Chapter 4
Cost-Sensitive Learning

Abstract Cost-sensitive learning is an aspect of algorithm-level modifications for class imbalance. Here, instead of using a standard error-driven evaluation (or 0–1 loss function), a misclassification cost is being introduced in order to minimize the conditional risk. By strongly penalizing mistakes on some classes, we improve their importance during classifier training step. This pushes decision boundaries away from their instances, leading to improved generalization on these classes. In this chapter we will discuss the basics of cost-sensitive methods, introduce their taxonomy, and describe how to deal with scenarios in which misclassification cost is not given beforehand by an expert. Then we will describe most popular cost-sensitive classifiers and talk about the potential for hybridization with other techniques. Section 4.1 offers background and taxonomy of cost-sensitive classification algorithms. The important issue of how to obtain the cost matrix is discussed in Sect. 4.2. Section 4.3 describes MetaCost, a popular wrapper approach for adapting any classifier to a cost-sensitive setting, while Sect. 4.4 discusses various aspects of cost-sensitive decision trees. Other cost-sensitive classification models are described in Sect. 4.5, while Sect. 4.6 shows the potential advantages of using hybrid cost-sensitive algorithms. Finally Sect. 4.7 concludes this chapter and presents future challenges in the field of cost-sensitive solutions to class imbalance.

4.1 Introduction

Cost-sensitive learning refers to a specific set of algorithms that are sensitive to different costs associated with certain characteristics of considered problems. These costs can originate from various aspects related to a given real-life problem and be provided by a domain expert, or learned during the classifier training phase. Two distinct views on cost-sensitive classifiers exist in the literature. On the one hand ‘cost associated with features and, on the other hand cost associated with classes.

1. **Cost associated with features.** This scenario assumes that acquiring a certain feature is connected with a given cost, being also known as test cost [47].

A. Fernández et al., *Learning from Imbalanced Data Sets*,
https://doi.org/10.1007/978-3-319-98074-4_4

This can be viewed from the monetary perspective (e.g., a feature is more expensive to extract, as it requires additional resources or laboratory tests), time perspective (e.g., a feature takes more time to extract and therefore may cause a bottleneck in the classification system), or other difficulties perspective (e.g., obtaining a feature involves invasive tests on humans, or the measurement procedure is unpleasant, painful, or difficult to perform) [65].

This aspect of cost-sensitive learning aims at creating a classifier that obtains the best possible predictive performance, while utilizing features that can be obtained at lowest possible cost (or the sum of costs being below a given threshold) [68].

This can be seen as a multi-objective learning, where we try to strike a balance between performance and cost of used features [29, 60]. In many cases they more costly features offer higher predictive power, leading to a problem of whether to use several cheaper features or few more expensive ones [23].

This can also be viewed as a feature selection task, but many cost-sensitive classifiers (e.g., decision trees) have the cost optimization procedure inbuilt [48].

2. **Cost associated with classes.** This scenario assumes that making errors on instances coming from certain classes causes is connected with a higher cost.

   This can be viewed from a monetary perspective (e.g., giving a credit to a person with a bad credit score will potentially cause higher loses to a bank than declining credit to a person with a good score), or priority/health/ethical issues (e.g., sending a sick patient home is much more costly and dangerous for a hospital than assigning additional tests to a healthy person) [32].

   This aspect of cost-sensitive learning aims to train a classifier in such a way that it will focus on classes that have higher costs assigned to them. They can be seen as priority ones and we want to influence the training procedure by treating them differently.

   While over the last decade cost-sensitive learning gained most attention for problems with skewed class distributions [39], it is also often used in balanced scenarios, where incorrect classification outcomes may lead to severe consequences.

In the context of class imbalance, the cost-sensitive learning can be seen as a specific type of algorithm-level approach [27, 42]. It assumes asymmetric misclassification costs between classes, defined in a form of a cost matrix. Standard machine learning methods most commonly use so-called 0–1 loss function, which assigns value 0 to a correctly classified instance and value 1 to an incorrectly classified one. Then the training procedure aims at minimizing the overall cost, i.e., minimizing the number of wrong predictions. As 0–1 loss function uses the same cost associated with a wrong classification for all classes considered, it is highly susceptible to skewed class distributions [36]. The 0–1 loss function over imbalanced data can be easily minimized by focusing on majority class and overlooking (or in extreme cases even completely ignoring) minority class. This problem is getting more prevalent with increasing imbalance ratio.

