Research Square

Preprints are preliminary reports that have not undergone peer review.
They should not be considered conclusive, used to inform clinical practice,
or referenced by the media as validated information.

Reject Inference in P2P Lending Credit Scoring: A Semi-Supervised Logistic–CEM Approach

Abderrahim El amrani

aelamrani@insea.ac.ma

Institut National de Statistique et d'Economie Appliquée

Badreddine Benyacoub

Institut National de Statistique et d'Economie Appliquée

Mohammed El Haj Tirari

Institut National de Statistique et d'Economie Appliquée

Research Article

Keywords: Credit scoring, Reject inference, Semi-Supervised Learning, P2P lending, Logistic-Regression, MAR

Posted Date: June 11th, 2026

DOI: https://doi.org/10.21203/rs.3.rs-9945746/v1

License: This work is licensed under a Creative Commons Attribution 4.0 International License.
Read Full License

Additional Declarations: No competing interests reported.

Reject Inference in P2P Lending Credit Scoring: A Semi-Supervised Logistic–CEM Approach

Abderrahim EL AMRANI[1*], Badreddine BENYACOUB[2] and Mohammed EL HAJ TIRARI[3]

[1,2,3]Research Laboratory in Applied Methods in Statistics, Actuarial Science, Finance and Quantitative Economics, INSEA, Avenue Allal El Fassi, , P.O. Box 6217 , Rabat, Morocco.

*Corresponding author(s). E-mail(s): aelamrani@insea.ac.ma;
Contributing authors: bbenyacoub@insea.ac.ma; tirari@insea.ac.ma;

Abstract

Credit scoring models usually train only on accepted applicants, leading to sample selection bias. In peer-to-peer lending, where many applicants get rejected, this bias becomes a major issue because the training data does not accurately reflect the entire applicant population. Reject inference methods aim to estimate outcomes for rejected applicants and integrate them into model training. Most current methods depend on interpolation, which allows some of the original bias from the accepted sample to remain. This paper builds on Anderson's modified logistic regression method for reject inference by adding the CEM algorithm to directly include rejected applicants in the training process. Unlike Anderson's study, which used simulated rejection with German credit data, this research works with a real peer-to-peer lending dataset that includes actual rejected applications. The results demonstrate strong performance in both reject inference and credit scoring tasks.

Keywords: Credit scoring, Reject inference, Semi-Supervised Learning, P2P lending, Logistic-Regression, MAR

1 Introduction

The rise of peer-to-peer (P2P) lending platforms has changed the credit ecosystem. They connect borrowers and lenders through online platforms. LendingClub has invested hundreds of billions of dollars in loans worldwide, providing an alternative to traditional banks. According to 2024 report, the total loans granted increased by 13 percent, reaching 1.85 billion dollars compared to 1.63 billion dollars the previous year[1]. While this option benefits borrowers, it poses challenges for lenders and investors, as they might lose their equity. Lending platforms need to create a model that predicts whether a loan applicant will repay or default. Building such a model is essential for researchers. In practical applications, even a 1% improvement in identifying borrowers with risky credit profiles can greatly improve capital security for both P2P lending platforms and investors [1]. To tackle this problem, lending institutions use credit risk scoring. This process assigns a score to each client to indicate their eligibility for a loan. In business terms, this is known as the probability of default.

Various approaches can be used for credit scoring, including expert judgment, statistical methods, and machine learning. The first approach can be sensitive to subjective biases Thomas [2]. To overcome this limitation, statistical/machine learning methods were introduced such as discriminant analysis, linear regression, and logistic regression, have been proposed Hand and Henley [1] Bijak and Thomas [3]. From a statistical/machine learning perspective, credit risk scoring is a binary classification problem where borrowers are classified into two categories: '0' for good payers, who pose minimal risk to the platform, and '1' for bad payers, who are likely to cause financial losses. Numerous methods have been studied in the literature to improve credit risk scoring, including linear regression Horrigan [4], West [5], discriminant analysis Altman [6], neural networks Khashman [7], decision trees Hongxia et al. [8], probit and logit regression Hwang et al. [9], Kaplan and Urwitz [10], and support vector machines (SVM) Bellotti and Crook [11], Chen and Li [12]. Credit scoring models are often trained only on accepted datasets. This approach ignores important information in rejected datasets, leading to sample bias. The dataset we are using has an 8% acceptance rate, which means after preprocessing the data, we get less than 8% of the entire population to be included in training the model. As a result, the sample does not truly represent the overall population. To address this issue, we use reject inference. This process involves using information from rejected loan applicants to improve credit scoring models. It helps to correct the bias that comes from training solely on accepted data. Several methods can be used, including reweighting Bücker et al. [13], augmentation Banasik and Crook [14], and semi-supervised learning Li et al. [15].

Choosing semi-supervised learning for this problem was driven by the large number of unlabeled data points (27,648,741) compared to the number of labeled data points (2,260,701). Semi-supervised learning (SSL) includes various algorithms, approaches, and methods Hady and Schwenker [16].

One of the most common SSL methods relies on generative models Zhu and Goldberg [17]. These models aim to learn the data distribution by assuming it comes

[1] https://ir.lendingclub.com/news/news-details/2025/LendingClub-Reports-Fourth-Quarter-and-Full-Year-2024-Results/

out of several probability densities, with each density weighted to reflect its importance in the overall population Hand and Everitt [18]. The learning process uses the Expectation-Maximization (EM) algorithm Newcomb [19].

Although mixture models are primarily unsupervised and used for clustering, we can modify them for classification based on the semi-supervised cluster assumption. This means we assume that each cluster mostly contains data points from the same class. By adding a classification step (C-step) to the EM algorithm, we can use the mixture model to both cluster the data and assign class labels simultaneously [20].

As the dimensionality increases, generative models become less efficient. One proposed method involves transforming them into discriminative models Vittaut et al. [21], particularly with logistic regression Amini and Gallinari [22]. This paper builds on both works: semi-supervised logistic regression Amini and Gallinari [22] and [23], by applying the latter to a real-world dataset. The main difference between our study and theirs is in the rejected dataset's nature: their dataset does not include rejected observations, and the reject sample is created artificially. In contrast, our framework is evaluated directly on a real reject inference dataset from a peer-to-peer (P2P) lending context. Results from the numerical analysis let us reconsider the use of this framework in reject inference scenarios.

To explore these aspects, we first present the theoretical foundations of semi-supervised logistic regression, followed by an applied framework. The structure of the paper is organized as follows. **Section 1** reviews the relevant literature by discussing prior research in the field of reject inference and credit scoring. **Section 2** delves into the methodology of semi-supervised logistic regression. **Section 3** presents the numerical experiments, where we evaluate the proposed model on the Lending Club peer-to-peer lending dataset through repeated training on randomly balanced samples. We compare the results with several widely used credit scoring models and assess statistical significance using the Friedman test and Nemenyi post-hoc analysis. This section also examines how different amounts of unlabeled data affect the results under two distinct scenarios. Finally, **Section 4** concludes the paper by summarizing the main findings and outlining directions for future research.

2 Literature review

Credit scoring methods have improved significantly over the years, with researchers exploring various methodologies to improve accuracy and precision. Early approaches relied heavily on statistical methods, including linear regression Horrigan [4], West [5], and discriminant analysis Altman [6]. As computational capabilities advanced, more sophisticated algorithms have been proposed, such as neural networks Khashman [7], decision trees Hongxia et al. [8], probit and logit regression Hwang et al. [9], Kaplan and Urwitz [10], and support vector machines (SVM) Bellotti and Crook [11], Chen and Li [12].

