Universiteit Antwerpen

This item is the archived peer-reviewed author-version of:

Robust instance-dependent cost-sensitive classification

Reference:
De Vos Simon, Vanderschueren Toon, Verdonck Tim, Verbeke Wouter.- Robust instance-dependent cost-sensitive classification
Advances in data analysis and classification - ISSN 1862-5355 - Heidelberg, Springer heidelberg, (2023), p. 1-23
Full text (Publisher's DOI): https://doi.org/10.1007/S11634-022-00533-3
To cite this reference: https://hdl.handle.net/10067/1933900151162165141

uantwerpen.be
Institutional repository IRUA

Robust Instance-Dependent Cost-Sensitive Classification

Simon De Vos[1], Toon Vanderschueren[1], Tim Verdonck[2,3] and Wouter Verbeke[1]

[1*]Faculty of Economics and Business, Leuven. AI, KU Leuven, Naamsestraat 69, Leuven, 3000, Belgium.
[2]Department of Mathematics, University of Antwerp, Middelheimlaan 1, Antwerp, 2020, Belgium.
[3]Department of Mathematics, KU Leuven, Celestijnenlaan 200B, Leuven, 3001, Belgium.

Contributing authors: simon.devos@kuleuven.be; toon.vanderschueren@kuleuven.be; tim.verdonck@uantwerpen.be; wouter.verbeke@kuleuven.be;

Abstract

Instance-dependent cost-sensitive (IDCS) learning methods have proven useful for binary classification tasks where individual instances are associated with variable misclassification costs. However, we demonstrate in this paper by means of a series of experiments that IDCS methods are sensitive to noise and outliers in relation to instance-dependent misclassification costs and their performance strongly depends on the cost distribution of the data sample. Therefore, we propose a generic three-step framework to make IDCS methods more robust: (i) detect outliers automatically, (ii) correct outlying cost information in a data-driven way, and (iii) construct an IDCS learning method using the adjusted cost information. We apply this framework to cslogit, a logistic regression-based IDCS method, to obtain its robust version, which we name r-cslogit. The robustness of this approach is introduced in steps (i) and (ii), where we make use of robust estimators to detect and impute outlying costs of individual instances. The newly proposed r-cslogit method is tested on synthetic and semi-synthetic data and proven to be superior in terms of savings compared to its non-robust counterpart for variable levels of noise and outliers. All our code is made available online at https://github.com/SimonDeVos/Robust-IDCS.

Keywords: Cost-sensitive learning, Instance-dependent costs, Classification, Outliers, Regression diagnostics, Logistic regression

Statements and declarations

The authors have no competing interests to declare that are relevant to the content of this article. No funds, grants, or other support was received.

1 Introduction

Classification is a well-studied machine learning task that involves the assignment of instances to a predefined set of outcome classes. Cost-sensitive classification methods take into account asymmetric costs related to incorrectly classifying instances across various classes (Elkan, 2001; Verbeke, Olaya, Berrevoets, Verboven, & Maldonado, 2020). Such misclassification costs may either be class-dependent, i.e., equal for all instances of a class, or instance-dependent, i.e., vary across instances.

Classification methods are adopted to support or automate business decision-making, e.g., for credit scoring (Petrides, Moldovan, Coenen, Guns, & Verbeke, 2022) or customer churn prediction (Lessmann, Haupt, Coussement, & De Bock, 2021). Note that in both applications, misclassified instances involve variable costs. For instance, the cost of a misclassified churner equals the future customer lifetime value, whereas a misclassified non-churner typically involves a much smaller cost, i.e., the cost of targeting the customer with the retention campaign. Either or both may be instance-dependent or class-dependent depending on the characteristics of the particular application setting.

A broad variety of cost-sensitive (CS) and instance-dependent cost-sensitive (IDCS) classification methods have been proposed in the literature as reviewed and experimentally evaluated by Petrides and Verbeke (2022) and Vanderschueren, Verdonck, Baesens, and Verbeke (2022). A prominent approach that is adopted by both CS and IDCS methods for taking misclassification costs into account is to weigh instances proportionally with the misclassification cost involved when learning a classification model.

In this article, we raise the question of whether IDCS classification methods are sensitive to outliers and noise in the data. No prior work seems to have addressed this question, which nonetheless is of significant practical importance given the broad adoption and potential monetary impact of using biased classification models for decision-making.

To address these shortcomings, we present the results of a series of experiments to evaluate the robustness of IDCS classification methods with respect to outlying costs in the data, which highlight the potential bias and vulnerability of IDCS classification methods. We propose a robust approach to IDCS classification by extending the existing cslogit approach (Höppner, Baesens, Verbeke, & Verdonck, 2022). An important benefit is the automatic and reliable detection of outliers in the data. These outliers may not only spoil the resulting analysis (as illustrated in this article) but can also contain valuable information. A robust analysis can thus provide better insight into the structure of the data.

The following section outlines related work on IDCS learning and discusses both cslogit and robustness. Next, in Section 3, a series of simulations on synthetic data is presented that motivate the need for robust IDCS learning which we develop in Section 4. Section 5 presents the results of a series of experiments that illustrate the excellent performance of the proposed robust IDCS learning method, denoted r-cslogit, in comparison with both logit and cslogit. We conclude and present directions for future research in Section 6.

2 Related work

Elkan (2001) introduces a learning paradigm where different misclassification errors incur different penalties depending on the predicted and actual class, with applications to, for example, detecting transaction fraud and credit scoring. The benefits and costs of different predictions can be summarized in a two-dimensional instance-dependent cost matrix with one dimension for the predicted value and another dimension for the ground truth. Given these benefits and costs, each new instance should be assigned to the class that leads to the lowest expected cost, which is calculated by means of conditional probabilities.

2.1 IDCS learning, cslogit, and robustness

For certain applications, benefits and costs depend not only on the class but also on the instance itself. Therefore, instance-dependent cost-sensitive learning considers a more detailed, lower level of granularity than class-dependent costs. For these applications, using instance-dependent costs instead of class-dependent costs leads to a decreased total misclassification cost (Brefeld, Geibel, & Wysotzki, 2003; Vanderschueren et al., 2022).

Several instance-dependent cost-sensitive methodologies have been proposed in the literature, with recent overviews given by Petrides and Verbeke (2022) and Vanderschueren et al. (2022). Especially relevant to our work are methodologies that adjust the learning algorithm to incorporate instance-dependent costs. Instance-dependent cost-sensitive variants have been proposed for several common machine learning classifiers, such as boosting (Fan, Stolfo, Zhang, & Chan, 1999; Höppner et al., 2022; Zelenkov, 2019), support vector machines (Brefeld et al., 2003), decision trees (Bahnsen, Aouada, & Ottersten, 2015; Sahin, Bulkan, & Duman, 2013), and logistic regression (Bahnsen, Aouada, & Ottersten, 2014; Höppner et al., 2022).

In this work, we will build upon an instance-dependent cost-sensitive version of logistic regression. Following Höppner et al. (2022), we will refer to this method as cslogit. Logistic regression is a widely used method for binary classification tasks. To extend logistic regression to its IDCS counterpart, Bahnsen, Aouada, Stojanovic, and Ottersten (2016) and Höppner et al. (2022) propose an objective function that combines both cost-sensitivity and instance-dependent learning, resulting in instance-dependent costs for optimization. The application of this objective function yields significant improvements in terms of higher savings compared to cost-insensitive or class-dependent cost-sensitive models in the context of, for example, credit scoring and transaction fraud detection.

Classical nonrobust methods for regression, such as least squares or maximum likelihood techniques, try to fit the model optimally to all the data. As a result, these methods are heavily influenced by data outliers. This implies that outliers may bias the parameter estimates and confidence intervals and thus hypothesis tests may become unreliable and/or uninformative. In contrast, robust methods can resist the effect of outliers to avoid distorted results and

false conclusions. As an important benefit, they allow the automatic detection of outliers as observations that deviate substantially from the robust fit. It is important to note that the detected outliers are not necessarily errors in the data. The presence of outliers may reveal that the data are more heterogeneous than has been assumed and that it can be handled by the original statistical model. Outliers can be isolated or may come in clusters, indicating that there are subgroups in the population that behave differently. Many different approaches to robust regression have been proposed and a good overview can be found in reference works such as Huber and Ronchetti (2009), Maronna, Martin, Yohai, and Salibián-Barrera (2019) and Rousseeuw and Leroy (1987). In the context of generalized linear models (GLMs), various robust alternatives have been presented, such as Cantoni and Ronchetti (2001), Bergesio and Yohai (2011), Valdora and Yohai (2014), Ghosh and Basu (2016) and Štefelová, Alfons, Palarea-Albaladejo, Filzmoser, and Hron (2021). Robust logistic regression has been studied by Künsch, Stefanski, and Carroll (1989), Morgenthaler (1992), Carroll and Pederson (1993), Bianco and Yohai (1996), Croux and Haesbroeck (2003), Bondell (2005), Bondell (2008), Monti and Filzmoser (2021) and Hosseinian and Morgenthaler (2011).

