Cost Matters: A New Example-Dependent Cost-Sensitive Logistic Regression Model

Nikou Günnemann(✉) and Jürgen Pfeffer

Technical University of Munich, Munich, Germany
{Nikou.Guennemann,Juergen.Pfeffer}@tum.de

Abstract. Connectivity and automation are evermore part of today's cars. To provide automation, many gauges are integrated in cars to collect physical readings. In the automobile industry, the gathered multiple datasets can be used to predict whether a car repair is needed soon. This information gives drivers and retailers helpful information to take action early. However, prediction in real use cases shows new challenges: misclassified instances have not equal but different costs. For example, incurred costs for not predicting a necessarily needed tire change are usually higher than predicting a tire change even though the car could still drive thousands of kilometers. To tackle this problem, we introduce a new *example-dependent cost sensitive prediction model* extending the well-established idea of logistic regression. Our model allows different costs of misclassified instances and obtains prediction results leading to overall less cost. Our method consistently outperforms the state-of-the-art in example-dependent cost-sensitive logistic regression on various datasets. Applying our methods to vehicle data from a large European car manufacturer, we show cost savings of about 10%.

1 Introduction

Automation has become of prime importance to improve the quality of our life. An example from the vehicle industry, where predictive maintenance [17] looms large, is to predict whether the tires of a car need to be changed soon. Goals are (i) providing customers services with less latency for tire change, and (ii) forecasting tire delivery in correct number for all customers with a tire change need.

Given historical data, such potential 'malfunctions' (required tire change) can be predicted based on binary classification algorithms like logistic regression, support vector machines, ARIMA models or neural networks etc. [2–5]. Such approaches, however, often do not meet the real world use cases since intuitively they try to minimize the so called zero-one loss with the assumption that all misclassified instances have equal cost. Meaning correct classifications lead to a cost of zero and misclassification gets a cost of one [6].

In many applications, however, the costs for misclassified instances might vary significantly from one instance to the other. Predicting tire change is ranked among these applications. The cost associated with, e.g., an incorrect early tire

J. Kim et al. (Eds.): PAKDD 2017, Part I, LNAI 10234, pp. 210–222, 2017.
DOI: 10.1007/978-3-319-57454-7_17

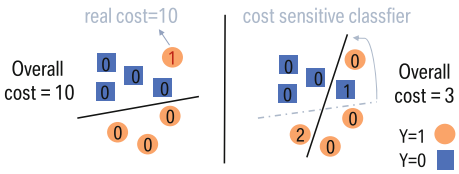

Fig. 1. Left: result of a cost-insensitive classifier; right: result of a cost-sensitive classifier with smaller overall cost

	True Pos. ($y_i = 1$)	True Neg. ($y_i = 0$)
Predicted Pos. ($\hat{y}_i = 1$)	0	${c_i}^{FP}$
Predicted Neg. ($\hat{y}_i = 0$)	${c_i}^{FN}$	0

Fig. 2. Cost matrix for cost-sensitive classification

change prediction is smaller than the expected cost for not predicting an imminent tire change at all, which could even cause customer dissatisfaction. Also, instances where the tire change has been predicted too early might show different costs. In some cases the tires might actually be used for further 20,000 km compared to only 1,000 km. Hence, in practical applications different misclassified instances can cause different costs. Since standard binary classifiers are not suited for such scenarios, *example-dependent cost-sensitive classification* [7] has been introduced, considering the different costs of instances during learning.

Technically, to distinguish between different misclassified instances, a predefined cost value can be assigned to each instance in the dataset. Figure 2 shows the cost for an instance according to their actual class vs. their predicted class. When the instance is a false positive, we have cost of c_i^{FP}, if it is a false negative, the cost is c_i^{FN}. If the instance is correctly classified (i.e. true positive or true negative), we assign a cost of zero. Note that each instance i might get different cost (indicated by the index i). Having assigned costs to each instance i, a possible way to estimate the overall misclassification cost is

$$Cost = \sum_{i}^{m} y_i \cdot (1 - \hat{y}_i) \cdot c_i^{FN} + (1 - y_i) \cdot \hat{y}_i \cdot c_i^{FP} \tag{1}$$

where $y_i \in \{0, 1\}$ is the observed and $\hat{y}_i \in \{0, 1\}$ the predicted label of instance i, with m as the number of overall instances [3]. Accordingly, instead of considering each instance equally, our goal is to train a classifier that takes the overall misclassification cost into account. The benefit of such an approach is shown in Fig. 1. On the left, the cost-insensitive classifier misclassified only one instance but its associated cost is very high (10). On the right, the potential result of a cost-sensitive classifier is shown. Although now two instances are classified false, they have only very low cost of 1 and 2; the previously misclassified instance with cost of 10 is classified correctly. Thus, the overall misclassification cost is only 3. Based on this motivation, in this work, we focus on the well established classification model of logistic regression – and we extend it by the principle of example-dependent cost sensitive learning. The contributions of this paper are:

– We propose an enhanced binary classification model that includes the individual costs of instances while fitting a model to the training set – thus, costs are not simply used in a post-processing step but during training.

