategies in determining medoids. Specifically, all-case clustering utilizes all cases for medoid identification, whereas positive-case clustering solely relies on positive cases. The rationale for the latter approach is grounded in the heightened importance of accurately identifying the positive cases, which often constitute the minority class. We conduct a comprehensive analysis to compare the outcomes of these clustering strategies.

Interestingly, both the default rates and entropies observed with positive-case clustering are akin to those with all-case clustering. This similarity in results also holds when employing quadratic features; therefore, we have elected to omit the corresponding figure for the sake of conciseness.

Table 2 juxtaposes the AUC obtained from XGBoost and Logistic Regression through positive-case clustering. This comparison incorporates EW-, REG-, LR-, and MI-rescaled original and quadratic feature sets for the test dataset. When contrasted with Table 1, both positive-case and all-case clustering strategies exhibit analogous AUCs. This suggests that focusing on clustering solely the positive cases, as opposed to all cases, attains comparable predictive accuracy.

In terms of XGBoost, incorporating quadratic features or escalating the number of clusters, even when features are rescaled, does not augment the AUC. Conversely, Logistic Regression demonstrates an enhancement in AUC when employing the quadratic feature set. Notably, with LR-rescaled features applied to both original and quadratic feature sets, Logistic Regression progressively increases its AUC as the value of

Table 2
AUC values utilizing the EW, REG, LR, and MI rescalers with positive-case clustering applied to the original and quadratic feature sets in the PAK dataset's testing set. The peak AUC for XGBoost and Logistic Regression are highlighted in bold. An asterisk beside values for Logistic Regression indicates configurations that outperform the highest AUC achieved by XGBoost.

K	XGBoost				Logistic Regression			
	EW	REG	LR	MI	EW	REG	LR	MI
The original feature set								
1	60.59	61.20	61.31	**61.32**	60.16	58.22	60.14	60.19
2	60.59	59.33	59.48	60.00	60.16	58.26	60.56	60.22
3	60.59	59.90	59.24	59.78	60.16	57.32	60.50	60.16
4	60.59	59.18	58.81	59.01	60.16	57.79	61.07	60.44
The quadratic feature set								
1	59.94	59.94	59.94	60.38	61.45*	62.00*	**61.99***	59.96
2	58.84		59.07	59.33	60.08		61.78*	59.77
3	59.11		58.04	58.77	59.58		61.89*	59.34
4	57.95		58.00	57.55	59.42		61.77*	58.79

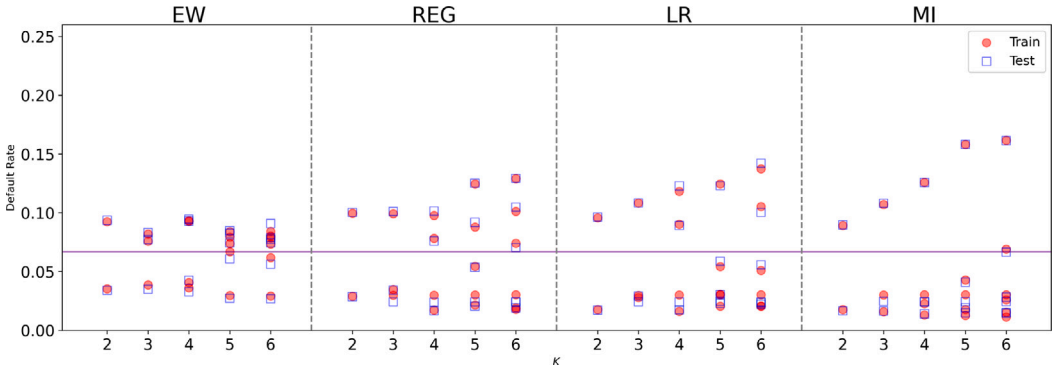

Fig. 11. Default rates comparison for $K = 2, \ldots, 6$ clusters using EW-, REG-, LG-, and MI-rescaled features in the GMC dataset. Solid circles and hollow squares denote training and testing sets, respectively. The horizontal black line represents the overall default rate of 6.7%.

K escalates. However, when using EW-, REG-, and MI-rescaled features, Logistic Regression's AUC remains relatively invariant. This implies that the selection of an apt rescaling technique is pivotal in harnessing the benefits for Logistic Regression.

4. Experiment of the GMC dataset

For the GMC dataset, we adopt the methodology outlined in Section 3 and extend it by incorporating cubic features and interactions. Specifically, the cubic feature set enhances the quadratic set by appending x_i^3, $x_i^2 x_j$, $x_i x_j^2$, and $x_i x_j x_k$ for $i = 1, \ldots, p$, $1 \le i < j \le p$, and $1 \le i < j < k \le p$. We employ the rescaled cluster-then-predict approach and juxtapose the outcomes of positive-clustering with all-case clustering.

4.1. Data exploration and clustering

Fig. 11 illustrates the default rates across clusters for $K = 2$ to $K = 6$ using solid circles for the training set and hollow squares for the testing set. A horizontal black line denotes the aggregate default rate of 6.7%. The proximity of solid and hollow symbols indicates that training and testing sets yield consistent and homogeneous clusters with comparable default rates. Similar observations hold for quadratic and cubic feature sets, so additional figures are omitted for brevity.

Fig. 12 portrays the entropy of clustering all cases for $K = 2, \ldots, 6$ after applying EW, REG, LG, and MI rescaling to the original feature set. The solid and dashed lines correspond to the entropy computed from the training and testing sets, respectively. Their similarity suggests a consistency in the clustering behavior across the training and testing sets.

As K increases, entropy diminishes, implying a reduction in data randomness with a higher number of clusters. Upon closer examination, we observe that clustering the EW-rescaled features results in higher entropy compared to REC-, LG-, and MI- rescaled features, suggesting that EW-rescaled features are less effective in minimizing randomness.

Both Figs. 11 and 12 illustrate that rescaled feature clustering can potentially decrease data randomness. This prompts a further investigation into whether constructing separate models based on these clusters can enhance prediction accuracy.

4.2. Evaluating the impact of clustering on prediction accuracy

Fig. 13 allows us to examine whether XGBoost or Logistic Regression is prone to overfitting. Each row presents the AUC for varying cluster numbers, $K = 1, 2, 3, 4$. XGBoost exhibits a significantly higher AUC in the training data than the testing data, suggesting overfitting, while Logistic Regression demonstrates robustness. This pattern remains consistent for $K = 1, 2, 3, 4$.

Even with the quadratic and cubic feature sets, XGBoost continues to show signs of overfitting, evidenced by a notably higher AUC in the training set compared to the testing set. Meanwhile, Logistic Regression maintains its robustness, delivering similar AUCs in both training and testing sets. These observations further underscore XGBoost's overfitting issue and Logistic Regression's reliability. As the AUCs for the quadratic and cubic feature sets closely resemble Fig. 13, they have been omitted for brevity.