2.2 Preliminaries

The dataset $\mathcal{D}$ consists of N observed predictor-response pairs $\{(\boldsymbol{x}_i, y_i)\}_{i=1}^{N}$ and is used to train a binary classification model $s(.)$. The costs C_i correspond to the cost matrix defined in Table 1.

This binary classification model predicts a probability score $s_i \in [0, 1]$ for each instance i based on the features $\boldsymbol{x}_i$. Depending on the classification threshold t_i^*, s_i is converted to a predicted class $\widehat{y}_i \in \{0, 1\}$.

For models trained with AEC (Equation (3)), savings remain relatively stable across different thresholding strategies (Vanderschueren et al., 2022). Therefore, we use a default threshold of 0.5.

A binary logistic regression predicts a probability score that an observation belongs to the positive class. This probability score is calculated by Equation (1), where β_0 is the bias term, $\beta_1 \dots \beta_d$ the learned weights and $\boldsymbol{x}_i$ are the features of a particular observation i:

$$s_i = s_{(\beta_0, \boldsymbol{\beta})}(\boldsymbol{x}_i) = \frac{1}{1 + e^{-z}} \text{ where } z = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \ldots + \beta_d x_{id}. \quad (1)$$

This probability score is then compared to a threshold to categorize each of these observations into classes. The objective function of a logistic regression is the likelihood that is maximized or the cross-entropy loss that is minimized. For a single sample with true label $y_i \in \{0, 1\}$ and a probability score $s_i = P(Y = 1)$, the cross-entropy loss is presented by Equation (2):

$$L_{\log}(y_i, s_i) = -\Big(y_i \log(s_i) + (1 - y_i) \log(1 - s_i)\Big). \quad (2)$$

Note that this equation does not take into account any costs. Because this objective function assigns equal weights to each misclassification, it does not necessarily correspond to the underlying business problem where costs are to be minimized. The reason for this is twofold: misclassification costs are different per class and per instance. The real business objective is to minimize the average expected total cost of the binary classifier.

We build upon the instance-dependent cost-sensitive logistic (cslogit) model as proposed by Bahnsen et al. (2016) and Höppner et al. (2022). Cslogit minimizes an instance-dependent cost-sensitive objective function corresponding to the real business objective of minimizing costs in domains such as customer churn prediction, credit scoring, and direct marketing (Claude Sammut, 2017; Thai-Nghe, Gantner, & Schmidt-Thieme, 2010). Dependent on this business objective, also other cost matrices can be considered. For example, Höppner et al. (2022) propose the cost matrix $C_i(0 \mid 0) = 0$, $C_i(0 \mid 1) = A_i$, $C_i(1 \mid 0) = c_f$, and $C_i(1 \mid 1) = c_f$ for the detection of transfer fraud where c_f is a fixed administrative fee. Alternatively, Bahnsen et al. (2014) propose the cost matrix $C_i(0 \mid 0) = 0$, $C_i(0 \mid 1) = L_{gd}$, $C_i(1 \mid 0) = r_i + C^a_{FP}$, and $C_i(1 \mid 1) = 0$ for credit scoring where L_{gd} is the loss given default, r_i is the loss in profit by rejecting what could have been a good customer, and C^a_{FP} is the cost related to the assumption that the financial institution will not keep the amount of the declined applicant unused. However, the reason for this work is to address the need for robustness and to propose a solution to solve this potential issue in a generic way, regardless of its application. Therefore, to present an application-agnostic methodology and preferring the most simple cost matrix, this work utilizes a symmetric cost matrix.

Equation (3) shows the average expected cost (AEC), the cost-sensitive objective function that is used by cslogit, given a symmetric cost matrix, as shown in Table 1:

$$\begin{aligned} AEC(s(\mathcal{D})) =& \frac{1}{N} E[\text{Cost}(s(\mathcal{D})) \mid \boldsymbol{X}] \\ =& \frac{1}{N} \sum_{i=1}^{N} \Big(y_i [s_i C_i(1 \mid 1) + (1 - s_i) C_i(0 \mid 1)] \\ & + (1 - y_i)[s_i C_i(1 \mid 0) + (1 - s_i) C_i(0 \mid 0)] \Big) \\ =& \frac{1}{N} \sum_{i=1}^{N} \Big(A_i (y_i (1 - s_i) + (1 - y_i) s_i) \Big). \end{aligned} \tag{3}$$

In Equation (3), each observation i is a pair of d features $\boldsymbol{x}_i = (x_{i1}, ..., x_{id})$ and a binary response label $y_i \in \{0, 1\}$.

Across multiple models, the total cost as a metric is not unambiguously interpretable, as datasets with high instance-dependent costs might have a higher total misclassification cost but still have a better relative score. Proceeding with the idea of normalizing the total classification costs of a model presented

Table 1: Symmetric cost matrix for cslogit

	Actual 0	Actual 1
Predicted as 0	$C_i(0 \mid 0) = 0$	$C_i(0 \mid 1) = A_i$
Predicted as 1	$C_i(1 \mid 0) = A_i$	$C_i(1 \mid 1) = 0$

in Whitrow, Hand, Juszczak, Weston, and Adams (2009), Bahnsen et al. (2014) introduce a more interpretable metric: *Savings*. This metric represents the relative improvement of the cost of a newly proposed model, $Cost(s(\mathcal{D}))$, compared to the cost of using an empty model that assigns all instances to a single class, $Cost_{empty}(\mathcal{D})$. $Cost_{empty}(\mathcal{D})$ is calculated by taking the minimum of the costs incurred when classifying all instances as either belonging to the negative or positive class:

$$Cost_{empty}(\mathcal{D}) = \min \left\{ Cost\left(s_0(\mathcal{D})\right), Cost\left(s_1(\mathcal{D})\right) \right\}. \tag{4}$$

Using the $Cost_{empty}(\mathcal{D})$ of an empty model as a factor to normalize total costs, *Savings* of the model $s(\mathcal{D})$ are calculated by Equation (5):

$$Savings(s(\mathcal{D})) = 1 - \frac{Cost(s(\mathcal{D}))}{Cost_{empty}(\mathcal{D})}. \tag{5}$$

3 Sensitivity analysis

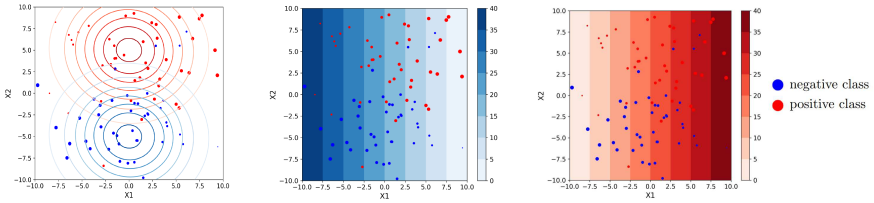
Data can contain outliers in terms of misclassification costs due to various reasons, such as missing data, invalid observations, or typos. By incorporating instance-dependent costs in the learning algorithm, outliers in these misclassification costs could potentially have a large impact on instance-dependent cost-sensitive learning methodologies such as cslogit. Therefore, we test the sensitivity of cslogit to these outliers and examine to what extent this is a shortcoming of this method.

3.1 Simulation setup

We analyze the sensitivity to outlying costs through a series of simulations on synthetic data. The different synthetic datasets all share the following properties. Each observation is visualized by a dot, with the size of the dot corresponding to its misclassification cost. The positive class is presented in red and the negative class in blue. Each observation has, other than its misclassification cost and label, two features: X_1 and X_2. X_1 is the feature for the misclassification cost A. For the positive class, this cost is positively related to X_1. Cases of the negative class have a negative relation between X_1 and their cost. The underlying function is given by Equation (6):

$$A_i = \begin{cases} 20 + 2x_{1i} & \text{for the positive class,} \\ 20 - 2x_{1i} & \text{for the negative class.} \end{cases} \tag{6}$$

Panel (b) and (c) in Figure 1 visualize this equation.



(a) Class distribution (b) Generated costs: negative class (c) Generated costs: positive class

Fig. 1: **The setup for synthetic data.** Panel (a) displays the distribution of the negative and positive class, dependent on X_2. Panels (b) and (c) represent the misclassification costs of observations from the negative and positive classes as a linear function of X_1, generated by Equation (6). The three panels all show a sample of size 50 per class.

X_2 is the feature that determines the two distributions of classes 0 and 1. The two class distributions are a 2-dimensional Gaussian, sharing the same standard deviations. Observations from the negative class are sampled from $N(\mu_0, \sigma_0^2, \nu_0, \tau_0^2, \rho)$ and observations from the positive class from $N(\mu_1, \sigma_1^2, \nu_1, \tau_1^2, \rho)$.

μ_0 and μ_1 are both equal to 0, while $\nu_0 = -5$ and $\nu_1 = 5$. The variances $\sigma_0^2, \tau_0^2, \sigma_1^2$ and τ_1^2 are equal to 4. As there is no correlation between the two dimensions X_1 and X_2, ρ is equal to 0. The cases of the positive class have a higher X_2 value than the cases of the negative class. Panel (a) in Figure 1 displays these class distributions by which data are generated.

