7

COST-SENSITIVE LOGISTIC REGRESSION

OUTLINE

In this chapter, we propose a cost-sensitive logistic regression algorithm. The model consists in a new logistic regression cost function, one that takes into account the real costs due to misclassification and correct classification. First, in Section 7.1, we give the background behind logistic regression. Then, in Section 7.2, we describe the cost-sensitive logistic regression. For this, we carry a deep analysis of the logistic regression implicit misclassification costs in Section 7.2.1. Then in Section 7.2.2, we give a new version of the logistic regression cost function. Finally, in Section 7.3, we compare the results of the proposed algorithm against state-of-the-art methods, using the five real-world cost-sensitive databases presented in Part II.

7.1 LOGISTIC REGRESSION

Logistic regression is a classification model that, in the specific context of binary classification, estimates the posterior probability of the positive class, as the logistic sigmoid of a linear function of the feature vector [Bishop, 2006]. The estimated probability is evaluated as

$$\hat{p}_i = P(y = 1|\mathbf{x}_i) = h_\theta(\mathbf{x}_i) = g\left(\sum_{j=1}^{k} \theta^j x_i^j\right), \tag{7.1}$$

where $h_\theta(\mathbf{x}_i)$ refers to the hypothesis of i given the parameters θ, the feature vector $\mathbf{x}_i$ for example i and $g(\cdot)$ is the logistic sigmoid function defined as

$$g(z) = \frac{1}{(1 + e^{-z})}. \tag{7.2}$$

In Figure 7.1, the logistic sigmoid function is shown.

The problem then becomes on finding the right parameters that minimize a given cost function. Usually, in the case of logistic regression, the cost function $J(\theta)$ refers to the negative logarithm of the likelihood, such that

$$J(\theta) = \frac{1}{N}\sum_{i=1}^{N} J_i(\theta), \tag{7.3}$$

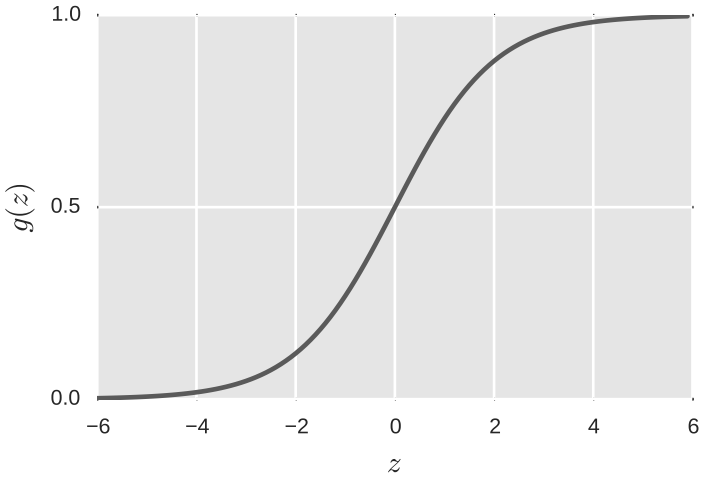

Figure 7.1: Sigmoid function

where

$$J_i(\theta) = -y_i \log(h_\theta(\mathbf{x}_i)) - (1 - y_i) \log(1 - h_\theta(\mathbf{x}_i)). \tag{7.4}$$

Therefore, the parameters are estimated by minimizing (7.4) $\hat{\theta} = \arg\min_\theta J(\theta)$.

There are several methods used to estimate the logistic regression, in particular, maximum likelihood [Hastie et al., 2009], Newton, coordinate descent [Murphy, 2012] and dual coordinate descent [Yu et al., 2011]. Nevertheless, all these methods rely on the assumption of convexity of the logistic regression cost function $J(\theta)$.

7.2 COST-SENSITIVE LOGISTIC REGRESSION

In this section we present our cost-sensitive logistic regression algorithm. First, we motivate the need to modify the logistic regression, as we analyze the implicit costs that the logistic regression assigned to each misclassification error during the estimation of the parameters θ. Then, we show our proposed algorithm.