Table 3 presents a comparison of the AUC between XGBoost and Logistic Regression on the testing set, using original, quadratic, and cubic feature sets, each rescaled by EW, REG, LR, and MI methods.

For the original feature set, while XGBoost yields a higher AUC than Logistic Regression for $K = 1$, further clustering does not enhance its AUC. As K increases, XGBoost's AUC declines. This pattern holds true for the quadratic and cubic feature sets, underscoring the difficulty in augmenting XGBoost performance using polynomial transformations or clustering, characteristic of its nature as an ensemble method.

Contrarily, Logistic Regression, with the original feature set, demonstrates an increase in AUC as K increases when EW and LR rescaling methods are used. In the context of the quadratic feature set, despite

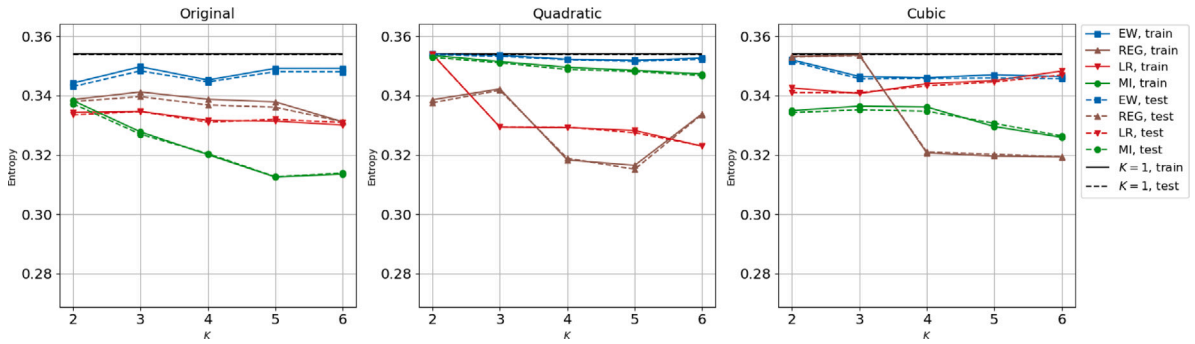

Fig. 12. Comparison of total entropies for $K = 2, \ldots$ clusters, utilizing EW-, REG-, LG-, and MI-rescaled features in the GMC dataset. Solid and dashed lines represent training and testing sets, respectively.

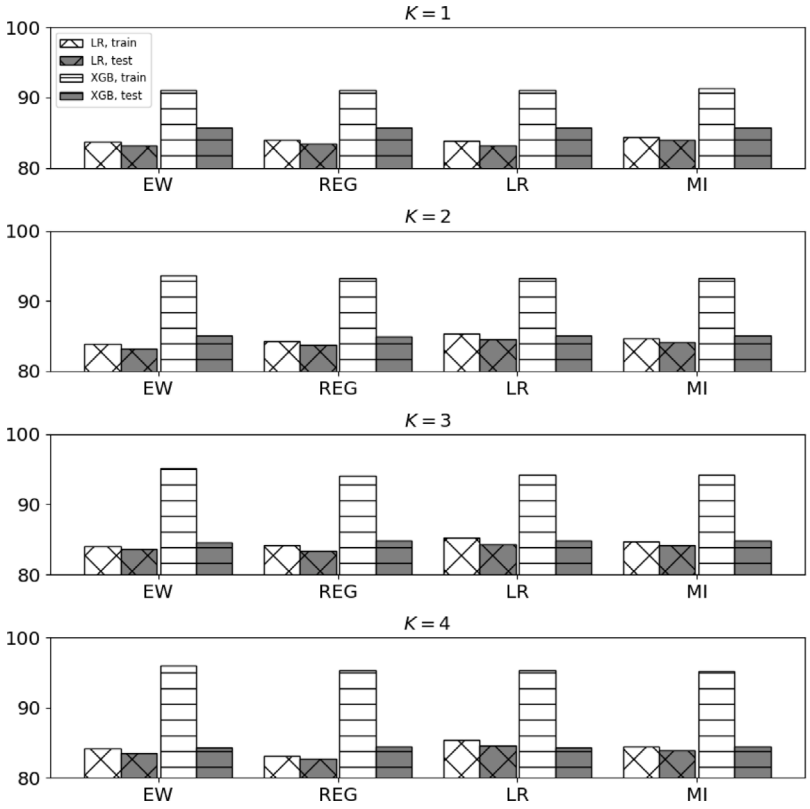

Fig. 13. Comparative analysis of AUC between XGBoost and Logistic Regression across various cluster sizes ($K = 1, 2, 3, 4$) using the EW, REG, LR, and MI rescaling on the PAK Dataset.

XGBoost's superior AUC at $K = 1$, an increase in K leads to a decline in XGBoost's AUC and an elevation in Logistic Regression's AUC, especially with LR rescaled features, which achieve the highest AUC at $K = 2, 3, 4$.

In the context of the cubic feature set, XGBoost initially outshines Logistic Regression when using EW, REG, and MI rescaled features at $K = 1$. However, as we increment K, the AUC for Logistic Regression enhances, reaching its zenith with LR rescaled features at $K = 2, 3, 4$. Notably, the maximum AUC is yielded by Logistic Regression when the LR rescalor is utilized with $K = 3$, resulting in an AUC of 85.52. This figure stands in close proximity to the highest AUC produced by XGBoost, which is 85.73.

Table 3 also substantiates that the efficacy of clustering for prediction enhancement hinges on (1) the selected machine learning algorithm and (2) the rescaling method employed before clustering. It highlights the decrease in XGBoost's AUC with an increase in clusters or complexity of quadratic and cubic features, while Logistic Regression's AUC improves. Of note, Logistic Regression with LR-rescaled features consistently outperforms its counterparts rescaled with EW, REG, and MI.

4.3. Evaluating clustering strategies: All cases versus positive cases

As seen in Section 3.3, we will now assess whether clustering solely positive cases can achieve a predictive accuracy akin to that obtained by clustering all cases. The default rates when implementing EW, REG, LR, and MI-rescaled original feature set clustering in the GMC dataset for $K = 2, \ldots, 6$ are depicted in Fig. 14. It is important to note that clustering based on positive cases results in a broader variety of default rates as opposed to all-case clustering, as illustrated in Fig. 11.

The entropy reduction using positive clustering proves to be more substantial compared to all-case clustering as seen in Fig. 9: The entropy drops to 0.28 when only positive cases are clustered, whereas it descends to about 0.32 in the case of all-case clustering. As evidenced

Table 3
AUC measurements using EW, REG, LR, and MI rescalors applied to the original, quadratic, and cubic feature sets in the GMC dataset's testing set. The benchmark AUCs for XGBoost and Logistic Regression, derived when $K = 1$ and utilizing the EW rescalor, are underlined. The highest AUC values for both XGBoost and Logistic Regression are highlighted in bold. For Logistic Regression, a plus denotes a configuration that yield the highest AUC.