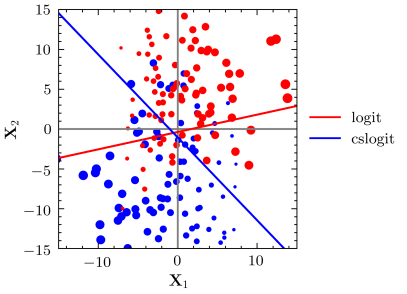
To generate outliers, both in the synthetic setup (Sections 3 and 5.1) and in the sensitivity analysis on real data (Section 5.2), the multivariate distribution of $\mathbf{X}$ remains unchanged since we only focus on outliers in the observed costs of instances. We use the *Tukey-Huber* contamination model to generate the cost distribution with outliers where the Dirac function is applied to generate costs of any size Maronna et al. (2019).

Given these settings for instance-dependent costs and class distribution, observations of the negative class with a high associated cost are expected to be located in the third quadrant and observations of the positive class with a high associated cost in the first quadrant. The symmetric cost matrix used for the examples on synthetic data is presented in Table 1 as introduced in Section 2.2.

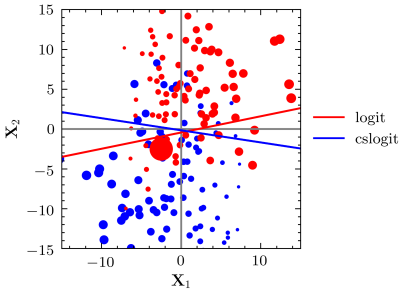
3.2 Results

Within each setting, two classifiers are compared: logit and cslogit. They are both linear classifiers and propose a distinctly different decision boundary based on the training data. Since the data are only two-dimensional, these decision

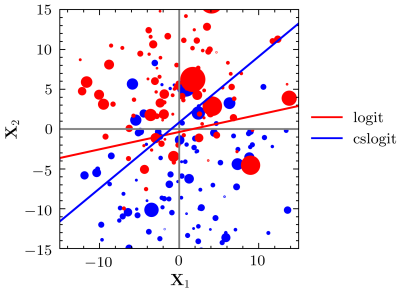
boundaries can be visually represented by lines. The logit and cslogit model's proposed boundaries are respectively coloured in red and blue. The normal behavior of both models in the default settings of examples on synthetic data is visualized in Panel a of Figure 2.



(a) Example on synthetic data: *Optimal* behavior of logit and cslogit



(b) Example on synthetic data: Influence of outliers on cslogit



(c) Example on synthetic data: Influence of noise on cslogit

Fig. 2: **The instability of cslogit's decision boundary.** This figure motivates the need for a robust version of cslogit. Three examples of synthetic data where logit and cslogit are tested are shown. Panel a shows the normal behavior of cslogit and logit in the default case. Panel b displays the case where a large outlier is added. The blue decision boundary of cslogit shifts, while the red decision boundary of logit remains stable. Note that the effect of the outlier can be subtle, i.e. it *pulls* on the decision boundary resulting in a slight rotation, without actually being classified correctly. Panel c displays the case where random noise is added to the misclassification costs. The decision boundary of cslogit shifts even further, resulting in an almost perpendicular boundary in comparison with Panel a. We further elaborate on the exact setting of the examples on synthetic data in Section 5.1.

4 Robust IDCS

To overcome the sensitivity of instance-dependent cost-sensitive classifiers to outlying costs, we introduce a three-step framework to make IDCS methods robust by detecting outliers and adapting their cost matrix. Hence, the final model will be trained using a less volatile and more rigid set of costs. The resulting robust classification model will also yield automatic outlier detection. The concrete implementation of this framework is represented by Algorithm 1.

Algorithm 1 Robust IDCS

Input: $\mathcal{D} = \{(\mathbf{x}_i, A_i, y_i) : i = 1, \ldots, N\}$ where $\mathbf{x}_i$ is a feature vector, A_i is the associated misclassification cost and $y_i \in \{0, 1\}$ is the response label of an observation i.

Output: Robust IDCS predictions $\hat{y}$ of label y

Step 1: Detect outliers.
Train a linear regression model with Huber loss so that: $\hat{A} = f(\mathbf{X}, Y)$
Initialize set $\mathcal{S}_{outlier} := \emptyset$
for each observation i **do**
    if absolute value of the standardized residuals > 2.5 **then**
        add observation $(\mathbf{x}_i, A_i, y_i)$ to set $S_{outlier}$
        remove observation $(\mathbf{x}_i, A_i, y_i)$ from $\mathcal{D}$
    end if
end for
Step 2: Impute instance-dependent misclassification cost.
for each observation i in $\mathcal{S}_{outlier}$ **do**
    replace A_i with $\hat{A}_i$
end for
$\mathcal{D}' := \mathcal{D} \cup \mathcal{S}_{outlier}$
Step 3: Apply the IDCS method.
Apply cslogit to the new set $\mathcal{D}'$.

To estimate the misclassification costs of observations in a robust manner in step 1, a regression with Huber loss is applied. Concretely, we estimate the cost A_i of observation i as a function of its features $\mathbf{x}_i$ and label y_i with a linear regression with a Huber loss function: $\hat{A}_i = f(\boldsymbol{x}_i, y_i)$. A formalization of robustness in statistics started with the work of Huber (1964). Interestingly, his ground-breaking results and well-known loss function are still widely used today in the field of statistics and machine learning. The Huber loss function is defined by Equation (7). This results in a regression that is less sensitive to outliers than traditional regression methods, which often use a squared error loss.

$$L_\delta(a) = \begin{cases} \frac{1}{2}a^2 & \text{for } |a| \leq \delta, \\ \delta\left(|a| - \frac{1}{2}\delta\right) & \text{otherwise.} \end{cases} \tag{7}$$

Next, to detect outliers, we compare the absolute value of the standardized residuals with a cutoff value of a normal distribution (Rousseeuw & Hubert,

2011). If this value exceeds 2.5, we consider it an outlier and add it to the initially empty set $\mathcal{S}_{outlier}$.

The observed cost A_i of observation i is operationally defined as an outlier if, for an estimator $\hat{A} = f(X, Y)$, the absolute value of the standardized residual ϵ_i is larger than 2.5.

Figure 3 further clarifies the concept of a conditional outlier on the setting of synthetic data where noise is added to the observed costs, as is displayed in Figure 2c. In this figure, the black line displays the estimated costs, as predicted by the linear regression model with Huber loss (Algorithm 1, line 2). The red dots represent the costs of observations in function of X_1. Consider the two observations A and B. To check whether their observed costs are outliers, we look at the standardized residuals, represented by the grey vertical lines for those two observations. Both observations A and B have an observed cost of 50. The standardized residual of A exceeds 2.5, whereas for B it is smaller than 2.5. Consequently, although both observations have the same cost, the cost of A is considered an outlier, whereas the cost of B is not.

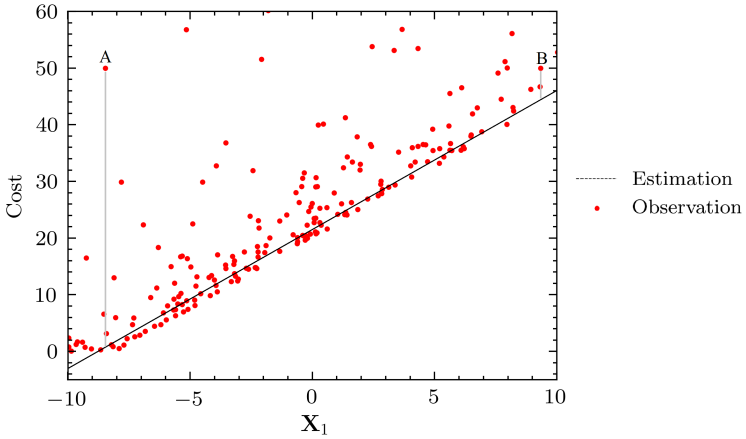


Fig. 3: Cost estimation in function of X_1.

By doing so, the costs A that are outliers, conditional on their features $\mathbf{X}$ and label Y, are detected. In step 2, the observed outlying costs A of all observations in $\mathcal{S}_{outlier}$ are imputed with their estimated counterpart $\hat{A}$. This results in a robust cost matrix (Table 2).

Equation (8) retakes the cost-sensitive objective function AEC, given by Equation (3), but adapts it to the new robust cost-matrix given by Table 2. Indicator function $\mathbb{1}_o$ takes value 1 if the cost A_i of observation i is classified as an outlier.