Semi-supervised learning (SSL) has gained attention as an alternative for credit scoring, particularly for leveraging unlabeled data. Kim [24] demonstrated that SSL significantly improves default risk prediction in P2P lending even with a limited number of labeled samples. Similar to other statistical models, we use the probabilistic

framework where we model the conditional density $p(y|x)$, which can give us the default probability of a credit seeker to be a good or a bad payer. Since it's a partially semi-supervised framework, we can see this reject inference framework as a missing-data problem where we model $p(z|x, y)$ where z is a latent variable that represents the class label. There are three primary types of missing data mechanisms:

- **MNAR (Missing Not at Random):** $p(z|x, y) \neq p(z|x)$ indicates that the missing data is related to the unobserved value itself.
- **MAR (Missing at Random):** $p(z|x, y) = p(z|x)$ means that the likelihood of missing data may be influenced by observed factors, such as income, age and loan amount.
- **MCAR (Missing Completely at Random):** $p(z|x, y) = p(z)$, meaning the probability of missing data is unrelated to both observed and unobserved variables. It typically arises from random errors or accidental data loss.

The dominant assumption underlying most proposed solutions is that rejections follow a Missing at Random (MAR) mechanism: lenders deny credit based on observed applicant characteristics and internal thresholds, not on latent factors invisible to the model Feelders [25] was among the first to formalise a solution under this assumption, using mixture modelling estimated via the Expectation–Maximisation (EM) algorithm to infer the default risk of rejected applicants. Anderson and Hardin [23] later extended this line of work by combining EM with logistic regression and testing it on real consumer lending data, moving beyond the synthetic datasets used in earlier studies.

Semi-supervised learning has since become a popular alternative framing. Rather than imputing outcomes for rejected cases, semi-supervised methods treat them as unlabelled examples and exploit their feature distributions during training. Li et al. [15] and Shen et al. [26] both developed semi-supervised SVM (S3VM) approaches for reject inference and reported gains over standard supervised classifiers. Kennedy et al. [27] took a different angle, applying a one-class classifier to the accept/reject setting and showing it outperformed the conventional two-class approach. Liu et al. [28] went further by combining multiple classifiers with clustering and a dynamic selection mechanism that picks the best classifier per instance, improving on baseline reject inference methods.

Several researchers have applied Bayesian and probabilistic frameworks to the problem. Mancisidor et al. [29] proposed two Bayesian formulations that incorporate rejected applications into model training and reported better performance than classical benchmarks. Anderson [30] used Bayesian networks to similar effect. Hájek and Olej [31] built a kernel-based hybrid of supervised and semi-supervised learning for corporate and municipal credit scoring. In the peer-to-peer lending context,[32] used Graph-based semi-supervised learning approaches, including label propagation and label spreading credit scoring by modeling rejected applicants as unlabeled nodes in a similarity graph and propagating labels from accepted applicants, El Annas et al. [33] proposed a Hidden Markov Model-based semi-supervised framework and showed it outperformed standard methods.

Other contributions have focused on feature selection and graph-based representations. Xia et al. [34] combined pessimistic likelihood estimation with gradient boosting

and found that selectively pruning low-confidence samples improved performance across several datasets. Xiao et al. [35] proposed a semi-supervised feature selection method using the Group Method of Data Handling (GMDH). Kang et al. [36] introduced a graph-based semi-supervised model and pointed out a problem that often goes unacknowledged: standard semi-supervised methods can be biased in the reject inference setting because rejected applicants are not drawn from the same distribution as accepted ones. Most classical frameworks assume uniform error behaviour across both groups, which does not hold in practice.

3 Semi-supervised Logistic Classification Expectation Maximisation (CEM) algorithm

3.1 Formulation of the Log-Likelihood for EM-Algorithm

Let $D_l = \{(x_i, z_i)\}_{i=1}^{n_l}$ represent the labeled dataset, where $x_i \in \mathbb{R}^d$ is the feature vector and $z_i \in \{1, \ldots, M\}$ is the class label. Let $D_u = \{x_i\}_{i=n_l+1}^{n}$ denote the unlabeled dataset, and let M represent the number of classes.

Under the cluster assumption in semi-supervised learning Zhu and Goldberg [17], we assume that if two points x_i and x_j lie in the same cluster, they are likely to belong to the same class. Therefore, we treat the classes as clusters.

The log-likelihood function for the entire dataset is given by:

$$LL_c = \sum_{i=1}^{n} \sum_{m=1}^{M} z_{im} \log \left(\pi_m p(x_i|\theta_m) \right), \tag{1}$$

where z_{im} represents the binary indicator variable for the class membership of x_i, and π_m is the prior probability of class m.

For the labeled data, where z_{im} is known for $i = 1, \ldots, n_l$, we can rewrite the log-likelihood as Dempster et al. [37]:

$$LL_c = \sum_{i=1}^{n_l} \sum_{x_i \in P_m} \log \left(\pi_m p(x_i|\theta_m) \right) + \sum_{i=n_l+1}^{n} \sum_{m=1}^{M} z_{im} \log \left(\pi_m p(x_i|\theta_m) \right). \tag{2}$$

Generative models often do not work well in high-dimensional data. To deal with this, we use a discriminative approach, which models the conditional distribution directly and performs better in such cases. **J.-N. Vittant et al.** [21] showed that the Classification Expectation-Maximization (CEM) algorithm can be combined effectively with discriminative models. Then, the complete log-likelihood can be written as:

$$LL_c = \underbrace{\sum_{m=1}^{M} \sum_{x_i \in P_m} \log P(P_m \mid x_i, \theta_m) + \sum_{m=1}^{M} \sum_{i=n_l+1}^{n} z_{im} \log P(P_m \mid x_i, \theta_m)}_{\widehat{LL_c}} + \sum_{i=1}^{n} \log P(x_i), \tag{3}$$

The corresponding penalized log-likelihood is given by:

$$\widehat{LL_c} = \sum_{m=1}^{2} \sum_{x_i \in P_m} \log P(P_m \mid x_i, \theta_m) + \sum_{m=1}^{2} \sum_{i=n_l+1}^{n} z_{im} \log P(P_m \mid x_i, \theta_m) - \lambda \sum_{m=1}^{2} ||\theta_m||^2 \tag{4}$$

The inclusion of ℓ_1-regularization serves to control model complexity and to select only the most informative features.

By defining the posterior probability as:

$$P(P_k \mid \mathbf{x}_i, \boldsymbol{\theta}_k) = \frac{1}{1 + \exp\Big(- \boldsymbol{\beta}_k^\top \mathbf{x}_i \Big)} = \frac{1}{1 + \exp\Big(- \big(\beta_{0,k} + \sum_{j=1}^{d} \beta_{j,k} x_{ij}\big)\Big)} \tag{5}$$

where:

- $\mathbf{x}_i = (x_{i1}, \ldots, x_{id})$ is the feature vector of observation i,
- $\beta_{0,k}$ is the intercept for class k,
- $\beta_{j,k}$ is the coefficient for feature j and class k,
- $\boldsymbol{\theta}_k = (\beta_{0,k}, \ldots, \beta_{d,k})$ collects all parameters for class k.

We obtain a semi-supervised logistic regression model. This formulation was originally proposed by **Amini & Gallinari** [22] as an extension of the Classification Expectation-Maximization algorithm. By integrating a logistic regression-based discriminative approach within the CEM framework.
The principle of the CEM-algorithm is summarized in three steps: the E-step is obtained using the formula 5, the C-step is deduced from the same formula, and the M-step is obtained by maximizing the formula 4.

