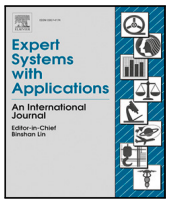
Expert Systems With Applications 272 (2025) 126708

Contents lists available at ScienceDirect

Expert Systems With Applications

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



Cost-sensitive reinforcement learning for credit risk

Jorge C-Rella [a,c,*], David Martinez Rego [b], Juan M. Vilar [a]

[a] Department of Mathematics, Research Group MODES, CITIC, University of A Coruña, A Coruña, Spain
[b] DataSpartan, London, United Kingdom
[c] Department of Risks, ABANCA Financial Services, A Coruña, Spain

ARTICLE INFO

Keywords:
Cost-sensitive classification
Reinforcement learning
Bandit algorithms
Online learning
Credit risk
Decision making

ABSTRACT

Credit risk problems are dynamic because customer behavior is not stable, and they are cost-sensitive because the impact of a decision depends on the amount of the loan. Online learning algorithms, which evolve as more information becomes available, are an appropriate tool to study these dynamic problems. However, only information on approved transactions is available, which can lead to unfair biases and opportunity costs. Within reinforcement learning, bandit algorithms address this by balancing exploitation (acting according to the current model) and exploration (considering an action with limited information to improve predictions). The only remaining gap is to address the problem taking into account the different classification costs. This paper introduces cost-sensitive reinforcement learning algorithms to solve the credit risk problem from a dynamic perspective maximizing long-term benefits, proposing a cost-sensitive passive-aggressive algorithm and a cost-sensitive logistic bandit. Experiments on benchmark datasets and extensive simulation studies demonstrate the effectiveness and efficiency of the proposed algorithms.

1. Introduction

Lenders and investors face the challenge of selecting potential borrowers to whom they should extend credit in order to control risk while maximizing profit (Bahnsen et al., 2014; Lessmann et al., 2015; Lu et al., 2013). Traditional credit scoring systems assign scores based on socio-economic characteristics, with higher scores indicating a higher risk of default. Comprehensive experimental benchmarking studies have recently been conducted by Lessmann et al. (2015) and Vanderschueren et al. (2022), showing that the industry standard, logistic regression, is significantly outperformed by more advanced methods, which also have limitations. Their reliance on historical data results in a lack of information about potential borrowers who were previously deemed ineligible for credit. This can lead to customers being unfairly misjudged, opportunity costs and dangerous errors when the underlying relationships between variables change (concept drift). In addition, credit risk leads to an exploration vs. exploitation dilemma (Sutton & Barto, 2018): should a new customer be employed to gain more information and improve predictions at the potential cost of incurring losses, or should the current model's assessment dictate the decision? As a result, financial institutions face two challenges that are often overlooked: adapting to dynamic environments while optimizing both immediate risk and long-term benefit.

Online learning (OL) addresses the first challenge of dealing with dynamic environments by continuously updating models as new information becomes available. The fundamental difference between OL and classical offline methods lies in their approach to data handling. Static learning algorithms, such as logistic regression or decision trees, use a fixed set of data for model training, achieving high accuracy in offline settings. However, they cannot adapt to evolving patterns. OL processes data incrementally, learning continuously as new observations become available. As a result, models remain relevant and effective over time, even as data distributions and patterns evolve (Hoi et al., 2018; Lattimore & Szepesvári, 2020; Sutton & Barto, 2018). This continuous learning loop also leads to superior resource efficiency with minimal retraining requirements as noted in Li et al. (2010). The most prominent algorithms in the OL literature are the Online Gradient Descent algorithm (Zhao et al., 2015), the Passive-Aggressive algorithm (Crammer et al., 2006), the perceptron (Rosenblatt, 1958), and many other proposals that follow the principle of large-margin learning (Crammer et al., 2006; Dredze et al., 2008; Kivinen et al., 2004; Shalev-Shwartz, 2012; Wang et al., 2012).

OL deals with dynamic settings but does not consider exploration. Within the reinforcement learning paradigm, bandit algorithms are designed to balance exploration and exploitation. They assume that multiple actions with different rewards are available. Then, an agent

* Corresponding author at: Department of Mathematics, Research Group MODES, CITIC, University of A Coruña, A Coruña, Spain.
E-mail addresses: jorge.crella@udc.es (J. C-Rella), david@dataspartan.com (D. Martinez Rego), juan.vilar@udc.es (J.M. Vilar).

https://doi.org/10.1016/j.eswa.2025.126708
Received 9 November 2024; Received in revised form 18 January 2025; Accepted 26 January 2025
Available online 4 February 2025

explores the different actions to decide on the optimal strategy that maximizes its reward in the long run. Due to the increasing number of problems in this context, several bandit algorithms have been developed in recent years, such as Epsilon-Greedy algorithms (Lattimore & Szepesvári, 2020; Sutton & Barto, 2018), Upper Confidence Bound (UCB) algorithms (Auer et al., 2002; Kaufmann et al., 2014) and sampling approaches (Kaufmann et al., 2012; Korda et al., 2013; Lattimore & Szepesvári, 2020; Russo et al., 2020; Thompson, 1933). The drawback of these algorithms is that they do not take into account any information other than the expected reward of each action. In practice, however, auxiliary information is usually available. For example, in credit risk, do you lend money to people regardless of their socioeconomic characteristics? Contextual bandits include the additional context of each action to make informed decisions. This approach has gained recognition in applications such as online advertising (Tang et al., 2013), recommender systems (Lai & Robbins, 1985; Li et al., 2010), personalized medicine (Rabbi et al., 2015), and credit risk (Shen et al., 2015; Visantavarakul, 2022), where tailoring decisions to specific situations leads to improved performance.

Most classification models, both online and offline, assume an uniform impact of misclassification errors. Nevertheless, this assumption can lead to suboptimal results in cost-sensitive (CS) scenarios where error costs vary across error types and instances. In credit risk, a borrower who defaults on a loan causes a greater loss than rejecting a good loan application, depending on the amount requested. By explicitly training models to adapt to different misclassification costs, they are aligned with the goal of maximizing benefits, leading to better results as shown in Bahnsen et al. (2014, 2013), C-Rella et al. (2025), Elkan (2001), Höppner et al. (2021), Vanderschueren et al. (2022) and Wang et al. (2012). Nevertheless, there is a lack of approaches for optimizing dynamic cost-sensitive problems as claimed in Achab et al. (2018) and Zhang et al. (2018).

In this paper, two novel algorithms are proposed to tackle the CS problem from a reinforcement learning perspective. An OL algorithm is proposed, extending the passive-aggressive (PA) algorithm introduced in Crammer et al. (2006). With this proposal, an updated model is obtained with all the information available to predict the optimal CS decision. In order to consider exploration, a CS version of the logistic bandit is also proposed, training a logistic model to maximize a reward function while considering exploration beyond the model's predictions. In this way, the risk of falling into bias or losing opportunities due to past decisions is minimized.

By combining a CS approach with the adaptability of OL and bandit algorithms, better results are expected with respect to current classification algorithms. This is empirically demonstrated with an extensive set of experiments on some real-world benchmark datasets and several simulations. Although exploration generally improves long-term performance, some companies may be more conservative or operate under stricter regulations that limit their ability to explore beyond the predictions of the current model. Therefore, both approaches are of practical interest. Throughout the paper the credit risk problem is considered to make the problem easier to understand, but note that all the proposals can be extended to any other CS problem by adapting the prediction costs specification.

The paper is organized as follows. Section 2 presents the CS classification problem to be solved. Section 3 introduces the online learning framework and the cost-sensitive passive-aggressive algorithm proposed in this paper. Section 4 presents state-of-the-art bandit algorithms and introduces the novel proposed CS logistic bandit. Section 5 summarizes the results of the simulation study and five real data sets. Conclusions and future extensions are included in Section 6.

2. Cost-sensitive classification

In cost-sensitive (CS) classification, the goal is to predict a binary response variable $Y \in \{0, 1\}$ (0 indicating a good operation and 1 default for the credit risk problem) from a set of features (e.g. income, credit history) summarized in a covariate vector $\mathbf{X} = (X_1, \dots, X_d)$ taking into account prediction error costs. These costs depend on an exogenous variable W (the loan amount in credit risk), which is assumed independent of $\mathbf{X}$ and Y according to Zadrozny and Elkan (2001). The objective is to maximize benefits.

Table 1
Instance-dependent cost matrix summarizing the benefit/cost of a true negative (TN), false positive (FP), false negative (FN) and a true positive (TP).

	$\hat{y}_i = 0$	$\hat{y}_i = 1$
$y_i = 0$	$C_i^{TN} = aw_i$	$C_i^{FP} = -cw_i - b$
$y_i = 1$	$C_i^{FN} = -aw_i$	$C_i^{TP} = -b$

Classical classification metrics are not valid in CS settings, as there is no direct relationship between accuracy-oriented metrics and CS metrics, as noted in C-Rella et al. (2024, 2025). Extending the proposals in Bahnsen et al. (2014, 2013), Elkan (2001) and Höppner et al. (2021), an instance-dependent *reward function* is defined from the cost matrix in Table 1 below. Considering the problem of credit risk, it is assumed that approving a good loan application implies a gain of aW, where a is the interest rate after deducting the operational costs. The impact of false negatives (FN) is given by cW, where c is the loss given default. Following the reasonableness condition introduced in Elkan (2001), it is assumed that $c > a$, i.e. that the impact of FN is greater than that of false positives (FP). Finally, there is a fixed cost b of labeling an observation as a case regardless of its true class, due to administrative costs. Negative values represent losses, positive values represent profits and it is assumed that $a, b, c > 0$.

Despite the cost matrix in Table 1 is very intuitive, we need a metric to evaluate model performance. By aggregating all the costs in Table 1 into a function, the reward function is defined as:

$$r(\hat{y}_i, w_i, y_i) = (1 - y_i)(1 - \hat{y}_i)C_i^{TN} + (1 - y_i)\hat{y}_i C_i^{FP} + y_i(1 - \hat{y}_i)C_i^{FN} + y_i\hat{y}_i C_i^{TP}. \quad (1)$$

The reward function in (1) is an individual metric. However, the goal is to maximize benefits over the entire population, for which model performance is evaluated considering the *mean reward*:

$$\mathcal{R} = \frac{1}{n}\sum_{i=1}^{n} r(\hat{y}_i, w_i, y_i), \quad (2)$$

which serves as an estimate of the expected benefit per operation and facilitates comparison between different models as it is a standardized metric. This is the function that is intended to be optimized throughout this work.

3. Online learning

Online learning (OL) algorithms incrementally learn a predictive model by sequentially observing data, making a prediction, receiving feedback, and adapting the model with the newly obtained information. This paper focuses on the Passive-Aggressive (PA) algorithm, a maximum margin classifier as introduced in Crammer et al. (2006), Shalev-Shwartz (2012) and Wang et al. (2012). This is motivated by the fact that it is generally a more effective approach than the Perceptron algorithm (see Zhao et al., 2015). Regarding previous CS OL approaches, the main proposals in the literature, introduced below in this section, are designed to deal with class unbalance considering constant entries in the cost matrix in Table 1. This is an oversimplification that may lead to suboptimal results, for which an instance-dependent cost-sensitive (IDCS) version of the PA algorithm is introduced to overcome the shortcomings of previous algorithms.

Throughout this section, for convenience, the binary response variable is represented as $Y \in \{-1, 1\}$, where 1 represents cases (default) and -1 non-cases (good operations). We restrict our discussion to classification functions of the form $\hat{Y} = sign(\theta'\mathbf{X})$, where $sign(z) = -1$

if $z < 0$ and $sign(z) = 1$ if $z \geq 0$, and $\theta \in \mathbb{R}^d$ is the parameter vector to be fitted.

At round t, the learner is given an observation $\mathbf{x}_t$ of the covariate vector $\mathbf{X}$ and makes a prediction $\hat{y}_t = sign(\theta' \mathbf{x}_t)$. The vector $\theta \in \mathbb{R}^d$ is estimated considering all the information available up to step t, denoted as $H_t = \{(\mathbf{x}_i, y_i, w_i)\}_{i=1}^{t-1}$. After the prediction $\hat{y}_t$ is made, the correct answer $y_t \in \{-1, 1\}$ is revealed and the learner suffers a loss $\ell(\theta' \mathbf{x}_t, y_t)$, which measures the discrepancy between his prediction and the correct label (Crammer et al., 2006; Shalev-Shwartz, 2012). The algorithm uses the newly obtained instance-label pair $(\mathbf{x}_t, y_t)$ to update the parameter vector θ for the next rounds, and the process is repeated. The goal of the learner is to minimize the cumulative loss along its path up to the horizon T, $\sum_{t=1}^{T} \ell(\hat{y}_t, w_t, y_t)$.

3.1. Passive-aggressive algorithm

The parameter θ in the linear classifier $sign(\theta' \mathbf{X})$ can be interpreted as the normal vector of an hyperplane dividing the data space in two half-spaces, where each side of the hyperplane corresponds to a class $\hat{Y} \in \{-1, 1\}$. The PA algorithm iteratively adjusts the parameter vector θ by maximizing the distance between the prediction for the last instance received, $\theta' \mathbf{x}_t$, and the hyperplane $\theta' \mathbf{X} = 0$. The goal is to get the maximum distance while maintaining proximity to the previous classifier, thereby ensuring a balance between adapting to new data and maintaining stability.

The value $|\theta' \mathbf{x}_t|$ represents the degree of confidence in the prediction (Crammer et al., 2006). The loss $\ell(\theta_t' \mathbf{x}_t, y_t)$ is proportional to the degree to which the prediction was wrong. If a prediction is correct, $y_t \hat{y}_t$ is positive. Consequently, the goal is to find $\theta \in \mathbb{R}^d$ such that $y_t \theta' \mathbf{x}_t > 0$. However, the focus remains on obtaining a classifier capable of predicting with high confidence. By imposing a margin on the classification, the following *hinge loss* function is obtained:

$$\ell_H(\theta; (\mathbf{x}_t, y_t)) = \begin{cases} 0, & y_t \theta' \mathbf{x}_t \geq 1 \\ 1 - y_t \theta' \mathbf{x}_t, & \text{otherwise.} \end{cases} \tag{3}$$

Considering the hinge loss function in Eq. (3), the parameter of the PA algorithm is actualized after each step t as the solution to the constrained optimization problem:

$$\theta_{t+1} = \arg\min_{\theta \in \mathbb{R}^d} \frac{1}{2} \|\theta - \theta_t\|^2 \text{ s.t. } \ell_H(\theta; (\mathbf{x}_t, y_t)) = 0, \tag{4}$$

where $\|\cdot\|$ represents the Euclidean L^2 norm. This notation is considered throughout the work for ease of reading.

If the prediction with the current parameter θ_t is correct, the hinge loss in (3) is zero and the algorithm remains passive, i.e. it keeps θ_t unchanged. If the prediction is wrong, $\ell_H(\theta_t; (\mathbf{x}_t, y_t)) > 0$ and the algorithm corrects θ_t in order to classify with margin, but changing the previous parameter θ_t as little as possible. This balances learning from new data while retaining knowledge from previous observations. The solution to the optimization problem in Eq. (4) is obtained in Crammer et al. (2006) as:

$$\theta_{t+1} = \theta_t + \tau_t y_t \mathbf{x}_t, \text{ where } \tau_t = \frac{\ell_H(\theta_t; (\mathbf{x}_t, y_t))}{\|\mathbf{x}_t\|^2}. \tag{5}$$

When an observation is misclassified, the update strategy in Eq. (5) updates the current estimate as necessary to assign the correct class to the last observation. Although reasonable, this can be too aggressive and is not robust to outliers. Therefore, following Vapnik (1998), it is considered a soft-margin approach introducing a slack variable ξ into the optimization problem in (4):

$$\theta_{t+1} = \arg\min_{\theta \in \mathbb{R}^d} \frac{1}{2} \|\theta - \theta_t\|^2 + C\xi \text{ s.t. } \ell_H(\theta; (\mathbf{x}_t, y_t)) \leq \xi \text{ and } \xi \geq 0. \tag{6}$$

where C is a positive constant that controls the effect of the slack variable. The updating process, outlined in Algorithm PA, shares the closed form of (5), but with a different τ_t term as demonstrated in Crammer et al. (2006):

$$\theta_{t+1} = \theta_t + \tau_t y_t \mathbf{x}_t, \text{ where } \tau_t = \min\left\{C, \frac{\ell_H(\theta_t; (\mathbf{x}_t, y_t))}{\|\mathbf{x}_t\|^2}\right\}. \tag{7}$$

Algorithm PA (Soft-margin) Passive-aggressive online learning.