K	XGBoost				Logistic Regression			
	EW	REG	LR	MI	EW	REG	LR	MI
The original feature set								
1	85.66	**85.73**	85.66	85.66	83.17	**84.02**	83.19	83.48
2	85.07	85.28	85.01	84.87	83.12	84.14	84.54	83.68
3	84.56	84.89	84.87	84.82	83.61	84.14	84.27	83.42
4	84.32	84.48	84.35	84.51	83.48	83.92	84.56	82.66
The quadratic feature set								
1	85.35	85.24	85.24	85.23	84.36	84.43	84.51	84.46
2	84.36	84.68	84.43	84.53	84.50	84.30	85.11	84.50
3	83.51	84.09	84.43	83.83	84.54	84.48	85.23	84.37
4	83.51	83.56	84.02	83.82	84.39	84.54	85.40	84.34
The cubic feature set								
1	84.87	84.87	84.77	84.87	83.44	84.72	85.26	82.51
2	84.05	84.17	83.78	84.24	83.67	84.78	85.51	82.55
3	83.59	83.24	83.15	83.49	83.44	84.75	85.52^{+}	82.63
4	82.65	82.85	82.60	82.86	83.48	84.81	85.42	84.04

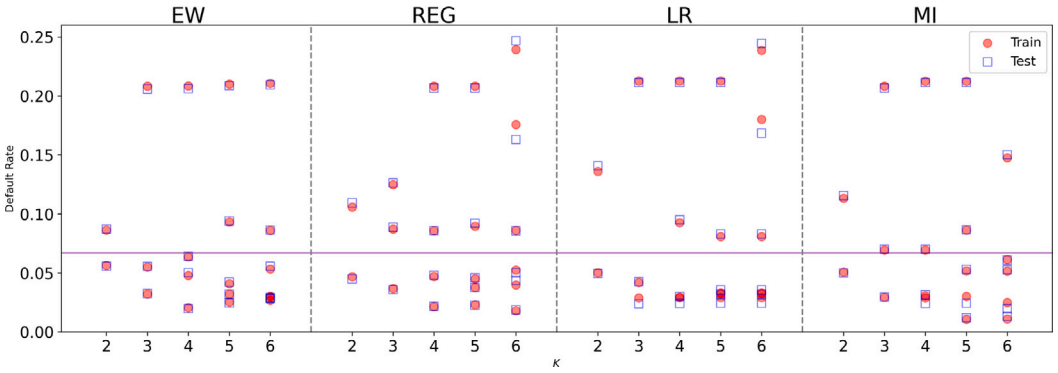

Fig. 14. Default rates comparison for $K = 2, \ldots, 6$ clusters using EW-, REG-, LG-, and MI-rescaled features in the GMC dataset when clustering only positive cases. Solid circles and hollow squares denote training and testing sets, respectively. The horizontal black line represents the overall default rate of 6.7%.

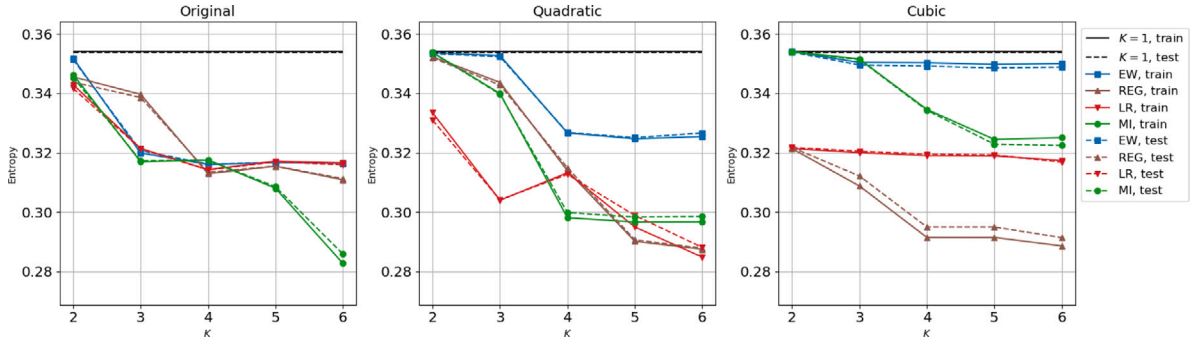

Fig. 15. Comparison of total entropies for $K = 2, \ldots$ clusters, utilizing EW-, REG-, LG-, and MI-rescaled features in the GMC dataset when clustering only positive cases. Solid and dashed lines represent training and testing sets, respectively.

by Figs. 14 and 15, exclusively clustering positive cases with rescaled features enhances the reduction in randomness to a notable extent.

Table 4 provides a comparison of AUC between XGBoost and Logistic Regression on the testing set, utilizing the original, quadratic, and cubic feature sets, each rescaled by EW, REG, LR, and MI methods when clustering only positive cases. This corroborates our conclusion from Table 3: XGBoost does not yield higher AUC values when using the quadratic or cubic feature sets, nor when K is increased. In contrast, Logistic Regression exhibits higher AUC values with the quadratic and cubic feature sets, particularly when the LR rescaling method is applied with increasing K. The highest achieved AUC value is 85.45, which is comparable to the highest AUC generated by XGBoost at 85.73.

5. Conclusion

In our study, we presented the "rescaled cluster-then-predict" method, which combines feature rescaling, informed by both features and the target variable, with clustering to develop bespoke predictive

Table 4
AUC measurements using EW, REG, LR, and MI rescalors applied to the original, quadratic, and cubic feature sets in the GMC dataset's testing set when clustering only positive cases. The benchmark AUCs for XGBoost and Logistic Regression, derived when $K = 1$ and utilizing the EW rescalor, are underlined. The highest AUC values for both XGBoost and Logistic Regression are highlighted in bold. For Logistic Regression, a plus denotes a configuration that yield the highest AUC.

Í	XGBoost				Logistic Regression			
K	EW	REG	LR	MI	EW	REG	LR	MI
The original feature set								
1	<u>85.66</u>	**85.73**	85.66	85.66	<u>83.17</u>	84.02	83.19	83.48
2	85.43	84.98	85.08	85.03	83.65	83.15	83.57	83.44
3	84.65	84.48	84.99	84.53	84.08	82.88	84.61	83.19
4	83.84	84.30	84.43	83.86	84.20	82.92	84.56	82.92
The quadratic feature set								
1	85.35	85.24	85.24	85.23	84.36	84.43	84.51	84.46
2	84.13	84.59	84.49	84.47	84.70	83.96	85.43	84.09
3	83.27	83.95	84.07	84.08	84.69	84.03	85.53	84.46
4	83.32	83.44	83.61	83.45	84.36	84.00	85.39	83.87
The cubic feature set								
1	84.87	84.87	84.77	84.87	83.44	84.72	85.26	82.51
2	84.05	84.17	83.78	84.24	83.00	84.66	85.27	82.74
3	83.59	83.24	83.15	83.49	83.14	84.73	85.39	82.55
4	82.65	82.85	82.60	82.86	83.17	84.84	**85.45**$^{+}$	82.60

models. This two-step method first rescales features using information from the target variable, then creates tailored models by clustering the data. Using two credit-scoring datasets, we revealed that clustering rescaled quadratic or cubic features boosts Logistic Regression's performance, even surpassing XGBoost's AUC on the test set. Additionally, we found that clustering only positive cases is as effective as clustering all cases, with the added advantage of computational efficiency.