Table 4.1 Cost matrix for a two-class problem

	True positive	True negative
Predicted positive	$C(0, 0)$	$C(0, 1)$
Predicted negative	$C(1, 0)$	$C(1, 1)$

Cost-sensitive learning aims at alleviate this problem by adapting a different loss function, with different costs associated with each class. Such a cost can be seen as a penalty factor introduced during a classifier training procedure (or in some cases during prediction step), aiming at increasing the importance of difficult (e.g., minority) classes. By stronger penalization of errors on a given class, we force the classifier training procedure (aiming to minimize the overall cost) to focus on instances coming from this distribution. An example of a cost matrix for a two-class problem is given in Table 4.1.

With a provided cost matrix, a new instance should be classified as the one belonging to a class characterized by the lowest expected cost. This is known as the minimum expected cost principle. The expected cost (conditional risk) $R(i|x)$ of classifying instance x as belonging to i-th class can be expressed as:

$$R(i|x) = \sum_{j=1}^{M} P(j|x) \cdot C(i, j), \tag{4.1}$$

where $P(j|x)$is the probability estimation of classifying instance x as belonging to class j from a set of M classes.

For a standard two-class problem a cost-sensitive classifier will classify given instance x as belonging to positive class if and only if:

$$P(0|x) \cdot C(1, 0) + P(1|x) \cdot C(1, 1) \leq P(0|x) \cdot C(0, 0) + P(1|x) \cdot C(0, 1), \tag{4.2}$$

which is equivalent to:

$$P(0|x) \cdot (C(1, 0) - C(0, 0)) \leq P(1|x) \cdot (C(0, 1) - C(1, 1)) . \tag{4.3}$$

This shows that any cost matrix can work under an assumption that $C(0, 0) = C(1, 1) = 0$ (and analogically for multi-class problems). This allows to reduce the number of cost parameters to be established, as one is only interested in misclassification cost among classes.

Following this assumption, a cost-sensitive classifier will classify given instance x as belonging to positive class if and only if:

$$P(0|x) \cdot C(1, 0) \leq P(1|x) \cdot C(0, 1). \tag{4.4}$$

By following the fact that $P(0|x) = 1 - P(1|x)$, we may obtain a threshold $p*$ for classifying an instance x as belonging to positive class if $P(1|x) \geq p*$, where:

$$p* = \frac{C(1,0)}{C(1,0) - C(0,1)}. \tag{4.5}$$

Cost-sensitive learning algorithms can be separated into two main groups:

- **Direct approaches.** This methodology is based on directly introducing the misclassification cost into the training procedure of a classifier. This directly corresponds to other algorithm-level approaches, with difference of utilizing the cost matrix.
- **Meta-learning approaches.** This methodology is based on modifying either the training data or the outputs of a classifier. It does not modify the underlying training algorithm, thus making this a suitable approach for almost any type of a classifier. Meta-learning solutions can be applied during two different steps of the classification process:
  - **Preprocessing.** Here we aim at modifying the training set, similarly to data-level solutions discussed in previous chapters. The most popular approach includes weighting the instances according to a provided cost matrix, thus allowing for assigning higher importance to minority objects.
  - **Postprocessing.** Here we aim at modifying the outputs of a classifier during the classification phase. It does not involve any modification before or during training and the entire effort is moved to introducing the cost factor when a decision about a new instance is being made.

4.2 Obtaining the Cost Matrix

The effectiveness of cost-sensitive learning relies strongly on the supplied cost matrix. Parameters provided there will be of crucial importance to both training and predictions steps. Incorrectly initialized costs can impair the learning process [64]. Too low costs will not allow to properly adjust the classification boundaries, while too high cost will lead to loss of generalization capabilities on the remaining classes. In case of class imbalance wrongly set costs can actually mirror a bias towards the majority class into a bias towards minority class – while we should aim to get a balanced performance on both of them. But how does one obtain such a cost matrix? There are two possible scenarios:

1. **Cost matrix provided by an expert.** In this case the supplied data is accompanied by the cost matrix that comes directly from the nature of a problem. This usually requires an access to a domain expert that can assess the most realistic cost values. As an example of an application with a predefined cost matrix we may take credit card fraud detection [45]. Here cost is given directly as an average monetary loss in a case of accepting a fraudulent transaction (this is $C(i, j)$) and in case of losing a customer after rejecting a valid transaction (this is $C(j, i)$).