Input Parameter $C > 0$
Initialize $\theta_1 = (0, \cdots, 0)$
for $t = 1, \cdots, T$ **do**
  Receive covariate, $\mathbf{x}_t$
  Compute prediction, $\hat{y}_t = sign(\theta_t' \mathbf{x}_t)$
  Receive feedback, y_t
  Compute $\ell_H(\theta_t; (\mathbf{x}_t, y_t))$ as defined in Equation (3)
  Compute $\tau_t = \min\left\{C, \frac{\ell_H(\theta_t; (\mathbf{x}_t, y_t))}{\|\mathbf{x}_t\|^2}\right\}$
  Update $\theta_{t+1} = \theta_t + \tau_t y_t \mathbf{x}_t$
end for
Output Parameter vector of the PA classifier, θ_T

The introduced Algorithm PA consists of a linear predictor, which may seem restrictive. However, it can be further extended to non-linear predictors using Mercer kernels as demonstrated in Vapnik (1998). Recall that the classifier can be represented as a sum of inner products:

$$\theta_t' \mathbf{x}_t = \sum_{i=1}^{t-1} \tau_i y_i (\mathbf{x}_i' \mathbf{x}_t),$$

and consider a Mercer kernel K such that $K(\mathbf{x}, \mathbf{x}') = \boldsymbol{\Phi}(\mathbf{x})' \boldsymbol{\Phi}(\mathbf{x}')$, where $\boldsymbol{\Phi}$ is a "feature map" mapping $\mathbf{x}$ to another space (possibly with higher dimensions). An explicit representation for $\boldsymbol{\Phi}$ is not necessary as long as it defines an inner product. Replacing the inner products in Algorithm PA with $K(\mathbf{x}, \mathbf{x}')$ gives a nonlinear version of the PA algorithm, represented as:

$$\hat{y}_t = sign(\theta_{t-1}' \boldsymbol{\Phi}(\mathbf{x}_t)), \text{ where } \theta_{t-1}' \boldsymbol{\Phi}(\mathbf{x}_t) = \sum_{i=1}^{t-1} \tau_i y_i K(\mathbf{x}_i, \mathbf{x}_t). \tag{8}$$

3.2. State-of-the-art cost-sensitive online learning algorithms

The PA algorithm solves the problem of updating the estimation of θ as new information becomes available. However, it overlooks the impact of different misclassification error costs, which can lead to suboptimal results in cost-sensitive settings as noted in Bahnsen et al. (2014), Elkan (2001), Höppner et al. (2021) and Vanderschueren et al. (2022). Consequently, CS algorithms appear in the OL literature, which are introduced in this section.

3.2.1. Passive-aggressive prediction-based algorithm

The passive-aggressive prediction-based (PAPB) algorithm proposed in Crammer et al. (2006) extends Algorithm PA considering a classification margin defined by an asymmetric loss function. The PAPB algorithm considers the following optimization problem:

$$\theta_{t+1} = \arg\min_{\theta \in \mathbb{R}^d} \frac{1}{2} \|\theta - \theta_t\|^2 \text{ s.t. } y_t \theta' \mathbf{x}_t - \hat{y}_t \theta' \mathbf{x}_t \geq \sqrt{\ell_{PB}(\hat{y}_t, y_t)}, \tag{9}$$

where ℓ_{PB} is a loss function with different values depending on the error type (FN or FP). This corresponds to the constant version of the loss function in (13) below. Considering a soft-margin approach as in (6) in the optimization problem in (9) leads to the update strategy (Crammer et al., 2006):

$$\theta_{t+1} = \theta_t + \tau_t (y_t - \hat{y}_t) \mathbf{x}_t, \quad \tau_t = \min\left\{C, \frac{(\hat{y}_t - y_t)\theta_t' \mathbf{x}_t + \sqrt{\ell_{PB}(\hat{y}_t, y_t)}}{\|(\hat{y}_t - y_t)\mathbf{x}_t\|^2}\right\}. \tag{10}$$

3.2.2. Cost-sensitive online gradient-descend algorithm

The cost-sensitive online gradient descent (CSOGD) algorithm, studied in Wang et al. (2012) and Zhao et al. (2015), extends the on-

line gradient descent approach introduced in Hazan et al. (2007) by considering misclassification costs in the parameter updating process:

$$\theta_{t+1} = \begin{cases} \theta_t + \lambda \rho_t y_t \mathbf{x}_t, & \ell_{CH}(\theta_t; (\mathbf{x}_t, y_t)) > 0 \\ \theta_t, & \text{otherwise.} \end{cases} \quad (11)$$

where I is the characteristic function, equals 1 when the expression between brackets is true and 0 otherwise, $\rho_t = \rho I(y_t = 1) + I(y_t = -1)$, $\rho = \frac{C^{FN}}{C^{FP}}$ is the ratio between the impact of a FN error and a FP error, $\lambda > 0$ is a learning rate parameter and $\ell_{CH}(\theta_t; (\mathbf{x}_t, y_t))$ is the cost-sensitive version of the hinge loss function in (3):

$$\ell_{CH}\left(\theta_t; (\mathbf{x}_t, y_t)\right) = \left(\rho I(y_t = 1) + I(y_y = -1)\right) \max\left\{0, 1 - y_t \theta_t' \mathbf{x}_t\right\}$$

3.2.3. Perceptron algorithm with uneven margin

The perceptron algorithm with uneven margin (PAUM) proposed in Li et al. (2002) is the CS version of the perceptron algorithm (Rosenblatt, 1958). The predicted class for an observation $\mathbf{x}_t$ is assigned as:

$$\hat{y}_t = sign(\theta' \mathbf{x}_t + b)$$

where $\theta \in \mathbb{R}^d$ and $b \in \mathbb{R}$. Considering different margins for the positive and negative classes, τ_{+1} and τ_{-1} respectively, $R = \max_t \|\mathbf{x}_t\|$ and a given learning rate η, starting with all variables equal to 0, the updating process of the PAUM algorithm is given by:

$$\theta_{t+1} = \begin{cases} \theta_t + \eta y_t \mathbf{x}_t, & y_t(\theta_t' \mathbf{x}_t + b_t) \le \tau_{y_t} \\ \theta_t, & \text{otherwise.} \end{cases} \quad (12)$$

$$b_{t+1} = \begin{cases} b_t + \eta y_t R^2, & y_t(\theta_t' \mathbf{x}_t + b_t) \le \tau_{y_t} \\ b_t, & \text{otherwise.} \end{cases}$$

3.3. Instance-dependent cost-sensitive passive-aggressive algorithm

Algorithm PA has shown good performance in some settings, such as online advertising, where the agent is only concerned with whether the customer clicks or not (Li et al., 2010). As for the CS approaches presented previously, they are designed to solve the class imbalance problem considering constant costs depending on the type of error (FP or FN). However, in settings where the misclassification cost depends on the characteristics of the instance, these assumptions are too restrictive and naive.

This section proposes a novel instance-dependent cost-sensitive passive-aggressive (IDCSPA) algorithm that extends the previously introduced PAPB algorithm to the IDCS setting. This approach retains the advantages of OL, but further incorporates the instance-dependent impact of misclassification errors. This is expected to result in a proficient model.

Since the PA algorithm, by definition, focuses only on losses, the reward function in Eq. (1) is adapted to consider only misclassification costs:

$$\ell_{CS} = I(y_i = -1) I(\hat{y}_i = 1) C_i^{FP} + I(y_i = 1) I(\hat{y}_i = -1) C_i^{FN}, \quad (13)$$

where C_i^{FP} and C_i^{FN} are the FP and FN costs respectively, as defined in Table 1. The loss function ℓ_{CS} in Eq. (13) is introduced into the PAPB optimization problem (9) to obtain the proposed IDCSPA algorithm. Thus, at each step t, the parameter θ_t is fitted by solving the optimization problem:

$$\theta_{t+1} = \arg\min_{\theta \in \mathbb{R}^d} \frac{1}{2} \|\theta - \theta_t\|^2 \text{ s.t. } y_t \theta' \mathbf{x}_t - \hat{y}_t \theta' \mathbf{x}_t \ge \sqrt{\ell_{CS}(\hat{y}_t, w_t, y_t)}, \quad (14)$$

where $\hat{y}_t = sign(\theta_t' \mathbf{x}_t)$ as for Algorithm PA. The optimization problem in (14) retains the idea of varying the current estimate as little as possible, while satisfying a certain margin, which in this case depends on the loss function ℓ_{CS} in (13). The optimization problem in Eq. (14) has a close form solution generalizing the solution developed in Crammer et al. (2006):

$$\theta_{t+1} = \theta_t + \tau_t (y_t - \hat{y}_t) \mathbf{x}_t, \text{ where } \tau_t = \frac{(\hat{y}_t - y_t)\theta_t' \mathbf{x}_t + \sqrt{\ell_{CS}(\hat{y}_t, w_t, y_t)}}{\|(\hat{y}_t - y_t)\mathbf{x}_t\|^2}. \quad (15)$$

As for the PA algorithm in Equation (5), the update strategy in (15) may be too aggressive, especially since we include the loss function in the estimation. To consider a soft-margin approach, extending the proposal in Eq. (7), the IDCSPA updating rule is defined by:

$$\theta_{t+1} = \theta_t + \tau_t (y_t - \hat{y}_t) \mathbf{x}_t, \quad \tau_t = \min\left\{ C, \frac{(\hat{y}_t - y_t)\theta_t' \mathbf{x}_t + \sqrt{\ell_{CS}(\hat{y}_t, w_t, y_t)}}{\|(\hat{y}_t - y_t)\mathbf{x}_t\|^2} \right\}, \quad (16)$$

where $C > 0$ is the parameter that controls the aggressiveness of the algorithm. The resulting classifier is shown in Algorithm IDCSPA, which as for the PA algorithm, can be extended to non-linear predictors by replacing the inner products with a Mercer kernel, similar to what was presented in (8).

Algorithm IDCSPA Instance-dependent cost-sensitive (soft-margin) passive-aggressive online learning (IDCSPA)

1: **Input** Parameter $C > 0$
2: **Initialize** $\theta_1 = (0, \cdots, 0)$
3: **for** $t = 1, \cdots, T$ **do**
4: **Receive** covariate $\mathbf{x}_t$
5: **Compute** prediction $\hat{y}_t = sign(\theta_t' \mathbf{x}_t)$
6: **Receive** feedback, y_t
7: **Compute** $\tau_t = \min\left\{ C, \frac{(\hat{y}_t - y_t)\theta_t' \mathbf{x}_t + \sqrt{\ell_{CS}(\hat{y}_t, w_t, y_t)}}{\|(\hat{y}_t - y_t)\mathbf{x}_t\|^2} \right\}$, with ℓ_{CS} as in (13)
8: **Update** $\theta_{t+1} = \theta_t + \tau_t (y_t - \hat{y}_t) \mathbf{x}_t$
9: **end for**
10: **Output** Vector classifier θ_T

4. Bandit algorithms

The OL framework aims to optimize exploitation with all the available information, but does not consider exploration beyond the predictions of the model. However, many real-world problems as credit risk (Achab et al., 2018; Visantavarakul, 2022) or online advertising (Li et al., 2010) require choosing among multiple options and learning the best one through successive trial and error. Bandit algorithms are a class of reinforcement learning algorithms designed to balance the process of seeking new information (exploration) with maximizing rewards based on existing knowledge (exploitation) in sequential decision problems. Since exploration and exploitation cannot be done simultaneously, in Sutton and Barto (2018) it is concluded that there is a trade-off between these two goals. Thus, at each step, the agent must decide whether to exploit the known option with the highest expected reward or to explore a less known option, potentially discovering better strategies for future gains.

Bandit algorithms receive their name from a very illustrative example represented in Fig. 1, where there are multiple bandit machines with different (unknown) reward distributions (Achab et al., 2018). A gambler (agent) stares at this series of one-armed bandits (actions), each promising unknown payouts. The agent is faced with the dilemma of pulling the familiar lever that is currently spitting out coins (exploitation) or trying a new one for potentially bigger wins (exploration).

The agent's goal is to learn the reward distribution of each arm and find a strategy (or policy) in order to maximize the cumulative reward over time. A naive approach would be to try all arms once and then choose the one with the highest reward. Nevertheless, this can be misleading, as the initial result could be pure luck. On the other

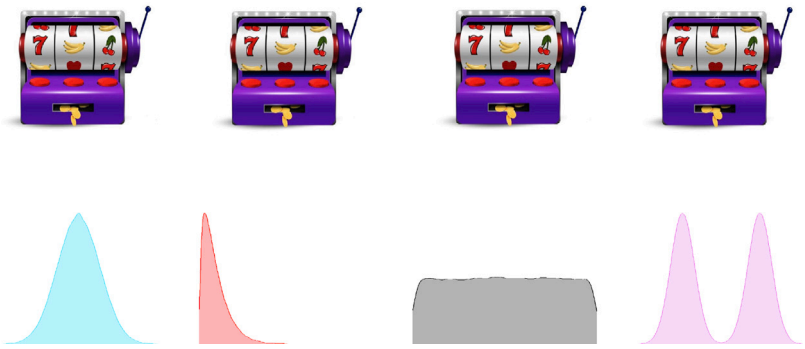


Fig. 1. Bandit machines with different reward distributions (bottom graphs).

hand, endless exploration will lead to missing the benefits of the best option. Bandit algorithms address this exploitation-exploration trade-off by strategically choosing between exploring less known options and exploiting the seemingly best one. Considering that instead of multiple slot machines there is a single one with multiple arms to pull, the problem is usually known as the multi-arm bandit (MAB) problem.

In this section, state-of-the-art MAB algorithms are presented along their drawbacks, motivating the novel CS logistic bandit algorithm proposed in this paper. There are three main bandit algorithms in the literature. The Epsilon-Greedy algorithm (see Sutton & Barto, 2018), a greedy algorithm allowing for exploration, is an intuitive way of forcing exploration. At each step, it exploits the most lucrative arm with probability $1-\epsilon$, and explores a random new arm with probability ϵ. However, this algorithm does not explore the environment efficiently, since it selects new actions with equal probability, regardless of whether it is actually a good action to explore. Two approaches are proposed to overcome this drawback. The first is the Upper Confidence Bound (UCB) algorithm (Auer et al., 2002; Kaufmann et al., 2014; Lattimore & Szepesvári, 2020), which prioritizes regions with high potential rewards while accounting for uncertainty through confidence bounds. Based on optimism in the face of uncertainty, the action with the highest upper confidence bound for the expected reward is selected. In this way, UCB exploits the best arm while exploring those with promising rewards, minimizing the risk of overlooking hidden riches. The other approach is Thompson Sampling (TS), introduced in Thompson (1933) and further studied in Kaufmann et al. (2012), Korda et al. (2013) and Russo et al. (2020). It embraces the probabilistic nature of the problem by treating models as probabilistic terms rather than deterministic truths. TS samples model parameters from their posterior distribution, updated with the observations collected up to time t. This uncertainty-aware approach allows TS to adapt to dynamic scenarios while exploring the arms for which there is less information.

4.1. Contextual bandits

The previously introduced MABs blindly explore actions with unknown rewards. Nevertheless, real-world scenarios often provide additional information about each option. For example, the socioeconomic characteristics of the customer in credit risk or the search history in advertisement recommendation. This additional knowledge is called *context*, and it can change the optimal choice as the best action may differ in different contexts (Langford & Zhang, 2007; Pandey et al., 2007; Wang et al., 2005; Zhou, 2016).

Contextual multi-armed bandits (CMAB) take this context into account, tailoring predictions to each unique situation. Thus, improving performance and allowing the generalization of the information across actions (Russo et al., 2020). As a result, CMABs have found applications in many different areas, such as recommendation engines (Lai & Robbins, 1985; Li et al., 2010; Tang et al., 2013), (personalized) healthcare (Rabbi et al., 2015), and portfolio selection (Shen et al., 2015), among others.

A CMAB, B, is defined by a set of available arms $\mathcal{A} \subseteq \{1, \dots, K\}$. The set of arms $\mathcal{A}$ is assumed to be fixed at each step $t = 1, \dots, T$. Each arm $k \in \mathcal{A}$ is described by a reward function r_k mapping d-dimensional context vectors $\mathbf{x}_t$ to some reward $r_{t,k} = r_k(\mathbf{x}_t)$ (Auer et al., 2002; Langford & Zhang, 2007). A policy π observes the current state of the world represented by the context feature $\mathbf{x}_t$. Using some arm-selection strategy based on the previous information, $H_t = \left\{(\mathbf{x}_i, r_{i,a_i})\right\}_{i=1}^{t-1}$, the policy chooses one of the available arms, defined as taking action $a_t \in \mathcal{A}$. Then, the policy gets the reward of the chosen arm, r_{t,a_t}. With the new observation $(\mathbf{x}_t, r_{t,a_t})$ the policy updates its strategy. The goal of the policy π is to maximize its cumulative reward $\sum_{t=1}^{T} r_{t,a_t}$. This is the unscaled version of the expected reward in (2).

One approach when having different contexts might be to apply a non-contextual MAB algorithm to each context. However, the relationship between the contexts is lost, so learning one context does not help learning the others as noted in Zhou (2016). In addition, this approach assumes that the contexts can be enumerated, which is not true when they are continuous. The most extended approach to solving CMAB problems is to consider stochastic bandit algorithms, which assume that the reward of each arm follows an unknown probability distribution depending on the context. The main state-of-the-art proposals are presented in the following subsections.

4.1.1. Contextual Epoch-Greedy

The Contextual Epoch-Greedy (CEG) algorithm studied in Langford and Zhang (2007) considers two well differentiated steps of exploration and exploitation. During exploration, it gathers information by randomly selecting each of the available arms to obtain information about their rewards along the contextual variables. The goal is to create unbiased samples by randomly pulling all arms to improve the accuracy of learning. This step leads to large immediate regret, but can potentially be reduced for the exploitation step. With the information obtained, a model $s : \mathbf{X} \mapsto \mathcal{A}$ is estimated in a class S of models to select the optimal action a_t given a context $\mathbf{x}_t$.