The effectiveness of our "rescaled cluster-then-predict" method is intrinsically linked to the quality and representativeness of the training data. As data volumes escalate, the clustering aspect can become more computationally demanding, potentially delaying the credit scoring process—this challenge may be accentuated in real-time situations or when evaluating large sets of loan applications. However, our recommendation to cluster solely positive cases aims to mitigate these scalability concerns. Our method, while effective for credit scoring, might require adjustments for other domains. Data clusters may not remain stable if data distribution shifts, necessitating periodic retraining. With the financial realm's evolving nature, introducing new variables, such as digital transactions, requires method re-evaluation. Additionally, as financial regulations around AI evolve, continuous updates will be essential for compliance and ethics.

With growing demand for transparent yet efficient AI in finance, our research fills a timely gap. Traditional methods often sacrifice either predictability or transparency. Our approach balances both, standing out in the credit-scoring field. Unlike prior research that clusters uniformly treated features, we uniquely combine feature rescaling with target-based clustering.

Our "rescaled cluster-then-predict" method is versatile. While proven effective for specific credit products, its adaptability to various financial offerings, like mortgages or auto loans, is yet to be tested. It is suitable for binary tasks like fraud detection, but is not restricted to them; it can handle both categorical and continuous targets. This broadens its potential use to pricing, returns prediction, and portfolio or wealth management. While its adaptability across domains is uncharted, upcoming research could delve into these realms or introduce advanced rescaling techniques, elevating our method's relevance in credit scoring.

To highlighting the practical, social, and academic implications of our study, we note that financial institutions can enhance loan approval accuracy, increasing profitability and avoiding overloading potential defaulters. On a societal level, precise credit scoring grants deserving parties access to essential financial resources. Our approach, by reducing default risks and focusing on positive case clustering, promotes financial stability and expedited loan approvals, benefiting borrowers in urgent needs. Academically, our research presents a unique merger of feature rescaling and clustering, addressing a literature gap. By evaluating its performance against algorithms like XGBoost, we provide insights into modern credit scoring methods. Our focus on transparency and explainability also supports the growing interest in ethical and explainable AI in academia.

Funding

The authors gratefully acknowledge the financial support they received for this work. The National Yang Ming Chiao Tung University's Higher Education Sprout Project, which was funded by the Ministry of Education of Taiwan, provided essential resources. Additionally, the National Science and Technology Council of Taiwan significantly contributed to the funding through Grants 110-2115-M-A49-010-MY2, 111-2118-M-A49-007, and 112-2118-M-A49-001-MY2.

CRediT authorship contribution statement

Huei-Wen Teng: Conceptualization, Data curation, Formal analysis, Funding acquisition, Investigation, Methodology, Resources, Supervision, Validation, Formal analysis, Investigation, Writing – original draft, Writing – review & editing. **Ming-Hsuan Kang:** Formal analysis, Funding acquisition, Investigation, Methodology, Writing – original draft, Writing – review & editing. **I-Han Lee:** Data curation, Formal analysis, Investigation, Software, Validation, Visualization, Writing – original draft. **Le-Chi Bai:** Software, Validation, Visualization.

Data availability

Data will be made available on request.

Declaration of Generative AI and AI-assisted technologies in the writing process

During the preparation of this work the authors used in order to improve language and readability. After using this tool/service, the authors reviewed and edited the content as needed and take full responsibility for the content of the publication.

Appendix A. The PAK dataset

The PAK dataset comprises 50,000 cases with one default target and 53 features. Out of these, 40 are categorical and 13 are numerical. With 13,041 default cases, the default rate stands at 26.1%, indicating an imbalanced data issue. Table 5 describes each feature in detail. For the 40 categorical features, we clean data as follows. First, features with a single unique value, such as *CLERK_TYPE*, *EDUC_LEVEL*,

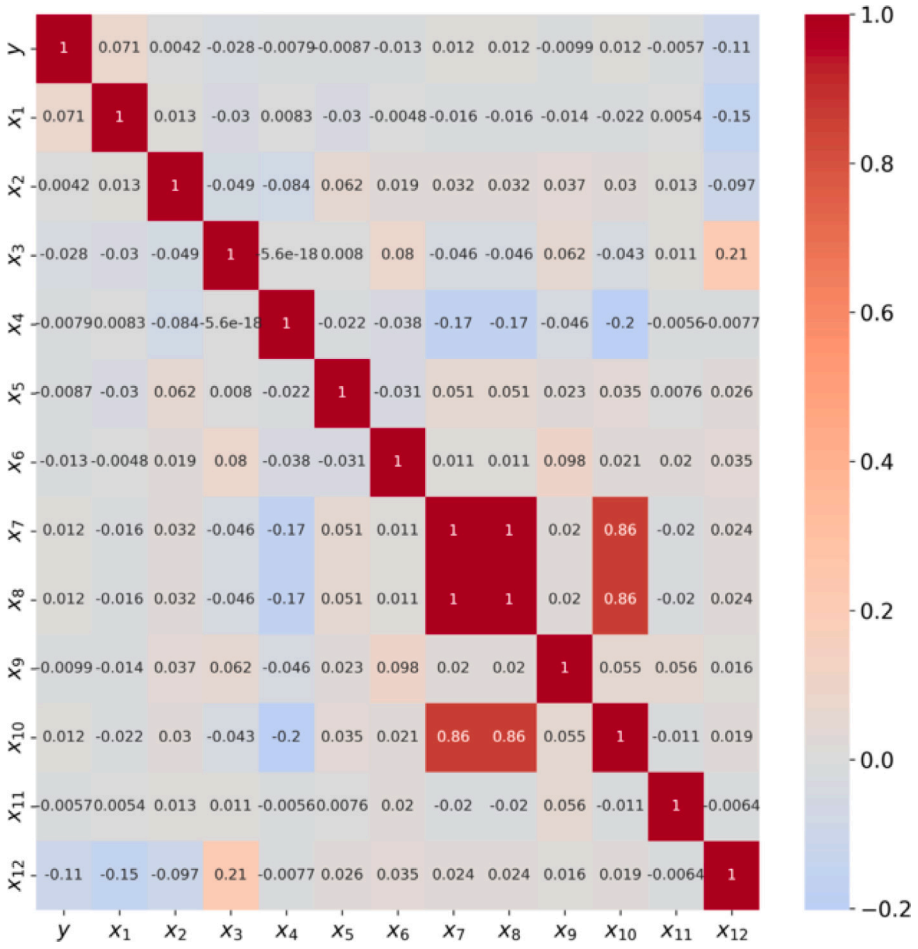

Fig. 16. A heatmap of pairwise correlations between features in PAK.