– We propose four different variants of our model each extending the sound principle of logistic regression but considering different properties.
– We perform experiments on multiple real-world datasets including data from a leading European car manufacturer showing that our methods successfully lower misclassification costs.

2 Background

In a binary classification problem, input vectors $X = \{x_1, \ldots, x_m\}$ with $x_i \in \mathbb{R}^d$, and class labels $Y = \{y_1, \ldots, y_m\}$ with $y_i \in \{0, 1\}$ are given. Here, x_i is the i-th instance described by d features, and $y_i = 1$ represents the class of instances with a certain issue (tire change required) and $y_i = 0$ the opposite. In our scenario, each instance i is associate with a certain predefined cost c_i.[1] The higher the cost, the worse is a potential misclassification of this instance. Our goal is to fit a model on observed data X to predict Y denoted by $\hat{Y}$ at best. More precisely, our aim is to find a model that leads to small overall misclassification cost.

2.1 Logistic Regression and Important Properties

Logistic regression treats the binary classification problem from a probabilistic perspective. Given the instance x, the probability of the occurrence of an issue (i.e. $y = 1$) is denoted by $p(y = 1 \mid x)$, and $p(y = 0 \mid x) = 1 - p(y = 1 \mid x)$ respectively for $y = 0$. Here, $p(y = 1 \mid x)$ is defined as the sigmoid function, known as logit:

$$p(y = 1 \mid x) = f(g(x, \beta)) = \frac{1}{1 + e^{-g(x,\beta)}} \tag{2}$$

where $0 \leq f(g(x, \beta)) \leq 1$ and $g(x, \beta) = \beta_0 + \sum_{j=1}^{m} \beta_j \cdot x_j$ is a linear expression of Eq. 2 including the explanatory features and the regression coefficients β. Considering the sigmoid equation, the question is how to estimate β in $g(x, \beta)$ to make $f(g(x, \beta)) = \hat{y}$ close to y? To formalize this, and assuming that the m samples in the data are independent, we can write $p(Y|X; \beta)$ as a product, leading to the following overall Likelihood function:

$$L(Y, X, \beta) = \prod_{i=1}^{m} f(g(x_i, \beta))^{y_i} \cdot (1 - f(g(x_i, \beta)))^{1-y_i} \tag{3}$$

The β in logistic regression can be obtained by maximizing Eq. 3, i.e. it corresponds to the maximum likelihood estimate. Obviously, instead of maximizing $L(Y, X, \beta)$, we can equivalently minimize the negative log likelihood given by

$$l(Y, X, \beta) = \sum_{i=1}^{m} y_i \cdot (-\log f(g(x_i, \beta))) + (1 - y_i) \cdot (-\log(1 - f(g(x_i, \beta)))) \tag{4}$$

[1] Note that we do not have to explicitly distinguish between c_i^{FP} and c_i^{FN}. If $y_i = 0$, then $c_i^{FP} = c_i$, if $y_i = 1$, then $c_i^{FN} = c_i$. For a single instance, c_i^{FP} and c_i^{FN} can never occur together.

which is the logistic loss function. Figure 3 shows the logistic loss function for $y_i = 1$ (e.g. tire change)[2]: $y_i \cdot (-\log f(g(x_i, \beta)))$. Clearly, if $f(g(x_i, \beta)) = 1$ the prediction is correct and we have zero loss. For $f \to 0$, in contrast, the loss will increase. Thus, minimizing the loss means lowering the prediction error.

Loss of Correctly Classified Instances: In logistic regression the return values of the sigmoid function are between 0 and 1. Therefore, we have no deterministic decision which samples are classified correctly and which are classified wrong. To turn the predicted probabilities into binary responses, a threshold is used. Based on the probabilistic view and as default in literature, we choose 0.5 as threshold. That is, if $f(g(x_i, \beta)) \geq 0.5$, the predicted class is 1, otherwise 0. The resulting observation is that *even for correctly classified instances the logistic loss is not zero*. This becomes obvious in Fig. 3: e.g. an instance with $f(g(x_i, \beta)) = 0.9$ has a loss of 0.05 even if it is correctly classified.