Once a satisfactory regret is obtained with the estimated model, the algorithm switches to the exploitation phase up to the horizon T. For a Bernoulli bandit, the expected regret in the exploration step can be as large as $O(1)$ when all chosen actions have a reward of 0 and the optimal arm has a reward of 1. In the exploitation step, the regret is expected to be much smaller. Considering n exploration steps, where the average regret is e_n, the total regret is bounded by $n + (T - n)e_n$. Consequently, the optimal strategy is to switch from exploration to exploitation at the step n that minimizes the previous expression, as demonstrated in Langford and Zhang (2007).

4.1.2. LinUCB

The Linear Upper Confidence Bound (LinUCB) algorithm, as proposed in Chu et al. (2011) and Li et al. (2010), extends the UCB algorithm introduced in Auer et al. (2002) to accommodate to context information. It assumes that the expected reward of an arm k is linear with respect to the feature vector $\mathbf{x}_t$ through an unknown coefficient vector θ_k:

$$E[r_{t,k} \mid \mathbf{x}_t] = \theta_k' \mathbf{x}_t \tag{17}$$

In Li et al. (2010) it is shown that, under the linear assumption in (17), a confidence interval for the expected reward can be efficiently computed. Considering ridge regression, the parameter vectors θ_k are estimated as:

$$\widehat{\theta}_k = \left(\mathbf{D}_k' \mathbf{D}_k + I_d\right)^{-1} \mathbf{D}_k' \mathbf{r}_k$$

where $\mathbf{D}_k \in \mathcal{M}^{(t-1)\times d}$ is the design matrix whose rows correspond to the observed instances $\{\mathbf{x}_i\}_{i=1}^{t-1}$ for the arm k, $\mathbf{r}_k$ the analog for the rewards, and I_d is the $d \times d$ identity matrix. Then, as shown in Li et al. (2010):

$$\mathbb{P}\left(|\widehat{\theta}_k \mathbf{x}_t - E[r_{t,k} \mid \mathbf{x}_t]| \le \alpha \sqrt{\mathbf{x}_t' (\mathbf{D}_k' \mathbf{D}_k + I_d)^{-1} \mathbf{x}_t}\right) \ge 1 - \delta \tag{18}$$

for any $\delta > 0$, where $\alpha = 1 + \sqrt{\ln(2/\delta)/2}$. The above inequality gives a bound on the expected reward of an arm k. Extending the UCB logic of optimism in the face of uncertainty, the policy in LinUCB selects the arm with the largest expected reward bound (18) as shown in Algorithm LinUCB.

Algorithm LinUCB LinUCB algorithm

Input Confidence parameter $\delta \in [0, 1]$
Initialize $\mathbf{A}_a \leftarrow I_d$
Initialize $\mathbf{b}_a \leftarrow \mathbf{0}_d$
for $t = 1, \cdots, T$ **do**
  for $a \in A_t$ **do**
    Compute $\widehat{\theta}_a \leftarrow \mathbf{A}_a^{-1} \mathbf{b}_a$
    Compute $r_{t,a} \leftarrow \widehat{\theta}_a' \mathbf{x}_t + \left(1 + \sqrt{\ln(2/\delta)/2}\right) \sqrt{\mathbf{x}_t' (\mathbf{x}_t \mathbf{x}_t' + I_d)^{-1} \mathbf{x}_t}$
  end for
  Compute $a_t = \text{argmax}_{a \in A_t} r_{t,a}$
  Receive Reward r_{a_t}
  Update $\mathbf{A}_a \leftarrow \mathbf{A}_a + \mathbf{x}_t \mathbf{x}_t'$
  Update $\mathbf{b}_a \leftarrow \mathbf{b}_a + r_{a_t} \mathbf{x}_t$
end for
Output Reward parameters $\widehat{\theta}_k$

4.1.3. Logistic bandits

In the Bernoulli bandit problem, the agent chooses an action a_t and receives a binary feedback $y_t \in \{0, 1\}$ Bernoulli distributed. The logistic bandit (LB) model extends Bernoulli bandits by incorporating generalization across actions through the context $\mathbf{x}_t$. As in logistic regression, the conditional probability of the response variable is modeled with a logistic function through a linear combination of the context vector. For each arm k, the probability of receiving a reward is modeled as $P(Y = 1 \mid \mathbf{X} = \mathbf{x}) = s(\mathbf{x}; \theta_k) = (1 + \exp(-\theta_k' \mathbf{x}))^{-1}$, where $\theta_k \in \mathbb{R}^d$ is the parameter vector to be estimated.

At each step t, the parameter θ_k associated with each arm k is estimated as the maximizer of the likelihood function given the available information:

$$\widehat{\theta}_{k,t} = \underset{\theta}{\text{argmax}} \ \mathcal{L}_{k,t}^L(\theta) \tag{19}$$

where

$$\mathcal{L}_{k,t}^L(\theta) = \frac{1}{t} \sum_{i=1}^{t} I(a_i = k) \left\{ y_i \ln[s(\mathbf{x}_i; \theta)] + (1 - y_i) \ln[1 - s(\mathbf{x}_i; \theta)] \right\} \tag{20}$$

is the likelihood function for an arm k and information up to the step t. The main difference with the classical logistic model is the exploration carried out by the logistic bandit and the continuous updating as more information is available. The exploration is imposed by considering Thompson Sampling (TS) as introduced in Thompson (1933). There are other proposals for allowing the agent to explore (Dumitrascu et al., 2018; Faury et al., 2020; Zhang et al., 2016). However, in Sutton and Barto (2018) TS is found to be more effective, for which further approaches are not considered.

In TS, at each step t, the parameter vector $\theta_{k,t}$ is treated as a random variable sampled from the posterior distribution of $\widehat{\theta}_{k,t}$, estimated from the available information. The sampled vectors are used to estimate the reward probability on each arm k as $s(\mathbf{x}; \theta_{k,t})$, and the action that maximizes the expected reward is selected. Feedback on the selected action is received, and the new information is used to update the posterior distribution of $\widehat{\theta}_{k,t}$ using Bayes' rule (Dumitrascu et al., 2018; Russo et al., 2020). For unexplored arms, the tails of the distribution are heavier, favoring exploration. As more information is obtained, the uncertainty is reduced, favoring the exploitation of the most profitable arms.

Fig. 2 shows an example of different steps of TS. The dotted line represents the estimator at each step and the curves represent the density over which it is resampled. In the early steps, there is a lot of uncertainty, favoring exploration. As more information becomes available, the density of the estimator becomes more concentrated, favoring exploitation.

The drawback of the logistic bandit algorithm is that it results in a computationally intractable posterior, which makes the implementation of TS challenging as noted in Dumitrascu et al. (2018). To address this, the Laplace approximation is used, which is based on a Taylor series expansion of the distribution function up to the second order. The Laplace approximation is expected to work well in the logistic framework, with a smooth density peaked around its mode (Dumitrascu et al., 2018; Gamerman & Lopes, 2006). There are other approaches to performing the sampling process, such as Metropolis Hasting or Langevin Monte Carlo Markov Chain. However, in Visantavarakul (2022) no significant improvement is observed with these approaches, for which no further algorithms are considered.

For the logistic bandit, given the estimator $\widehat{\theta}_{k,t}$ of θ_k as in Eq. (19), considering the Laplace sampling approximation, the parameter vector used in the step $t + 1$ is resampled from the distribution:

$$\theta_{k,t+1} \sim N_d(\widehat{\theta}_{k,t}, H_{k,t}) \tag{21}$$

where $H_{k,t}$ is the Hessian of the likelihood function in (20) evaluated at $\widehat{\theta}_{k,t}$. This process is analogous to the one represented in Fig. 2. Considering the Laplace sampling approximation in Eq. (21), the logistic bandit algorithm is illustrated in Algorithm LB.

Algorithm LB Logistic bandit algorithm

Initialize $\theta_{k,0} \leftarrow \mathbf{0}_d$
Initialize $H_0 \leftarrow I_d$
for $t = 1, \cdots, T$ **do**
  Compute $\theta_{k,t} \sim N_d(\widehat{\theta}_{k,t-1}, H_{k,t-1})$, $k \in A_t$
  Compute $a_t = \text{argmax}_{a \in \mathcal{A}} s(\theta_{a,t}; \mathbf{x})$
  Receive Result y_t
  Update $\widehat{\theta}_{k,t} = \text{argmax}_\theta \ \mathcal{L}_{k,t}^L(\theta)$, where $\mathcal{L}_{k,t}^L$ as defined in (20)
  Update $H_{k,t} = H\mathcal{L}_{k,t}^L(\widehat{\theta}_{k,t})$
end for
Output Parameters $\widehat{\theta}_{k,T}$

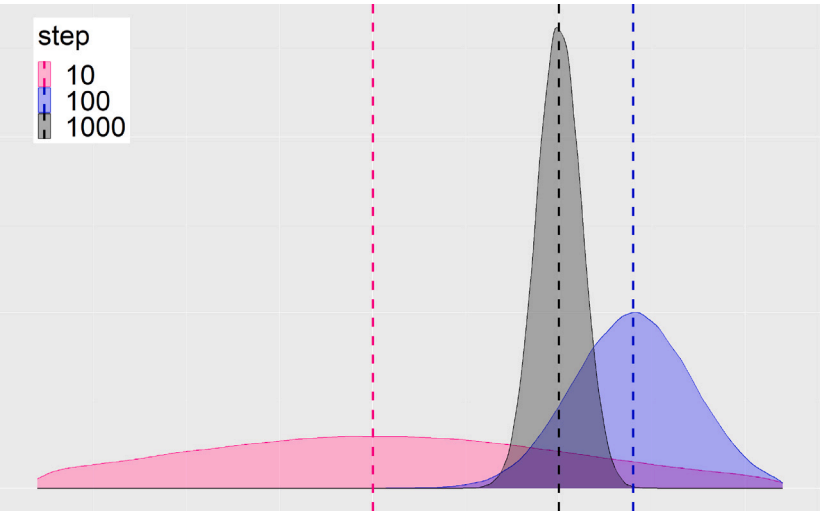


Fig. 2. Thompson sampling resampling densities for different steps.

4.2. Cost-sensitive logistic bandit

In this section, motivated by the success of the MAB policies recalled above, a cost-sensitive logistic bandit (CSLB) is proposed, extending the logistic bandit introduced in Algorithm LB, to solve the credit risk problem (a CS binary classification problem as introduced in Section 2).

The set of actions in credit risk is binary, $\hat{Y} \in \{0, 1\}$, for which it can be thought of as a Bernoulli one-armed bandit where the agent has to decide whether to pull the arm or not. If the operation is approved, the customer's behavior is observed; otherwise, no information is obtained. The goal is to maximize the total reward, which depends on the characteristics of each instance as defined by the reward function in Eq. (1).

Considering the logistic classifier, assume that $E[Y \mid \mathbf{X} = \mathbf{x}] = s(\mathbf{x}; \theta) = (1 + \exp(-\theta'\mathbf{x}))^{-1}$, where $\theta \in \mathbb{R}^d$. The goal is to estimate the parameter θ maximizing the expected reward $E[r(\hat{Y}, W, Y)]$. Assuming that $\hat{Y} = \mathbb{I}(s(\mathbf{X}) > u)$, with $u \sim U[0, 1]$, then $E[\hat{Y} \mid \mathbf{X} = \mathbf{x}] = P(\hat{Y} = 1 \mid \mathbf{x}) = s(\mathbf{x}; \theta)$ as introduced in C-Rella et al. (2025). Thus, the *average expected reward* (*AER*) (Bahnsen et al., 2014; Höppner et al., 2021) of a classifier s over the set sample $\mathcal{Z}_t = \{\mathbf{z}_i = (\mathbf{x}_i, w_i, y_i)\}_{i=1}^t$ can be defined as:

$$\mathcal{L}_t^{CS}(\theta) = \frac{1}{t} E\Big[\sum_{i=1}^{t} r(\hat{y}_i, w_i, y_i) \mid \mathcal{Z}_t\Big] = \frac{1}{t} \sum_{i=1}^{t} r(s(\mathbf{x}_i; \theta), w_i, y_i) \tag{22}$$

where r is the reward function in (1). The AER in (22) corresponds to the empirical version of $E[r(\hat{Y}, W, Y)]$ given the sample $\mathcal{Z}_t = \{\mathbf{z}_i = (\mathbf{x}_i, w_i, y_i)\}_{i=1}^t$. Thus, the CS estimator of θ is defined as:

$$\widehat{\theta} = \arg\min_{\theta} \mathcal{L}_t^{CS}(\theta) \tag{23}$$

The objective function in (22) is usually not convex nor has a unique minimum due to its dependence on W, Y and θ through $\hat{Y}$. To allow solving the optimization problem, a regularizer is added to the objective function to enforce convexity as suggested in Bahnsen et al. (2013), Höppner et al. (2021) and Stripling et al. (2018). This makes it easier for numerical optimization techniques to avoid suboptimal solutions and improves generalization performance. Therefore, it is considered the regularized AER:

$$\mathcal{L}_t^{\lambda}(\theta) = \frac{1}{t} \sum_{i=1}^{t} r(s(\mathbf{x}_i; \theta), w_i, y_i) + \lambda \|\theta\|_1 \tag{24}$$

where $\lambda \in \mathbb{R}^+$ is the parameter that controls the regularization. We consider a lasso regularization as proposed in C-Rella et al. (2025) and Höppner et al. (2021), without loss of generality. The corresponding estimator is defined as:

$$\widehat{\theta} = \arg\min_{\theta} \mathcal{L}_t^{\lambda}(\theta) \tag{25}$$

As the set of actions is assumed binary, the problem can be modeled with a single parameter θ such that $\hat{y}_t = I(s(\theta; \mathbf{x}_t) > h_t)$, where h_t is the decision threshold for the instance $(\mathbf{x}_t, w_t)$. For a reward function as in (1), considering the expected reward of an action,

$$R(\hat{y} \mid \mathbf{x}, w) = r(\hat{y}, w, 0)(1 - s(\mathbf{x})) + r(\hat{y}, w, 1)s(\mathbf{x})$$

the arm is pulled if the expected reward is greater than not pulling the arm, i.e. if $R(0 \mid \mathbf{x}, w) \geq R(1 \mid \mathbf{x}, w)$. This logic leads to the theoretically optimal decision threshold (Bahnsen et al., 2013; Elkan, 2001):

$$h_i = \frac{C_i^{FP} - C_i^{TN}}{C_i^{FP} - C_i^{TN} - C_i^{TP} + C_i^{FN}} \tag{26}$$

Introducing the regularized AER in (24) as the objective function in the logistic bandit in Algorithm LB and considering the decision rule in (26), the CS logistic bandit (CSLB) is represented in Algorithm CSLB. At each step, the CSLB selects the optimal action based on the estimator in (25) and the decision threshold in (26) considering all the information available. Similarly, by considering the TS sampling strategy, it efficiently explores all contexts to enhance long-term rewards. By combining exploration with a CS approach, the CSLB algorithm is expected to outperform previous models.

Algorithm CSLB Cost-sensitive logistic bandit algorithm

Input Reward function $r(\hat{y}_i, w_i, y_i)$ as in Equation (1)
Initialize $\theta_0 \leftarrow \mathbf{0}_d$
Initialize $H_0 \leftarrow I_d$
for $t = 1, \cdots, T$ **do**
  Compute $\theta_t \sim N_d(\widehat{\theta}_{t-1}, H_{t-1})$
  Receive Context $(\mathbf{x}_t, w_t)$
  Compute $h_t = \frac{C_t^{FP} - C_t^{TN}}{C_t^{FP} - C_t^{TN} - C_t^{TP} + C_t^{FN}}$
  Compute $a_t = I\left(s(\theta_t; \mathbf{x}_t) \geq h_t\right)$
  Receive Result y_t
  Update $\widehat{\theta}_t = \text{argmax}_{\theta}\ \mathcal{L}_t^{\lambda}(\theta)$, where $\mathcal{L}_t^{\lambda}$ as defined in (24)
  Update $H_t = H\mathcal{L}_t^{\lambda}(\widehat{\theta}_t)$
end for
Output Reward parameter $\widehat{\theta}_T$

5. Experimental results

The performance of the introduced OL and bandit algorithms are evaluated considering a wide range of simulated settings and the analysis of five real credit risk data sets. The objective is to understand how the different algorithms behave as the proportion of cases, the dependency relationship and the cost specification vary.

Online learning and bandit algorithms represent two different methodologies. Therefore, the algorithms are evaluated separately. After introducing the datasets, the results obtained for the different algorithms are presented, together with some comments on the performance of the classifiers. The optimization process for all the algorithms is implemented using the open source statistical software R (R Core Team, 2021).

Regarding OL algorithms, the passive-aggressive (PA) algorithm detailed in Algorithm PA, the CSOGD algorithm in Eq. (11), the PAUM algorithm as depicted in Eq. (12), the PAPB algorithm described in (10) and the proposed Algorithm IDCSPA are considered. The selection of the aggressiveness parameter C in the PA algorithms is done by cross-validation (CV) over the grid $C \in (0, 0.001, 0.01, 0.1, 1, 10, 100)$. In addition, we evaluate a static approach (Static) considering a linear model, $sign(\theta'\mathbf{X})$, previously fitted to a static training sample minimizing the hinge loss in (3), in line with the classical approach in credit risk.

Regarding bandit algorithms, we consider the logistic bandit described in Algorithm LB, the LinUCB algorithm summarized in Algorithm LinUCB and the CEG algorithm described in Section 4.1.1. We also consider a classical (Static) approach in the simulation study. This consists of training a logistic model on a training sample and then considering the fitted model on the new samples. All these algorithms are compared with the proposal in Algorithm CSLB, with the regularization parameter λ estimated by cross-validation over the grid $\lambda \in (0, 0.001, 0.01, 0.1, 1, 2, 5, 10, 20)$.