2. **Cost matrix estimated using training data.** In many cases we do not have an access to a domain expert and no a priori information on cost matrix is available during classifier training. This is a common scenario when we want to apply cost-sensitive learning as a method for solving imbalanced problems, especially over a wide range of different datasets. This requires either heuristic setting of cost values or learning them from training data.

The most popular heuristic approach lies in utilizing the IR as a direct way to estimate the cost. In this set-up $C(i, j) = \text{IR}$ and $C(j, i) = 1$, where minority is the i-th and majority is the j-th class (to allow for cases with multiple classes). The reasoning behind this approach is that the higher IR the more difficult the learning problem. While this is very easy to apply and gives the cost matrix very quickly, one must be aware of significant limitations of it. The major one lies in the fact that IR is not the sole source of learning difficulties in imbalanced data [27]. One must take into an account instance-level characteristics, such as small sample size, class overlapping [56], or presence of noisy and borderline instances [58]. Therefore, two problems with similar IR may pose drastically different challenges to a classifier and using similar cost matrices for them will be an oversimplification.

Popular way of training a cost-sensitive classifier without know cost matrix is to put emphasis on modifying the classification outputs when predictions are being made on new data. This is usually done by setting a threshold on the positive class, below which the negative one is being predicted. The value of this threshold is optimized using a validation set and thus the cost matrix can be learned from training data [18]. This approach has been criticized for creating a division between training and cost-sensitive evaluation, as the trained classifier in the first phase (when costs are unknown) is error-driven and not cost-driven [6]. Therefore, the estimation of the cost parameters is initialized with a method that is not cost-sensitive and the outcome may be biased.

This problem has been addressed by incorporating ROC-based criterion for classifier training. As ROC analysis allows to handle performance on both classes simultaneously, it is a highly suitable tool for cost-sensitive learning. Values of cost matrix are then found using a ROC space with iso-performance lines [17]. The best threshold (cost parameter) is defined by the operating point for which the iso-performance line is tangent to the ROC curve. ROC-based training can be easily applied to various classification algorithms, making their cost-sensitive adaptations possible in scenarios without explicitly stated cost matrix [19, 46]. At the same time, we must remember that ROC-based cost tuning is sub-optimal, as there are no guarantees for the obtained classifier to be optimal over all possible misclassification costs.

This problem can be alleviated by using an ensemble-based strategy and training a pool of classifiers, where each individual one is being specialized in a given misclassification cost settings. Additionally, this allows to predict multiple potential scenarios in the prediction phase, allowing for handling cases where testing set has different properties than training set (which is known as dataset shift). Then we may select single best classifier that is most suitable for discovered cost matrix, or

combine more classifiers to achieve better classification performance by exploring diversity among their cost settings [31]. One of the first ensemble approaches was based on generating a grid of cost pairs and training an individual classifier on each of them [3]. A multi-objective genetic algorithm is applied for optimizing base classifiers with respect to estimated cost matrix. This idea was extended as ROC Front [8], where authors proposed to use multi-objective optimization to train ensemble of SVMs by adapting hyperparameter values to the misclassification cost. An interesting alternative was proposed in [53], where authors used Precision-Recall Operating Characteristic (P-ROC).