3.2 Derivation of the Discriminative Log-Likelihood

Starting from the complete-data log-likelihood in the mixture model framework:

$$LL_c = \sum_{i=1}^{n} \sum_{m=1}^{M} z_{im} \log \left[\pi_m p(x_i \mid \theta_m)\right]$$

Using Bayes' theorem, we can rewrite:

$$\pi_m p(x_i \mid \theta_m) = p(P_m, x_i) = P(P_m \mid x_i) \cdot p(x_i)$$

Taking logarithms:

$$\log \left[\pi_m p(x_i \mid \theta_m)\right] = \log P(P_m \mid x_i) + \log p(x_i)$$

Substituting back:

$$LL_c = \sum_{i=1}^{n} \sum_{m=1}^{M} z_{im} \log P(P_m \mid x_i) + \sum_{i=1}^{n} \sum_{m=1}^{M} z_{im} \log p(x_i)$$

Since $\sum_{m=1}^{M} z_{im} = 1$ for each i:

$$LL_c = \sum_{i=1}^{n} \sum_{m=1}^{M} z_{im} \log P(P_m \mid x_i) + \sum_{i=1}^{n} \log p(x_i)$$

The first term is the discriminative log-likelihood (LL_c), while the second term is constant with respect to the model parameters β, thus can be ignored during optimization.

The proposed framework begins with two cleaned datasets: the labeled and the unlabeled samples. Initially, a supervised logistic regression model is trained on the labeled data to obtain a preliminary estimation of the regression coefficients $\boldsymbol{\beta}$. This initial model is then used to generate a first partition of the data (C-step) through the logistic function $\sigma(\cdot)$. Subsequently, the E-step is performed using the same probabilistic outputs $\sigma(\cdot)$, allowing the computation of the expected complete-data log-likelihood. The complete likelihood function is then reformulated to incorporate both the labeled and unlabeled observations. Finally, during the M-step, this likelihood is maximized to re-estimate the optimal parameters $\boldsymbol{\beta}$.

To determine convergence, we employ an early stopping strategy based on two criteria with a small tolerance parameter ϵ. The algorithm is terminated when either the absolute change in the estimated complete-data log-likelihood or the absolute change in the model coefficients falls below ϵ for three consecutive iterations:

$$\left|\widehat{LL_c}^{(t+1)} - \widehat{LL_c}^{(t)}\right| < \epsilon \quad \text{or} \quad \left|\beta_k^{(t+1)} - \beta_k^{(t)}\right| < \epsilon. \tag{6}$$

Now that we've seen an overview of the methodology, let's take a closer look at the core of the approach. Here is the detailed algorithm 1 that implements this process:

Algorithm 1 Penalized Logistic-CEM Model

Initialization: Train a penalized logistic model $G^{(0)}(x)$ over D_l:

$$G^{(0)}(x) = \sigma\left(\beta_0^{(0)} + \sum_{i=1}^{d} \beta_i^{(0)} x_i\right)$$

Where $\sigma(z) = \frac{1}{1+e^{-z}}$

Let $P^{(0)}$ be the initial partition of $D_l \cup D_u$ based on $G^{(0)}$. Initialize $\widehat{LL}_C^{(0)}$ as the log-likelihood of this initial model.

While: $|\widehat{LL}_C^{(j+1)} - \widehat{LL}_C^{(j)}| > \epsilon$, **for each iteration** $0 \leq j \leq J_{max}$:

- **E-Step:** Estimate posterior probabilities for each $x_i \in D_u$, $i = n+1, \ldots, n+m$:

$$z^{(j)}(x_i) = \beta_0^{(j)} + \sum_{k=1}^{d} \beta_k^{(j)} x_{ik}$$

- **C-Step:** Assign each x_i to the cluster $P_k^{(j+1)}$ based on maximum posterior:

$$z_{ki}^{(j+1)} = \begin{cases} 1 & \text{if } p_k^{(j)}(x_i) > 0.5 \\ 0 & \text{otherwise} \end{cases} \quad \text{for } k = 1, 2$$

- **M-Step:** Maximize the complete log-likelihood:

$$\begin{aligned} \widehat{LL}_C(P^{(j+1)}, \beta^{(j)}) = & \sum_{x_i \in P_1} \log\Big(\sigma(z^{(j)}(x_i))\Big) \\ & + \sum_{x_i \in P_2} \log\Big(1 - \sigma(z^{(j)}(x_i))\Big) \\ & + \lambda \sum_{i=1}^{d} \left\| \beta_i^{(j)} \right\|^2 . \end{aligned}$$

- **Early Stopping C :** Stop if

$$|\widehat{LL}_C^{(j+1)} - \widehat{LL}_C^{(j)}| < \epsilon$$

  is verified for three consecutive iterations.

Return: $\beta_0^{final}, \beta_1^{final}, \ldots, \beta_N^{final}$

4 Experimental setup

4.1 Data, features, and preprocessing

Our numerical experiment is based on a dataset from Lending Club, spanning the period from 2007 to 2018. The data includes both accepted and rejected loan applications with 2.260.701 accepted loans and 27.648.741 rejected loans.
the accepted dataset contains 150 features [38], and the rejected dataset contains only 6 features:

- **Loan Amount:** The total amount of the loan requested by the borrower. If the credit department reduces the loan amount at any point, this adjustment will be reflected in the recorded value.
- **Employment Length:** The duration of the borrower's employment, measured in years. Values range from 0 (less than one year) to 10 (ten or more years).
- **Debt-to-Income Ratio (DTI):** The ratio of the borrower's total monthly debt obligations (excluding mortgage and the requested loan) to their reported monthly income.
- **FICO Score:** A credit score used by lenders to evaluate a borrower's creditworthiness. It is a numerical representation of the borrower's credit history and ability to repay loans.
- **Loan Purpose:** The reason for the loan application, such as debt consolidation, home improvement, or other personal needs.
- **Address State:** The U.S. state in which the borrower resides. This is a categorical variable that helps indicate geographical location, which can be relevant for loan risk evaluation.

In order to utilize both datasets simultaneously, we retain only the common features. Although removing 144 features represents a significant loss of information, we will focus on feature engineering to extract the maximum possible information from the remaining features. The target variable, *loan_status*, contains the following categories:

Table 1 Loan Status Distribution

Loan Status	**Count**
Fully Paid	1,076,751
Current	878,317
Charged Off	268,559
Late (31-120 days)	21,467
In Grace Period	8,436
Late (16-30 days)	4,349
Does not meet the credit policy. Status: Fully Paid	1,988
Does not meet the credit policy. Status: Charged Off	761
Default	40

For our analysis, we will retain only two classes: 'Fully Paid,' which will be encoded as '0', and 'Charged Off,' which will be encoded as '1'. The final data set resulted

in this study contain 2;064;314 rejected loans and 1;266;782 accepted loans, including 247;426 default loans, here we can see that Class 0 forms 0.804681% of the labels and Class 1 forms 0.195319% and that a highly imbalanced dataset. The accepted data set contains data from 2007 to 2018, and the rejected data set only contains data from 2007 to 2017. The table 2 presents the statistical tables of the accepted and rejected datasets.