By considering static approaches, the improvement obtained with reinforcement learning algorithms is evaluated with respect to a static approach. Similarly, by considering cost-insensitive models such as the PA algorithm or the logistic bandit, a comparison is also made of the improvement obtained by considering a CS approach. Finally, this study evaluates the behavior of the proposed algorithms and compares them with the state-of-the-art.

For the OL algorithms, for a set $\{(\hat{y}_i, y_i, w_i)\}_{i=1}^n$ of the predicted label, the case indicator and the exogenous variable, the savings (Bahnsen et al., 2014; Höppner et al., 2021) are considered to evaluate model performance:

$$\text{Savings} = 1 - \frac{\sum_{i=1}^n \ell(\hat{y}_i, w_i, y_i)}{\sum_{i=1}^n y_i w_i} \tag{27}$$

where ℓ is the loss function defined in Eq. (13). This metric provides a standardized measure of the losses prevented compared to the base case scenario, allowing for a reasonable comparison across different models and problems. The maximum value this metric can take is 1, in which case all observations are well classified. For the bandit algorithms, we evaluate model performance considering the AER introduced in Eq. (22).

Since a scaled version of the same objective function is obtained with the same a/c and b/c ratios in the cost matrix in Table 1, the values $b = 10$ (the analysis cost) and $c = 0.75$ (the loss given default) are fixed and $a = \{0.05, 0.1, 0.2\}$ (the benefit per operation). This provides valuable insights into model performance under different false positive and false negative rates.

Model performance is evaluated using both cost-sensitive and classification metrics, taking into account the savings in (27) or the AER in (22) as appropriate, the positive predicted proportion (PP), the precision ($PPV = TP/PP$), the recall $= TP/(TP + FN)$ and the F score.

Table 2
Data generation models for the simulation study, where μ is the link function, $V = \theta_0'\mathbf{X}$ a linear combination of the covariates and ϵ an error term.

Model	Index	$\mu(V)$	ϵ
Logit	$V = \theta_0'\mathbf{X}$	$\frac{e^V}{1+e^V}$	$N(0,1)$
Squared	$V = \theta_0'\mathbf{S}$	$\frac{e^V}{1+e^V}$	$N(0,1)$
Qubic	$V = \theta_0'\mathbf{Q}/5$	$V(V-2)(V+2)$	$N(0,1)$
Ker & Sam	$V = \theta_0'\mathbf{X}$	V	$.5N(3,1) + .5N(-3,1)$
Scobit	$V = \theta_0'\mathbf{X}$	$1 - \frac{1}{(1+e^V)^{1}0}$	$N(0,1)$
Klein & Spady	$V = \theta_0'\mathbf{X}$	V	$N(0, \sqrt{0.1}(1+V^2))$

5.1. Simulation study

This section explores the behavior of the algorithms presented in a simulation study motivated by the credit risk problem introduced in Section 2. By controlling the data generation process we can observe and compare the strengths and weaknesses of each algorithm, providing valuable guidance for selecting the most appropriate method for specific real-world applications.

5.1.1. Simulation models

The data is simulated based on six different models, which are summarized in Table 2. For all simulations, the underlying model is given by $P(Y = 1) = \mu(v+\epsilon)$, where μ is a link function and $V = \theta_0'\mathbf{X}$ is a linear combination of the predictor covariates $\mathbf{X} = (1, X_1, \ldots, X_6) \in \mathbb{R}^7$. For the Logit model, $X_i \sim N(0,1)$, $\mu(v) = \frac{e^v}{1+e^v}$ and $\epsilon \sim N(0,1)$. For the Squared model, $V = \theta_0'\mathbf{S}$ where $\mathbf{S} = (1, S_1, \ldots, S_6)$ and $S_j = X_j^2$. The Qubic model is constructed similarly, but defines $V = \theta_0'\mathbf{Q}$, with $\mathbf{Q} = (1, Q_1, \ldots, Q_6)$ and $Q_j = X_j^3/5$. For the Ker & Sam model, introduced in Ker and Sam (2018), $X_j \sim N(0,1)$, $j = 1,3,5$, $X_j \sim \chi_3^2$, $j = 2,4,6$ and ϵ is drawn from a normal mixture such that $\epsilon \sim N(3,1)$ or $\epsilon \sim N(-3,1)$ with probability 0.5. This corresponds to an homoscedastic model but with a bimodal error term. The Scobit model is defined as the skewed version of the Logit model, considering the link function $\mu(v) = 1 - \frac{1}{(1+e^v)^6}$. For the Klein & Spady model, introduced in Klein and Spady (1993), a heteroskedastic error term $\epsilon \mid V \sim N(0, \sigma(V))$ is considered, with $\sigma^2(V) = 0.1(1+V^2)^2$. For all simulations, $\theta_0 = (\theta_{00}, \theta_{01}, \ldots, \theta_{06}) = (\theta_{00}, 1, 2, -1, -2, 3, -3)'$ and two scenarios are considered. In the normal scenario, θ_{00} is chosen so that $\bar{y} \approx 0.1$. In the risky scenario, θ_{00} is chosen so that $\bar{y} \approx 0.25$. Thus, model performance is also explored at different levels of case proportions.

The exogenous variable is assumed to be independent from the covariates following Zadrozny and Elkan (2001). It is simulated as $W = 100W_1W_2$, with $W_1 \sim \chi_4^2$ and $W_2 \sim N(7, 0.8)$, where W_1 and W_2 are independent. The idea is to replicate the asymmetry commonly encountered in financial problems such as the one motivating this work.

Six models (introduced in Table 2), three cost settings ($a = \{0.05, 0.1, 0.2\}$) and two scenarios (normal and risky) are combined to generate 36 simulation settings. This provides a comprehensive test of the performance of the introduced classifiers under different dependency relationships and cost settings. For each scenario, samples are simulated with $n = 10{,}000$ and the algorithms are trained as introduced in Sections 3 and 4. This is repeated 20 times for each simulation scenario and the results are summarized by averaging over these 20 runs. Due to the number of simulations performed, the results are summarized in in Appendix. A summary of the results obtained with each of the algorithms is included in the form of graphs in Figs. 3–6. In this way we have a quick and visual representation of the performance of the models and the details are left for the Appendix.

5.1.2. Online learning algorithms

This section evaluates the performance of the OL algorithms presented in Section 3 over the simulated datasets. Figs. 3 and 4 show the box plot of the savings in (27) obtained for the 6 simulation models and

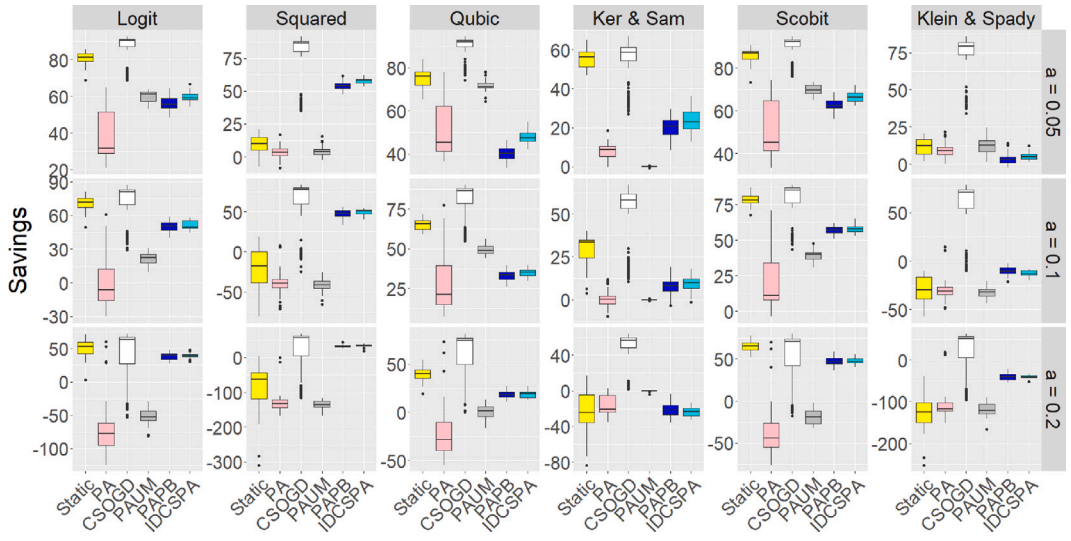


Fig. 3. Box plots of the savings in (27) for the OL algorithms in the normal scenario.

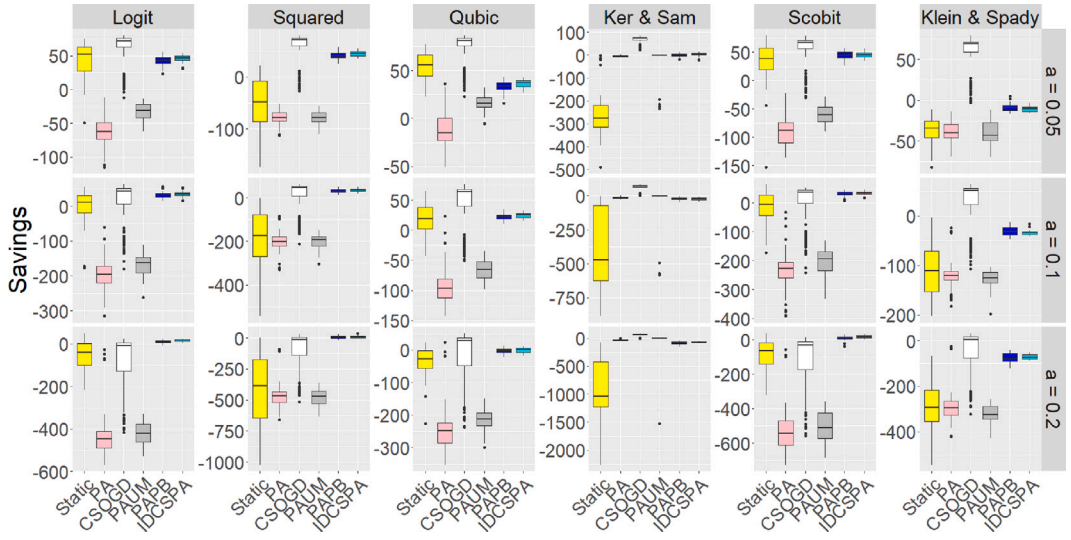


Fig. 4. Box plots of the savings in (27) for the OL algorithms in the risky scenario.

the 3 cost specifications for the normal and risky scenario, respectively. The effect of parameter a on model performance can be observed. This is due to the greater impact of false positives. Since OL algorithms only take losses into account, their performance decreases as the value of a increases. Something similar can be observed depending on the case proportion scenario. As expected, the more cases there are in the sample, the lower the savings is.

The static alternative performs well in many scenarios. However, the fact that the model is not updated with the new observations received implies a poor performance when the underlying model deviates from linearity. Similarly, the PA algorithm has the worst performance among the OL classifiers because it does not take costs into account.

The CSOGD model is the one that performs best in some scenarios, but still has a large variance in some settings. This is particularly the case for simulation models that are furthest from a linear model and where the impact of errors is greater. This variability across scenarios means that while it may be a good alternative, it must be treated with caution. Something similar happens with the PAUM model, although to a lesser extent.

As for the CS passive-aggressive algorithms, namely PAUM, PAPB, and IDCSPA, their behavior is generally quite similar, with a general improvement with the instance-dependent algorithm. The IDCSPA algorithm does not perform better than the state-of-the-art in the normal scenario. This is because the model tends to mark many observations as cases relative to the true proportion of cases as it can be seen in . As the proportion of cases increases, the model has more information to update its predictions and achieves better performance. Therefore, the proposed model should be considered with caution in highly imbalanced scenarios. Nevertheless, it is the best in most of simulated settings and gives stable results with the least variability regardless of the problem setting as shown in Figs. 3 and 4. Lastly, it is worth noting that the proposed IDCSPA algorithm consistently performs better than the other passive-aggressive algorithms. Taking this into account, Algorithm IDCSPA is considered to be the best performing model in the simulation study. Therefore, it is an interesting model when dealing with dynamic problems, especially in scenarios where interpretability is important, such as the credit risk problem, as it provides an interpretable model with a proficient performance.

5.1.3. Bandit algorithms

This section evaluates the behavior of the introduced bandit algorithms over the simulated datasets. Figs. 5 and 6 show the mean

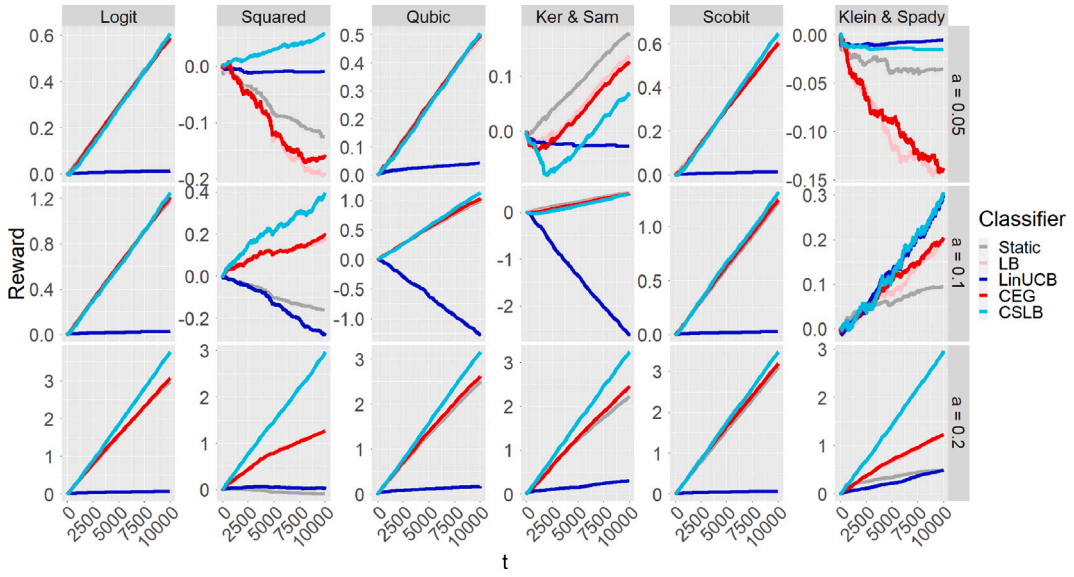


Fig. 5. Cumulative reward (in millions) for the bandit algorithms in the normal scenario.

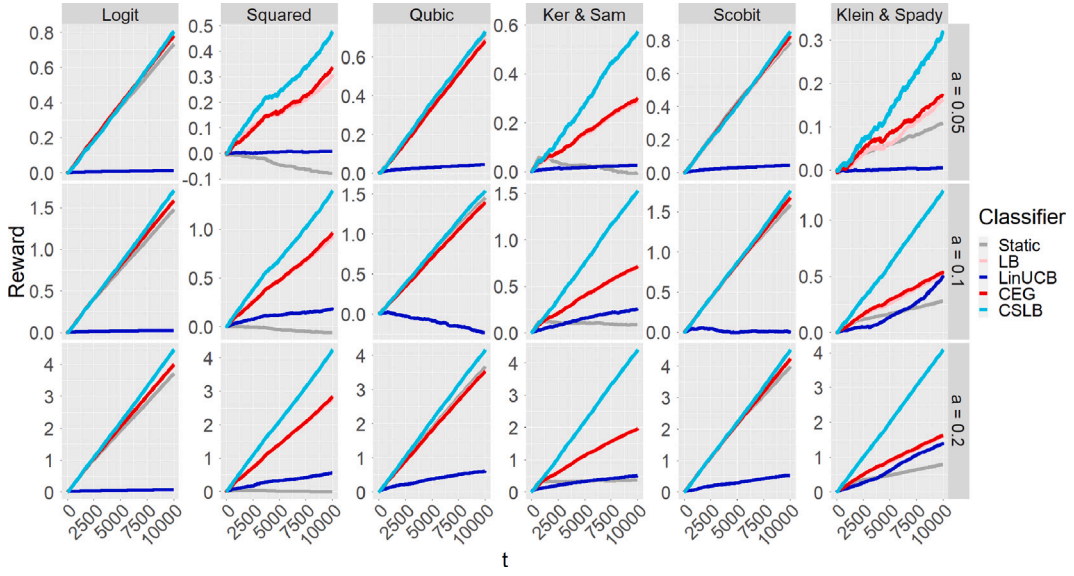


Fig. 6. Cumulative reward (in millions) for the bandit algorithms in the risky scenario.

cumulative reward obtained over the 20 simulations for the 6 simulation models and the 3 cost specifications for the normal and risky scenario, respectively. The X-axis represents the step and the Y-axis represents the cumulative reward.

For the bandit algorithms, as for the OL algorithms, we can see how the proportion of cases and the cost specification significantly affect the performance of the models. In this case, since we include both profits and losses in the objective function, the average reward increases as a does, which is the profit made on an approved good operation.

The LinUCB algorithm has a poor performance due to the fact that this algorithm only considers the context to model the expected reward. The exogenous variable is assumed to be independent of the covariates. Therefore, the model makes a naive prediction in all scenarios. Nevertheless, this optimistic approach appears effective in certain scenarios.