ROC-based tuning of cost matrix has also been considered in multi-class scenarios, by working in a $M \times (M - 1)$ dimensional ROC space (for M classes). Then weights are assigned to individual outputs per class in order to control the decision making process and make it cost-sensitive. However, the number of possible weight combinations with different values of thresholds becomes computationally prohibitive [5, 35]. Therefore, using the divide-and-conquer approach with a pairwise combination of classes attracted more attention [26]. This approach has been used for Volume Under the Surface estimation [21], as well as for optimization of parameters used in classifier training [33, 34]. Ensemble-based approaches for multi-class cost matrix estimation are not that popular, yet one should point out to very interesting works by Everson and Fieldsen [16], as well as by Bernard et al. [4]

4.3 MetaCost

MetaCost algorithm [12] will be discussed firstly, due to its unique flexibility in adapting classifiers to cost-sensitive scenarios. It works as a wrapper method that can be applied to any type of classifier, regardless of the type of output it returns (either class labels or probability estimates). This makes it stand out from the remaining algorithms discusses in this chapter, as they are focusing on modifying only a specific type of classifiers.

MetaCost is based on the assumption that with the introduction of a cost matrix, the classification boundary should be adapted in favor of the classes with a higher cost assigned. This translates to expanding regions in the decision space assigned to these classes, even if the a priori probabilities do not change. Hence, class labels provided for the instances in the training set may in fact coincide with optimal predictions for them according to a provided cost matrix. MetaCost postulates that if these instances would be relabeled to their optimal classes suggested by the cost matrix, then there is no further need for data preprocessing and a standard classifier using 0–1 loss function can be used. Modified training set should allow any classifier to find optimal decision boundaries that will minimize the misclassification cost.

MetaCost is a preprocessing meta-learning approach that utilizes ensemble-based data manipulation [20]. The original training set is used to learn multiple classifiers by bootstrapping instances (following Bagging idea). Then a probability for each instance belonging to each of classes is being estimated using a fraction of votes

Algorithm 1 MetaCost algorithm

Require: S: Training set; S': relabeled training set; L: Number of Bagging iterations; n: Bootstrap size; Ψ: classification algorithm; M: # of classes.

for $l = 1$ to L **do**
  $S_l \leftarrow$ RandomSampleReplacement(n,S)
  $\Psi_l \leftarrow$ train Ψ on S_l
end for

for $s = 1$ to $|S|$ **do**
  for $i = 1$ to M **do**
    if Ψ returns probabilistic outputs **then**
      obtain $P(i|x_s, \Psi_l)$
    else
      $P(i|x_s, \Psi_l) = 1$ for the class predicted and 0 for others
    end if
    if use all bootstraps for each instance **then**
      l ranges over all Ψ_l
    else
      l ranges over all Ψ_l such that $x_s \notin S_l$
    end if
    $P(i|x_s) = \frac{1}{\sum_l 1} \sum_l P(i|x_s, \Psi_l)$
  end for
  $x'_s \leftarrow$ relabel x_s according to $\text{argmin}_l \sum_j^M P(m|x_s)C(i, j)$
  $S' \leftarrow S' \cup x'_s$
end for

$\Psi_{final} \leftarrow$ train Ψ on S'

it receives from the ensemble. Then training instances are relabeled to minimize the conditional risk (see Eq. 4.1). Finally, the ensemble is being discarded (as it was used only for preprocessing step) and a new classifier is being trained on the modified set of instances. The pseudo-code of MetaCost is given in Algorithm 1.

4.4 Cost-Sensitive Decision Trees

Among all of the classifiers, induction of cost-sensitive decision trees has arguably gained the most attention [13, 37, 52, 57]. This can be attributed to the ease of modification of their training [38, 59] and pruning algorithms [14, 25, 66], as well as plethora of ways to apply meta-learning principles. Let us provide general frameworks for two most important approaches to cost-sensitive decision tree induction: splitting criterion modification (Sect. 4.4.1) and instance weighting (Sect. 4.4.2).

4.4.1 Direct Approach with Cost-Sensitive Splitting

Induction of a cost-sensitive tree takes into account two different costs – test cost of a-th feature $tc(a)$ and misclassification cost of instance $mc(x)$ [38]. Both of these costs are to be minimized, allowing to eliminate the skew towards majority class, while reducing the cost of used features [40, 41]. As for many imbalanced problems we do not have costs assigned to features, then one should assume an uniform value of $tc(a)$, which will allow for the training algorithm to focus on minimization of $mc(x)$. The following five-step process allows for the computation of the average total cost associated with a given decision tree.