Table 2: Symmetric cost matrix for r-cslogit

	$y=0$	$y=1$
$\hat{y}=0$	$C_i(0 \mid 0)=0$	$C_i(0 \mid 1)=\begin{cases}\hat{A}_i=f(\boldsymbol{x}_i,y_i) & \text{if outlier,}\\ A_i & \text{otherwise}\end{cases}$
$\hat{y}=1$	$C_i(1 \mid 0)=\begin{cases}\hat{A}_i=f(\boldsymbol{x}_i,y_i) & \text{if outlier,}\\ A_i & \text{otherwise}\end{cases}$	$C_i(1 \mid 1)=0$

$$\begin{aligned}\mathrm{AEC}(s(\mathcal{D})) =& \frac{1}{N} E[\mathrm{Cost}(s(\mathcal{D})) \mid \boldsymbol{X}] \\ =& \frac{1}{N}\sum_{i=1}^{N}\Bigg[\mathbb{1}_o\Big(\hat{A}_i\left(y_i\left(1-s_i\right)+\left(1-y_i\right)s_i\right)\Big) \\ &+\left(1-\mathbb{1}_o\right)\Big(A_i\left(y_i\left(1-s_i\right)+\left(1-y_i\right)s_i\right)\Big)\Bigg]\end{aligned} \tag{8}$$

5 Results

This section discusses the performance of logit, cslogit and the novel r-cslogit on synthetic data and tests their sensitivity on real data with additional outliers. In the reported experiments, symmetric cost matrices are taken into account. However, the use of alternative cost matrices as presented in Section 2.2 yields similar results concerning robustness.

The performance of binary classification algorithms is typically measured by labeling one class as positive and the other class as negative and constructing a confusion matrix. Positive classes are typically used to describe the minority class and negative classes are used to describe the majority class. From the confusion matrix, we count the following numbers. True Negatives (TN) is the number of correctly classified negative cases. False Positives (FP) is the number of negative cases incorrectly classified as positive. False Negatives (FN) is the number of positive cases incorrectly classified as negative. True Positives (TP) is the number of correctly classified positive cases. With these numbers, we define *Sensitivity* or *Recall*, *Specificity*, and *Precision* by Equations 9, 10, and 11:

$$Sensitivity = Recall = \frac{TP}{TP+FN} \tag{9}$$

$$Specificity = \frac{TN}{TN+FP} \tag{10}$$

$$Precision = \frac{TP}{TP+FP} \tag{11}$$

Equation 12 defines $F1$-measure, which is the Harmonic mean of precision and recall.

$$F_1 = \frac{2 \cdot precision \cdot recall}{precision + recall} \tag{12}$$

The area under the receiver operating curve (AUC) of a classifier can be interpreted as a measure of the probability that a randomly chosen minority case is predicted to have a higher score than a randomly chosen majority case. Therefore, a higher AUC indicates better classification performance (Höppner et al., 2022). Class distributions and misclassification costs are not taken into account in calculating the AUC.

Equation 13 defines the Brier score, where s_i is the predicted probability and y_i is the observed outcome. This metric measures the mean squared difference between the predicted probability and the actual outcome and is used to assess whether the model's predictions are calibrated probabilities. A lower score is better.

$$Brier = \frac{1}{N} \sum_{i=1}^{N} (s_i - y_i)^2 \tag{13}$$

5.1 Synthetic data

In this subsection, we reuse the examples on synthetic data introduced in Section 3.1 to demonstrate how the possible shortcomings of cslogit can be countered by deploying the more robust r-cslogit. Figure 4 displays the decision boundaries of logit, cslogit and r-cslogit in red, blue and green, respectively.

Synthetic data: Three settings. The basic setting of the examples on synthetic data in Panel a is the same as explained in Subsection 3.1. In Panel b, an additional outlier of the positive class is added in the third quadrant with a cost equal to 400. The robust method first estimates its cost with a linear Huber regression to be 13.75 and flags it as an outlier. Next, the cost for this instance of 400 is changed to its estimated cost of 13.75. In Panel c, we add noise to the costs. Hence, the misclassification costs are generated by Equation (14), where the noise ϵ_i is sampled from a lognormal distribution with parameters $\mu = 2$ and $\sigma = 1.5$.

$$A_i = \begin{cases} 20 + 2x_{1i} + \epsilon_i & \text{for the positive class,} \\ 20 - 2x_{1i} + \epsilon_i & \text{for the negative class.} \end{cases} \tag{14}$$

Description of results. Figure 4 visualizes the decision boundaries of the three models. In Panel a, the decision boundaries of cslogit and r-cslogit overlap as regression with Huber Loss can perfectly predict the underlying function of associated misclassification costs as a function of X_1 in the absence of noise or outliers. Panel b displays the case where one outlier is added. The decision boundary of the logit model is not affected by the size of misclassification costs. Hence, it is not influenced by the outlier and remains unchanged, demonstrating normal behavior as defined before. The blue decision boundary of the cslogit model is strongly influenced by outliers. The objective function takes into account the full misclassification costs of the observations in the training set,

including the excessive outliers. As a consequence, the behavior of the cslogit model has been completely disrupted. This is strongly in conflict with its normal behavior, as the decision boundary is almost tilted by a quarter turn. This tilted decision boundary results in poor predictive classification power, making the cslogit model to be of inferior quality. The green decision boundary of r-cslogit remains largely unchanged, as it is robust against the single added outlier.

Performance metrics are summarized in Table 3. We consider the cost-sensitive metric *Savings* introduced in Section 2.2 and cost-independent metrics *Sensitivity*, *Specificity*, $F1$, *AUC*, and *Brier* score.

r-cslogit outperforms logit and cslogit in terms of Savings when we add an outlier and noise. Moreover, the performance in terms of Savings remains unchanged after adding an outlier. In the default case of setting one, r-cslogit and cslogit are equivalent, as they make the exact same predictions. When considering cost-insensitive metrics, logit performs best. A full analysis on synthetic data where we experiment with different settings of class imbalance and outlier size can be found in Appendix A.

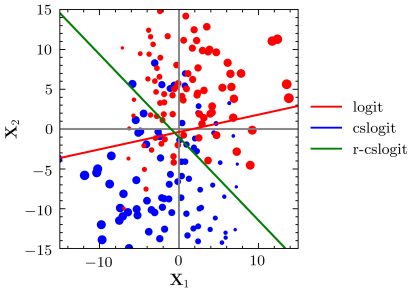
		Savings	$F1$	*AUC*	*Sensitivity*	*Specificity*	*Brier*
default setting	logit	0.6845 ± 0.08	$\mathbf{0.8444 \pm 0.04}$	$\mathbf{0.8551 \pm 0.04}$	$\mathbf{0.8321 \pm 0.04}$	$\mathbf{0.8781 \pm 0.05}$	$\mathbf{0.1433 \pm 0.04}$
	cslogit	$\mathbf{0.8024 \pm 0.05}$	0.7650 ± 0.05	0.7739 ± 0.06	0.7821 ± 0.03	0.7656 ± 0.11	0.2267 ± 0.06
	r-cslogit	$\mathbf{0.8024 \pm 0.05}$	0.7650 ± 0.05	0.7739 ± 0.06	0.7821 ± 0.03	0.7656 ± 0.11	0.2267 ± 0.06
with outlier	logit	0.6845 ± 0.08	$\mathbf{0.8444 \pm 0.04}$	$\mathbf{0.8551 \pm 0.04}$	$\mathbf{0.8321 \pm 0.04}$	$\mathbf{0.8781 \pm 0.05}$	$\mathbf{0.1433 \pm 0.04}$
	cslogit	0.6461 ± 0.15	0.8182 ± 0.05	0.8270 ± 0.05	0.8121 ± 0.05	0.8219 ± 0.07	0.1733 ± 0.05
	r-cslogit	$\mathbf{0.8024 \pm 0.05}$	0.7650 ± 0.05	0.7739 ± 0.06	0.7821 ± 0.03	0.7656 ± 0.11	0.2267 ± 0.06
with noise	logit	0.6845 ± 0.08	$\mathbf{0.8444 \pm 0.04}$	$\mathbf{0.8551 \pm 0.04}$	$\mathbf{0.8321 \pm 0.04}$	$\mathbf{0.8781 \pm 0.05}$	$\mathbf{0.1433 \pm 0.04}$
	cslogit	0.5666 ± 0.09	0.7648 ± 0.06	0.7703 ± 0.05	0.7657 ± 0.06	0.8253 ± 0.04	0.1683 ± 0.05
	r-cslogit	$\mathbf{0.7778 \pm 0.06}$	0.8182 ± 0.03	0.8308 ± 0.04	0.8142 ± 0.05	0.8575 ± 0.05	0.1469 ± 0.03

Table 3: Results on synthetic data (i) in the default setting, (ii) with an outlier, and (iii) with additional noise added to the amounts. The sample size is set to 300, i.e. 150 per class. The data is generated according to the setting as explained in Section 3.1. We apply a 2×5-fold cross validation procedure with a train/test split ratio of 0.8/0.2. The best performing methods are indicated in bold. A full analysis on synthetic data with different settings for class imbalance and outlier size can be found in Appendix A. We report the average together with the standard deviation over these 10 runs.