Table 2 Descriptive Statistics of Accepted and Rejected Datasets

	Accepted				**Rejected**			
	loan_amnt	**issue_d**	**dti**	**fico_score**	**loan_amnt**	**issue_d**	**dti**	**fico_score**
count	1,266,782	1,266,782	1,266,782	1,266,782	2,064,314	2,064,314	2,064,314	2,064,314
mean	14,601.12	2014.95	18.13	698.13	15,315.66	2013.73	32.32	675.99
std	8,746.53	1.65	9.57	31.66	10,786.62	1.44	49.51	37.82
min	500	2007	-1.00	627.00	1,000.00	2007	-1.00	627.00
25%	8,000	2014	11.76	672.00	6,000.00	2013	11.79	648.00
50%	12,075	2015	17.52	692.00	12,000.00	2014	24.32	668.00
75%	20,000	2016	23.91	712.00	24,000.00	2014	39.44	696.00
max	40,000	2018	999.00	847.50	40,000.00	2017	998.46	990.00

The loans accepted dataset initially included a variety of attributes describing borrower characteristics, loan details, and financial history. To create a clear credit scoring model, we applied several preprocessing steps. For missing values, since the percentage was under 3%, we chose to remove them from the dataset. We standardized numerical features. At the same time, we converted categorical variables, like loan purpose and state, into numerical forms. After these preprocessing steps, the final dataset for this study included 2,064,314 rejected loans and 1,266,782 accepted loans, which contained 247,426 defaulted loans. One common variable between the accepted and rejected loan datasets is addr state, which shows the borrower's state of residence. Since regional economic differences can affect credit risk, we examined and adjusted this variable before adding it to our model. The dataset includes 51 states. Using the label encoding method would turn these 51 categories into numerical values, which could cause the model to see them as continuous variables. In contrast, the dummy encoding method would create 51 dummy variables, complicating the model's learning process and exponentially increasing computing time.

To understand how location affects loan performance, we counted the number of good payers and bad payers in each state within the accepted loans dataset. We then visualized these counts in a bar chart 1. This one indicated that some states had higher default rates, which suggests that geographic location influences borrower risk [39]. Instead of encoding addr_ state as a categorical variable with many levels, we grouped the states into three categories based on the quartiles of the default counts distribution. This change helps capture location-related risk effectively without adding too much complexity. The categorization was as follows:

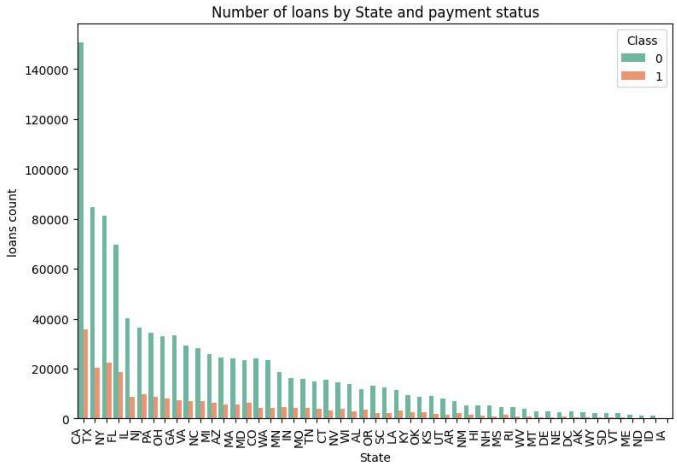


Fig. 1 Counts of good payers and bad payers on each state

- **d1 (High risk)**: CA, TX, NY, FL, IL, NJ, PA, GA, OH, VA, NC, MI, AZ, MA, CO, MD, WA
- **d2 (Medium Risk)**: MN, IN, MO, CT, TN, NV, WI, OR, SC, AL, LA, KY, KS, OK, UT, AR, NM
- **d3 (Low risk)**: NH, HI, RI, MS, WV, MT, DC, DE, NE, AK, WY, VT, SD, ME, ID, ND, IA

Each loan was then assigned three dummy variables (**d1**, **d2**, **d3**), where only one of them is equal to 1, depending on the state's default risk category. This approach effectively reduces dimensionality while preserving crucial geographical information related to credit risk, allowing the model to incorporate location-based variations in default probability.

4.2 Performance evaluation

Model performance is assessed using standard credit scoring metrics: **accuracy, precision, recall, AUC, Brier score, KS statistic, and G-Mean**. These metrics capture both classification performance and probabilistic calibration.

Accuracy, **precision**, and **recall** are defined as:

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}, \quad \text{Precision} = \frac{TP}{TP + FP}, \quad \text{Recall} = \frac{TP}{TP + FN} \tag{7}$$

These quantities are derived from the confusion matrix in Table 3.

Brier Score evaluates probabilistic calibration via mean squared error:

$$BS = \frac{1}{N} \sum_{i=1}^{N} (f(x_i) - y_i)^2 \tag{8}$$

Table 3 Confusion matrix for credit scoring

Observed	Predicted: Good	Predicted: Bad
Good (Positive)	TP	FN
Bad (Negative)	FP	TN

where $f(x_i)$ is the predicted probability and $y_i \in \{0, 1\}$ is the true label.

KS statistic measures class separation:

$$KS = \max_t \left| F^+(t) - F^-(t) \right| \tag{9}$$

where F^+ and F^- are empirical CDFs of positive and negative scores.

G-Mean balances sensitivity and specificity:

$$\text{G-Mean} = \sqrt{\frac{TP}{TP + FN} \cdot \frac{TN}{TN + FP}} \tag{10}$$

AUC quantifies ranking ability independent of threshold:

$$AUC = \frac{1}{N^+ N^-} \sum_{i \in M^+} \sum_{j \in M^-} \mathbb{I}\big(f(x_i) > f(x_j)\big) \tag{11}$$

4.3 Statistical tests for model comparison

Machine learning studies often compare several models, but meaningful conclusions require statistical validation rather than just looking at performance metrics. Choosing the right statistical test is challenging because parametric methods assume normality. For this reason, non-parametric tests are often used because they have fewer assumptions and are more robust.

In this paper, we use Friedman's test [40], a non-parametric alternative to repeated-measures ANOVA, to evaluate statistical differences among models based on accuracy, recall, precision, and AUC. It is computed as follows:

$$\chi_F^2 = \frac{12N}{k(k+1)} \sum_{j=1}^{k} R_j^2 - 3N(k+1) \tag{12}$$

Where:

- k is the number of models,
- N is the number of datasets (or observations),
- R_j is the sum of the ranks assigned to model j.

If **Friedman's test rejects the null hypothesis**, we conduct a **Nemenyi post-hoc test**[40] by computing the **critical difference (CD)** to identify which models differ significantly. The critical difference is given by:

$$CD = q_\alpha \sqrt{\frac{k(k+1)}{6N}} \tag{13}$$

where q_α is the critical value from the **Studentized range statistic**. We also perform the **Wilcoxon signed-rank test** to compare pairs of models. Given two related samples X and Y, we compute the differences $d_i = x_i - y_i$, rank the absolute differences $|d_i|$, and sum the ranks of positive differences to obtain the test statistic:

$$W = \sum_{\{i:d_i>0\}} R_i$$

where R_i is the rank of $|d_i|$. A small p-value indicates a significant difference between the two models.

4.4 Experiment Design

All models were evaluated under a repeated random subsampling protocol of 30 independent runs. In each run we drew a balanced sample of 3 500 labelled observations from the accepted-loan pool (1 750 per class) and 25 000 unlabelled observations from the rejected-loan pool. The labelled data were split 80/20 into training and test sets.

Seven performance metrics were computed on the held-out test set at each run: Area Under the ROC Curve (AUC), Accuracy, Precision, Recall, F1-score, Kolmogorov–Smirnov statistic (KS) and Brier Score. Median values and median absolute deviations (MAD) are reported across the 30 runs. Statistical comparisons were carried out with the Friedman test ($\alpha = 0.05$) followed by the post-hoc Nemenyi test (critical distance CD = 2.193), as implemented in the `autorank` library [41].