Assuming the correctly classified instances get a probability $f(g(x_i, \beta))$ uniformly random between 0.5 and 1, then the average loss of a correctly classified instance is proportional to $T_{log} := \int_{0.5}^{1} y_i \cdot (-\log f(g(x_i, \beta))) \, \mathrm{d}f \approx 0.15$.[3] Here, T_{log} can also be illustrated as the area under the 'logistic loss'-curve as shown in Fig. 3. Likewise, the incorrectly classified instances get an average loss proportional to $F_{log} := \int_{0}^{0.5} y_i \cdot (-\log f(g(x_i, \beta))) \, \mathrm{d}f \approx 0.85$ where F_{log} represents the area from 0 to 0.5. Also note that $T_{log} + F_{log} = 1$. That is, the average loss assigned to an instance (independent if correctly or incorrectly classified) is 1. Obviously $F_{log} > T_{log}$, which means that a correct prediction actually leads to smaller loss. However, the two loss terms F_{log} and T_{log} are constant and identical for each instance. That is, the standard logistic loss function does not distinguish between different losses caused by different instances with different costs.

3 Example Dependent Cost-Sensitive Logistic Regression

The above discussion leads to the core motivation of our paper: How can we adapt the logistic loss function in a sound way, so that different samples having different costs are treated differently? How can we define a loss function to make sure that instances with higher costs are more likely to be predicted correctly?

General Framework. To answer these questions, we adapt the standard logistic loss function to a cost sensitive one in four different ways. The general framework we explore in these versions is to minimize the loss function $l(Y, X, \beta)$ defined as

$$\sum_{i=1}^{m} a_i \cdot y_i \cdot (-\log f(g(x_i, \beta))^{b_i}) + a_i \cdot (1 - y_i) \cdot (-\log(1 - f(g(x_i, \beta)))^{b_i}) \quad (5)$$

[2] The case $y_i = 0$ is equivalent; only mirrored. W.l.o.g. we consider in the following only $y_i = 1$.

[3] More precise, the average loss for correctly classified instances would be $2 \cdot T_{log}$.

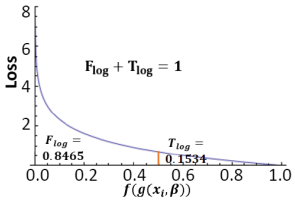

Fig. 3. Loss function of standard logistic regression for $y = 1$.

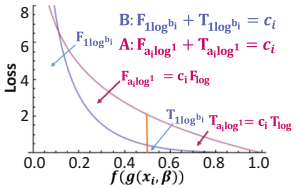

Fig. 4. Loss function for A & B. The loss ratio for B is smaller. (Color figure online)

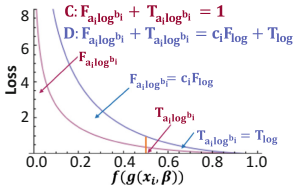

Fig. 5. Loss function for variant C & D. Both variants control the loss ratio.

where a_i and b_i depend on c_i. That is, $a_i = a(c_i)$ and $b_i = b(c_i)$ based on functions $a : \mathbb{R}^+ \to \mathbb{R}^+$ and $b : \mathbb{R}^+ \to \mathbb{R}^+$. As shown next, given different choices of a and b, we realize different properties. For convenience, let us already introduce the following notation: $F_{a_i log^{b_i}} := \int_0^{0.5} a_i \cdot (y_i \cdot (-\log f(g(x_i, \beta)))^{b_i})\ \mathrm{d}f$ represents the average loss for misclassified instances and $T_{a_i log^{b_i}} := \int_{0.5}^{1} a_i \cdot (y_i \cdot (-\log f(g(x_i, \beta)))^{b_i})\ \mathrm{d}f$ is the average loss for correctly classified instances.

Variant A: Weighting the Logistic Loss Function. The first, simplest way is to weight the logistic loss function depending on the cost value c_i by setting the area under the curve – representing the average loss – equal to the cost.

$$\int_0^1 a_i \cdot (y_i \cdot (-\log f(g(x_i, \beta))))\ \mathrm{d}f \stackrel{!}{=} c_i \tag{6}$$

This way, instances with a higher cost will get a higher average loss value. Since in standard logistic regression the area is 1, it obviously holds that the weight factor a_i needs to be equal to c_i. The variable b_i is equal to 1. The purple curve in Fig. 4 shows the plot for $c_i = 3$. Clearly, by weighting the loss function, we oversample the instances proportional to their costs, i.e. an instance with cost 2 is basically considered twice. But this solution has one drawback: By weighting the loss, not only the misclassification loss $F_{a_i log^1}$ but also the 'correct' loss $T_{a_i log^1}$ will be higher. *Correctly classified instances are penalized by this version, too.* In particular, the ratio between $F_{a_i log^1}$ and $T_{a_i log^1}$ does *not* change. Thus, for every instance a misclassification has always $\frac{F_{log}}{T_{log}} \approx 5.5$ higher loss than a correct classification. Thus, this solution might not well represent the intuition that the cost of *misclassification* will be higher.