7.2.1 *Implicit costs of the logistic regression*

The logistic regression cost function, as described in (7.4), implicitly assume that false positives and false negatives have the same costs, i.e. $C_{FP_i} = C_{FN_i}\ \forall i \in \{1, \cdots, N\}$. This can be easily shown by analyzing the logistic cost function for both values of y_i and the algorithm prediction $h_\theta(\mathbf{x}_i)$):

- If $y_i = 0$ and $h_\theta(\mathbf{x}_i) \approx 0$, then

$$\begin{aligned} J_i(\theta) &= -y_i \log(h_\theta(\mathbf{x}_i)) - (1 - y_i) \log(1 - h_\theta(\mathbf{x}_i)) \\ &\approx -(0) \log((0)) - (1 - (0)) \log(1 - (0)) \\ &\approx 0. \end{aligned}$$

- If $y_i = 0$ and $h_\theta(\mathbf{x}_i) \approx 1$, then

$$\begin{aligned} J_i(\theta) &\approx -(0) \log((1)) - (1 - (0)) \log(1 - (1)) \\ &\approx \infty. \end{aligned}$$

- If $y_i = 1$ and $h_\theta(\mathbf{x}_i) \approx 0$, then

$$\begin{aligned} J_i(\theta) &\approx -(1) \log((0)) - (1 - (1)) \log(1 - (0)) \\ &\approx \infty. \end{aligned}$$

- If $y_i = 1$ and $h_\theta(\mathbf{x}_i) \approx 1$, then

$$\begin{aligned} J_i(\theta) &\approx -(1) \log((1)) - (1 - (1)) \log(1 - (1)) \\ &\approx 0. \end{aligned}$$

Then, we collect the previous results in a cost matrix:

	Actual Positive $y_i = 1$	Actual Negative $y_i = 0$
Predicted Positive $c_i = 1$	$C_{TP_i} \approx 0$	$C_{FP_i} \approx \infty$
Predicted Negative $c_i = 0$	$C_{FN_i} \approx \infty$	$C_{TN_i} \approx 0$

Table 7.1: Logistic regression cost matrix

This confirms that the logistic regression cost function implicitly assume that false positives and false negatives have the same costs. However, as discussed in Chapter 4 and Chapter 5, this is not the case in several real-world applications.

7.2.2 *Cost-sensitive logistic regression cost function*

In order to incorporate the different real costs, as showed in Table 3.1, into the logistic regression, we start by analyzing the expected costs that a modified logistic regression cost function should make for each misclassification and correct classification case.

$$J_i^c(\theta) = \begin{cases} C_{TP_i} & \text{if } y_i = 1 \text{ and } h_\theta(\mathbf{x}_i) \approx 1 \\ C_{TN_i} & \text{if } y_i = 0 \text{ and } h_\theta(\mathbf{x}_i) \approx 0 \\ C_{FP_i} & \text{if } y_i = 0 \text{ and } h_\theta(\mathbf{x}_i) \approx 1 \\ C_{FN_i} & \text{if } y_i = 1 \text{ and } h_\theta(\mathbf{x}_i) \approx 0. \end{cases}$$

Then, as we already have the real costs, we create a new cost-sensitive logistic regression cost function, by including the different costs into the logistic function,

$$J^c(\theta) = \frac{1}{N}\sum_{i=1}^{N}\Big(y_i(h_\theta(\mathbf{x}_i)C_{TP_i} + (1-h_\theta(\mathbf{x}_i))C_{FN_i}) + (1-y_i)(h_\theta(\mathbf{x}_i)C_{FP_i} + (1-h_\theta(\mathbf{x}_i))C_{TN_i})\Big). \quad (7.5)$$

Since this new cost function is not convex, we estimate the parameters θ with genetic algorithms, as this optimization heuristic does not require the underlying function to be differentiable or convex [Haupt and Haupt, 2004].

7.3 EXPERIMENTS

For the experiments we use five datasets from four different real world example-dependent cost-sensitive problems: Credit card fraud detection (see Section 4.1), credit scoring (see Section 4.2), churn modeling (see Section 5.1) and direct marketing (see Section 5.2). The different datasets are summarized in Table 4.7 and Table 5.3.

In particular, we are interested in comparing the results of the different logistic regression models. First we train a logisitc regression (LR) using the training (t), under-sampled (u), cost-proportionate rejection-sampling (r) [Zadrozny et al., 2003] and cost-proportionate over-sampling (o) [Elkan, 2001] datasets. Afterwards, we evaluate the results of the algorithms using BMR, see Chapter 6. Lastly, we calculate the cost-sensitive logistic regression (CSLR). We use the logistic regression implementation of *Scikit-learn* [Pedregosa et al., 2011], and the *CostCla* library, see Appendix A, for the cost-sensitive algorithms. Unless otherwise stated, the random selection of the training set was repeated 50 times, and in each time the models were trained and results collected, this allows us to measure the stability of the results.