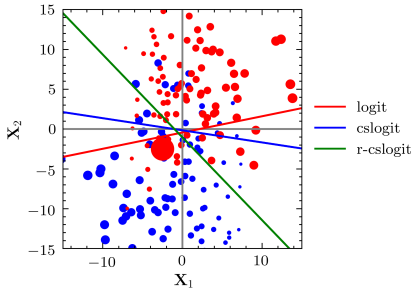
5.2 Sensitivity analysis on real data

In this subsection, we analyze the sensitivity of the three methods in an experiment with real data where we add an additional outlier, gradually increasing in size. To add outliers, we randomly select an observation and change its class label and instance-dependent misclassification cost.

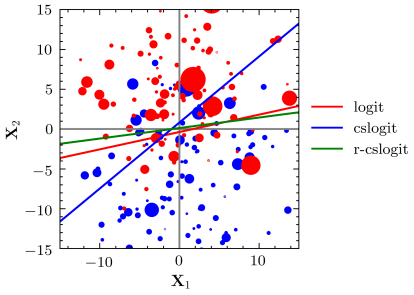
This setup is similar to the second setup with synthetic data as presented in the previous subsection. The performance is measured by the cost-sensitive metric *Savings* as described before as well as the cost-independent metrics



(a) Example on synthetic data: Default case



(b) Example on synthetic data: One outlier



(c) Example on synthetic data: Additional noise

Fig. 4: **Superiority of r-cslogit.** The red decision boundary of logit remains unchanged, as it is not cost-sensitive. The blue decision boundary of cslogit differs strongly per example, as the model is prone to outliers and noise. The green decision boundary of r-cslogit is stable against outliers and handles noise in misclassification costs quite well. In Panel a, the blue and green decision boundaries coincide.

Sensitivity, *Specificity*, $F1$, *AUC*, and *Brier* score. The measurement of performance makes use of five-fold cross-validation with a stratified split on class distribution that is repeated twice with a different random initialization.

Description of the dataset. The dataset on which the three methods are tested is the Kaggle Credit Card Fraud Detection dataset (ULB, 2018). The dataset dates from September 2013 and contains transactions made by European credit cardholders. A total of 492 out of 284,807 transactions are fraudulent, resulting in a high class imbalance. The numerical input features $V1, V2, \ldots, V28$ are the results of a PCA transformation to anonymize the dataset. *Time* and *Amount* have not been transformed. The feature *Time* is

not taken into consideration in this experiment and is therefore dropped in the preprocessing phase. The feature *Amount* is the transaction amount, which is of high importance in cost-sensitive instance-dependent learning and translates into our setting as the instance-dependent misclassification cost. The feature $Class \in \{0, 1\}$ indicates whether a transaction is fraudulent or not.

Outlier size	Method	*Savings*	$F1$	*AUC*	*Sensitivity*	*Specificity*	*Brier*
0	logit	0.1310 ± 0.14	0.7197 ± 0.01	$\mathbf{0.9715 \pm 0.01}$	0.6204 ± 0.04	$\mathbf{0.9998 \pm 0.00}$	$\mathbf{0.0005 \pm 0.00}$
	cslogit	$\mathbf{0.6060 \pm 0.06}$	0.8100 ± 0.01	0.9290 ± 0.02	0.7817 ± 0.04	0.9997 ± 0.00	0.0006 ± 0.00
	r-cslogit	$\mathbf{0.6060 \pm 0.06}$	$\mathbf{0.8120 \pm 0.02}$	0.9280 ± 0.02	$\mathbf{0.7828 \pm 0.04}$	0.9997 ± 0.00	0.0006 ± 0.00
10 K	logit	0.1310 ± 0.14	0.7197 ± 0.01	$\mathbf{0.9715 \pm 0.01}$	0.6204 ± 0.04	$\mathbf{0.9998 \pm 0.00}$	$\mathbf{0.0005 \pm 0.00}$
	cslogit	$\mathbf{0.6094 \pm 0.06}$	$\mathbf{0.8103 \pm 0.01}$	0.9259 ± 0.02	$\mathbf{0.7849 \pm 0.04}$	0.9997 ± 0.00	0.0006 ± 0.00
	r-cslogit	0.5916 ± 0.07	0.8092 ± 0.01	0.9264 ± 0.02	0.7817 ± 0.04	0.9997 ± 0.00	0.0006 ± 0.00
100 K	logit	0.1310 ± 0.14	0.7197 ± 0.01	$\mathbf{0.9715 \pm 0.01}$	0.6204 ± 0.04	$\mathbf{0.9998 \pm 0.00}$	$\mathbf{0.0005 \pm 0.00}$
	cslogit	0.6071 ± 0.06	$\mathbf{0.8097 \pm 0.01}$	0.9278 ± 0.02	0.7839 ± 0.04	0.9997 ± 0.00	0.0006 ± 0.00
	r-cslogit	$\mathbf{0.6091 \pm 0.06}$	0.8084 ± 0.01	0.9285 ± 0.02	$\mathbf{0.7849 \pm 0.04}$	0.9997 ± 0.00	0.0006 ± 0.00
1 M	logit	0.1310 ± 0.14	0.7197 ± 0.01	$\mathbf{0.9715 \pm 0.01}$	0.6204 ± 0.04	$\mathbf{0.9998 \pm 0.00}$	$\mathbf{0.0005 \pm 0.00}$
	cslogit	0.5438 ± 0.06	0.7217 ± 0.03	0.8921 ± 0.04	0.7837 ± 0.04	0.9992 ± 0.00	0.0012 ± 0.00
	r-cslogit	$\mathbf{0.6034 \pm 0.6}$	$\mathbf{0.8100 \pm 0.01}$	0.9266 ± 0.02	$\mathbf{0.7849 \pm 0.03}$	0.9997 ± 0.00	0.0006 ± 0.00
10 M	logit	0.1310 ± 0.14	0.7197 ± 0.01	$\mathbf{0.9715 \pm 0.01}$	0.6204 ± 0.04	$\mathbf{0.9998 \pm 0.00}$	$\mathbf{0.0005 \pm 0.00}$
	cslogit	0.4102 ± 0.13	0.6618 ± 0.04	0.8716 ± 0.05	0.7570 ± 0.06	0.9989 ± 0.00	0.0015 ± 0.00
	r-cslogit	$\mathbf{0.6072 \pm 0.06}$	$\mathbf{0.8094 \pm 0.01}$	0.9278 ± 0.02	$\mathbf{0.7849 \pm 0.04}$	0.9997 ± 0.00	0.0006 ± 0.00
100 M	logit	0.1310 ± 0.14	0.7197 ± 0.01	$\mathbf{0.9715 \pm 0.01}$	0.6204 ± 0.04	$\mathbf{0.9998 \pm 0.00}$	$\mathbf{0.0005 \pm 0.00}$
	cslogit	0.3216 ± 0.24	0.6245 ± 0.04	0.8544 ± 0.05	0.7269 ± 0.09	0.9988 ± 0.00	0.0016 ± 0.00
	r-cslogit	$\mathbf{0.6036 \pm 0.06}$	$\mathbf{0.8096 \pm 0.01}$	0.9277 ± 0.02	$\mathbf{0.7817 \pm 0.04}$	0.9997 ± 0.00	0.0006 ± 0.00
1 B	logit	0.1310 ± 0.14	0.7197 ± 0.01	$\mathbf{0.9715 \pm 0.01}$	0.6204 ± 0.04	$\mathbf{0.9998 \pm 0.00}$	$\mathbf{0.0005 \pm 0.00}$
	cslogit	0.2877 ± 0.27	0.6284 ± 0.05	0.8706 ± 0.04	0.7559 ± 0.05	0.9987 ± 0.00	0.0017 ± 0.00
	r-cslogit	$\mathbf{0.6092 \pm 0.06}$	$\mathbf{0.8093 \pm 0.01}$	0.9268 ± 0.02	$\mathbf{0.7839 \pm 0.04}$	0.9997 ± 0.00	0.0006 ± 0.00

Table 4: Sensitivity analysis on real data resulting from a two times five-fold cross validation procedure on the Kaggle Credit Card Fraud Detection dataset. The size of the outlier is gradually increased. Each metric is based on 10 (2×5) out-of-sample performance estimates over the 10 test sets. We report the average together with the standard deviation over these 10 runs.

Results. Table 4 contains the results of a 2×5-fold cross validation procedure for the Kaggle Credit Card Fraud Detection dataset. We measure each classifier's performance averaged over the ten (2×5) test sets with the metrics *Savings*, $F1$, *AUC*, *Sensitivity*, *Specificity*, and *Brier* score where instance-independent thresholds are applied.