Variant B: Logistic Loss Function to the Power of b. To avoid penalizing correctly classified instances, we exchange the weighting in Eq. 6 by an exponentiation of the logistic loss function to the power of b. That is, we increase the average loss from 1 to c_i by using the term b_i and keeping $a_i = 1$:

$$\int_0^1 y_i \cdot (-\log f(g(x_i, \beta)))^{b_i}\ \mathrm{d}f \stackrel{!}{=} c_i \tag{7}$$

Since Eq. 7 is equal to $\Gamma(b_i + 1)$, the solution for b_i given a specific c_i is equal to

$$b_i = \Gamma^{-1}(c_i) - 1 \tag{8}$$

Γ^{-1} is the inverse of the Gamma function Γ which can be computed numerically.

Figure 4 shows the corresponding loss function by the blue curve with $c_i = 3$. While the loss area $T_{1log^{b_i}}$ is pressed downwards, the loss area of instances in $F_{1log^{b_i}}$ wins on more importance since instances with high costs are more important to be classified correct. Thus, not only the average loss increases for these instances but also the ratio between $F_{1log^{b_i}}$ and $T_{1log^{b_i}}$. A potential drawback is that the ratio $F_{1log^{b_i}}/T_{1log^{b_i}}$ is not controlled explicitly.

Variant C: Controlling the Ratio - I. We aim to control the ratio between the loss area of $F_{a_i log^{b_i}}$ and $T_{a_i log^{b_i}}$. That is, for an instance with cost c_i we want to ensure $\frac{F_{a_i log^{b_i}}}{T_{a_i log^{b_i}}} \stackrel{!}{=} \frac{F_{log}}{T_{log}} \cdot c_i$. The ratio between the loss of false and correct classification is c_i times higher than for an instance with cost 1. Simultaneously, the average loss of the instances should be independent of c_i. The motivation is that in average each instance is equally important, but for some of them the *misclassification* should be penalized stronger. That is, the area under the curve has to be equal to 1, meaning $F_{a_i log^{b_i}} + T_{a_i log^{b_i}} \stackrel{!}{=} 1$. This constraint implies that

$$a_i = \frac{1}{\Gamma(b_i + 1)} \tag{9}$$

The value of $b_i > 0$ can be computed numerically by solving (see Appendix)

$$\frac{\Gamma(b_i + 1)}{\Gamma(b_i + 1, 0.6931)} = 1 + \frac{T_{log}}{c_i \cdot F_{log}} \tag{10}$$

where $\Gamma(s, r)$ is the incomplete gamma function. The effect of this variant is shown in Fig. 5 as variant C, again for $c_i = 3$. Here, we have $F \approx 0.94$ and $T \approx 0.057$. Thus, the ratio is c_i times higher than in the standard case. Still the average loss is identical (i.e. equal to 1).

Variant D: Controlling the Ratio - II. In version C, we kept the average loss at 1 but increased the ratio between false and correct classification; thus, the area $T_{a_i log^{b_i}}$ needs to decrease. Accordingly, instances with a high costs will not only have higher *misclassification loss* but also lower *correct classification loss* compared to instances with low costs – which again, might not be intended since the costs for correct classification is constant. Therefore, we introduce our last version which (i) directly controls the ratio, and (ii) ensures that the correct classification loss stays constant. This idea can be transformed to $\frac{F_{a_i log^{b_i}}}{T_{a_i log^{b_i}}} \stackrel{!}{=} \frac{F_{log}}{T_{log}} \cdot c_i$ and $T_{a_i log^{b_i}} \stackrel{!}{=} T_{log}$. Solving this, we obtain for b_i the identical solution as in variant C; only the weighting a_i changes to

$$a_i = \frac{T_{log}}{\Gamma(b_i + 1) - \Gamma(b_i + 1, 0.6931)} \tag{11}$$

variant	a_i	b_i	avg. loss	ratio F/T	T
LR	1	1	1	constant	T_{log}
A	c_i	1	c_i	constant	$c_i \cdot T_{log}$
B	1	Eq. 8	c_i	adaptive	adaptive
C	Eq. 9	Eq. 10	1	$\propto c_i$	adaptive
D	Eq. 11	Eq. 10	$T_{log} + c_i \cdot F_{log}$	$\propto c_i$	T_{log}