It is worth highlighting the good performance of the static model in some scenarios, such as Logit, Qubic or Scobit, despite the fact that it does not update predictions or take exploration into account. In the case of the Logit scenario, this good behavior is justified because the model is well specified. In the other two, despite the fact that the model is not well specified, the link function is similar to the assumed by the logistic model, which also makes the model behave well. As the misspecification of the logistic model is further broken down, its behavior deteriorates significantly, as can be seen in the Squared or Klein & Spady scenarios. In these simulations, the exploration performed by the LB and CEG models allows them to adapt much better to the problem. This is reflected in the higher expected rewards despite not considering a CS approach for model fitting.

With respect to the proposed CSLB algorithm, it can be seen that it practically achieves the best results in all simulations when compared to the state-of-the-art algorithms, both cost-insensitive and cost-sensitive. The biggest differences are observed as the value of a increases, due to a higher potential profit per good operation. It is noteworthy that the CSLB model, unlike the other models, obtains better results in the most risky scenario. This is because the model explores a greater number of cases in the first steps. Thus, in the long run, it has a more representative sample and consequently more accurate and better results are obtained.

The only simulation scenario where the proposed model performs worse than state-of-the-art models is the Ker & Sam simulation with $a = 0.05$ in the normal scenario. This is due to a significant loss in the early iterations, and although it is corrected, it is not compensated in the run considered. Perhaps, given the results obtained in the other simulation models, the CSLB will outperform the other approaches over a longer time frame.

Taking into account that the CSLB obtains the best results regardless of the proportion of cases, relationship dependencies, and cost specification, it is considered that the proposed CSLB algorithm achieves satisfactory performance and is an interesting alternative for cost-sensitive dynamic problems.

5.2. Real datasets study

This section examines the behavior of the reinforcement learning algorithms presented on various real credit risk data sets. The objective is to evaluate the performance of the different algorithms in real scenarios, after having studied their performance in simulated environments. After introducing the data sets, the results obtained for the different algorithms are presented, together with some comments on their performance.

The different datasets are selected to investigate the impact of class imbalance and the number of covariates on model performance. As none of the state-of-the-art datasets provide guidance on the cost of misclassification errors, several cost specifications are considered. Likewise, the effect of different ratios of false positive and false negative costs on model performance is studied. The values for the parameters of the cost matrix in Table 1 are taken in line with those considered in the previous section, i.e. $a \in \{0.05, 0.10, 0.20\}$, $b = 10$ and $c = 0.75$.

Five real credit risk data sets are considered, where some covariates are available to model a response variable (fraud or default). The objective is to minimize losses/maximize benefits, which depend on the amount of the transaction and the cost specification. For each data set, a 5-fold cross-validation is performed, stratified by case proportion and amount as suggested in Höppner et al. (2021). The mean results over the test sets are summarized in Tables 3–5 for the OL algorithms, and in Fig. 7 and in Tables A.12–A.14 in the Appendix for the bandit algorithms.

5.2.1. Datasets

The first data set (Credit Card[1] data set) is a fraud data set containing 28 variables resulting from a PCA along with a "Time" variable. The data set is undersampled so that they remain 15,000 transactions containing 465 (3.19%) frauds. The second one (GCD[2] data set) is a real credit risk data set of size $n = 1000$ with a case proportion of 30% and 24 variables available to model the default indicator. The third one (PAKDD[3] data set, as introduced in Linhart et al., 2009) consists of 11,111 prospective clients for which 4 numerical variables are available to model their probability of default. There are 3,047 (27.42%) defaults in the sample. The amount of the loan is simulated as $W \sim 100\chi_3$ in order to replicate the asymmetry commonly encountered in credit risk problems. The fourth one (ASF data set) is loaned by a collaborating bank (Abanca Servicios Financieros). It is composed of 23,377 loan requests collected between January 2022 and December 2022 with a default percentage of 3.06%. To preserve confidentiality, the number of registers in the original dataset is truncated, and so is the default proportion. It consists on 13 numerical variables which include socioeconomic information of the client and the point of sale where the credit is requested and characteristics of the operation. The fifth one (Drift data set) consists of 60,000 synthetically generated observations from the ASF data set, where concept and covariate drifts are introduced to test the performance of the different models in the presence of drifts. To do this, we follow (Nowok et al., 2016) with the syn function of the R package synthpop. First, a sample of size 15,000 is synthetically generated from the ASF data. To this data set we add 15,000 observations generated with a different distribution for the covariates, but preserving the dependence between $\mathbf{X}$ and Y. Another 15,000 instances are added introducing concept drift, generating the response variable with a different model than the one considered for the first 30,000 observations. Finally, 15,000 synthetically generated observations from the original ASF data set are added to form the final sample.

Table 3
Savings (Sav), accuracy (Acc), recall (Rec), positive predicted value (PPV), positive predicted proportion (PP) and F-score (FS) for the OL algorithms for the real datasets and cost setting $a = 0.05$.

Data set	Classifier	Sav	Acc	Rec	PPV	PP	FS
Credit card	Static	−52.36	43.23	11.18	0.66	54.30	1.24
	PA	−20.05	52.65	52.90	3.55	47.54	6.65
	CSOGD	63.71	73.93	98.44	11.36	29.16	20.28
	PAUM	−10.85	50.14	68.39	4.27	51.03	8.04
	PAPB	62.44	96.72	79.78	49.14	5.17	60.82
	IDCSPA	63.68	96.91	78.28	50.98	4.89	61.75
GCD	Static	27.03	53.65	25.00	23.96	31.33	24.47
	PA	31.82	65.57	35.67	41.47	25.83	38.35
	CSOGD	74.96	91.42	79.58	90.69	26.35	84.78
	PAUM	0.49	69.67	0.67	28.57	0.70	1.30
	PAPB	34.17	63.16	45.00	39.94	33.83	42.32
	IDCSPA	34.78	62.86	45.33	39.65	34.33	42.30
PAKDD	Static	34.96	54.85	42.63	28.21	41.13	33.95
	PA	22.76	61.01	26.28	27.43	26.08	26.84
	CSOGD	76.10	86.98	76.77	75.72	27.60	76.24
	PAUM	−0.01	72.77	0.00	0.00	0.01	
	PAPB	0.04	72.77	0.03	33.33	0.02	0.05
	IDCSPA	19.95	63.78	22.08	28.60	21.02	24.92
ASF	Static	−32.71	68.59	32.03	3.23	30.31	5.87
	PA	−17.03	61.64	58.47	5.05	38.93	9.31
	CSOGD	58.72	82.74	87.96	13.83	19.59	23.9
	PAUM	−3.85	63.75	77.78	6.31	37.96	11.67
	PAPB	−4.44	90.56	14.81	6.27	7.27	8.82
	IDCSPA	−4.40	90.50	14.81	6.23	7.33	8.77
Drift	Static	−22.70	83.35	5.33	1.44	13.41	2.27
	PA	−3.78	92.18	3.81	3.10	4.47	3.42
	CSOGD	72.87	98.03	71.95	73.36	3.56	72.65
	PAUM	−1.42	95.27	0.55	1.75	1.14	0.84
	PAPB	27.44	92.99	35.77	21.71	5.98	27.02
	IDCSPA	39.52	93.00	50.92	26.16	7.07	34.56

5.2.2. Online learning algorithms

This section evaluates the performance of the online learning algorithms presented in Section 3 over the real datasets previously introduced. Tables 3–5 summarize the savings obtained with the 6 algorithms compared for the 5 datasets and the 3 cost specifications. It can be observed how the performance of the models varies significantly depending on the value of the parameter a and the case proportion as in the simulation study. As expected, when there are more bad requests in the sample, the savings obtained is lower.

In line with the findings of the simulation study, the CSOGD algorithm consistently demonstrates the best overall performance. The proposed IDCSPA model is the next best performing algorithm across all scenarios. Although the IDCSPA algorithm does not outperform the CSOGD, it achieves the best results within the passive-aggressive models. This suggests that incorporating the instance-dependent loss function into the training process improves the performance of OL algorithms. These results corroborate in practical scenarios the results of the simulation study and indicate the potential for extending other

[1] https://kaggle.com/mlg-ulb/creditcardfraud.

[2] https://github.com/JLZml/Credit-Scoring-Data-Sets/tree/master/1.%20UCI%20Repository/German.

[3] https://github.com/JLZml/Credit-Scoring-Data-Sets/tree/master/2.%20PAKDD%202009%20Data%20Mining%20Competition.

Table 4
Savings (Sav), accuracy (Acc), recall (Rec), positive predicted value (PPV), positive predicted proportion (PP) and F-score (FS) for the OL algorithms for the real datasets and cost setting $a = 0.1$.

Data set	Classifier	Sav	Acc	Rec	PPV	PP	FS
Credit card	Static	−151.97	42.72	12.69	0.74	54.90	1.39
	PA	−103.49	51.51	50.32	3.31	48.51	6.20
	CSOGD	30.12	72.32	98.82	10.77	30.80	19.35
	PAUM	−69.31	48.34	72.26	4.34	53.08	8.19
	PAPB	54.48	97.22	79.78	54.32	4.68	64.63
	IDCSPA	59.12	97.10	79.14	53.03	4.76	63.50
GCD	Static	5.74	59.26	15.67	23.38	20.12	18.76
	PA	21.18	64.96	35.33	40.46	26.23	37.72
	CSOGD	70.22	90.22	75.00	90.82	24.80	82.15
	PAUM	−0.13	69.87	0.00	0.00	0.10	
	PAPB	26.96	62.36	43.00	38.62	33.43	40.69
	IDCSPA	27.35	62.76	43.67	39.22	33.43	41.32
PAKDD	Static	12.69	60.18	23.01	24.93	25.12	23.93
	PA	17.52	61.42	25.76	27.63	25.38	26.66
	CSOGD	73.77	86.98	77.24	75.51	27.84	76.36
	PAUM	−0.01	72.74	0.03	14.29	0.05	0.05
	PAPB	−0.00	72.77	0.00	0.00	0.01	
	IDCSPA	14.16	63.76	21.45	28.21	20.70	24.37
ASF	Static	−64.39	78.35	12.31	1.95	19.34	3.36
	PA	−91.36	60.75	61.98	5.34	40.07	9.83
	CSOGD	30.67	84.11	86	13.67	17.94	23.59
	PAUM	−75.38	66	66	5.39	34.91	9.97
	PAPB	−12.8	91.19	18	7.35	6.99	10.43
	IDCSPA	−6.13	91.99	21	9.42	6.36	13
Drift	Static	−49.56	83.35	5.33	1.44	13.41	2.27
	PA	−10.99	92.18	3.81	3.10	4.47	3.42
	CSOGD	71.37	98.03	71.95	73.36	3.56	72.65
	PAUM	−3.21	95.27	0.55	1.75	1.14	0.84
	PAPB	17.72	92.54	36.96	20.59	6.52	26.45
	IDCSPA	29.05	92.69	52.11	25.35	7.46	34.10

Table 5
Savings (Sav), accuracy (Acc), recall (Rec), positive predicted value (PPV), positive predicted proportion (PP) and F-score (FS) for the OL algorithms for the real datasets and cost setting $a = 0.2$.

Data set	Classifier	Sav	Acc	Rec	PPV	PP	FS
Credit card	Static	−178.65	56.18	9.68	0.75	41.25	1.39
	PA	−239.81	51.99	50.32	3.34	48.03	6.26
	CSOGD	−53.32	71.78	98.06	10.55	31.28	18.97
	PAUM	−187.69	47.95	80.43	4.75	53.99	8.97
	PAPB	35.53	97.05	78.92	52.43	4.80	63.00
	IDCSPA	45.16	97.59	80.65	58.96	4.36	68.12
German	Static	14.14	59.76	44.00	36.07	36.64	39.64
	PA	13.62	65.17	34.67	40.62	25.63	37.41
	CSOGD	70.71	90.47	77.83	89.04	26.25	83.06
	PAUM	−0.16	69.77	0.00	0.00	0.20	
	PAPB	15.26	63.96	43.33	40.62	32.03	41.94
	IDCSPA	14.65	61.46	48.67	38.73	37.74	43.13
PAKDD	Static	7.96	56.07	33.49	26.09	34.94	29.33
	PA	7.75	61.79	25.38	27.84	24.81	26.56
	CSOGD	69.83	86.75	76.28	75.36	27.55	75.82
	PAUM	0.24	72.73	0.35	39.39	0.25	0.70
	PAPB	−0.00	72.77	0.00	0.00	0.01	
	IDCSPA	6.89	63.03	21.48	27.27	21.44	24.03
ASF	Static	−235.03	65.91	29.65	2.76	32.85	5.05
	PA	−265.14	60.87	59.84	5.23	39.82	9.62
	CSOGD	−42.39	82.89	84.3	14.93	19.48	25.37
	PAUM	−243.08	64.06	68.6	6.36	37.22	11.64
	PAPB	−49.54	89.79	17.36	7.53	7.96	10.5
	IDCSPA	−42.01	90.7	12.4	6.38	6.7	8.43
Drift	Static	−103.27	83.35	5.33	1.44	13.41	2.27
	PA	−25.41	92.18	3.81	3.10	4.47	3.42
	CSOGD	68.37	98.03	71.95	73.36	3.56	72.65
	PAUM	−6.80	95.27	0.55	1.75	1.14	0.84
	PAPB	−2.16	92.54	36.96	20.59	6.52	26.45
	IDCSPA	7.93	92.82	51.56	25.67	7.29	34.27

online learning algorithms to cost-sensitive settings to achieve further improvements.

5.2.3. Bandit algorithms

This section evaluates the behavior of the bandit algorithms introduced in Section 4. Results are summarized in Fig. 7 and detailed in the Appendix in Tables A.12–A.14. Fig. 7 shows the mean cumulative reward obtained with the 5 considered algorithms for the 5 datasets and the 3 different cost specifications as in Fig. 5. As for the OL algorithms, we can see how the proportion of cases, the number of covariates and the cost specification significantly affect model performance.

As in the simulation study, the LinUCB algorithm is the worst performing approach. Regarding the performance of the Static approach, excluding the LinUCB algorithm, its performance is the worse. Thus, it is concluded that updating the models with all the available information and performing exploration significantly improves model performance, especially in real problems as those considered in this study.

With respect to the proposed CSLB algorithm, it can be seen that it achieves the best results in all the datasets and cost settings when compared to both cost-insensitive and cost-sensitive approaches. As for the simulation study, the biggest differences are observed as the value of a increases, where the CSLB algorithm is able to increase the profits even more thanks to a higher profit per good operation granted.

Regarding the Drift data set, it can be clearly seen a decline in model performance for all the algorithms as the drifts appear, specially in the step where it starts the concept drift. Nevertheless, the proposed CSLB is able to correct its estimates thanks to the exploration considered and obtain the best aggregated results in all the cost settings.

The proposed Algorithm CSLB obtains the best results in all the problems, regardless of the proportion of cases, sample size, relationship dependencies, and cost specification. Taking this into account, it is considered that the proposed model achieves satisfactory performance and is an interesting alternative for cost-sensitive dynamic problems as the credit risk problem.

6. Conclusions and future work

Motivated by the credit risk problem, this paper presents two novel algorithms designed to tackle cost-sensitive dynamic classification. In this paper, the PA algorithm (Crammer et al., 2006), is extended to the CS setting in Algorithm IDCSPA. In addition, the logistic bandit approach is extended with Algorithm CSLB for efficient CS learning considering exploration. The performance of the proposed models is empirically tested considering a wide range of simulations and the study of five real datasets. Promising experimental results demonstrate the effectiveness and efficiency of the proposed models compared to state-of-the-art algorithms. Furthermore, this study provides practitioners with valuable insights into the algorithmic performance in terms of case proportions, dependency relationships, and cost specifications.

Regarding OL algorithms, CSOGD and PAUM perform well in some scenarios but poorly in others, especially when misclassification error costs are larger. While PAPB and IDCSPA show similar behavior, the proposed IDCSPA algorithm emerges as the best model overall, offering consistent performance across different scenarios.

As for the bandit algorithms, the static approach gives a good performance in some simulations. Nevertheless, it suffers a performance decline in the real scenarios. Bandit algorithms such as ML and CEG correct model misspecification through exploration, leading to better adaptation and higher rewards. The proposed CS logistic bandit consistently outperforms state-of-the-art algorithms particularly excelling as the profit per good operation increases. This shows that the exploration performed by CSLB combined with a CS approach can significantly improve the potential benefits.