In Table 7.2, we show the results of each algorithm in the different databases measured by savings. Firstly, the proposed CSLR has the

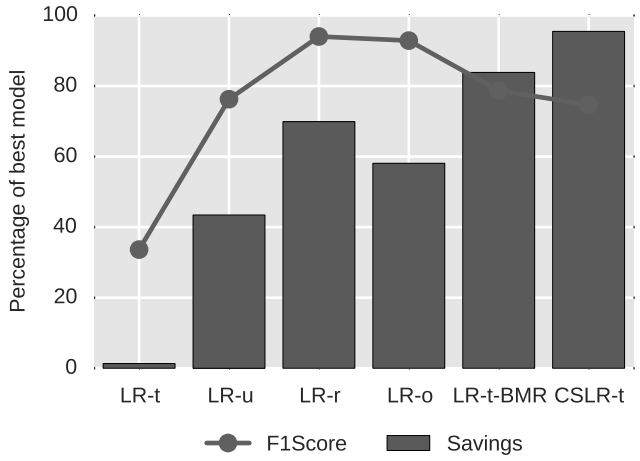

Figure 7.2: **Comparison of the average savings and F_1Score of the algorithms versus the the best model.** The models that perform the best measured by F_1Score are not the best in terms of savings.

highest savings in the fraud, churn and credit 1 databases, moreover, in the credit 2 and marketing databases, it is the second best model. It is interesting to see how different the results from a standard logistic regression and the cost-sensitive logistic regression are. Moreover, we also calculate the results of the F_1Score for each model, as shown in Table 7.3. It is observed that the CSLR algorithm is not the one that gives the best results measured by F_1Score. In fact, there is not a clear relation between the results measured by savings or F_1Score.

Finally, we compute the perBest statistic for the savings and the F_1Score. This statistic is calculated as the average result of each algorithm compared with the best in each set. The results are shown in Figure 7.2. On average, the CSLR is the best model, as it yield to 95.5% of the best model in the different databases. Nevertheless, when measured by F_1Score, it only yield to 74.7% of the best model. This leads to the conclusion of the importance of using a cost-sensitive measure such as savings when evaluating real-world example-dependent cost-sensitive problems.

Moreover, it is observed that the standard logistic regression trained using the training set is the model that performs the worst measured by both savings and F_1Score. This is related not only to the cost-sensitivity of the problems, but also to the highly unbalanced distribution of positive and negatives presented in all the databases. This is the reason why by using an under-sampling procedure, the results are improved by both measures.

Family	Algorithm	Fraud	Churn	Credit 1	Credit 2	Marketing
CI	LR-t	0.0092±0.0002	-0.0001±0.0002	0.0177±0.0126	0.0039±0.0012	-0.2931±0.0602
	LR-u	0.1243±0.0387	0.0039±0.0492	0.4118±0.0313	0.1850±0.0231	0.2200±0.0376
CPS	LR-r	0.3077±0.0301	0.0484±0.0375	0.3965±0.0263	0.2650±0.0115	0.4210±0.0267
	LR-o	0.2793±0.0185	0.0316±0.0228	0.3301±0.0109	0.2554±0.0090	0.3129±0.0277
BMR	LR-t-BMR	0.4552±0.0203	0.1082±0.0316	0.2189±0.0541	**0.3148±0.0094**	**0.4973±0.0084**
CST	CSLR-t	**0.6113±0.0262**	**0.1118±0.0484**	**0.4554±0.1039**	0.2748±0.0069	0.4484±0.0072

(Models with the highest savings are marked in bold)

Table 7.2: Results of the algorithms measured by savings

Family	Algorithm	Fraud	Churn	Credit 1	Credit 2	Marketing
CI	LR-t	0.1531±0.0045	0.0000±0.0000	0.0494±0.0277	0.0155±0.0037	0.2702±0.0125
	LR-u	0.0241±0.0163	0.1222±0.0098	0.3160±0.0314	**0.3890±0.0053**	0.3440±0.0083
CPS	LR-r	0.1846±0.0123	0.1258±0.0111	0.3597±0.0156	0.3793±0.0049	0.3374±0.0101
	LR-o	0.1776±0.0117	0.1085±0.0203	**0.3769±0.0067**	0.3804±0.0044	**0.3568±0.0102**
BMR	LR-t-BMR	0.1384±0.0044	**0.1370±0.0150**	0.1915±0.0340	0.3572±0.0045	0.2954±0.0079
CST	CSLR-t	**0.2031±0.0065**	0.1134±0.0151	0.1454±0.0517	0.3363±0.0045	0.2339±0.0051

(Models with the highest F_1 Score are marked in bold)

Table 7.3: Results of the algorithms measured by F_1 Score