FLAG_MOBILE_PHONE, *FLAG_ADDRESS_DOCU*, *FLAG_RG*, *FLAG_CPF*, *FLAG_INCOME_PROOF*, and *FLAG_ACSP_RECORD*, are removed. Second, region-related features like *STATE_BIRTH*, *CITY_BIRTH*, *NACIONALITY*, *RESID_STATE*, *RESID_CITY*, *RESID_BOROUGH*, *RESID_PHONE_CODE*, *PROF_STATE*, *PROF_CITY*, *PROF_BOROUGH*, *PROF_PHONE_CODE*, *RESID_ZIP*, and *PROF_ZIP* are excluded, given their common analysis with graphical models. The two numerical features, *ID* and *N_ADD_CARDS*, are removed. *ID* is irrelevant to the target response, and *N_ADD_CARDS* has only one unique value.

Table 6 summarizes descriptive statistics of used features. Most features have complete values, except that *MONTHS_IN_RESID* have a missing ratio of 7.6%.

Based on Section 2.4, we apply the following data pre-processing:

1. Features like *N_DEPENDANTS*, *MONTH_INCOME*, *OTHER_INCOMES*, *ASSETS*, and *MONTHS_IN_JOB* with kurtosis over 10 undergo log transformation.
2. Original numerical features receive z-score scaling.
3. For *MONTHS_IN_RESID* with missing values, we set them to zero and introduce a dummy variable to indicate their absence.

Table 7 lists the transformed features post-processing.

Table 8 summarizes the post-processed numerical features.

Using the heatmap in Fig. 16, we identify high pairwise correlations: x_7 and x_8 have a correlation of 1, while both pairs x_{10} and x_7, and x_{10} and x_8, have a correlation of 0.86. Thus, we remove x_8 and x_{10}.

Appendix B. The GMC dataset

The GMC dataset has 150,000 cases, a default target, and ten numerical features. The competition aims to predict the default probability, assisting banks in loan decisions. Of the 150,000 cases, 10,026 are defaults, resulting in a 6.7% rate, highlighting an imbalanced data issue. Table 9 details the features.

Table 10 summarizes the raw data. While most features are complete, *MonIncome* and *NOfDenpendents* have missing rates of 19.8% and 2.6%, respectively.

Data pre-processing is vital for effective modeling. Following Section 2.4, we outline our observations:

1. To address high skewness and kurtosis, features like *RevolveUtilize*, *30-59DaysPastDue*, *DebtRatio*, *MonIncome*, *90DaysLate*, *NRealEstateLoans*, and *60-89DaysPastDue* with kurtosis over 10 undergo log transformation.
2. We apply z-score scaling.
3. For *MonIncome* and *NOfDependents* with missing data, we introduce indicators x_6 and x_{12} respectively, and replace missing values with zeros.

Post-processed features are detailed in Table 11.

Table 12 presents statistics for the target (y) and the 12 post-processed features ($x_1, \ldots, x_{12}$).

We examine pairwise correlations using the heatmap in Fig. 17. While x_4 and x_6 have a correlation of 0.74, it is less concerning as x_6 is a manually created dummy variable.

Table 5
Descriptions and types of each feature in the original data.

No	Feature	Description	Type
1	*ID*	Sequential number for the applicant (to be used as a key)	Numerical
2	*CLERK_TYPE*	Not informed	Categorical
3	*PAY_DAY*	Day of the month for bill payment, chosen by the applicant	Numerical
4	*SUBMISSION_TYPE*	Indicates if the application was submitted via the internet or in person/posted	Categorical
5	*N_ADD_CARDS*	Quantity of additional cards asked for in the same application form	Numerical
6	*ADDRESS_TYPE*	Indicates if the address for posting is the home address or other. Encoding not informed.	Categorical
7	*SEX*	Not informed	Categorical
8	*MARITAL*	Encoding not informed	Categorical
9	*N_DEPENDANTS*	Not informed	Numerical
10	*EDUC_LEVEL*	Educational level in gradual order not informed	Categorical
11	*STATE_BIRTH*	Not informed	Categorical
12	*CITY_BIRTH*	Not informed	Categorical
13	*NACIONALITY*	Country of birth. Encoding not informed but Brazil is likely to be equal 1.	Categorical
14	*RESID_STATE*	State of residence	Categorical
15	*RESID_CITY*	City of residence	Categorical
16	*RESID_BOROUGH*	Borough of residence	Categorical
17	*FLAG_RESID_PHONE*	Indicates if the applicant possesses a home phone	Categorical
18	*RESID_PHONE_CODE*	Three-digit pseudo-code	Categorical
19	*RESID_TYPE*	Encoding not informed. In general, there are the types: owned, mortgage, rented, parents, family etc.	Categorical
20	*MONTHS_IN_RESID*	Time in the current residence in months	Numerical
21	*FLAG_MOBILE_PHONE*	Indicates if the applicant possesses a mobile phone	Categorical
22	*FLAG_EMAIL*	Indicates if the applicant possesses an e-mail address	Categorical
23	*MONTH_INCOME*	Applicant's personal regular monthly income in Brazilian currency (R$)	Numerical
24	*OTHER_INCOMES*	Applicant's other incomes monthly averaged in Brazilian currency (R$)	Numerical
25	*FLAG_VISA*	Flag indicating if the applicant is a VISA credit card holder	Categorical
26	*FLAG_MASTERCARD*	Flag indicating if the applicant is a MASTERCARD credit card holder	Categorical
27	*FLAG_DINERS*	Flag indicating if the applicant is a SINERS credit card holder	Categorical
28	*FLAG_AMER_EXP*	Flag indicating if the applicant is an AMERICAN EXPRESS credit card holder	Categorical
29	*FLAG_OTHER_CARDS*	Despite being label "FLAG", this field presents three values not explained	Categorical
30	*N_BANK_ACCOUNTS*	Not informed	Numerical
31	*N_SPECIAL_ACCOUNTS*	Not informed	Numerical
32	*ASSETS*	Total value of the personal possessions such as houses, cars etc. in Brazilian currency (R$).	Numerical
33	*N_CARS*	Quantity of cars the applicant possesses	Numerical
34	*COMPANY*	If the applicant has supplied the name of the company where he/she formally works	Categorical
35	*PROF_STATE*	State where the applicant works	Categorical
36	*PROF_CITY*	City where the applicant works	Categorical
37	*PROF_BOROUGH*	Borough where the applicant works	Categorical
38	*FLAG_PROF_PHONE*	Indicates if the professional phone number was supplied	Categorical
39	*PROF_PHONE_CODE*	Three-digit pseudo-code	Categorical
40	*MONTHS_IN_JOB*	Time in the current job in months	Numerical
41	*PROF_CODE*	Applicant's profession code. Encoding not informed	Categorical
42	*OCCUP_TYPE*	Encoding not informed	Categorical
43	*MATE_PROF_CODE*	Mate's profession code. Encoding not informed	Categorical
44	*MATE_EDUC_LEVEL*	Mate's educational level in gradual order not informed	Categorical
45	*FLAG_ADDRESS_DOCU*	Flag indicating documental confirmation of home address	Categorical
46	*FLAG_RG*	Flag indicating documental confirmation of citizen card number	Categorical
47	*FLAG_CPF*	Flag indicating documental confirmation of tax payer status	Categorical
48	*FLAG_INCOME_PROOF*	Flag indicating documental confirmation of income	Categorical
49	*PRODUCT*	Type of credit product applied. Encoding not informed	Categorical
50	*FLAG_ACSP_RECORD*	Flag indicating if the applicant has any previous credit delinquency	Categorical
51	*AGE*	Applicant's age at the moment of submission	Numerical
52	*RESID_ZIP*	Three most significant digits of the actual home zip code	Categorical
53	*PROF_ZIP*	Three most significant digits of the actual job zip code	Categorical
54	*TARGET*	Target Variable: BAD=1, GOOD=0	Categorical