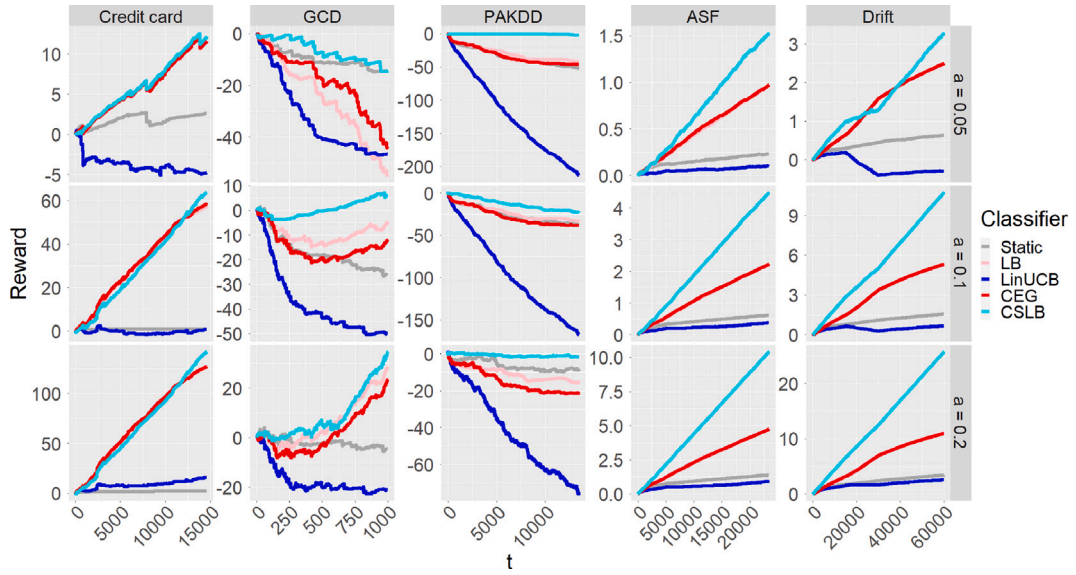


Fig. 7. Real datasets results (cumulative reward, in thousands for the credit card, GCD and PAKDD datasets, and in millions for the ASF and drift datasets) for the bandit algorithms.

The CS algorithms proposed in this paper prove to be effective and stable in most scenarios, making them good approaches for cost-sensitive dynamic problems. By proposing one approach that considers exploration and another that does not, the full range of alternatives is covered from the perspective of reinforcing learning in cost-sensitive classification problems. Thus, the dynamic cost-sensitive learning problem is solved for any setting.

Lastly, the good performance of the proposed algorithms opens the door to extend more reinforcement learning algorithms to the CS setting. In addition, one can consider extending algorithms that are not necessarily from the reinforcement learning literature. For example, consider the thresholding stage with a dynamic approach, updating the decision as more information becomes available. For this, it would be interesting to consider an empirical algorithm such as the one proposed in C-Rella et al. (2024) and C-Rella and Vilar (2024). The possible extensions are innumerable combining a cost-sensitive and dynamic approach.

CRediT authorship contribution statement

Jorge C-Rella: Conceptualization, Methodology, Software, Investigation, Writing. **David Martinez Rego:** Conceptualization, Validation, Investigation. **Juan M. Vilar:** Conceptualization, Validation.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This research is part of the grants PID2020-113578RB-I00 and PID2023-147127OB-I00 "ERDF/EU", funded by MCIN/AEI/10.13039/501100011033/. It has also been supported by the Xunta de Galicia, Spain (Grupos de Referencia Competitiva ED431C-2024/14) and by CITIC as a center accredited for excellence within the Galician University System and a member of the CIGUS Network, receives subsidies from the Department of Education, Science, Universities, and Vocational Training of the Xunta de Galicia. Additionally, it is co-financed by the EU through the FEDER Galicia 2021–27 operational program (Ref. ED431G 2023/01). The first author was financed by the Axencia Galega de Innovación Industrial PhD Grant 14-IN606D-2021-2607768.

Appendix. Experimental results

See Tables A.6–A.14.

Table A.6
Savings (Sav), accuracy (Acc), recall (Rec), positive predicted value (PPV), positive predicted proportion (PP) and F-score (FS) for the OL algorithms for the cost setting $a = 0.05$ in the simulated datasets.

Model	Classifier	Normal						Risky					
		Sav	Acc	Rec	PPV	PP	FS	Sav	Acc	Rec	PPV	PP	FS
Logit	Static	82.81	90.80	94.55	54.03	18.45	68.46	68.42	93.02	94.33	34.51	10.29	50.41
	PA	39.26	55.65	77.83	16.42	50.26	27.11	−62.73	50.83	68.03	5.11	50.54	9.50
	CSOGD	87.74	87.67	99.03	49.18	22.71	65.02	61.12	86.71	97.19	25.19	16.87	39.44
	PAUM	61.37	60.25	97.98	20.78	49.91	34.28	−30.05	52.81	96.60	7.22	50.73	13.43
	PAPB	57.86	89.71	62.47	51.25	12.95	56.22	43.25	93.99	54.38	32.92	6.35	40.71
	IDCSPA	58.66	88.02	66.03	44.67	15.31	53.28	44.47	93.90	55.30	32.08	6.50	40.34
Squared	Static	3.67	74.67	30.81	16.85	21.74	19.80	−6.55	88.18	30.99	11.53	10.49	16.57
	PA	2.50	49.46	50.07	9.87	50.56	16.49	−82.80	49.99	50.64	3.72	50.06	6.92
	CSOGD	79.10	86.68	92.34	45.98	21.77	60.84	57.62	86.72	94.58	24.55	16.56	38.52
	PAUM	3.39	50.24	49.97	10.01	49.75	16.67	−87.50	49.66	47.42	3.48	50.15	6.49
	PAPB	54.88	89.43	60.74	47.72	12.72	53.36	42.72	94.05	54.17	32.13	6.28	40.11
	IDCSPA	59.19	88.19	66.64	43.94	15.13	52.96	48.35	94.03	61.25	34.03	6.83	43.74

(*continued on next page*)

Table A.6 (*continued*).

Model	Classifier	Normal						Risky					
		Sav	Acc	Rec	PPV	PP	FS	Sav	Acc	Rec	PPV	PP	FS
Qubic	Static	73.08	81.65	88.66	43.94	29.69	58.63	62.73	87.95	94.42	29.84	16.70	45.24
	PA	51.55	58.17	76.87	22.70	49.72	35.04	−17.16	52.17	68.77	7.65	49.90	13.75
	CSOGD	90.72	88.40	98.23	58.47	25.75	72.72	73.22	87.05	96.85	33.37	18.15	48.88
	PAUM	73.16	64.44	96.97	28.84	49.34	44.45	11.86	54.91	94.53	10.47	50.01	18.83
	PAPB	39.66	81.21	47.21	38.57	17.98	42.41	33.73	90.45	43.78	27.65	8.90	33.65
	IDCSPA	48.76	79.39	57.79	37.31	22.91	45.34	33.50	90.78	41.42	28.29	8.31	33.31
Ker & Sam	Static	53.61	79.36	84.52	25.84	26.15	39.56	−3.21	86.32	57.76	10.41	14.03	17.60
	PA	8.10	87.22	16.33	24.87	7.38	16.01	−6.62	95.60	1.86	–	2.14	–
	CSOGD	56.82	96.31	56.79	95.15	4.78	70.69	67.31	99.00	66.39	89.38	1.76	74.78
	PAUM	−0.17	91.62	0.33	1.38	0.42	–	−44.11	80.86	17.46	0.55	17.77	–
	PAPB	20.71	83.48	35.95	20.18	14.27	25.75	−0.34	91.35	25.49	8.03	7.49	12.10
	IDCSPA	23.49	81.43	43.00	19.53	17.48	26.86	−0.32	92.49	19.91	7.92	6.07	11.24
Scobit	Static	85.76	92.67	92.91	65.38	18.39	76.66	68.31	92.71	96.28	32.73	10.24	47.81
	PA	50.80	58.22	80.90	20.95	49.74	33.28	−94.51	51.21	66.64	4.22	49.84	7.94
	CSOGD	90.00	88.15	99.04	54.71	24.49	69.86	52.86	86.73	97.88	21.94	16.30	35.39
	PAUM	70.26	62.64	98.96	25.50	49.97	40.54	−61.87	53.15	98.25	6.24	49.91	11.72
	PAPB	61.34	89.19	66.58	56.97	15.09	61.35	47.78	95.22	57.96	35.16	5.30	43.47
	IDCSPA	65.69	88.10	72.28	52.24	17.54	60.64	46.95	95.35	56.80	35.62	5.08	43.53
Klein & Spady	Static	−1.47	49.33	51.32	11.43	50.96	18.39	−47.29	60.62	43.17	5.90	38.71	10.21
	PA	11.67	50.45	49.93	11.15	49.53	18.22	−38.73	51.55	49.57	5.40	48.39	9.73
	CSOGD	73.50	85.32	85.60	44.93	22.57	58.46	58.25	85.56	85.28	27.84	18.16	41.51
	PAUM	15.99	49.82	55.43	11.94	51.36	19.62	−39.25	50.61	51.17	5.48	49.54	9.89
	PAPB	4.12	74.76	18.77	11.33	18.33	14.05	−7.81	83.66	13.14	5.63	12.45	7.77
	IDCSPA	11.40	66.92	27.23	12.70	27.28	17.31	−1.70	87.43	10.76	6.72	8.43	8.27

Table A.7
Savings (Sav), accuracy (Acc), recall (Rec), positive predicted value (PPV), positive predicted proportion (PP) and F-score (FS) for the OL algorithms for the cost setting $a = 0.10$ in the simulated datasets.

Model	Classifier	Normal						Risky					
		Sav	Acc	Rec	PPV	PP	FS	Sav	Acc	Rec	PPV	PP	FS
Logit	Static	82.59	91.59	91.41	57.26	17.32	70.19	75.78	94.57	91.72	42.21	8.64	57.28
	PA	3.08	56.32	78.43	16.23	49.52	26.89	−201.20	50.76	67.35	4.50	50.39	8.43
	CSOGD	72.13	86.48	98.96	47.07	23.56	62.87	4.84	85.09	97.60	21.86	18.13	35.12
	PAUM	19.45	60.19	97.62	20.18	49.58	33.44	−204.16	52.49	94.33	6.30	50.50	11.80
	PAPB	50.13	90.00	62.59	51.00	12.62	56.11	27.37	94.31	53.34	30.82	5.92	38.78
	IDCSPA	53.35	89.67	65.80	50.27	13.66	56.92	29.90	94.77	51.45	33.49	5.32	40.43
Squared	Static	9.64	80.10	30.05	21.71	16.24	23.24	−28.40	81.77	34.35	9.56	17.11	14.26
	PA	−26.22	50.23	50.44	10.26	49.86	17.04	−174.26	49.69	50.14	3.89	50.33	7.22
	CSOGD	63.96	85.07	90.00	44.38	23.03	58.78	14.68	85.00	91.79	23.83	18.27	37.33
	PAUM	−36.17	50.28	51.15	10.38	49.96	17.25	−194.67	50.43	53.48	4.21	49.86	7.79
	PAPB	48.48	89.71	61.84	49.53	12.69	54.91	33.12	94.06	56.14	34.29	6.44	42.37
	IDCSPA	51.10	89.29	64.79	48.53	13.74	55.44	35.35	94.33	56.16	36.40	6.17	44.05
Qubic	Static	74.91	82.85	89.08	45.84	28.65	60.48	64.01	87.11	92.48	29.55	17.53	44.44
	PA	31.04	58.01	77.12	23.04	50.12	35.48	−69.26	52.35	69.67	8.10	49.93	14.51
	CSOGD	81.52	87.51	97.57	57.86	26.74	71.88	50.15	85.80	97.25	33.11	19.68	48.54
	PAUM	49.88	63.88	97.30	28.98	50.30	44.65	−49.41	55.95	97.51	11.42	49.57	20.44
	PAPB	31.16	80.73	46.54	38.27	18.23	41.95	24.60	90.44	46.12	29.63	9.10	35.88
	IDCSPA	33.62	79.15	52.24	36.38	21.53	42.88	25.15	90.40	46.24	29.15	9.17	35.69
Ker & Sam	Static	51.51	77.39	88.73	24.05	28.47	37.69	−15.34	87.35	46.71	11.17	12.45	17.15
	PA	1.07	87.45	15.03	16.93	7.12	–	−13.01	95.26	3.09	–	2.26	–
	CSOGD	50.86	96.06	52.33	94.09	4.30	65.72	63.02	98.81	63.82	89.39	1.94	70.12
	PAUM	−0.16	92.16	0.04	1.43	0.07	–	−113.28	82.71	15.34	0.49	15.39	–
	PAPB	10.02	84.20	36.93	20.93	13.75	26.60	−26.87	91.05	26.17	9.17	7.71	13.46
	IDCSPA	10.60	83.84	38.27	20.58	14.34	26.62	−20.54	92.91	17.92	8.33	5.47	11.36
Scobit	Static	86.04	92.75	92.50	66.02	17.95	76.65	78.40	96.53	92.60	47.58	6.11	62.42
	PA	22.12	58.09	81.15	20.61	49.79	32.87	−181.28	50.58	67.14	4.48	50.58	8.40
	CSOGD	78.17	87.04	99.08	52.82	25.38	68.08	12.56	85.01	97.54	21.76	18.20	35.01
	PAUM	37.06	62.64	98.66	25.12	49.68	40.03	−170.94	52.62	95.78	6.41	50.48	12.01
	PAPB	56.96	89.65	67.76	57.74	14.86	62.30	32.68	94.65	55.54	33.22	5.72	41.28
	IDCSPA	57.81	89.03	70.34	54.76	16.08	61.58	35.69	95.33	53.58	37.55	4.91	43.91
Klein & Spady	Static	−5.89	69.60	24.67	9.68	25.31	13.20	−60.74	57.44	44.28	5.20	42.07	9.03
	PA	−24.46	51.43	47.22	10.57	47.99	17.26	−106.75	50.51	46.95	4.96	49.17	8.97
	CSOGD	57.93	84.25	83.02	43.10	22.84	56.19	29.12	84.00	82.40	26.06	19.37	39.09
	PAUM	−32.25	48.83	52.28	10.85	51.66	17.97	−124.72	48.55	52.23	5.26	51.69	9.55
	PAPB	−8.89	75.48	19.80	11.76	18.06	14.67	−30.33	83.58	11.71	4.79	12.45	6.72
	IDCSPA	−8.33	75.20	20.68	12.11	18.47	15.11	−20.52	86.40	10.53	6.59	9.10	8.11

Table A.8
Savings (Sav), accuracy (Acc), recall (Rec), positive predicted value (PPV), positive predicted proportion (PP) and F-score (FS) for the OL algorithms for the cost setting $a = 0.20$ in the simulated datasets.

Model	Classifier	Normal						Risky					
		Sav	Acc	Rec	PPV	PP	FS	Sav	Acc	Rec	PPV	PP	FS
Logit	Static	81.48	90.78	93.37	53.77	18.11	67.90	67.02	92.29	93.07	34.87	10.96	49.80
	PA	−57.09	56.50	79.02	16.42	49.47	27.17	−393.00	51.67	66.37	4.94	49.55	9.19
	CSOGD	42.10	85.18	98.78	45.39	24.85	61.11	−86.76	83.74	97.44	22.82	19.77	36.17
	PAUM	−55.21	60.19	98.96	20.40	49.88	33.80	−441.41	52.71	93.45	6.84	50.52	12.74
	PAPB	36.03	89.38	60.65	48.72	12.84	53.95	10.94	94.25	55.83	33.72	6.20	41.75
	IDCSPA	37.19	89.79	59.90	50.22	12.27	54.56	19.41	94.84	53.71	36.44	5.43	43.29
Squared	Static	13.99	82.04	31.72	20.07	14.64	24.46	−31.40	83.01	26.35	6.92	15.29	10.56
	PA	−99.85	50.63	51.48	10.32	49.70	17.18	−394.73	49.88	54.04	3.79	50.41	7.08
	CSOGD	26.00	83.51	88.48	42.14	24.14	56.28	−90.93	83.28	90.50	21.25	19.58	33.82
	PAUM	−130.53	50.23	49.39	9.90	49.67	16.48	−492.41	49.09	48.82	3.39	50.81	6.34
	PAPB	32.61	89.51	60.97	47.89	12.69	53.55	−3.81	93.66	50.24	28.20	6.35	35.87
	IDCSPA	32.75	89.80	57.44	48.89	11.71	52.75	0.48	94.14	45.36	30.78	5.49	36.59
Qubic	Static	75.33	81.90	90.53	44.38	30.00	59.52	65.21	86.91	95.12	30.93	18.45	46.51
	PA	−8.59	57.98	77.71	23.02	50.28	35.51	−191.01	53.20	73.42	8.76	49.58	15.63
	CSOGD	61.68	86.15	97.82	55.72	28.09	70.04	−9.77	84.36	97.53	32.37	21.27	47.51
	PAUM	1.97	63.85	97.73	28.88	50.35	44.58	−203.64	55.91	98.19	11.66	49.79	20.83
	PAPB	17.07	80.96	47.49	38.73	18.29	42.62	−5.76	89.68	42.65	26.88	9.46	32.79
	IDCSPA	18.21	80.70	47.14	39.03	18.45	42.62	−4.17	90.06	39.10	27.20	8.68	32.04
Ker & Sam	Static	50.78	80.68	80.24	26.43	24.01	39.50	−20.45	85.81	51.14	9.89	14.25	16.26
	PA	−13.29	87.32	13.63	20.25	7.08	–	−26.63	95.40	3.42	–	2.23	–
	CSOGD	47.98	95.93	49.60	94.84	4.03	62.29	57.90	98.78	60.89	–	1.78	–
	PAUM	−0.66	92.16	0.16	1.73	0.14	–	−305.93	78.35	20.00	0.44	19.97	–
	PAPB	−23.87	83.88	35.44	19.80	13.86	25.28	−85.31	90.99	28.55	9.36	7.92	13.98
	IDCSPA	−20.31	85.08	32.40	20.46	12.21	25.02	−56.68	92.77	24.06	10.63	5.89	14.74
Scobit	Static	85.18	93.58	91.35	69.79	17.25	78.97	73.57	95.33	91.63	42.03	7.30	56.93
	PA	−24.53	57.82	80.60	20.32	49.89	32.44	−456.24	51.24	64.69	4.07	49.70	7.65
	CSOGD	54.19	85.60	99.06	50.71	26.74	66.07	−110.00	83.80	97.40	20.24	19.16	32.76
	PAUM	−21.29	61.69	98.63	24.55	50.55	39.31	−511.68	52.55	94.87	5.92	50.25	11.12
	PAPB	47.11	89.44	66.34	56.97	14.67	61.25	6.45	94.91	53.84	31.71	5.36	39.59
	IDCSPA	53.41	90.50	67.11	60.87	13.77	63.80	16.23	95.60	47.87	35.16	4.27	40.54
Klein & Spady	Static	2.59	61.62	38.57	12.17	36.05	17.56	−59.06	54.38	51.33	5.83	45.78	10.34
	PA	−90.00	50.44	49.32	10.83	49.41	17.75	−228.43	51.99	47.45	5.31	47.73	9.55
	CSOGD	23.84	82.24	82.40	40.95	24.79	54.01	−36.87	82.67	81.83	25.71	20.74	38.54
	PAUM	−116.74	48.39	52.85	10.97	52.21	18.15	−301.98	48.86	54.23	5.63	51.60	10.19
	PAPB	−40.58	75.11	18.22	10.89	18.00	13.56	−77.89	83.11	11.24	4.66	12.74	6.51
	IDCSPA	−39.83	75.05	16.28	10.92	17.15	13.00	−54.13	86.63	8.13	4.87	8.90	6.09

Table A.9
Average expected reward (AER), accuracy (Acc), recall (Rec), positive predicted value (PPV), positive predicted proportion (PP) and F-score (FS) for the bandit algorithms for the cost setting $a = 0.05$ in the simulated datasets.