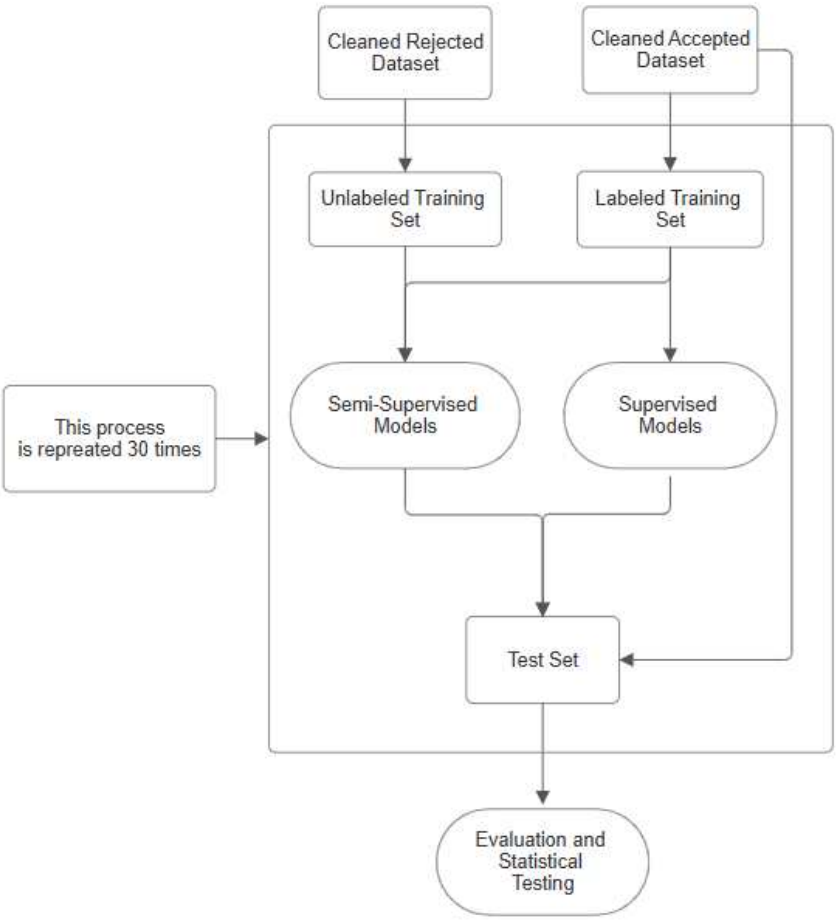


Fig. 2 First Experiment Design

Hyperparameters

The models were optimized using a 5-fold cross-validation grid search. The hyperparameter search spaces are presented in Table 4.

4.5 Numerical Results

4.5.1 Model Comparison

Table 5 summarises the average performance of all nine Models over the 30 runs. Models are sorted in descending order of AUC. We distinguish three groups: six fully supervised baselines (Logistic Regression, SVM, Random Forest, MLP, LightGBM, XGBoost), two standard semi-supervised baselines (Self-Training LR and Label Spreading), and the proposed method (LogisticCEM).

Table 4 Hyperparameters

Model	Hyperparameter	Search Space
Logistic Regression	C	$\{10^{-3}, 10^{-2}, 10^{-1}, 1, 10, 100\}$
	penalty	{l1, l2}
SVM	C	$\{10^{-2}, 10^{-1}, 1, 10, 100\}$
	kernel	{linear, rbf}
	γ	{scale, auto}
Self-Training LR	C	$\{10^{-3}, 10^{-2}, 10^{-1}, 1, 10\}$
	threshold	$\{0.6, 0.7, 0.8, 0.9\}$
Logistic CEM	C	$\{10^{-3}, 10^{-2}, 10^{-1}, 1, 10, 100\}$
	class_weight	{None, balanced}
Label Spreading	kernel	{rbf, knn}
	α	$\{0.2, 0.5, 0.8\}$
Random Forest	$n_{\text{estimators}}$	{100, 200, 500}
	max_depth	{None, 10, 20, 30}
	min_samples_split	{2, 5, 10}
	min_samples_leaf	{1, 2, 5}
MLP	hidden_layer_sizes	{(64), (128), (64,64), (128,64), (128,128)}
	α	$\{10^{-5}, 10^{-4}, 10^{-3}, 10^{-2}\}$
	activation	{relu, tanh}
LightGBM	$n_{\text{estimators}}$	{100, 200, 500}
	learning_rate	{0.01, 0.05, 0.10}
	num_leaves	{31, 63, 127}
	subsample	{0.8, 1.0}
XGBoost	$n_{\text{estimators}}$	{100, 200, 500}
	learning_rate	{0.01, 0.05, 0.10}
	max_depth	{3, 5, 7}
	subsample	{0.8, 1.0}

Table 5 Performance comparison of models.

Model	AUC	Accuracy	Precision	Recall	F1	KS	Brier
SelfTrainingLR	**0.635650**	0.596286	**0.591910**	0.620190	0.605460	**0.223048**	0.236214
LogisticCEM	0.635439	**0.598143**	0.591530	**0.634952**	**0.612134**	0.220667	**0.236176**
LogReg	0.635049	0.595905	0.591454	0.620286	0.605222	0.221238	0.236357
SVM	0.611529	0.578714	0.577806	0.583905	0.580236	0.188190	0.240810
MLP	0.600813	0.568429	0.568280	0.575810	0.569955	0.173524	0.254128
LightGBM	0.596696	0.567571	0.566212	0.580095	0.572728	0.163905	0.254921
RF	0.591407	0.565571	0.566164	0.561619	0.563639	0.161619	0.253432
LabelSpreading	0.581730	0.557714	0.571802	0.465048	0.510945	0.150095	0.247668
XGBoost	0.579910	0.554667	0.553764	0.564667	0.558964	0.140286	0.277517

The top three positions are occupied by the two semi-supervised methods and standard Logistic Regression, which are essentially tied in AUC (0.6350–0.6357). All three comfortably outpace the remaining supervised models, with AUC gaps exceeding 0.02 with respect to SVM and up to 0.056 compared with XGBoost. This suggests that the features contain only a small but real signal for distinguishing outcomes. Models that can use the structure of the data, either by spreading labels or by learning from the unlabeled data, are able to make better use of this information.

LogisticCEM ranks second in AUC (0.6354) but achieves the highest Accuracy (0.598), F1-score (0.612) and Recall (0.635) among all models. The recall advantage is particularly meaningful in a credit-risk context: the model is better at catching actual charge-offs without sacrificing precision to the same degree as naive methods. The Brier Score of LogisticCEM (0.2362) matches the best value in the table (SelfTraining LR) and is substantially lower than all fully supervised classifiers except plain Logistic Regression, confirming better probability calibration.

Against SelfTraining LR, the AUC difference is negligible (< 0.001), but LogisticCEM leads on F1 (+0.007) and Recall (+0.015). Against Label Spreading, the gap is wider across the board: +0.054 in AUC, +0.101 in F1, and +0.170 in Recall. Label Spreading also exhibits a severe precision–recall imbalance (precision 0.572 vs recall 0.465), indicating that its label propagation strategy fails in this high-dimensional, tabular setting. LogisticCEM sidesteps this by treating unlabelled examples as training instances rather than labeling them and retrain the model on the full pool.

Random Forest, LightGBM, and XGBoost perform poorly relative to the simpler linear models, ranking 7^{th}, 6^{th}, and 9^{th} in AUC, respectively. Their Brier Scores are also the worst in the table (XGBoost: 0.2775, LightGBM: 0.2549). This is somewhat unexpected and likely reflects the linear nature of the decision boundary.

The KS statistic shows that SelfTrainingLR and LogisticCEM achieve the strongest discriminatory power (KS = 0.22), while tree-based and boosting models perform lower. Overall, discrimination remains moderate, reflecting the difficulty of reject inference under selection bias. The Brier score confirms that linear and semi-supervised logistic models provide better calibrated probabilities than ensemble methods. In contrast, XGBoost shows the weakest calibration and ranking performance, indicating sensitivity to biased and noisy training data.