Table 6
Summary statistics of the original data in PAK.

Feature	Mean	Std	Min	50%	Max	Skewness	Kurtosis
TARGET	0.3	0.4	0.0	0.0	1.0	1.1	−0.8
PAY_DAY	12.9	6.6	1.0	10.0	25.0	0.5	−0.6
N_DEPENDANTS	0.7	1.2	0.0	0.0	53.0	4.1	83.2
MONTHS_IN_RESID	9.7	10.7	0.0	6.0	228.0	1.9	9.1
MONTH_INCOME	886.7	7.8×10^3	60.0	500.0	9.6×10^5	85.7	9×10^3
OTHER_INCOMES	35.4	891.5	0.0	0.0	1.9×10^5	207.3	4.5×10^4
N_BANK_ACCTS	0.4	0.5	0.0	0.0	2.0	0.6	−1.6
N_SPEC_ACCTS	0.4	0.5	0.0	0.0	2.0	0.6	−1.6
ASSETS	2.3×10^3	4.2×10^4	0.0	0.0	6×10^6	103.7	1.3×10^4
N_CARS	0.3	0.5	0.0	0.0	1.0	0.7	−1.5
MONTHS_IN_JOB	0.0	0.4	0.0	0.0	35.0	62.4	4.5×10^3
AGE	43.2	15.0	6.0	41.0	106.0	0.5	−0.4

Table 7
Notations, feature, and pre-processing method for each features in PAK.

Syntax	Feature	Description
y	*TARGET*	Original target
x_1	*PAY_DAY*	Taken z-score scaling
x_2	*LogN_DEPENDANTS*	Taken log transformation and z-score scaling
x_3	*MONTHS_IN_RESID*	Taken z-score scaling
x_4	*ISNA_MONTHS_IN_RESID*	Missing indicator for *MONTHS_IN_RESID*
x_5	*LogMONTH_INCOME*	Taken log transformation and z-score scaling
x_6	*LogOTHER_INCOMES*	Taken log transformation and z-score scaling
x_7	*N_BANK_ACCTS*	Taken z-score scaling
x_8	*N_SPEC_ACCTS*	Taken z-score scaling
x_9	*LogASSETS*	Taken log transformation and z-score scaling
x_{10}	*N_CARS*	Taken z-score scaling
x_{11}	*LogMONTHS_IN_JOB*	Taken log transformation and z-score scaling
x_{12}	*AGE*	Taken z-score scaling

Table 8
Processed PAK dataset: A summary statistics overview.

	Mean	Std	Min	50%	Max	Skewness	Kurtosis
y	0.3	0.4	0.0	0.0	1.0	1.1	−0.8
x_1	0.0	1.0	−1.8	−0.4	1.8	0.5	−0.6
x_2	0.0	1.0	−0.7	−0.7	1.9	0.7	−1.4
x_3	0.0	1.0	−0.9	−0.3	20.5	2.0	10.1
x_4	0.1	0.3	0.0	0.0	1.0	3.2	8.3
x_5	0.0	1.0	−3.7	−0.2	12.0	1.7	7.4
x_6	−0.0	1.0	−0.2	−0.2	5.7	3.9	13.3
x_7	−0.0	1.0	−0.7	−0.7	3.4	0.6	−1.6
x_8	−0.0	1.0	−0.7	−0.7	3.4	0.6	−1.6
x_9	0.0	1.0	−0.2	−0.2	5.9	4.3	16.4
x_{10}	0.0	1.0	−0.7	−0.7	1.4	0.7	−1.5
x_{11}	0.0	1.0	−0.0	−0.0	28.3	25.7	658.5
x_{12}	0.0	1.0	−2.5	−0.2	4.2	0.5	−0.4

Table 9
Feature in the GMC dataset.

No	Feature	Description	Type
1	*Target*	Person experienced 90 days past due delinquency or worse	Binary
2	*RevolveUtilize*	Total balance on credit cards and personal lines of credit except real estate and no installment debt like car loans divided by the sum of credit limits	Numerical
3	*Age*	Age of borrower in years	Numerical
4	*30-59DaysPastDue*	Number of times borrower has been 30-59 days past due but no worse in the last 2 years	Numerical
5	*DebtRatio*	Monthly debt payments, alimony,living costs divided by monthly gross income	Numerical
6	*MonIncome*	Monthly income	Numerical

(*continued on next page*)

Table 9 (*continued*).

No	Feature	Description	Type
7	*NOfCredit&Loan*	Number of Open loans (installment like car loan or mortgage) and Lines of credit (e.g. credit cards)	Numerical
8	*90DaysLate*	Number of times borrower has been 90 days or more past due	Numerical
9	*NRealEstateLoans*	Number of mortgage and real estate loans including home equity lines of credit	Numerical
10	*60-89DaysPastDue*	Number of times borrower has been 60-89 days past due but no worse in the last 2 years.	Numerical
11	*NOfDependents*	Number of dependents in family excluding themselves (spouse, children etc.)	Numerical

Table 10
Summary statistics of the raw data in the GMC dataset.

Feature	Mean	Std	Min	50%	Max	Skewness	Kurtosis
Target	0.1	0.2	0.0	0.0	1.0	3.5	10.0
RevolveUtilize	6.0	249.8	0.0	0.2	5.1×10^4	97.6	14 544.7
Age	52.3	14.8	0.0	52.0	109.0	0.2	−0.5
30-59DaysPastDue	0.4	4.2	0.0	0.0	98.0	22.6	522.4
DebtRatio	3.5×10^2	2.0×10^3	0.0	0.4	3.3×10^5	95.2	13 734.3
MonIncome	6.7×10^3	1.4×10^4	0.0	5400.0	3.0×10^6	114.0	19 504.7
NOfCredit&Loan	8.5	5.1	0.0	8.0	58.0	1.2	3.1
90DaysLate	0.3	4.2	0.0	0.0	98.0	23.1	537.7
NRealEstateLoans	1.0	1.1	0.0	1.0	54.0	3.5	60.5
60-89DaysPastDue	0.2	4.2	0.0	0.0	98.0	23.3	545.7
NOfDependents	0.8	1.1	0.0	0.0	20.0	1.6	3.0

Table 11
Gost-processed features in GMC dataset.