Fig. 6. Proposed variants and their properties

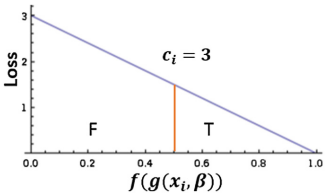

Fig. 7. Loss function used in [7].

Indeed, what we observe is that for this variant we have $F_{a_i log^{b_i}} = c_i \cdot F_{log}$. Thus, we only increase the average loss for the misclassified instances by a factor of c_i. This effect is shown in Fig. 5 as variant D. As one can also observe, the area $F_{a_i log^{b_i}}$ in variant D is equal to the area $F_{a_i log^{b_i}}$ of variant A; in both variants the area increases by a factor of c_i compared to standard logistic regression. While in variant A, however, also the area $T_{a_i log^{b_i}}$ increases, it stays constant in variant D. Thus variant D better captures the increased costs for *misclassification*.

Summary and Algorithmic Solution. Figure 6 summarizes our different variants. While variants A and B focus on increasing the average loss according to the costs, variants C and D focus on increasing the fraction between false and correct classification loss.

Our final goal is to find the parameter β that minimizes the loss function in Eq. 5: $\beta^* = \arg\min_\beta l(Y, X, \beta)$. For this purpose we exploit a gradient descent search. Starting from a random solution, we iteratively follow the steepest descent direction: $\beta^{t+1} \leftarrow \beta^t - \alpha \nabla l(\beta)$ where α is the learning rate.

4 Related Work

Various research papers are published with focus on cost sensitivity [1,7–13]. Often, the main objective is predicting potential customers with financial obligation based on their existing financial experience. While [11–14] use constant costs for misclassified instances, the authors in [9] propose a Bayes minimum risk classifier including the financial costs of credit card fraud detection in order to have a cost sensitive detection system. Another interesting approach is introduced in [15], by presenting a taxonomy of cost-sensitive decision tree algorithm using the class-dependent cost. An extension of [15] with focus on example-dependent cost for decision trees is published by [16].

The only method similar to ours is [7], which proposes an example-dependent cost sensitive logistic regression. Here the loss function of logistic regression is changed to a cost sensitive one by integrating the cost as a factor into its calculation. A drawback of [7] is that the loss function is no longer a logarithmic function but linear. That is, for the case that correct classification has 0 cost, [7] uses $\frac{1}{m} \sum_i^m y_i(1 - f(g(x_i, \beta)))c_i + (1 - y_i) f(g(x_i, \beta))c_i$. Thus, the loss decreases linearly: starting from c_i to 0 (see Fig. 7). Using a linear loss function causes weak differentiation between false and correctly classified instances. The two areas marked by F and T in Fig. 7 show this problem. As we will see in our experimental analysis, this principle will often perform worse than our technique.

5 Experimental Analysis

In this section we compare our four variants A–D with standard logistic regression LR and the competition model proposed in [7]. For this purpose, we test our designed models on the basis of three different datasets: (i) a vehicle dataset from a large European car manufacturer for predicting tire change service, (ii) the dataset breast cancer[4] to predict whether a patient is affected by breast cancer or not, and (iii) data from the 2011 Kaggle competition Give Me Some Credit[5] to predict whether a customer will experience financial distress in the next two years.

Our main goal is to achieve low overall misclassification cost (see. Eq. 1). Thus, a technique is successful if it obtains the lowest overall cost. As an evaluation measure we compute the savings of our techniques w.r.t. logistic regression $savings = \frac{Cost_{LR} - Cost_x}{Cost_{LR}}$ where $Cost_{LR}$ is the obtained misclassification cost (Eq. 1) based on the result of logistic regression and $Cost_x$ the cost based on the result of the technique x. In each scenario we used $\frac{2}{3}$ of the data for training our models and $\frac{1}{3}$ to evaluate them.

5.1 Tire Change Service

The vehicle dataset is a binary classification dataset containing 1,800 instances, each with 40 features. The features are indirectly influenced by tire wear and which, thus, indicate a resultant tire change. An example of such features could be acceleration.[6] Important to mention is that features resulting through, e.g. a sensor which directly measures the tire tread to asses a tire wear are not considered here. The target variable is whether a vehicle needs a tire change: yes = 1 or no = 0. Instances requiring a tire change account to $\sim 15\%$ amount of the whole data. Each instance is assigned with a cost; the higher the cost value, the more urgently a tire change is needed. The degree of urgency was determined by the domain experts.