4.5.2 Statistical Significance

The Friedman test rejects the null hypothesis of equal central tendency across all models (all $p < 0.001$), indicating statistically significant differences in performance. Post-hoc Nemenyi analysis (critical difference $CD = 2.193$) reveals three homogeneous groups for AUC. The top-performing group consists of LogisticCEM, SelfTrainingLR, and Logistic Regression, with no significant differences among them. SVM forms an intermediate group, while XGBoost, Label Spreading, Random Forest, LightGBM, and MLP constitute the lowest-performing cluster, within which pairwise differences are not statistically significant.

The same three-way partition holds for Accuracy, F1, KS, and Brier Score. For Recall, LogisticCEM achieves the lowest mean rank (1.5, i.e. the best), while Label Spreading is the worst ranked method (8.67) with a large and statistically significant gap relative to all other models. Critical difference diagrams are shown in Figure 9.

4.5.3 ROC Analysis: Effect of Unlabelled Pool Size

To investigate how the size of the unlabelled rejected-loan pool N_R affects the quality of the learned classifier, we conducted two additional experiments in which N_A (the number of labelled accepted-loan examples) was held fixed while N_R was varied across a wide range.

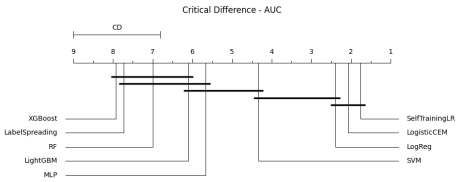


Fig. 3 *

(a) AUC

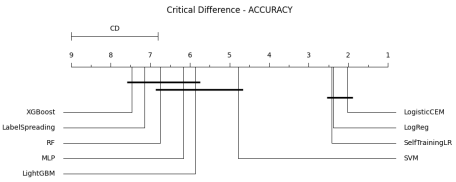


Fig. 4 *

(b) Accuracy

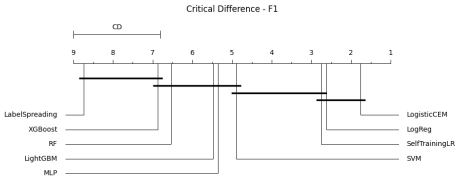


Fig. 5 *

(c) F1-score

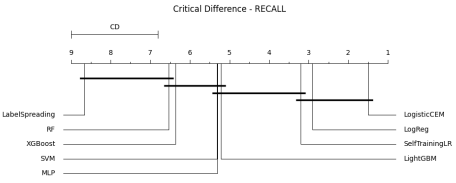


Fig. 6 *

(d) Recall

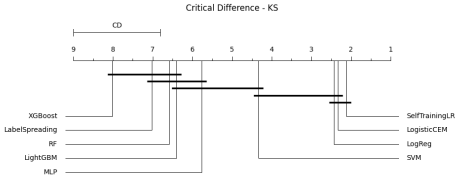


Fig. 7 *

(e) KS

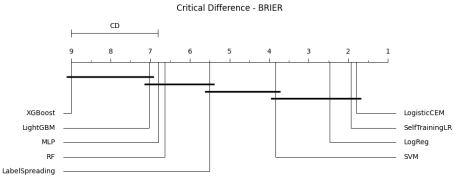


Fig. 8 *

(f) Brier Score

Fig. 9 Critical difference diagrams (Nemenyi post-hoc test, $\alpha = 0.05$, CD = 2.193) for six evaluation metrics over 30 runs.

Experiment 1: small labelled set

We fixed the labelled pool to 3 500 balanced observations (2 800 training samples after the split) and varied $N_R \in \{1\,000,\ 2\,000,\ 5\,000,\ 10\,000,\ 15\,000,\ 20\,000,\ 25\,000\}$.

From Figure 10, one can draw the following observations. (i) Even the smallest unlabelled pool ($N_R = 1\,000$) yields a classifier that is clearly better than the random baseline. (ii) AUC increases steadily as N_R grows from 1 000 to $\approx 5\,000$, after which gains diminish, suggesting a saturation effect: once the unlabelled mass represents a ratio of roughly $N_R/N_A \approx 1.8$ additional unlabelled data contribute marginal discriminative information relative to the labelled set. (iii) The shape of the ROC curves is consistent across all values of N_R: the true-positive rate rises steeply at low false-positive rates, which is the practically relevant operating region for a credit scoring system aiming to reject a small fraction of applicants.

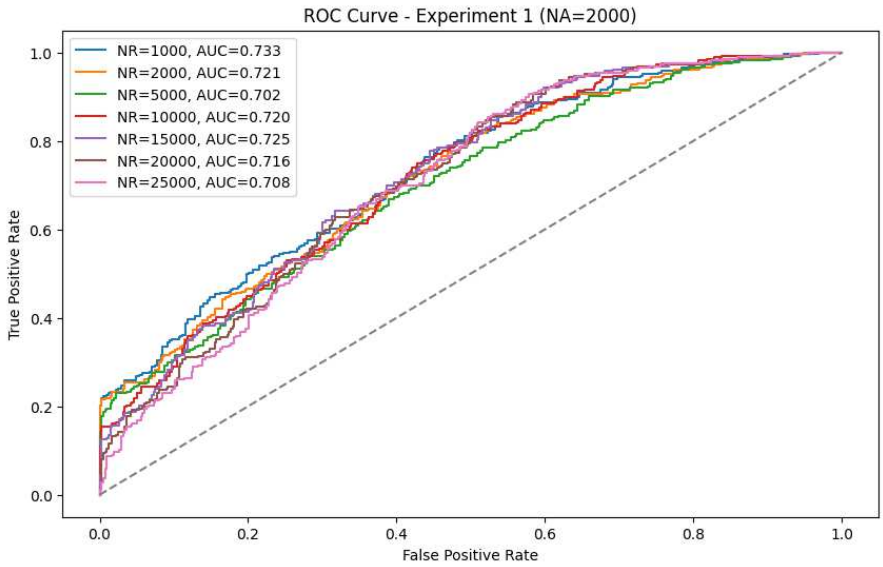


Fig. 10 ROC curves of LogisticCEM with a training set fixed at $N_A = 2\,800$.

Experiment 2: large labelled set

We used the full balanced accepted-loan ($N_A = 2\,038\,712$ after SMOTE; $1\,427\,098$ training samples) and varied $N_R \in \{1\,000, 10\,000, 100\,000, 1\,000\,000, 2\,064\,314\}$.

Figure 11 reveals a qualitatively different picture compared to Experiment 1. (i) At large N_A, a small unlabelled pool ($N_R = 1\,000$) already produces a competitive classifier, confirming that labelled data quality dominates when the labelled set is large. (ii) Increasing N_R from $1\,000$ to $100\,000$ produces a measurable improvement in AUC; further growth up to $N_R = 2\,064\,314$ yields diminishing returns, and the curves for $N_R \geq 1\,000\,000$ are nearly indistinguishable. (iii) The absolute AUC values in Experiment 2 are noticeably higher than in Experiment 1, illustrating the expected benefit of a large labelled pool. Taken both Experiments suggest that LogisticCEM is robust to the ratio N_R/N_A: it extracts useful signal from the rejected pool regardless of whether the labelled set is small or large, and does not degrade when $N_R \gg N_A$.