Notation	Feature	Description
y	*Target*	Original target
x_1	*LogRevolveUtilize*	Taken log transformation and z-score scaling
x_2	*Age*	Taken z-score scaling
x_3	*Log30-59DaysPastDue*	Taken log transformation and z-score scaling
x_4	*LogDebtRatio*	Taken log transformation and z-score scaling
x_5	*LogMonIncome*	Taken log transformation and z-score scaling
x_6	*ISNAMonIncome*	Dummy variable indicates whether MonIncome is missing value
x_7	*NOfCredit&Loan*	Taken z-score scaling
x_8	*Log90DaysLate*	Taken log transformation and z-score scaling
x_9	*LogNRealEstateLoans*	Taken log transformation and z-score scaling
x_{10}	*Log60-89DaysPastDue*	Taken log transformation and z-score scaling
x_{11}	*NOfDependents*	Taken z-score scaling
x_{12}	*ISNANOfDependents*	Dummy variable indicates whether NOfDependents is missing value

Table 12
Summary of post-processed GMC dataset statistics.

	Mean	Std	Min	50%	Max	Skewness	Kurtosis
y	0.1	0.2	0.0	0.0	1.0	3.5	10.0
x_1	−0.0	1.0	−3.1	0.3	3.9	−2.1	4.2
x_2	−0.0	1.0	−3.5	−0.0	3.8	0.2	−0.5
x_3	0.0	1.0	−0.4	−0.4	3.0	1.9	1.5
x_4	0.0	1.0	−2.4	−0.2	3.3	0.7	0.6
x_5	−0.0	0.9	−7.6	0.0	3.3	−6.9	54.4
x_6	0.2	0.4	0.0	0.0	1.0	1.5	0.3
x_7	−0.0	1.0	−1.6	−0.1	9.6	1.2	3.1
x_8	0.0	1.0	−0.2	−0.2	5.2	3.9	13.3
x_9	−0.0	1.0	−1.3	0.7	1.3	−0.5	−1.7
x_{10}	0.0	1.0	−0.2	−0.2	5.5	4.1	15.1
x_{11}	0.0	1.0	−0.7	−0.7	17.3	1.6	3.2
x_{12}	0.0	0.2	0.0	0.0	1.0	5.9	33.3

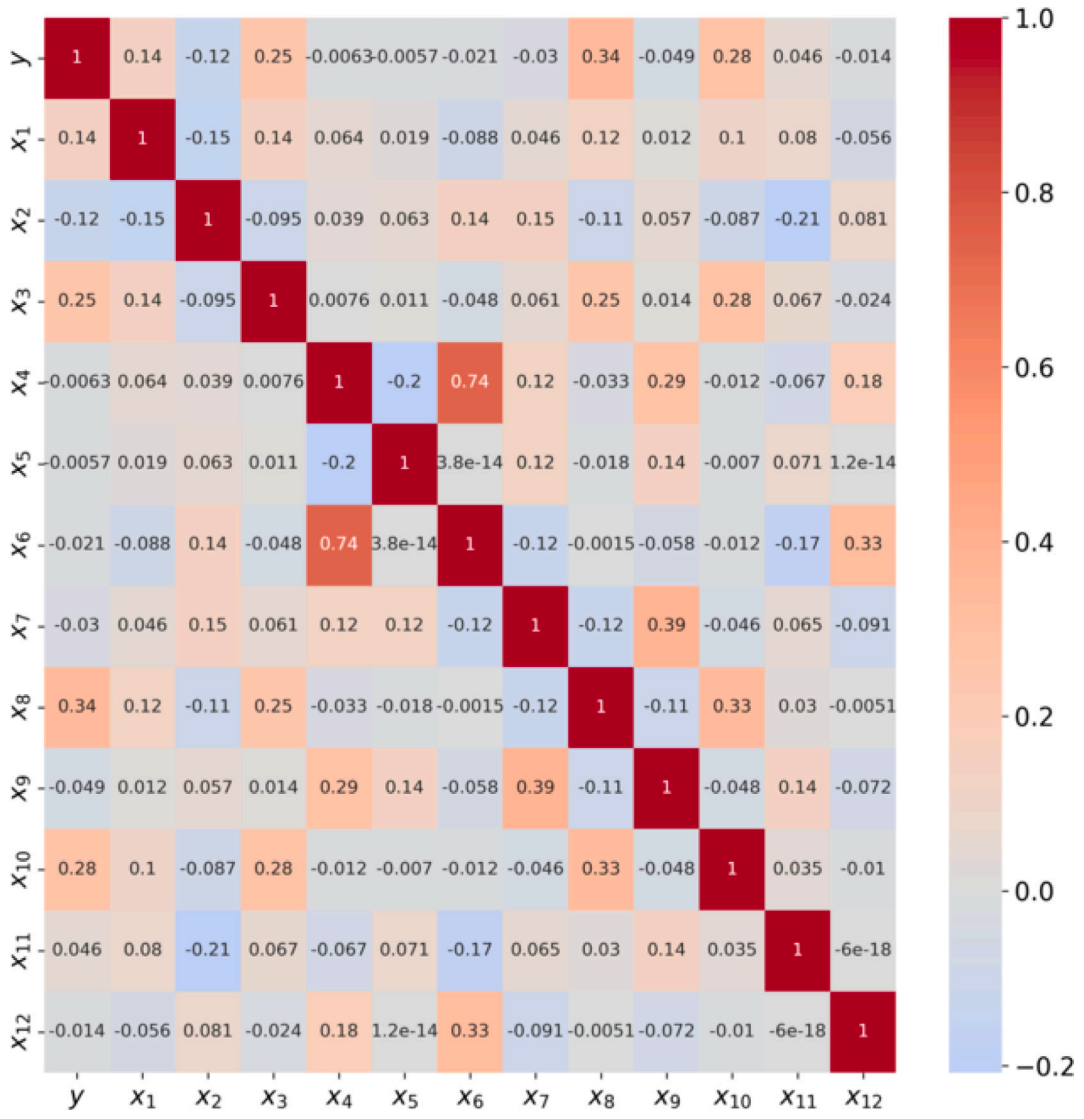

Fig. 17. Heatmap of pairwise correlations for post-processed GMC dataset features.

References

Aggarwal, C. C. (2015). *Data mining: The textbook*. Springer.

Arrieta, A. B., Díaz-Rodríguez, N., Del Ser, J., Bennetot, A., Tabik, S., Barbado, A., et al. (2020). Explainable Artificial Intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI. *Information Fusion*, *58*, 82–115.