Model	Classifier	Normal						Risky					
		AER	Acc	Rec	PPV	PP	FS	AER	Acc	Rec	PPV	PP	FS
Logit	Static	57.82	71.23	99.02	25.97	38.73	41.15	73.05	79.35	98.36	14.85	24.18	25.80
	LB	58.94	73.78	97.93	27.67	35.96	43.15	77.74	85.36	95.89	19.46	17.99	32.35
	LinUCB	−63.82	89.84	0.00	–	0.00	–	−0.61	5.86	94.52	3.54	97.39	6.83
	CEG	58.85	73.34	98.43	27.40	36.50	42.86	77.87	85.45	96.44	19.62	17.94	32.61
	CSLB	60.80	87.86	89.57	45.09	20.18	59.99	80.37	95.37	81.37	42.92	6.92	56.20
Squared	Static	−12.54	12.73	88.19	8.99	94.64	16.32	−7.95	4.56	81.90	2.92	97.66	5.64
	LB	−19.22	43.18	61.45	10.05	59.03	17.27	30.38	63.76	37.36	3.68	35.36	6.69
	LinUCB	−54.88	90.35	0.00	–	0.00	–	−2.63	5.57	92.53	3.31	97.39	6.38
	CEG	−16.15	42.60	63.63	10.23	60.03	17.62	33.55	64.38	39.08	3.90	34.86	7.09
	CSLB	5.80	19.95	96.99	10.50	89.12	18.95	47.66	92.29	16.95	10.91	5.41	13.27
Qubic	Static	48.86	66.30	99.25	30.34	48.22	46.47	71.86	80.85	98.27	22.94	24.72	37.19
	LB	48.97	68.49	98.37	31.68	45.77	47.93	69.37	78.49	97.05	20.79	26.94	34.24
	LinUCB	−129.75	85.26	0.00	–	0.00	–	−1.39	7.73	95.84	5.67	97.56	10.70
	CEG	49.80	68.77	98.58	31.90	45.55	48.20	68.07	78.46	96.01	20.63	26.85	33.97
	CSLB	50.41	75.72	95.32	37.33	37.64	53.65	73.00	89.43	86.66	33.78	14.80	48.61
Ker & Sam	Static	17.67	48.74	98.29	33.24	76.12	49.68	−0.57	11.56	75.48	2.19	89.77	4.26
	LB	13.56	51.46	96.62	34.28	72.54	50.61	28.90	44.08	66.67	3.06	56.79	5.86
	LinUCB	−305.73	74.26	0.00	–	0.00	–	1.03	4.73	96.93	2.59	97.72	5.04
	CEG	12.51	51.15	96.23	34.09	72.65	50.35	29.93	44.15	72.03	3.30	57.00	6.31
	CSLB	6.80	50.74	95.07	33.77	72.46	49.84	57.19	96.35	2.68	5.93	1.18	3.69

(*continued on next page*)

Table A.9 (*continued*).

Model	Classifier	Normal						Risky					
		AER	Acc	Rec	PPV	PP	FS	AER	Acc	Rec	PPV	PP	FS
Scobit	Static	60.08	76.37	99.06	35.06	36.20	51.79	78.65	84.45	98.73	16.66	18.61	28.51
	LB	61.23	78.92	98.13	37.62	33.41	54.39	81.72	89.56	94.90	22.47	13.26	36.34
	LinUCB	−103.19	87.19	0.00	–	0.00	–	−0.36	5.39	94.27	3.04	97.39	5.89
	CEG	60.20	78.56	98.28	37.24	33.81	54.01	82.69	89.67	96.50	22.87	13.25	36.97
	CSLB	64.68	89.47	94.46	55.20	21.92	69.68	85.32	94.77	92.36	36.76	7.89	52.58
Klein & Spady	Static	−3.56	22.14	87.64	10.22	85.35	18.30	10.77	21.53	88.47	4.60	81.74	8.75
	LB	−13.83	41.93	67.34	10.89	61.52	18.75	16.28	40.33	68.00	4.72	61.20	8.83
	LinUCB	−58.43	90.05	0.00	–	0.00	–	−0.15	6.71	97.18	4.24	97.30	8.13
	CEG	−13.81	41.82	66.43	10.76	61.45	18.52	17.22	40.55	69.65	4.84	61.12	9.06
	CSLB	−1.52	12.07	97.49	9.96	97.38	18.08	31.97	90.66	6.12	4.63	5.61	5.27

Table A.10
Average expected reward (AER), accuracy (Acc), recall (Rec), positive predicted value (PPV), positive predicted proportion (PP) and F-score (FS) for the bandit algorithms for the cost setting $a = 0.10$ in the simulated datasets.

Model	Classifier	Normal						Risky					
		AER	Acc	Rec	PPV	PP	FS	AER	Acc	Rec	PPV	PP	FS
Logit	Static	117.55	71.23	99.02	25.97	38.73	41.15	147.27	79.35	98.36	14.85	24.18	25.80
	LB	121.22	73.78	97.93	27.67	35.96	43.15	157.93	85.36	95.89	19.46	17.99	32.35
	LinUCB	−1.18	12.06	96.46	10.06	97.38	18.23	1.75	5.86	94.52	3.54	97.39	6.83
	CEG	120.78	73.34	98.43	27.40	36.50	42.86	157.91	85.45	96.44	19.62	17.94	32.61
	CSLB	124.96	92.01	74.41	58.38	12.95	65.43	170.03	95.80	80.82	45.74	6.45	58.42
Squared	Static	−16.24	10.76	86.22	8.65	96.23	15.72	−6.34	4.56	81.90	2.92	97.66	5.64
	LB	17.43	43.18	61.45	10.05	59.03	17.27	92.07	63.76	37.36	3.68	35.36	6.69
	LinUCB	−5.23	11.24	94.72	9.38	97.39	17.08	−0.40	5.57	92.53	3.31	97.39	6.38
	CEG	19.50	42.58	63.52	10.21	60.03	17.59	95.94	64.38	39.08	3.90	34.86	7.09
	CSLB	39.58	78.20	22.80	13.29	16.55	16.79	139.42	95.85	3.16	12.36	0.89	5.03
Qubic	Static	99.08	66.30	99.25	30.34	48.22	46.47	145.41	80.85	98.27	22.94	24.72	37.19
	LB	101.54	68.49	98.37	31.68	45.77	47.93	140.71	78.49	97.05	20.79	26.94	34.24
	LinUCB	−3.19	16.27	96.95	14.65	97.57	25.45	0.77	7.73	95.84	5.67	97.56	10.70
	CEG	102.64	68.77	98.58	31.90	45.55	48.20	139.33	78.46	96.01	20.63	26.85	33.97
	CSLB	112.41	79.71	92.40	41.54	32.79	57.31	153.93	90.85	77.99	36.35	12.38	49.59
Ker & Sam	Static	41.18	48.74	98.29	33.24	76.12	49.68	8.85	11.56	75.48	2.19	89.77	4.26
	LB	39.96	51.46	96.62	34.28	72.54	50.61	70.39	44.08	66.67	3.06	56.79	5.86
	LinUCB	−2.57	27.07	98.17	25.86	97.73	40.93	3.14	4.73	96.93	2.59	97.72	5.04
	CEG	38.84	51.15	96.23	34.09	72.65	50.35	71.46	44.15	72.03	3.30	57.00	6.31
	CSLB	37.19	60.03	90.75	38.33	60.95	53.89	152.71	96.93	1.92	8.93	0.56	3.15
Scobit	Static	122.28	76.37	99.06	35.06	36.20	51.79	158.33	84.45	98.73	16.66	18.61	28.51
	LB	126.03	78.92	98.13	37.62	33.41	54.39	166.88	89.56	94.90	22.47	13.26	36.34
	LinUCB	−2.86	14.50	96.33	12.67	97.37	22.40	2.01	5.39	94.27	3.04	97.39	5.89
	CEG	124.60	78.56	98.28	37.24	33.81	54.01	167.78	89.67	96.50	22.87	13.25	36.97
	CSLB	131.42	91.18	86.34	61.00	18.13	71.49	176.07	95.32	90.76	39.36	7.24	54.91
Klein & Spady	Static	9.40	22.14	87.64	10.22	85.35	18.30	27.69	21.53	88.47	4.60	81.74	8.75
	LB	20.23	41.93	67.34	10.89	61.52	18.75	52.37	40.33	68.00	4.72	61.20	8.83
	LinUCB	−0.78	12.11	97.19	9.94	97.28	18.04	2.27	6.71	97.18	4.24	97.30	8.13
	CEG	20.34	41.82	66.43	10.76	61.45	18.52	53.44	40.55	69.65	4.84	61.12	9.06
	CSLB	30.48	85.61	6.03	10.64	5.64	7.70	126.11	95.05	0.47	2.70	0.74	0.80

Table A.11
Average expected reward (AER), accuracy (Acc), recall (Rec), positive predicted value (PPV), positive predicted proportion (PP) and F-score (FS) for the bandit algorithms for the cost setting $a = 0.20$ in the simulated datasets.

Model	Classifier	Normal						Risky					
		AER	Acc	Rec	PPV	PP	FS	AER	Acc	Rec	PPV	PP	FS
Logit	Static	296.74	71.23	99.02	25.97	38.73	41.15	369.93	79.35	98.36	14.85	24.18	25.80
	LB	308.08	73.78	97.93	27.67	35.96	43.15	398.50	85.36	95.89	19.46	17.99	32.35
	LinUCB	5.41	12.06	96.46	10.06	97.38	18.23	8.84	5.86	94.52	3.54	97.39	6.83
	CEG	306.55	73.34	98.43	27.40	36.50	42.86	398.03	85.45	96.44	19.62	17.94	32.61
	CSLB	376.03	93.20	72.15	64.87	11.30	68.31	444.48	96.77	68.77	54.57	4.60	60.85
Squared	Static	−9.34	10.76	86.22	8.65	96.23	15.72	−1.50	4.56	81.90	2.92	97.66	5.64
	LB	127.38	43.18	61.45	10.05	59.03	17.27	277.15	63.76	37.36	3.68	35.36	6.69
	LinUCB	0.63	11.24	94.72	9.38	97.39	17.08	6.32	5.57	92.53	3.31	97.39	6.38
	CEG	126.64	42.58	63.52	10.21	60.03	17.59	283.09	64.38	39.08	3.90	34.86	7.09
	CSLB	296.66	89.18	1.76	11.26	1.51	3.05	423.09	96.16	2.01	14.00	0.50	3.52

(*continued on next page*)

Table A.11 (*continued*).

Model	Classifier	Normal						Risky					
		AER	Acc	Rec	PPV	PP	FS	AER	Acc	Rec	PPV	PP	FS
Qubic	Static	249.77	66.30	99.25	30.34	48.22	46.47	366.08	80.85	98.27	22.94	24.72	37.19
	LB	259.24	68.49	98.37	31.68	45.77	47.93	354.71	78.49	97.05	20.79	26.94	34.24
	LinUCB	2.52	16.27	96.95	14.65	97.57	25.45	7.26	7.73	95.84	5.67	97.56	10.70
	CEG	261.16	68.77	98.58	31.90	45.55	48.20	353.11	78.46	96.01	20.63	26.85	33.97
	CSLB	315.19	83.03	86.97	46.00	27.87	60.17	415.04	93.49	61.70	45.29	7.86	52.24
Ker & Sam	Static	221.66	54.23	96.67	15.18	53.63	26.24	37.11	11.56	75.48	2.19	89.77	4.26
	LB	245.58	59.03	92.99	16.24	48.21	27.65	194.85	44.08	66.67	3.06	56.79	5.86
	LinUCB	7.66	10.33	97.86	8.43	97.73	15.53	9.49	4.73	96.93	2.59	97.72	5.04
	CEG	244.16	59.27	93.23	16.35	48.01	27.82	196.06	44.15	72.03	3.30	57.00	6.31
	CSLB	323.20	91.37	0.48	13.79	0.29	0.92	439.25	97.10	0.77	6.06	0.33	1.36
Scobit	Static	308.88	76.37	99.06	35.06	36.20	51.79	397.39	84.45	98.73	16.66	18.61	28.51
	LB	320.44	78.92	98.13	37.62	33.41	54.39	422.34	89.56	94.90	22.47	13.26	36.34
	LinUCB	3.45	14.50	96.33	12.67	97.37	22.40	9.15	5.39	94.27	3.04	97.39	5.89
	CEG	317.79	78.56	98.28	37.24	33.81	54.01	423.06	89.67	96.50	22.87	13.25	36.97
	CSLB	346.91	93.34	62.14	81.47	9.77	70.50	450.84	97.22	66.56	54.71	3.82	60.06
Klein & Spady	Static	48.31	22.14	87.64	10.22	85.35	18.30	78.45	21.53	88.47	4.60	81.74	8.75
	LB	122.39	41.93	67.34	10.89	61.52	18.75	160.65	40.33	68.00	4.72	61.20	8.83
	LinUCB	5.90	12.11	97.19	9.94	97.28	18.04	9.54	6.71	97.18	4.24	97.30	8.13
	CEG	122.80	41.82	66.43	10.76	61.45	18.52	162.09	40.55	69.65	4.84	61.12	9.06
	CSLB	293.99	89.88	0.30	13.04	0.23	0.59	407.19	95.52	0.24	4.00	0.25	0.44

Table A.12
Average expected reward (AER), accuracy (Acc), recall (Rec), positive predicted value (PPV), positive predicted proportion (PP) and F-score (FS) for the bandit algorithms for the cost setting $a = 0.05$ in the real datasets.

Data set	Classifier	AER	Acc	Rec	PPV	PP	FS
Credit card	Static	0.18	10.77	97.42	3.37	92.25	6.51
	LB	0.80	52.82	89.25	5.72	49.68	10.76
	LinUCB	−0.33	5.15	64.73	2.15	95.79	4.17
	CEG	0.79	53.23	88.82	5.75	49.24	10.80
	CSLB	0.83	52.51	86.67	5.54	49.83	10.42
GCD	Static	−14.34	45.20	85.67	33.73	76.20	48.40
	LB	−54.57	55.50	64.33	36.35	53.10	46.45
	LinUCB	−46.57	43.90	67.67	30.43	66.70	41.99
	CEG	−43.84	59.00	70.00	39.62	53.00	50.60
	CSLB	−14.50	41.40	92.67	33.02	84.20	48.69
PAKDD	Static	−3.87	32.22	91.81	27.60	90.54	42.44
	LB	−3.12	30.09	93.97	27.26	93.84	42.26
	LinUCB	−15.85	41.39	66.95	26.86	67.83	38.34
	CEG	−3.46	29.66	93.50	27.07	94.03	41.98
	CSLB	−0.18	27.46	99.54	27.23	99.50	42.76
ASF	Static	9.87	15.37	90.21	3.17	87.09	6.12
	LB	41.24	50.80	77.34	4.65	50.88	8.77
	LinUCB	4.23	9.90	88.39	2.92	92.45	5.66
	CEG	41.56	51.16	77.90	4.71	50.55	8.89
	CSLB	65.03	96.91	0.00	0.00	0.03	–
Drift	Static	10.57	16.57	94.08	3.94	86.63	7.57
	LB	41.46	41.78	91.60	5.43	61.24	10.25
	LinUCB	−4.87	12.87	71.90	2.94	88.72	5.65
	CEG	41.50	41.51	91.51	5.40	61.50	10.20
	CSLB	54.51	96.21	0.09	1.96	0.17	0.18

Table A.13
Average expected reward (AER), accuracy (Acc), recall (Rec), positive predicted value (PPV), positive predicted proportion (PP) and F-score (FS) for the bandit algorithms for the cost setting $a = 0.10$ in the real datasets.