In terms of *Savings*, logit is always outperformed by cslogit and r-cslogit. When adding an outlier, r-cslogit outperforms cslogit. Note that the performance of r-cslogit remains stable for all considered metrics when increasing the size of the outlier. In terms of cost-insensitive metrics *AUC*, *Specificity*, and *Brier* score, logit performs best. In terms of $F1$ and *Sensitivity*, logit is outperformed by either cslogit or r-cslogit. This could be due to the effect of class imbalance and is in line with previous findings of Höppner et al. (2022) The results in terms of Savings are visualized in Figure 5. Since the logit model is not cost-sensitive, its performance remains constant after adding an outlier. The performance of cslogit is strongly disrupted after the cost of the outlier is set to 1 M or larger.

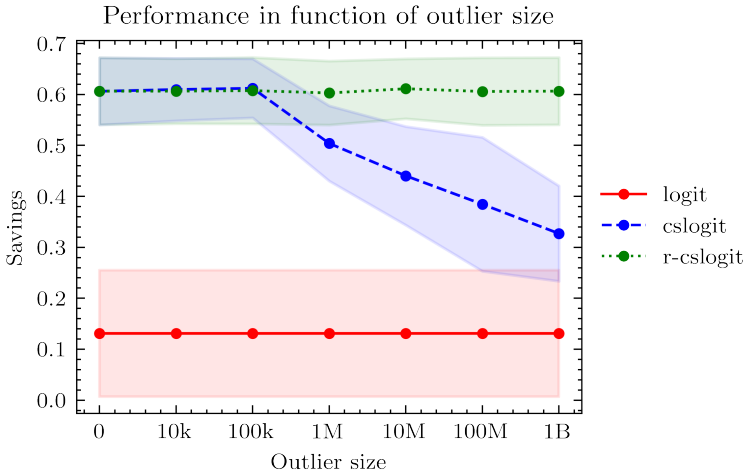


Fig. 5: Sensitivity analysis on real data.

This corresponds with the shift of the two-dimensional linear decision boundary, as shown by the findings of the examples on synthetic data. Even though the dataset contains over 280,000 instances, a single outlier, albeit a large outlier, can unhinge the cslogit method. When increasing the misclassification cost of a single outlier, the performance of r-cslogit remains stable. It is certainly more robust to this additional noise than its nonrobust counterpart, as the individual outlier is detected and its cost is imputed with an estimated, expected cost. The shaded areas in Figure 5 represent the variability of performance over different folds in cross-validation. In contrast to the variability of cslogit, which increases drastically, the variability in the performance of r-cslogit remains stable.

6 Conclusion

Instance-dependent cost-sensitive (IDCS) learning methods take into account variable misclassification costs across instances in the training data in learning a classification model. This allows for optimizing the performance of the resulting classification model in terms of the misclassification costs rather than the classification accuracy.

In this article, we present the results of a series of experiments on synthetic data to demonstrate the sensitivity of IDCS methods to outliers and noise in the data. We show that the resulting classification model may be highly sensitive to outlying instance-dependent costs, in learning an instance-dependent cost-sensitive classification model. Consequently, using existing cost-sensitive models in the presence of noise or outliers can result in large misclassification costs.

To address this potential vulnerability, we propose a generic, IDCS-method-independent, three-step framework to develop robust IDCS methods with respect to the effects of random variability and noise. In the first step, instances

with outlying misclassification costs are detected. In the second step, outlying costs are corrected in a data-driven way. In the third step, an IDCS learning method is applied using the adjusted instance-dependent cost information.

This generic framework is subsequently applied in combination with cslogit, which is a logistic regression-based IDCS method, to obtain its robust version named r-cslogit. The robustness of this approach is introduced in the first two steps of the generic framework by making use of robust estimators to detect and impute outlying costs of individual instances. The newly proposed r-cslogit method is tested on synthetic and semi-synthetic data. The results show that the proposed method is superior in terms of cost savings when compared to its non-robust counterpart for variable levels of noise and outliers.

References

Bahnsen, A.C., Aouada, D., Ottersten, B. (2014). Example-dependent cost-sensitive logistic regression for credit scoring. *2014 13th international conference on machine learning and applications* (p. 263-269). 10.1109/ICMLA.2014.48

Bahnsen, A.C., Aouada, D., Ottersten, B. (2015). Example-dependent cost-sensitive decision trees. *Expert Systems with Applications*, *42*(19), 6609–6619.

Bahnsen, A.C., Aouada, D., Stojanovic, A., Ottersten, B. (2016). Feature engineering strategies for credit card fraud detection. *Expert Systems with Applications*, *51*, 134–142.

Bergesio, A., & Yohai, V.J. (2011). Projection estimators for generalized linear models. *Journal of the American Statistical Association*, *106*(494), 661–671.

Bianco, A.M., & Yohai, V.J. (1996). Robust estimation in the logistic regression model. *Robust statistics, data analysis, and computer intensive methods* (pp. 17–34). Springer.

Bondell, H.D. (2005). Minimum distance estimation for the logistic regression model. *Biometrika*, *92*(3), 724–731.

Bondell, H.D. (2008). A characteristic function approach to the biased sampling model, with application to robust logistic regression. *Journal of Statistical Planning and Inference*, *138*(3), 742–755.

Brefeld, U., Geibel, P., Wysotzki, F. (2003). Support vector machines with example dependent costs. *European conference on machine learning* (pp. 23–34).

Cantoni, E., & Ronchetti, E. (2001). Robust inference for generalized linear models. *Journal of the American Statistical Association*, *96*(455), 1022–1030.

Carroll, R.J., & Pederson, S. (1993). On robustness in the logistic regression model. *Journal of the Royal Statistical Society: Series B (Methodological)*, *55*(3), 693–706.

Claude Sammut, G.I.W. (2017). *Encyclopedia of machine learning and data mining*. Springer US.

Croux, C., & Haesbroeck, G. (2003). Implementing the bianco and yohai estimator for logistic regression. *Computational statistics & data analysis*, *44*(1-2), 273–295.

Elkan, C. (2001). The foundations of cost-sensitive learning. *International joint conference on artificial intelligence* (Vol. 17, pp. 973–978).

Fan, W., Stolfo, S.J., Zhang, J., Chan, P.K. (1999). Adacost: misclassification cost-sensitive boosting. *Icml* (Vol. 99, pp. 97–105).

Ghosh, A., & Basu, A. (2016). Robust estimation in generalized linear models: the density power divergence approach. *TEST*, *25*(2), 269–290.

Höppner, S., Baesens, B., Verbeke, W., Verdonck, T. (2022). Instance-dependent cost-sensitive learning for detecting transfer fraud. *European Journal of Operational Research*, *297*(1), 291–300.

Hosseinian, S., & Morgenthaler, S. (2011). Robust binary regression. *Journal of statistical planning and inference*, *141*(4), 1497–1509.

Huber, P.J. (1964). Robust estimation of a location parameter. *The Annals of Mathematical Statistics*, *35*(1), 73–101. Retrieved from http://www.jstor.org/stable/2238020

Huber, P.J., & Ronchetti, E. (2009). Robust statistics. 2nd john wiley & sons. *Hoboken, NJ*, *2*.

Künsch, H.R., Stefanski, L.A., Carroll, R.J. (1989). Conditionally unbiased bounded-influence estimation in general regression models, with applications to generalized linear models. *Journal of the American Statistical Association*, *84*(406), 460–466.

Lessmann, S., Haupt, J., Coussement, K., De Bock, K.W. (2021). Targeting customers for profit: An ensemble learning framework to support marketing decision-making. *Information Sciences*, *557*, 286–301.

Maronna, R.A., Martin, R.D., Yohai, V.J., Salibián-Barrera, M. (2019). *Robust statistics: theory and methods (with r)*. John Wiley & Sons.

Monti, G.S., & Filzmoser, P. (2021, September). Robust logistic zero-sum regression for microbiome compositional data. *Adv. Data Anal. Classif.*.

Morgenthaler, S. (1992). Least-absolute-deviations fits for generalized linear models. *Biometrika*, *79*(4), 747–754.

Petrides, G., Moldovan, D., Coenen, L., Guns, T., Verbeke, W. (2022). Cost-sensitive learning for profit-driven credit scoring. *J. Oper. Res. Soc.*, *73*(2), 338-350.

Petrides, G., & Verbeke, W. (2022). Cost-sensitive ensemble learning: a unifying framework. *Data Mining and Knowledge Discovery*, *36*(1), 1–28.

Rousseeuw, P.J., & Hubert, M. (2011, January). Robust statistics for outlier detection. *Wiley Interdiscip. Rev. Data Min. Knowl. Discov.*, *1*(1), 73–79.

Rousseeuw, P.J., & Leroy, A.M. (1987). *Robust regression and outlier detection*. John Wiley & Sons, Inc.

Sahin, Y., Bulkan, S., Duman, E. (2013). A cost-sensitive decision tree approach for fraud detection. *Expert Systems with Applications*, *40*(15), 5916–5923.

Štefelová, N., Alfons, A., Palarea-Albaladejo, J., Filzmoser, P., Hron, K. (2021, December). Robust regression with compositional covariates including cellwise outliers. *Adv. Data Anal. Classif.*, *15*(4), 869–909.