4.5.4 Summary

Across all seven evaluation metrics and across both the small-scale benchmark and the large-scale ROC experiments, LogisticCEM consistently ranks among the top-performing models. Its advantages are most visible in Recall and F1-score, the metrics most relevant to a practitioner who needs to minimise missed defaults without inflating false alarms. Statistical testing confirms that these advantages are significant relative to all supervised methods and to Label Spreading, while the difference with Self-Training LR is not significant — the two semi-supervised methods occupy the same statistical group. The ROC analysis further demonstrates that LogisticCEM scales

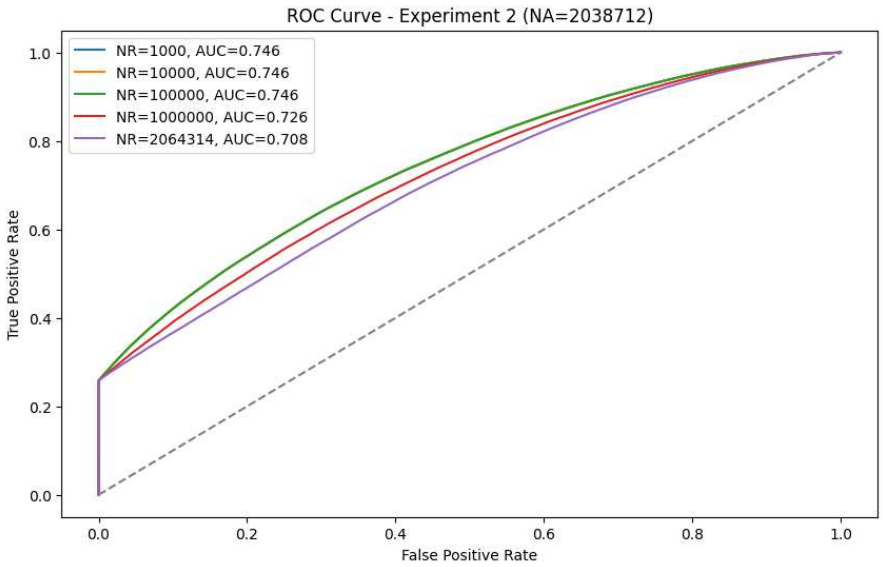


Fig. 11 ROC curves of LogisticCEM with a labelled training set fixed at $N_A \approx 1.43$ million.

gracefully with the size of the unlabelled pool, making it a practical choice when large volumes of rejected applications are available but carry no outcome label.

5 Conclusion and Discussion

The core idea behind LogisticCEM is to treat the rejected-loan population as a source of distributional information rather than a collection of unlabelled positives or negatives. Under the Classification Expectation Maximisation (CEM) framework, each rejected applicant is assigned a soft posterior probability of belonging to each outcome class at every E-step; the M-step then re-estimates the logistic parameters by weighting both the labelled and unlabelled contributions by these posteriors. This is fundamentally different from Self-Training, which commits to hard pseudo-labels above a confidence threshold and discards the remaining unlabelled examples.

The experimental results reflect this design choice in two ways. First, LogisticCEM achieves a higher Recall than any other model (0.635 vs 0.620 for Self-Training LR), without a corresponding drop in Precision. Hard pseudo-labelling in Self-Training tends to be overconfident on near-boundary examples, propagating label noise that suppresses recall on the minority class. By contrast, the E-step of LogisticCEM naturally hedges on ambiguous cases, allowing the M-step to extract partial information from them rather than silencing their contribution entirely. Second, the Brier Score of LogisticCEM (0.2362) is the joint lowest in the table, indicating that the predicted probabilities are well calibrated. Good calibration matters operationally: a lender who uses predicted default probabilities to set interest rates or credit limits needs scores that are reliable in absolute terms, not merely in rank order.

The KS statistic further corroborates this picture. At 0.221, LogisticCEM achieves near-identical separation between the cumulative score distributions of defaulters and non-defaulters as the two best linear baselines, while all tree-based models are substantially below (0.14–0.18). In practice, lenders frequently use the KS statistic as their primary scorecard validation criterion; by this measure, LogisticCEM passes the same threshold as the best competitors while also providing better calibrated probabilities.

The two ROC experiments in Section 4.5 put light on how the size of the rejected pool N_R interacts with the size of the labelled accepted pool N_A.

When N_A is small (Experiment 1, $N_A = 2\,800$), the model benefits noticeably from increasing N_R up to roughly 5 000, after which the ROC curves plateau. This saturation is consistent with what one would expect from an EM-type algorithm: additional unlabelled mass helps refine the posterior estimates of the latent class assignments, but only up to the point where the labelled set constrains the parameter space tightly enough that further unlabelled examples can no longer shift the decision boundary. Beyond that ratio, the bottleneck shifts back to the labelled data, and more rejected examples add noise as much as signal. When N_A is large (Experiment 2, $N_A \approx 1.43$ million), the picture changes qualitatively. Here, even $N_R = 1\,000$ already yields a strong classifierbecause the labelled set constrains the parameter estimates strongly. The unlabelled pool still provides a meaningful lift but saturation occurs earlier in the N_R/N_A ratio. Crucially, the model never degrades as N_R grows to over two million, which matters in a production setting where the historical rejected pool accumulates continuously and re-processing is not always feasible. A practical rule suggested by these experiments is that the effective ratio is approximately $N_R/N_A \approx 1$ to 3 for small labelled sets, while larger labelled sets can absorb higher ratios without degradation. Future work should formalise this relationship theoretically.

Reject inference has been studied since at least the 1990. Classical approaches include augmentation (assigning fixed pseudo-labels to rejects), parcelling (distributing rejects proportionally across classes), and extrapolation (fitting a model on accepts and extrapolating to rejects) [42].

LogisticCEM connects naturally to this literature by framing reject inference as a missing-data problem and solving it with the Expectation Maximisation algorithm, a principled choice that was noted as early as Feelders [25] but has seen limited follow-up with modern semi-supervised machinery. The contribution of this paper is to demonstrate that the logistic CEM can be applied in real-life reject inference problems and achieves competitive or superior performance compared with both classical supervised methods and modern semi-supervised algorithms, and does so with a model that remains fully interpretable (logistic regression coefficients) and fast to train (early stopping typically fires within a handful of iterations, as observed in the experiments).

Several limitations of this study should be noted. The model is based on a limited set of variables, which may restrict its performance compared to richer information sources. All experiments are also based on the Lending Club dataset, so the results may not directly generalise to other lenders, countries, or credit products. In addition, the model is kept intentionally linear for interpretability, although more flexible methods could achieve better performance in more complex settings. Finally, the current

approach does not explicitly correct for selection bias, and future work could improve this by adding bias correction, online learning, survival analysis, cross-lender data, and fairness constraints.

Funding

The author declares that no funding was received for this study.

Acknowledgements

The author would like to express sincere gratitude to their supervisors, Badreddine Benyacoub and Mohamed El Haj Tirari, for their guidance, support, and valuable feedback throughout this work.

Data and Code Availability

The data used in this study are publicly available from the Lending Club dataset hosted on Kaggle: https://www.kaggle.com/datasets/wordsforthewise/lending-club. The code used to support the findings of this study is available from the corresponding author upon reasonable request.