Figure 8 shows the results. Here, the threshold to cast the predicted probabilities to binary responses is set to 0.5. Generally all of our four versions obtain lower overall misclassification cost than traditional logistic regression. But the best savings are achieved by B, C, and D with a win of around 10%. Applying the competing variant from [7] shows even higher overall cost than our variants. As discussed, this is caused by the used non-discriminative loss function in their logistic regression model.

While a threshold of 0.5 is from a probabilistic view the correct one, it might, however, not lead to smallest misclassification cost. Thus, in Fig. 9 we report the results for each method when individually using the threshold that leads to lowermost misclassification cost. These 'optimal' thresholds are given in Table 1. Note that in practice such a tuning is not possible since we a-priori do not know the true class of an instance.

[4] https://goo.gl/U2Uwz2.

[5] http://www.kaggle.com/c/GiveMeSomeCredit/.

[6] Due to nondisclosure agreements we unfortunately can not provide more details on the dataset. The two other datasets studied in this work are publicly available.

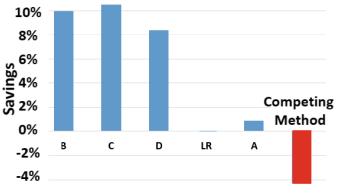

Fig. 8. Savings of the methods when threshold is 0.5. Our techniques significantly outperform logistic regression.

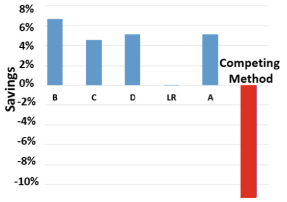

Fig. 9. Savings of the methods with variable threshold with minimal misclassification cost. Our techniques perform better

The results show that even in a scenario of complete knowledge our techniques significantly outperform logistic regression. Example dependent cost improves the ability to make less failure than the standard process. Also, the results from Table 1 are very close to $t = 0.5$. Thus, applying our models for $t = 0.5$ without tuning the threshold in practice would still obtain very good performance.

Table 1. Variable thresholds leading to minimal costs in vehicle datasets

LR	A	B	C	D	Competing
0.4542	0.4509	0.4950	0.5060	0.4775	0.5001

5.2 Breast Cancer

The Wisconsin breast cancer dataset is a binary classification dataset, e.g., available in scikit-learn. The total number of instances is 569, each with 30 attributes. 212 ($\sim$ 37.2%) of records are malignant denoted by 1 and 357 ($\sim$ 62.8%) records are benign presented by 0. Since the instances are not presented by different costs we randomly assigned each instance a cost between 1 to 5 to guaranty the fairness. To obtain reliable results, we generated $n = 10$ such datasets.

Table 2 shows the number of correctly classified instances and false classified ones (on avg. on the test data). Also, the average corresponding misclassification cost of the false classified instances are presented by the column *cost*. The left part of the table presents results for $t = 0.5$, the right part shows the results for the 'optimal'/tuned thresholds.

For $t = 0.5$, versions B and D return the lowest overall cost with a small number of false classified malignant instances. Surprisingly, not only the cost is lower in our variants, but also the classification accuracy increases. The same behavior can be seen for the variable threshold. In comparison to the competing variants LR and [7] our results are much better.

Figures 10 and 11 show the relative savings of the techniques w.r.t. logistic regression. Since we applied the algorithms 10 times based on different randomly assigned costs, different savings are observed. In Fig. 10, the bars show the mean savings for $t = 0.5$ achieved by each algorithm; the black lines represent the standard deviation over the 10 runs. In average, D as well as B save at most whereas the model in [7] cause even more loss than standard LR. *As shown by*

Table 2. Wisconsin breast cancer dataset. Left: threshold $t = 0.5$; right: variable thresholds with overall minimum misclassification cost.