Thai-Nghe, N., Gantner, Z., Schmidt-Thieme, L. (2010). Cost-sensitive learning methods for imbalanced data. *The 2010 international joint conference on neural networks (ijcnn)* (p. 1-8). 10.1109/IJCNN.2010.5596486

ULB, M.L.G. (2018). *Anonymized credit card transactions labeled as fraudulent or genuine*. https://www.kaggle.com/mlg-ulb/creditcardfraud.

Valdora, M., & Yohai, V.J. (2014). Robust estimators for generalized linear models. *Journal of Statistical Planning and Inference*, *146*, 31–48.

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.

Verbeke, W., Olaya, D., Berrevoets, J., Verboven, S., Maldonado, S. (2020, July). The foundations of cost-sensitive causal classification. https://arxiv.org/abs/2007.12582 [cs.LG]

Whitrow, C., Hand, D.J., Juszczak, P., Weston, D., Adams, N.M. (2009). Transaction aggregation as a strategy for credit card fraud detection. *Data mining and knowledge discovery*, *18*(1), 30–55.

Zelenkov, Y. (2019). Example-dependent cost-sensitive adaptive boosting. *Expert Systems with Applications*, *135*, 71–82.

Appendix A Results on synthetic data

		No outlier			$A_{outlier} = 100$		
Imbal.	Metric	logit	cslogit	r-cslogit	logit	cslogit	r-cslogit
50/50	*Savings*	0.6845 ± 0.08	$\mathbf{0.8024 \pm 0.05}$	$\mathbf{0.8024 \pm 0.05}$	0.6845 ± 0.08	$\mathbf{0.8024 \pm 0.05}$	$\mathbf{0.8024 \pm 0.05}$
	AUC	$\mathbf{0.8551 \pm 0.04}$	0.7739 ± 0.06	0.7739 ± 0.06	$\mathbf{0.8551 \pm 0.04}$	0.7739 ± 0.06	0.7739 ± 0.06
	F1	$\mathbf{0.8444 \pm 0.04}$	0.7650 ± 0.05	0.7650 ± 0.05	$\mathbf{0.8444 \pm 0.04}$	0.7650 ± 0.05	0.7650 ± 0.05
	Spec	$\mathbf{0.8781 \pm 0.05}$	0.7656 ± 0.11	0.7656 ± 0.11	$\mathbf{0.8781 \pm 0.05}$	0.7656 ± 0.11	0.7656 ± 0.11
	Sens	$\mathbf{0.8321 \pm 0.04}$	0.7821 ± 0.03	0.7821 ± 0.03	$\mathbf{0.8321 \pm 0.04}$	0.7821 ± 0.03	0.7821 ± 0.03
	Brier	$\mathbf{0.1433 \pm 0.04}$	0.2267 ± 0.06	0.2267 ± 0.06	$\mathbf{0.1433 \pm 0.04}$	0.2267 ± 0.06	0.2267 ± 0.06
60/40	*Savings*	0.6412 ± 0.11	$\mathbf{0.7465 \pm 0.06}$	$\mathbf{0.7465 \pm 0.06}$	0.6412 ± 0.11	$\mathbf{0.7465 \pm 0.04}$	$\mathbf{0.7465 \pm 0.04}$
	AUC	$\mathbf{0.8485 \pm 0.05}$	0.7646 ± 0.07	0.7646 ± 0.07	$\mathbf{0.8485 \pm 0.05}$	0.7661 ± 0.06	0.7646 ± 0.06
	F1	$\mathbf{0.8785 \pm 0.04}$	0.7864 ± 0.04	0.7864 ± 0.04	$\mathbf{0.8785 \pm 0.04}$	0.7883 ± 0.04	0.7864 ± 0.04
	Spec	$\mathbf{0.7930 \pm 0.06}$	0.7734 ± 0.13	0.7734 ± 0.13	$\mathbf{0.7930 \pm 0.06}$	0.7734 ± 0.12	0.7734 ± 0.12
	Sens	$\mathbf{0.9041 \pm 0.05}$	0.7558 ± 0.04	0.7558 ± 0.04	$\mathbf{0.9041 \pm 0.05}$	0.7587 ± 0.03	0.7558 ± 0.04
	Brier	$\mathbf{0.1433 \pm 0.05}$	0.2367 ± 0.06	0.2367 ± 0.06	$\mathbf{0.1433 \pm 0.05}$	0.2350 ± 0.06	0.2367 ± 0.05
70/30	*Savings*	0.5223 ± 0.05	$\mathbf{0.7040 \pm 0.05}$	$\mathbf{0.7040 \pm 0.05}$	0.5223 ± 0.05	0.6994 ± 0.05	$\mathbf{0.7040 \pm 0.05}$
	AUC	$\mathbf{0.8050 \pm 0.02}$	0.7438 ± 0.03	0.7438 ± 0.03	$\mathbf{0.8050 \pm 0.02}$	0.7413 ± 0.03	0.7438 ± 0.03
	F1	$\mathbf{0.8915 \pm 0.01}$	0.8444 ± 0.03	0.8444 ± 0.03	$\mathbf{0.8915 \pm 0.01}$	0.8417 ± 0.03	0.8444 ± 0.03
	Spec	$\mathbf{0.6750 \pm 0.03}$	0.6150 ± 0.03	0.6150 ± 0.03	$\mathbf{0.6750 \pm 0.03}$	0.6150 ± 0.03	0.6150 ± 0.03
	Sens	$\mathbf{0.9350 \pm 0.02}$	0.8725 ± 0.06	0.8725 ± 0.06	$\mathbf{0.9350 \pm 0.02}$	0.8675 ± 0.05	0.8725 ± 0.06
	Brier	$\mathbf{0.1517 \pm 0.02}$	0.2133 ± 0.04	0.2133 ± 0.04	$\mathbf{0.1517 \pm 0.02}$	0.2167 ± 0.04	0.2133 ± 0.04
80/20	*Savings*	0.3684 ± 0.10	$\mathbf{0.5826 \pm 0.15}$	$\mathbf{0.5826 \pm 0.15}$	0.3684 ± 0.10	$\mathbf{0.5826 \pm 0.15}$	$\mathbf{0.5826 \pm 0.15}$
	AUC	$\mathbf{0.7959 \pm 0.04}$	0.7396 ± 0.05	0.7396 ± 0.05	$\mathbf{0.7959 \pm 0.04}$	0.7396 ± 0.05	0.7396 ± 0.05
	F1	$\mathbf{0.9285 \pm 0.02}$	0.8974 ± 0.01	0.8974 ± 0.01	$\mathbf{0.9285 \pm 0.02}$	0.8974 ± 0.01	0.8974 ± 0.01
	Spec	$\mathbf{0.6406 \pm 0.06}$	0.5703 ± 0.13	0.5703 ± 0.13	$\mathbf{0.6406 \pm 0.06}$	0.5703 ± 0.13	0.5703 ± 0.13
	Sens	$\mathbf{0.9513 \pm 0.03}$	0.9089 ± 0.04	0.9089 ± 0.04	$\mathbf{0.9513 \pm 0.03}$	0.9089 ± 0.04	0.9089 ± 0.04
	Brier	$\mathbf{0.1150 \pm 0.04}$	0.1633 ± 0.01	0.1633 ± 0.01	$\mathbf{0.1150 \pm 0.04}$	0.1633 ± 0.01	0.1633 ± 0.01
90/10	*Savings*	0.2317 ± 0.12	$\mathbf{0.3615 \pm 0.11}$	$\mathbf{0.3615 \pm 0.11}$	0.2317 ± 0.12	$\mathbf{0.3844 \pm 0.14}$	$\mathbf{0.3844 \pm 0.14}$
	AUC	$\mathbf{0.6752 \pm 0.03}$	0.6162 ± 0.02	0.6162 ± 0.02	$\mathbf{0.6752 \pm 0.03}$	0.6551 ± 0.05	0.6420 ± 0.05
	F1	$\mathbf{0.9458 \pm 0.01}$	0.8874 ± 0.04	0.8874 ± 0.04	$\mathbf{0.9458 \pm 0.01}$	0.9097 ± 0.03	0.8945 ± 0.04
	Spec	$\mathbf{0.3824 \pm 0.06}$	0.3676 ± 0.06	0.3676 ± 0.06	0.3824 ± 0.06	$\mathbf{0.4118 \pm 0.11}$	$\mathbf{0.4118 \pm 0.11}$
	Sens	$\mathbf{0.9680 \pm 0.01}$	0.8647 ± 0.08	0.8647 ± 0.08	$\mathbf{0.9680 \pm 0.01}$	0.8985 ± 0.06	0.8985 ± 0.06
	Brier	$\mathbf{0.0983 \pm 0.02}$	0.1917 ± 0.07	0.1917 ± 0.07	$\mathbf{0.0983 \pm 0.02}$	0.1567 ± 0.05	0.1800 ± 0.06

Table A1: This table displays the results of tests on synthetic data with no outlier and an outlier of size 100. We apply a 2×5-fold cross validation procedure with a train/test split ratio of 0.8/0.2. We report the average together with the standard deviation over these 10 runs. Per row, the two classes become more and more imbalanced. The best performing methods are indicated in bold. In this table, r-cslogit always perform at least equally good in comparison to cslogit. In terms of the cost-sensitive metric *Savings*, logit is always outperformed by cslogit and r-cslogit. Logit performs best in terms of cost-insensitive metrics and its performance remains stable after increasing the size of the outlier given its cost-insensitive nature. An exception is *Specificity* with a 90/10 class imbalance and an outlier of 100. However, given the relatively high standard deviation of 0.11, these results are rather volatile because of the high class imbalance, which results in a small amount of observations in one class.