References

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

[2] Thomas, L.C.: A survey of credit and behavioural scoring: Forecasting financial risk of lending to consumers. International Journal of Forecasting **16**(2), 149–172 (2000) https://doi.org/10.1016/s0169-2070(00)00034-0

[3] Bijak, K., Thomas, L.C.: Does segmentation always improve model performance in credit scoring? Expert Systems with Applications **39**(3), 2433–2442 (2012) https://doi.org/10.1016/j.eswa.2011.08.093

[4] Horrigan, J.O.: The determination of long-term credit standing with financial ratios. Journal of Accounting Research **4**, 44–62 (1966)

[5] West, R.R.: An alternative approach to predicting corporate bond ratings. Journal of Accounting Research **8**, 118–125 (1970)

[6] Altman, E.I.: Financial ratios, discriminant analysis and the prediction of corporate bankruptcy. Journal of Finance **23**, 589–609 (1968)

[7] Khashman, A.: Neural networks for credit risk evaluation: Investigation of different neural models and learning schemes. Expert Systems with Applications **37**, 6233–6239 (2010)

[8] Hongxia, W., Xueqin, L., Yanhui, L.: Enterprise credit rating model based on fuzzy clustering and decision tree. In: Proceedings of the International Symposium on Information Science and Engineering (ISISE), pp. 105–108. IEEE, ??? (2010)

[9] Hwang, R.C., Cheng, K., Lee, C.F.: On multiple-class prediction of issuer credit ratings. Applied Stochastic Models in Business and Industry **25**, 535–550 (2009)

[10] Kaplan, R.S., Urwitz, G.: Statistical models of bond ratings: A methodological inquiry. Journal of Business **52**, 231–261 (1979)

[11] Bellotti, T., Crook, J.: Support vector machines for credit scoring and discovery of significant features. Expert Systems with Applications **36**, 3302–3308 (2009)

[12] Chen, C.-C., Li, S.-T.: Credit rating with a monotonicity-constrained support vector machine model. Expert Systems with Applications **41**, 7235–7247 (2014)

[13] Bücker, M., Kampen, M., Krämer, W.: Reject inference in consumer credit scoring with nonignorable missing data. Journal of Banking and Finance **37**(3), 1040–1045 (2013)

[14] Banasik, J., Crook, J.: Credit scoring, augmentation and lean models. Journal of the Operational Research Society **56**(9), 1072–1081 (2005)

[15] Li, Z., Tian, Y., Li, K., Zhou, F., Yang, W.: Reject inference in credit scoring using semi-supervised support vector machines. Expert Systems with Applications **70**, 201–210 (2017) https://doi.org/10.1016/j.eswa.2017.01.011

[16] Hady, M.F.A., Schwenker, F.: Semi-supervised learning. In: Bianchini, M., Maggini, M., Jain, L. (eds.) Handbook on Neural Information Processing. Intelligent Systems Reference Library, vol. 49, pp. 215–239. Springer, Berlin, Heidelberg (2013). https://doi.org/10.1007/978-3-642-36657-4_7

[17] Zhu, X., Goldberg, A.B.: Introduction to Semi-Supervised Learning. Synthesis Lectures on Artificial Intelligence and Machine Learning, vol. 1. Springer, Cham (2009). https://doi.org/10.1007/978-3-031-01548-9

[18] Hand, D.J., Everitt, B.S.: Finite Mixture Distributions. Chapman and Hall, London (1981)

[19] Newcomb, S.: A generalized theory of the combination of observations so as to obtain the best result. American Journal of Mathematics **8**(4), 343–366 (1886)

[20] Celeux, G., Govaert, G.: A classification em algorithm for clustering and two stochastic versions. Computational Statistics & Data Analysis **14**(3), 315–332 (1992) https://doi.org/10.1016/0167-9473(92)90042-E

[21] Vittaut, J.N., Amini, M.R., Gallinari, P.: Learning classification with both labeled and unlabeled data. In: Elomaa, T., Mannila, H., Toivonen, H. (eds.) Machine

Learning: ECML 2002. Lecture Notes in Computer Science, vol. 2430. Springer, Berlin, Heidelberg (2002). https://doi.org/10.1007/3-540-36755-1_39

[22] Amini, M.-R., Gallinari, P.: Semi-supervised logistic regression. In: Proceedings of the 15th European Conference on Artificial Intelligence (ECAI), pp. 390–394 (2002)

[23] Anderson, B., Hardin, J.M.: Modified logistic regression using the em algorithm for reject inference. International Journal of Data Analysis Techniques and Strategies **5**(4), 359–373 (2014) https://doi.org/10.1504/IJDATS.2013.058582

[24] Kim, H.: Semi-supervised learning to predict default risk for P2P lending. Journal of Digital Convergence **20**(4), 185–192 (2022) https://doi.org/10.14400/JDC.2022.20.4.185

[25] Feelders, A.J.: Credit scoring and reject inference with mixture models. Intelligent Systems in Accounting, Finance and Management **9**(1), 1–8 (2000) https://doi.org/10.1002/(SICI)1099-1174(200003)9:1⟨1::AID-ISAF177⟩3.0.CO;2-#

[26] Shen, F., Yang, Z., Zhao, X., Lan, D.: Reject inference in credit scoring using a three-way decision and safe semi-supervised support vector machine. Information Sciences **607**, 1392–1405 (2022) https://doi.org/10.1016/j.ins.2022.05.067

[27] Kennedy, K., Namee, B.M., Delany, S.J.: Using semi-supervised classifiers for credit scoring. Journal of the Operational Research Society **64**(4), 513–529 (2013)

[28] Liu, Y., Li, X., Zhang, Z.: A new approach in reject inference using ensemble learning based on a global semi-supervised framework. Future Generation Computer Systems **109**, 382–391 (2020)

[29] Mancisidor, R.A., Kampffmeyer, M., Aas, K., Jenssen, R.: Deep generative models for reject inference in credit scoring. Knowledge-Based Systems **196**, 105758 (2020)

[30] Anderson, B.: Using bayesian networks to perform reject inference. Expert Systems with Applications **137**, 349–356 (2019)

[31] Hájek, P., Olej, V.: Credit rating modelling by kernel-based approaches with supervised and semi-supervised learning. Neural Computing and Applications **20**(6), 761–773 (2011)

[32] Kang, Y., Jia, N., Cui, R., Deng, J.: A graph-based semi-supervised reject inference framework considering imbalanced data distribution for consumer credit scoring. Applied Soft Computing **105**, 107259 (2021) https://doi.org/10.1016/j.asoc.2021.107259

[33] El Annas, M., Benyacoub, B., Ouzineb, M.: Semi-supervised adapted HMMs for

P2P credit scoring systems with reject inference. Computational Statistics **38**(1), 149–169 (2023)

[34] Xia, Y., Yang, X., Zhang, Y.: A rejection inference technique based on contrastive pessimistic likelihood estimation for P2P lending. Electronic Commerce Research and Applications **30**, 111–124 (2018)

[35] Xiao, J., Cao, H., Jiang, X., Gu, X., Xie, L.: Gmdh-based semi-supervised feature selection for customer classification. Knowledge-Based Systems **132**, 236–248 (2017)

[36] Kang, Y., Chen, L., Jia, N., Wang, X., Liu, Z.: A CWGAN-GP-based multi-task learning model for consumer credit scoring. Expert Systems with Applications **206**, 117650 (2022)

[37] Dempster, A.P., Laird, N.M., Rubin, D.B.: Maximum likelihood from incomplete data via the EM algorithm. Journal of the Royal Statistical Society: Series B (Methodological) **39**(1), 1–38 (1977)

[38] Verthez, J.: Scoring models in peer-to-peer lending. Master's thesis, Ghent University (2017)

[39] Elliehausen, G., Lundquist, E.C., Staten, M.E.: The impact of credit counseling on subsequent borrower behavior. Journal of Consumer Affairs **41**(1), 1–28 (2007) https://doi.org/10.1111/j.1745-6606.2006.00069.x

[40] Demšar, J.: Statistical comparisons of classifiers over multiple data sets. Journal of Machine Learning Research **7**, 1–30 (2006)

[41] Herbold, S.: Autorank: A python package for automated ranking of classifiers. Journal of Open Source Software **5**(48), 2173 (2020) https://doi.org/10.21105/joss.02173

[42] Ehrhardt, A., Biernacki, C., Vandewalle, V., Heinrich, P., Beben, S.: Reject inference methods in credit scoring. Journal of Applied Statistics **48**(13–15), 2734–2754 (2021)