Method, t	Incorrect	Correct	Avg. cost	Method, t	Incorrect	Correct	Avg. cost
LR, 0.5	≈ 17	≈ 171	52.61	0.39	≈ 14	≈ 174	38.07
A, 0.5	≈ 13	≈ 175	40.29	0.52	≈ 11	≈ 177	36.23
B, 0.5	≈ **12**	≈ **176**	**36.96**	0.4857	≈ **11**	≈ **177**	**33.37**
C, 0.5	≈ 14	≈ 174	43.72	0.496	≈ 13	≈ 175	38.61
D, 0.5	≈ **12**	≈ **176**	**37.21**	0.496	≈ **11**	≈ **177**	**34.02**
Competing, 0.5	≈ 87	≈ 101	282.8	0.499	≈ 38	≈ 150	115.49

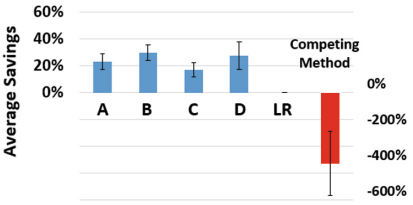

Fig. 10. Average savings when threshold t = 0.5. Average over 10 runs.

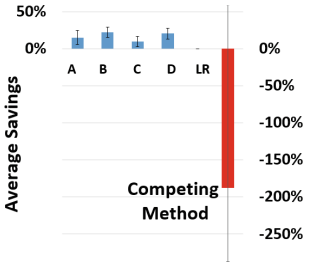

Fig. 11. Average savings when selecting variable t with minimal misclassification cost (see. Table 2). Average over 10 runs.

the black lines, these results are significant. While all our 4 versions return very similar good results, [7] shows bad performance and strong fluctuation. A similar behavior can be considered in Fig. 11 for variable thresholds.

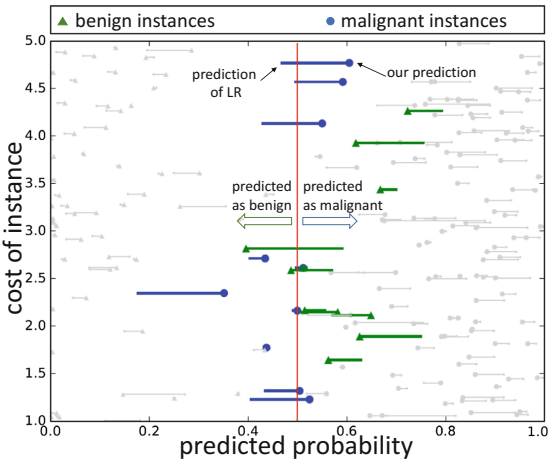

Fig. 12. Comparison between LR and D. The predicted probabilities of D (big end of each line) reflect better the true labels. Triangles should be on the left; circles on right. (Color figure online)

Finally, Fig. 12 shows why our techniques (here variant D) perform better than LR, that is we show the difference in their prediction. Each line in Fig. 12 represents one instance from the dataset. The start of each line indicates the predicted probability based on LR, while the end (shown by a circle/triangle) the probability assigned by our model. In an optimal classification, all triangles (green) should be on the left of the threshold line ($t = 0.5$); all circles (blue) on the right. We *grayed out* all instances which are correctly classified by both techniques;

thus, the *colored lines* show the interesting differences between LR and model D. As we can see, our method pushes more instances to the correct side of the line, i.e. classifies them correctly. Specifically, we also observe these changes for instances with high costs, thus, leading to overall lower misclassification cost.

5.3 Credit Datasets

The Kaggle Credit dataset is a bi-class dataset containing 112,915 credit borrowers as instances. Each instance has 10 features with a proportion of 6.74% positive examples. Some features are, e.g., monthly income or monthly debt. Our goal is to perform credit score prediction using our different versions of logistic regression. For assigning instances different costs, we took the same costs proportion as in [7]. Table 3 shows the corresponding results. In contrast to the first two experiments, the best models for Kaggle Credit data, are A and [7]. For $t = 0.5$, indeed version A has the best accuracy of around 93.0% but it saves slightly less than [7]. In contrast, using the optimal threshold, version A wins more on savings than [7] but the number of false classified instances is higher. This means that in model A, primarily instances with low costs are classified incorrectly. The model D performs good w.r.t. the savings and shows in both cases a low number of false classified instances.

Table 3. Results on credit dataset. Left: threshold $t = 0.5$; right: variable t with overall minimum misclassification cost.

Method, t	Incorrect	Correct	Cost	Savings	t	Incorrect	Correct	Cost	Savings
LR, 0.5	2470	34792	8260.35	-	0.39	2573	34689	8056.65	-
A, 0.5	**2444**	34818	7972.32	**3%**	0.38	**2586**	34676	7732.88	**4%**
B, 0.5	2474	34788	8248.05	0.1%	0.36	2483	34779	8056.65	−2%
C, 0.5	2471	34791	8261.57	−0.01%	0.40	2547	34715	8225.40	−2%
D, 0.5	2466	34796	8152.35	1%	0.33	2480	34782	7893.33	2%
Comp., 0.5	**2695**	34567	7823.53	**5%**	0.503	**2497**	34765	7766.05	**3%**

In summary, considering all datasets together, model D has consistently ranked among the best competing methods. Based on our model description (Sect. 3) it also very naturally captures example-dependent cost.