		$A_{outlier} = 1000$			$A_{outlier} = 10000$		
Imbal.	Metric	logit	cslogit	r-cslogit	logit	cslogit	r-cslogit
50/50	*Savings*	0.6845 ± 0.08	0.6461 ± 0.15	$\mathbf{0.8024 \pm 0.05}$	0.6845 ± 0.08	0.5819 ± 0.11	$\mathbf{0.8024 \pm 0.05}$
	AUC	$\mathbf{0.8551 \pm 0.04}$	0.8270 ± 0.05	0.7739 ± 0.06	$\mathbf{0.8551 \pm 0.04}$	0.8422 ± 0.03	0.7739 ± 0.06
	F1	$\mathbf{0.8444 \pm 0.04}$	0.8182 ± 0.05	0.7650 ± 0.05	$\mathbf{0.8444 \pm 0.04}$	0.8337 ± 0.03	0.7650 ± 0.05
	Spec	$\mathbf{0.8781 \pm 0.05}$	0.8219 ± 0.07	0.7656 ± 0.11	$\mathbf{0.8781 \pm 0.05}$	0.8344 ± 0.05	0.7656 ± 0.11
	Sens	$\mathbf{0.8321 \pm 0.04}$	0.8121 ± 0.05	0.7821 ± 0.03	0.8321 ± 0.04	$\mathbf{0.8500 \pm 0.07}$	0.7821 ± 0.03
	Brier	$\mathbf{0.1433 \pm 0.04}$	0.1733 ± 0.05	0.2267 ± 0.06	$\mathbf{0.1433 \pm 0.04}$	0.1583 ± 0.03	0.2267 ± 0.06
60/40	*Savings*	0.6412 ± 0.11	0.6864 ± 0.10	$\mathbf{0.7486 \pm 0.04}$	0.6412 ± 0.11	0.5844 ± 0.18	$\mathbf{0.7465 \pm 0.04}$
	AUC	$\mathbf{0.8485 \pm 0.05}$	0.8103 ± 0.07	0.7661 ± 0.06	$\mathbf{0.8485 \pm 0.05}$	0.8393 ± 0.06	0.7646 ± 0.06
	F1	$\mathbf{0.8785 \pm 0.04}$	0.8297 ± 0.08	0.7883 ± 0.04	$\mathbf{0.8785 \pm 0.04}$	0.8719 ± 0.05	0.7864 ± 0.04
	Spec	0.7930 ± 0.06	$\mathbf{0.8008 \pm 0.08}$	0.7734 ± 0.12	$\mathbf{0.7930 \pm 0.06}$	0.7773 ± 0.07	0.7734 ± 0.12
	Sens	$\mathbf{0.9041 \pm 0.05}$	0.8198 ± 0.13	0.7587 ± 0.03	$\mathbf{0.9041 \pm 0.05}$	0.9012 ± 0.06	0.7558 ± 0.04
	Brier	$\mathbf{0.1433 \pm 0.05}$	0.1883 ± 0.07	0.2350 ± 0.06	$\mathbf{0.1433 \pm 0.05}$	0.1517 ± 0.06	0.2367 ± 0.05
70/30	*Savings*	0.5223 ± 0.05	0.4240 ± 0.16	$\mathbf{0.7040 \pm 0.05}$	0.5223 ± 0.05	0.3216 ± 0.20	$\mathbf{0.7040 \pm 0.05}$
	AUC	$\mathbf{0.8050 \pm 0.02}$	0.7725 ± 0.05	0.7438 ± 0.03	$\mathbf{0.8050 \pm 0.02}$	0.7938 ± 0.06	0.7438 ± 0.03
	F1	$\mathbf{0.8915 \pm 0.01}$	0.8795 ± 0.02	0.8444 ± 0.03	$\mathbf{0.8915 \pm 0.01}$	0.8792 ± 0.03	0.8444 ± 0.03
	Spec	$\mathbf{0.6750 \pm 0.03}$	0.6050 ± 0.15	0.6150 ± 0.03	$\mathbf{0.6750 \pm 0.03}$	0.6400 ± 0.07	0.6150 ± 0.03
	Sens	$\mathbf{0.9350 \pm 0.02}$	0.9200 ± 0.06	0.8725 ± 0.06	$\mathbf{0.9350 \pm 0.02}$	0.8875 ± 0.06	0.8725 ± 0.06
	Brier	$\mathbf{0.1517 \pm 0.02}$	0.1717 ± 0.03	0.2133 ± 0.04	$\mathbf{0.1517 \pm 0.02}$	0.1617 ± 0.04	0.2133 ± 0.04
80/20	*Savings*	0.3684 ± 0.10	0.3741 ± 0.17	$\mathbf{0.5826 \pm 0.15}$	0.3684 ± 0.10	0.1022 ± 0.47	$\mathbf{0.5826 \pm 0.15}$
	AUC	$\mathbf{0.7959 \pm 0.04}$	0.7153 ± 0.04	0.7396 ± 0.05	$\mathbf{0.7959 \pm 0.04}$	0.7626 ± 0.08	0.7396 ± 0.05
	F1	$\mathbf{0.9285 \pm 0.02}$	0.9132 ± 0.03	0.8974 ± 0.01	$\mathbf{0.9285 \pm 0.02}$	0.9083 ± 0.04	0.8974 ± 0.01
	Spec	$\mathbf{0.6406 \pm 0.06}$	0.4688 ± 0.06	0.5703 ± 0.13	$\mathbf{0.6406 \pm 0.06}$	0.6016 ± 0.19	0.5703 ± 0.13
	Sens	0.9513 ± 0.03	$\mathbf{0.9619 \pm 0.07}$	0.9089 ± 0.04	$\mathbf{0.9513 \pm 0.03}$	0.9237 ± 0.08	0.9089 ± 0.04
	Brier	$\mathbf{0.1150 \pm 0.04}$	0.1433 ± 0.04	0.1633 ± 0.01	$\mathbf{0.1150 \pm 0.04}$	0.1450 ± 0.05	0.1633 ± 0.01
90/10	*Savings*	0.2317 ± 0.12	0.1687 ± 0.13	$\mathbf{0.3616 \pm 0.11}$	0.2317 ± 0.12	0.1911 ± 0.16	$\mathbf{0.3616 \pm 0.11}$
	AUC	$\mathbf{0.6752 \pm 0.03}$	0.6230 ± 0.05	0.6420 ± 0.05	$\mathbf{0.6752 \pm 0.03}$	0.6588 ± 0.06	0.6420 ± 0.05
	F1	$\mathbf{0.9458 \pm 0.01}$	0.9457 ± 0.00	0.8945 ± 0.04	$\mathbf{0.9458 \pm 0.01}$	0.9438 ± 0.01	0.8945 ± 0.04
	Spec	0.3824 ± 0.06	0.2647 ± 0.10	$\mathbf{0.4118 \pm 0.13}$	0.3824 ± 0.06	0.3382 ± 0.13	$\mathbf{0.4118 \pm 0.13}$
	Sens	0.9680 ± 0.01	$\mathbf{0.9812 \pm 0.06}$	0.8722 ± 0.08	0.9680 ± 0.01	$\mathbf{0.9793 \pm 0.06}$	0.8722 ± 0.08
	Brier	$\mathbf{0.0983 \pm 0.02}$	0.1000 ± 0.01	0.1800 ± 0.06	$\mathbf{0.0983 \pm 0.02}$	0.1133 ± 0.01	0.1800 ± 0.06

Table A2: This table displays the results of tests on synthetic data with an outlier of size 1000 and an outlier of size 10000. We apply a 2×5-fold cross validation procedure with a train/test split ratio of 0.8/0.2. We report the average together with the standard deviation over these 10 runs. Per row, the two classes become more and more imbalanced. The best performing methods are indicated in bold. In terms of *Savings*, r-cslogit always outperforms the other two methods and remains stable after increasing the size of the outlier. Also in terms of cost-insensitive metrics, the performance of r-cslogit remains stable. After increasing the outlier size, cslogit performs worse. This is analogous to the results as displayed in Figure 5. Logit performs best in terms of cost-insensitive metrics and, given its cost-insensitive nature, its performance remains stable after increasing the size of the outlier. The few times that logit is outperformed by either cslogit or r-cslogit in terms of cost-insensitive metrics, the performance scores have a rather high volatility. This is predominantly the case for tests with high class imbalance.