Data set	Classifier	AER	Acc	Rec	PPV	PP	FS
Credit card	Static	0.07	4.57	98.28	3.18	98.51	6.16
	LB	3.91	59.72	86.02	6.44	42.57	11.98
	LinUCB	0.05	5.23	62.37	2.08	95.56	4.03
	CEG	3.99	59.39	86.88	6.45	42.96	12.00
	CSLB	4.36	51.41	91.40	5.69	51.23	10.71
GCD	Static	−26.01	42.10	78.00	31.33	74.70	44.70
	LB	−4.82	57.30	79.33	39.47	60.30	52.71
	LinUCB	−49.88	42.00	62.67	28.66	65.60	39.33
	CEG	−11.96	57.90	76.00	39.51	57.70	52.00
	CSLB	5.97	41.90	96.33	33.64	85.90	49.87
PAKDD	Static	−2.79	32.22	91.81	27.60	90.54	42.44
	LB	−2.47	30.09	93.97	27.26	93.84	42.26
	LinUCB	−12.44	41.39	66.95	26.86	67.83	38.34
	CEG	−2.83	29.66	93.50	27.07	94.03	41.98
	CSLB	−1.74	30.44	94.43	27.42	93.74	42.50
ASF	Static	26.26	15.37	90.21	3.17	87.09	6.12
	LB	94.80	50.80	77.34	4.65	50.88	8.77
	LinUCB	15.82	9.90	88.39	2.92	92.45	5.66
	CEG	95.37	51.16	77.90	4.71	50.55	8.89
	CSLB	191.73	96.91	0.00	0.00	0.03	–
Drift	Static	25.87	16.57	94.08	3.94	86.63	7.57
	LB	88.59	41.78	91.60	5.43	61.24	10.25
	LinUCB	10.79	12.87	71.90	2.94	88.72	5.65
	CEG	88.27	41.51	91.51	5.40	61.50	10.20
	CSLB	178.46	95.24	0.55	1.71	1.17	0.83

Table A.14
Average expected reward (AER), accuracy (Acc), recall (Rec), positive predicted value (PPV), positive predicted proportion (PP) and F-score (FS) for the bandit algorithms for the cost setting $a = 0.20$ in the real datasets.

Data set	Classifier	AER	Acc	Rec	PPV	PP	FS
Credit card	Static	0.13	4.57	98.28	3.18	98.51	6.16
	LB	8.64	59.72	86.02	6.44	42.57	11.98
	LinUCB	1.07	5.23	62.37	2.08	95.56	4.03
	CEG	8.69	59.39	86.88	6.45	42.96	12.00
	CSLB	9.71	53.41	89.25	5.79	49.09	10.88
GCD	Static	−4.39	42.10	78.00	31.33	74.70	44.70
	LB	28.48	57.30	79.33	39.47	60.30	52.71
	LinUCB	−20.90	42.00	62.67	28.66	65.60	39.33
	CEG	23.43	57.90	76.00	39.51	57.70	52.00
	CSLB	34.59	57.80	82.00	40.07	61.40	53.83
PAKDD	Static	−0.64	32.22	91.81	27.60	90.54	42.44
	LB	−1.16	30.09	93.97	27.26	93.84	42.26
	LinUCB	−5.63	41.39	66.95	26.86	67.83	38.34
	CEG	−1.58	29.66	93.50	27.07	94.03	41.98
	CSLB	−0.13	28.24	98.28	27.28	98.05	42.71
ASF	Static	59.04	15.37	90.21	3.17	87.09	6.12
	LB	201.92	50.80	77.34	4.65	50.88	8.77
	LinUCB	39.01	9.90	88.39	2.92	92.45	5.66
	CEG	202.99	51.16	77.90	4.71	50.55	8.89
	CSLB	445.12	96.91	0.00	0.00	0.03	–
Drift	Static	56.48	16.57	94.08	3.94	86.63	7.57
	LB	182.85	41.78	91.60	5.43	61.24	10.25
	LinUCB	42.10	12.87	71.90	2.94	88.72	5.65
	CEG	181.82	41.51	91.51	5.40	61.50	10.20
	CSLB	428.17	95.24	0.55	1.71	1.17	0.83

Data availability

The data that has been used is confidential.

References

Achab, M., Clémençon, S., & Garivier, A. (2018). Profitable bandits. In *Proceedings of the 10th Asian conference on machine learning vol. 95*, 694–709, URL: https://proceedings.mlr.press/v95/achab18a.html.

Auer, P., Cesa-Bianchi, N., & Fischer, P. (2002). Finite-time analysis of the multiarmed bandit problem. *Machine Learning*, *47*, 235–256, URL: https://api.semanticscholar.org/CorpusID:207609497.

Bahnsen, A. C., Aouada, D., & Ottersten, B. (2014). Example-dependent cost-sensitive logistic regression for credit scoring. *vol. 1*, In *Proceedings - 2014 13th international conference on machine learning and applications* (pp. 263–269). http://dx.doi.org/10.1109/ICMLA.2014.48.

Bahnsen, A. C., Stojanovic, A., Aouada, D., & Ottersten, B. (2013). Cost sensitive credit card fraud detection using Bayes minimum risk. *vol. 1*, In *Proceedings - 2013 12th international conference on machine learning and applications* (pp. 333–338). http://dx.doi.org/10.1109/ICMLA.2013.68.

C-Rella, J., Cao, R., & Vilar, J. M. (2024). Cost-sensitive thresholding over a two-dimensional decision region for fraud detection. *Information Sciences*, *657*, Article 119956. http://dx.doi.org/10.1016/j.ins.2023.119956.

C-Rella, J., Claeskens, G., Cao, R., & Vilar, J. M. (2025). Instance-dependent cost-sensitive parametric learning. *Neurocomputing*, *615*, Article 128875. http://dx.doi.org/10.1016/j.neucom.2024.128875.

C-Rella, J., & Vilar, J. M. (2024). Flexible multi-class cost-sensitive thresholding. URL: https://papers.ssrn.com/sol3/papers.cfm?abstract_id=4897024.

Chu, W., Li, L., Reyzin, L., & Schapire, R. (2011). Contextual bandits with linear payoff functions. *vol. 15*, In *Proceedings of the fourteenth international conference on artificial intelligence and statistics* (pp. 208–214). Fort Lauderdale, FL, USA: URL: https://proceedings.mlr.press/v15/chu11a.html.

Crammer, K., Dekel, O., Keshet, J., Shalev-Shwartz, S., & Singer, Y. (2006). Online passive-aggressive algorithms. *Journal of Machine Learning Research*, *7*(19), 551–585, URL: http://jmlr.org/papers/v7/crammer06a.html.

Dredze, M., Crammer, K., & Pereira, F. (2008). Confidence-weighted linear classification. In *Proceedings of the 25th international conference on machine learning* (pp. 264–271). http://dx.doi.org/10.1145/1390156.1390190.

Dumitrascu, B., Feng, K., & Engelhardt, B. E. (2018). PG-ts: Improved thompson sampling for logistic contextual bandits. In *Proceedings of the 32nd international conference on neural information processing systems* (pp. 4629–4638). URL: https://proceedings.neurips.cc/paper/2018/file/ce6c92303f38d297e263c7180f03d402-Paper.pdf.

Elkan, C. (2001). The foundations of cost-sensitive learning. *vol. 2001*, In *International joint conference on artificial intelligence* (pp. 973–978). Lawrence Erlbaum Associates Ltd., URL: https://api.semanticscholar.org/CorpusID:16149383.

Faury, L., Abeille, M., Calauzenes, C., & Fercoq, O. (2020). Improved optimistic algorithms for logistic bandits. *vol. 119*, In *Proceedings of the 37th international conference on machine learning* (pp. 3052–3060). PMLR, URL: https://proceedings.mlr.press/v119/faury20a.html.

Gamerman, D., & Lopes, H. (2006). *Markov Chain Monte Carlo: Stochastic simulation for bayesian inference, second edition* (2nd ed.). Chapman and Hall/CRC, http://dx.doi.org/10.1201/9781482296426.

Hazan, E., Rakhlin, A., & Bartlett, P. (2007). Adaptive online gradient descent. *Advances in Neural Information Processing Systems*, *20*, URL: https://proceedings.neurips.cc/paper_files/paper/2007/file/afd4836712c5e77550897e25711e1d96-Paper.pdf.

Hoi, S. C. H., Sahoo, D., Lu, J., & Zhao, P. (2018). Online learning: A comprehensive survey. ArXiv abs/1802.02871. arXiv:1802.02871.

Höppner, S., Baesens, B., Verbeke, W., & Verdonck, T. (2021). Instance-dependent cost-sensitive learning for detecting transfer fraud. *European Journal of Operational Research*, *297*, 291–300. http://dx.doi.org/10.1016/j.ejor.2021.05.028.

Kaufmann, E., Cappé, O., & Garivier, A. (2014). On the complexity of best-arm identification in multi-armed bandit models. *Journal of Machine Learning Research*, *17*, 1–42, URL: https://jmlr.org/papers/volume17/kaufman16a/kaufman16a.pdf.

Kaufmann, E., Korda, N., & Munos, R. (2012). Thompson sampling: An asymptotically optimal finite time analysis. http://dx.doi.org/10.1007/978-3-642-34106-9_18, ArXiv1205.4217.

Ker, A. P., & Sam, A. G. (2018). Semiparametric estimation of the link function in binary-choice single-index models. *Computational Statistics*, *33*, 1429–1455. http://dx.doi.org/10.1007/s00180-017-0779-2.

Kivinen, J., Smola, A., & Williamson, R. (2004). Online learning with kernels. *IEEE Transactions on Signal Processing*, *52*(8), 2165–2176. http://dx.doi.org/10.1109/TSP.2004.830991.

Klein, R., & Spady, R. (1993). An efficient semiparametric estimator of the binary response models. *Econometrica*, *61*, 387–421. http://dx.doi.org/10.2307/2951556.

Korda, N., Kaufmann, E., & Munos, R. (2013). Thompson sampling for 1-dimensional exponential family bandits. *Advances in Neural Information Processing Systems*, *26*, URL: https://proceedings.neurips.cc/paper_files/paper/2013/file/aba3b6fd5d186d28e06ff97135cade7f-Paper.pdf.

Lai, T., & Robbins, H. (1985). Asymptotically efficient adaptive allocation rules. *Advanced in Applied Mathematics*, *6*, 4–22. http://dx.doi.org/10.1145/2505515.2514700.

Langford, J., & Zhang, T. (2007). The epoch-greedy algorithm for multi-armed bandits with side information. In J. Platt, D. Koller, Y. Singer, & S. Roweis (Eds.), *Advances in Neural Information Processing Systems*, *20*, 661–670, URL: https://proceedings.neurips.cc/paper_files/paper/2007/file/4b04a686b0ad13dce35fa99fa4161c65-Paper.pdf.

Lattimore, T., & Szepesvári, C. (2020). *Bandit algorithms*. Cambridge University Press, http://dx.doi.org/10.1017/9781108571401.

Lessmann, S., Baesens, B., Seow, H.-V., & Thomas, L. (2015). Benchmarking state-of-the-art classification algorithms for credit scoring: An update of research. *European Journal of Operational Research*, *247*, 124–136. http://dx.doi.org/10.1016/j.ejor.2015.05.030.

Li, L., Chu, W., Langford, J., & Schapire, R. (2010). A contextual-bandit approach to personalized news article recommendation. In *WWW '10: Proceedings of the 19th international conference on world wide web* (pp. 661–670). http://dx.doi.org/10.1145/1772690.1772758.

Li, Y., Zaragoza, H., Herbrich, R., Shawe-Taylor, J., & Kandola, J. (2002). The perceptron algorithm with uneven margins. *International Conference on Machine Learning*, URL: https://api.semanticscholar.org/CorpusID:14475371.

Linhart, C., Harari, G., Abramovich, S., & Buchris, A. (2009). PAKDD data mining competition 2009: New ways of using known methods. In *Proceedings of the 13th pacific-asia international conference on knowledge discovery and data mining: New frontiers in applied data mining PAKDD'09* (pp. 99–105). Springer-Verlag, http://dx.doi.org/10.5555/1883700.1883707.

Lu, F.-Q., Huang, M., Ching, W.-K., & Siu, T. K. (2013). Credit portfolio management using two-level particle swarm optimization. *Information Sciences*, *237*, 162–175. http://dx.doi.org/10.1016/j.ins.2013.03.005.

Nowok, B., Raab, G. M., & Dibben, C. (2016). Synthpop: Bespoke creation of synthetic data in R. *Journal of Statistical Software*, *74*, 1–26. http://dx.doi.org/10.18637/jss.v074.i11.

Pandey, S., Agarwal, D., Chakrabarti, D., & Josifovski, V. (2007). Bandits for taxonomies: A model-based approach. In *Proceedings of the 2007 SIAM International Conference on Data Mining (SDM)* (pp. 216–227). http://dx.doi.org/10.1137/1.9781611972771.20.

R Core Team (2021). *R: A language and environment for statistical computing*. Vienna, Austria: R Foundation for Statistical Computing, URL: https://www.R-project.org/.

Rabbi, M., Aung, M. H., Zhang, M., & Choudhury, T. (2015). Mybehavior: Automatic personalized health feedback from user behaviors and preferences using smartphones. In *Proceedings of the 2015 ACM international joint conference on pervasive and ubiquitous computing* (pp. 707–718). New York, NY, USA: Association for Computing Machinery, http://dx.doi.org/10.1145/2750858.2805840.

Rosenblatt, F. (1958). The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, *65*(6), 368–408. http://dx.doi.org/10.1037/h0042519.

Russo, D., Roy, B. V., Kazerouni, A., Osband, I., & Wen, Z. (2020). A tutorial on thompson sampling. arXiv:1707.02038.

Shalev-Shwartz, S. (2012). Online learning and online convex optimization. *Foundations and Trends in Machine Learning*, *4*, 107–194. http://dx.doi.org/10.1561/2200000018, URL: https://api.semanticscholar.org/CorpusID:51730029.

Shen, W., Wang, J., Jiang, Y.-G., & Zha, H. (2015). Portfolio choices with orthogonal bandit learning. In *Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence*. URL: https://www.ijcai.org/Proceedings/15/Papers/142.pdf.

Stripling, E., Broucke, S. v., Antonio, K., Baesens, B., & Snoeck, M. (2018). Profit maximizing logistic model for customer churn prediction using genetic algorithms. *Swarm and Evolutionary Computation*, *40*, 116–130. http://dx.doi.org/10.1016/j.swevo.2017.10.010.

Sutton, R. S., & Barto, A. G. (2018). *Reinforcement learning: An introduction*. Cambridge, MA, USA: A Bradford Book, ISBN : 978-0-262-19398-6.

Tang, L., Rosales, R., Singh, A., & Agarwal, D. (2013). Automatic ad format selection via contextual bandits. In *Proceedings of the 22nd ACM international conference on information and knowledge management (CIKM '13)* (pp. 1587–1594). http://dx.doi.org/10.1016/0196-8858(85)90002-8.

Thompson, W. R. (1933). On the likelihood that one unknown probability exceeds another in view of the evidence of two samples. *Biometrika*, *25*(3/4), 285–294. http://dx.doi.org/10.2307/2332286.

Vanderschueren, T., Verdonck, T., Baesens, B., & Verbeke, W. (2022). Predict-then-optimize or predict-and-optimize? An empirical evaluation of cost-sensitive learning strategies. *Information Sciences*, *594*, 400–415. http://dx.doi.org/10.1016/j.ins.2022.02.021.

Vapnik, V. N. (1998). *Statistical learning theory*. Wiley, ISBN: 978-0-471-03003-4.

Visantavarakul, K. (2022). *An application of reinforcement learning to credit scoring based on the logistic Bandit framework*. Chulalongkorn University Theses and Dissertations (Chula ETD 6050), URL: https://digital.car.chula.ac.th/chulaetd/6050.

Wang, C.-C., Kulkarni, S., & Poor, H. (2005). Bandit problems with side observations. *IEEE Transactions on Automatic Control*, *50*(3), 338–355. http://dx.doi.org/10.1109/TAC.2005.844079.

Wang, J., Zhao, P., & Hoi, S. C. (2012). Cost-sensitive online classification. In *2012 IEEE 12th International conference on data mining* (pp. 1140–1145). http://dx.doi.org/10.1109/ICDM.2012.116.

Zadrozny, B., & Elkan, C. (2001). Learning and making decisions when costs and probabilities are both unknown. *vol. 2001*, In *Proceedings of the seventh ACM SIGKDD international conference on knowledge discovery and data mining* (pp. 204–213). http://dx.doi.org/10.1145/502512.502540.

Zhang, Y., Wu, M., Hu, X., & Zhu, Y. (2018). An online transfer learning algorithm with adaptive cost. In *Proceedings of the 2018 international conference on signal processing and machine learning*. URL: https://api.semanticscholar.org/CorpusID:69168504.

Zhang, L., Yang, T., Jin, R., Xiao, Y., & Zhou, Z. (2016). Online stochastic linear optimization under one-bit feedback. *vol. 48*, In *Proceedings of the 33rd international conference on machine learning* (pp. 392–401). New York, New York, USA: PMLR, URL: https://proceedings.mlr.press/v48/zhangb16.html.

Zhao, P., Zhuang, F., Wu, M., Li, X., & Hoi, S. C. H. (2015). Cost-sensitive online classification with adaptive regularization and its applications. In *IEEE international conference on data mining ICDM 2015* (pp. 649–658). URL: https://ink.library.smu.edu.sg/sis_research/2923.

Zhou, L. (2016). A survey on contextual multi-armed bandits. arXiv:1508.03326.