6 Conclusion

In this paper we have presented four different extensions of logistic regression to a cost sensitive one, each using a different loss functions having different properties. We evaluated the impact of each model based on two different public datasets as well as on a vehicle dataset to predict tire change. Our results confirm

that a cost sensitive model not only classifies instances with higher importance better but can also improve the accuracy of classical logistic regression. For our use case on tire change service, we obtained significant savings of 10%.

A Appendix

$$\frac{F_{a_i log^{b_i}}}{F_{a_i log^{b_i}}} = \frac{a_i \Gamma(b_i + 1, 0.6931)}{a_i(\Gamma(b_i + 1) - \Gamma(b_i + 1, 0.6931))} \stackrel{!}{=} c_i \cdot \frac{F_{log}}{T_{log}}$$

$$\Leftrightarrow \Gamma(b_i + 1, 0.6931) \stackrel{!}{=} c_i \cdot \frac{F_{log}}{T_{log}} \cdot \Gamma(b_i + 1) - c_i \cdot \frac{F_{log}}{T_{log}} \Gamma(b_i + 1, 0.6931)$$

$$\Leftrightarrow \frac{\Gamma(b_i + 1)}{\Gamma(b_i + 1, 0.6931)} \stackrel{!}{=} \frac{1 + c_i \cdot \frac{F_{log}}{T_{log}}}{c_i \cdot \frac{F_{log}}{T_{log}}} = 1 + \frac{T_{log}}{c_i \cdot F_{log}}$$

References

1. Zadrozny, B., et al.: Cost-sensitive learning by cost-proportionate example weighting. In: ICDM, pp. 435–442 (2003)
2. Günnemann, N., et al.: Robust multivariate autoregression for anomaly detection in dynamic product ratings. In: WWW, pp. 361–372 (2014)
3. Murphy, K.P.: Machine Learning: A Probabilistic Perspective. MIT Press, Cambridge (2012)
4. Quinlan, J.R.: Induction of decision trees. Mach. Learn. **1**, 81–106 (1986)
5. Haykin, S.: A comprehensive foundation. Neural Netw. **2**, 41 (2004)
6. Weiss, G.M.: Learning with rare cases and small disjuncts. In: ICML, pp. 558–565 (1995)
7. Bahnsen, A.C., et al.: Example-dependent cost-sensitive logistic regression for credit scoring. In: ICMLA, pp. 263–269 (2014)
8. Anderson, R.: The Credit Scoring Toolkit: Theory and Practice for Retail Credit Risk Management and Decision Automation. Oxford University Press, Oxford (2007)
9. Bahnsen, A.C., et al.: Cost sensitive credit card fraud detection using Bayes minimum risk. In: ICMLA, pp. 333–338 (2013)
10. Bahnsen, A.C., et al.: Improving credit card fraud detection with calibrated probabilities. In: SIAM, pp. 677–685 (2014)
11. Alejo, R., García, V., Marqués, A.I., Sánchez, J.S., Antonio-Velázquez, J.A.: Making accurate credit risk predictions with cost-sensitive MLP neural networks. In: Casillas, J., Martínez-López, F., Vicari, R., De la Prieta, F. (eds.) Management Intelligent Systems. AISC, vol. 220, pp. 1–8. Springer, Heidelberg (2013). doi:10.1007/978-3-319-00569-0_1
12. Beling, P., et al.: Optimal scoring cutoff policies and efficient frontiers. J. Oper. Res. Soc. **56**(9), 1016–1029 (2005)
13. Oliver, R.M., et al.: Optimal score cutoffs and pricing in regulatory capital in retail credit portfolios. University of Southampton (2009)
14. Verbraken, T., et al.: Development and application of consumer credit scoring models using profit-based classification measures. Eur. J. Oper. Res. **238**(2), 505–513 (2014)

15. Lomax, S., et al.: A survey of cost-sensitive decision tree induction algorithms. CSUR **45**(2), 16 (2013)
16. Bahnsen, A.C., et al.: Ensemble of example-dependent cost-sensitive decision trees (2015). arXiv preprint arXiv:1505.04637
17. Mobley, R.K.: An Introduction to Predictive Maintenance. Butterworth-Heinemann, Oxford (2002)