Baesens, B., Van Gestel, T., Viaene, S., Stepanova, M., Suykens, J., & Vanthienen, J. (2003). Benchmarking state-of-the-art classification algorithms for credit scoring. *Journal of the Operational Research Society*, *54*(6), 627–635.

Berkhin, P. (2006). A survey of clustering data mining techniques. In *Grouping multidimensional data: Recent advances in clustering* (pp. 25–71). Springer.

Brundage, M., Avin, S., Clark, J., Toner, H., Eckersley, P., Garfinkel, B., et al. (2018). *The malicious use of artificial intelligence: Forecasting, prevention, and mitigation*: *Technical report*, arXiv preprint arXiv:1802.07228.

Bussmann, N., Giudici, P., Marinelli, D., & Papenbrock, J. (2020). Explainable AI in fintech risk management. *Frontiers in Artificial Intelligence*, *3*, 26.

Chang, Y. C., Chang, K. H., & Wu, G. J. (2018). Application of extreme gradient boosting trees in the construction of credit risk assessment models for financial institutions. *Applied Soft Computing*, *73*, 914–920.

Chen, C., Lin, K., Rudin, C., Shaposhnik, Y., Wang, S., & Wang, T. (2018). An interpretable model with globally consistent explanations for credit risk. arXiv preprint arXiv:1811.12615.

Chen, C., & Shyu, M. L. (2011). Clustering-based binary-class classification for imbalanced data sets. In *2011 IEEE international conference on information reuse & integration* (pp. 384–389). IEEE.

Choi, J., & Kwon, H. J. (2015). The information filtering of gene network for chronic diseases: Social network perspective. *International Journal of Distributed Sensor Networks*, *11*(9), Article 736569.

Demajo, L. M., Vella, V., & Dingli, A. (2020). Explainable AI for interpretable credit scoring. arXiv preprint arXiv:2012.03749.

Fernández, A., García, S., Galar, M., Prati, R. C., Krawczyk, B., & Herrera, F. (2018). *Learning from imbalanced data sets, vol. 10*. Springer.

Freund, Y., Schapire, R., & Abe, N. (1999). A short introduction to boosting. *Journal-Japanese Society for Artificial Intelligence*, *14*(771–780), 1612.

Friedman, J. H. (2001). Greedy function approximation: A gradient boosting machine. *Annals of Statistics*, *29*(5), 1189–1232.

García, S., Luengo, J., & Herrera, F. (2015). *Data preprocessing in data mining*. Springer.

Géron, A. (2022). *Hands-on machine learning with scikit-learn, keras, and tensorFlow*. O'Reilly Media, Inc.

Goodman, B., & Flaxman, S. (2017). European union regulations on algorithmic decision-making and a "right to explanation". *AI Magazine*, *38*(3), 50–57.

Gray, R. M. (2011). *Entropy and information theory*. Springer Science & Business Media.

Hand, D. J., & Henley, W. E. (1997). Statistical classification methods in consumer credit scoring: a review. *Journal of the Royal Statistical Society: Series A (Statistics in Society)*, *160*(3), 523–541.

Hastie, T., Tibshirani, R., Friedman, J. H., & Friedman, J. H. (2009). *The elements of statistical learning: Data mining, inference, and prediction, vol. 2*. Springer.

He, H., & Garcia, E. A. (2009). Learning from imbalanced data. *IEEE Transactions on Knowledge and Data Engineering*, *21*(9), 1263–1284.

Heaton, J. (2016). An empirical analysis of feature engineering for predictive modeling. In *SoutheastCon 2016* (pp. 1–6). IEEE.

James, G., Witten, D., Hastie, T., & Tibshirani, R. (2013). *An introduction to statistical learning, vol. 112*. Springer.

Krawczyk, B. (2016). Learning from imbalanced data: open challenges and future directions. *Progress in Artificial Intelligence*, *5*(4), 221–232.

Lessmann, S., Baesens, B., Seow, H. V., & Thomas, L. C. (2015). Benchmarking state-of-the-art classification algorithms for credit scoring: An update of research. *European Journal of Operational Research*, *247*(1), 124–136.

Liu, W., Fan, H., & Xia, M. (2022). Credit scoring based on tree-enhanced gradient boosting decision trees. *Expert Systems with Applications*, *189*, Article 116034.

Menard, S. (2002). *Applied logistic regression analysis, no. 106*. Sage.

Misheva, B. H., Osterrieder, J., Hirsa, A., Kulkarni, O., & Lin, S. F. (2021). Explainable AI in credit risk management. arXiv preprint arXiv:2103.00949.

Peikari, M., Salama, S., Nofech-Mozes, S., & Martel, A. L. (2018). A cluster-then-label semi-supervised learning approach for pathology image classification. *Scientific Reports*, *8*(1), 1–13.

Qin, C., Zhang, Y., Bao, F., Zhang, C., Liu, P., & Liu, P. (2021). XGBoost optimized by adaptive particle swarm optimization for credit scoring. *Mathematical Problems in Engineering*, *2021*, 1–18.

Qiu, W. (2019). Credit risk prediction in an imbalanced social lending environment based on xgboost. In *2019 5th international conference on big data and information analytics* (pp. 150–156). IEEE.

Radu, V., Katsikouli, P., Sarkar, R., & Marina, M. K. (2014). A semi-supervised learning approach for robust indoor-outdoor detection with smartphones. In *Proceedings of the 12th ACM conference on embedded network sensor systems* (pp. 280–294).

Ross, B. C. (2014). Mutual information between discrete and continuous data sets. *PLoS One*, *9*(2), Article e87357.

Schapire, R. E. (1999). A brief introduction to boosting. In *Ijcai, vol. 99* (pp. 1401–1406). Citeseer.

Selbst, A., & Powles, J. (2018). "Meaningful information" and the right to explanation. In S. A. Friedler, & C. Wilson (Eds.), *Proceedings of machine learning research: vol. 81*, *Proceedings of the 1st conference on fairness, accountability and transparency* (p. 48). PMLR, URL: https://proceedings.mlr.press/v81/selbst18a.html.

Sun, Y., Wong, A. K., & Kamel, M. S. (2009). Classification of imbalanced data: A review. *International Journal of Pattern Recognition and Artificial Intelligence*, *23*(04), 687–719.

Teng, H. W., & Lee, M. (2019). Estimation procedures of using five alternative machine learning methods for predicting credit card default. *Review of Pacific Basin Financial Markets and Policies*, *22*(03), Article 1950021.

Thomas, L., Crook, J., & Edelman, D. (2017). *Credit scoring and its applications*. SIAM.

Tsai, C. F. (2014). Combining cluster analysis with classifier ensembles to predict financial distress. *Information Fusion*, *16*, 46–58.

Xia, Y., Liu, C., Li, Y., & Liu, N. (2017). A boosted decision tree approach using Bayesian hyper-parameter optimization for credit scoring. *Expert Systems with Applications*, *78*, 225–241.