Step 1. Denote the inducted decision tree as T, training set as S, selected training instance as $x \in S$, and set of features describing this instance as $B(x)$.

Step 2. Calculate the test cost associated with x using the subset of features $B'(x)$ that are used by path from the root of T to one of its leaves to which x belongs:

$$tc(x) = tc\left(B'(x)\right) = \sum_{a \in S'} tc(a). \tag{4.6}$$

Step 3. Denote the set of instances in a given leaf node l as $S'(l)$ and the decision value of instance $x \in S'(l)$ as $d(x)$. Let $|S'_i(l)|$ and $|S'_j(l)|$ be the number of instances from i-th and j-th class in this l-th node. To minimize the misclassification cost, a one class $d_c(x)$ is assigned for all of instances in $S'(l)$, based on:

$$mc(S'(l)) = \min(|S'_i(l)| \times C(i, j), |S'_j(l)| \times C(j, i)). \tag{4.7}$$

Then for any $x \in S'(l)$ the assigned class is calculated as:

$$d_c(x) = \begin{cases} i\text{-th class} & \text{if } |S'_j(l)| \times C(j, i), \\ j\text{-th class} & \text{if } |S'_i(l)| \times C(i, j). \end{cases} \tag{4.8}$$

Step 4. Calculate the misclassification cost. Despite each leaf node l is now being associated with only a single class label, there still may be instances within it that have their true class labels different. We denote the true class label of x as $d_t(x)$, while the label assigned to it by a given leaf node as $d_c(x)$. For each instance x in S' that has different true class label than the label associated with this node, calculate:

$$mc(x) = \begin{cases} C(i, j) & \text{if } d_t(x) = i \text{ and } d_c(x) = j, \\ C(j, i) & \text{if } d_t(x) = j \text{ and } d_c(x) = i. \end{cases} \tag{4.9}$$

Step 5. Calculate the average total cost ATC that takes into account both the cost associated with misclassified instances ($mc(x)$) and features used by them

$(tc(x))$. This metric is calculated for the entire training set U:

$$ATC(U) = \frac{\sum_{x \in U} (tc(x) + mc(x))}{|S|}. \tag{4.10}$$

This approach can be used with any splitting measure that has been adapted to take into account feature costs.

4.4.2 Meta-learning Approach with Instance Weighting

An alternative solution to using the cost directly when creating splits lies in weighting the training instances [59]. Higher weights is assigned to instances coming from the class with higher value of misclassification cost. This is done instead of sampling procedures, thus the size of the training set is not altered. The following four-step process allows for modifying any decision tree induction scheme to take into account misclassification costs expressed as weighted instances.

Step 1. Convert the cost matrix into a cost parameter for each class. For a two-class problem $C_i = C(i, j)$ and $C_j = C(j, i)$, while for multi-class problems with M classes one may use the following conversion:

$$C(i) = \sum_{j}^{M} C(i, j). \tag{4.11}$$

Step 2. Calculate weight associated to instances coming from i-th class:

$$w_i = C_i \frac{|S|}{\sum_j C_j |S_j|}, \tag{4.12}$$

where $|S_j|$ is the number of training instances belonging to j-th class and $\sum_i w(i)|S_i| = |U|$. For $C(i) \geq 1$, w_i takes the smallest value bounded within $0 \leq \frac{|S|}{\sum_j C_j |S_j|}, \leq 1$ when $C_i = 1$, and takes the following largest value when $C_i = \max_j C_j$:

$$w_i = \frac{C_i |S_i|}{\sum_j C_j |S_j|} \geq 1. \tag{4.13}$$

Step 3. Calculate the ratio of the total weight of i-th class in leaf node l to the overall total weight of all instances in l:

$$p_w(i|l) = \frac{W_i(l)}{\sum_j W_j(l)} = \frac{w_i |S'_i(l)|}{\sum_j w_j |S'_j(l)|}. \tag{4.14}$$

Step 4. Use any selected training procedure for induction decision trees. $W_i(l)$ must be used instead of $|S'_j(l)|$ when calculating splitting criterion value in each node in the tree growing process, as well as and the error estimation in the pruning process. Therefore, no algorithm-level modifications are required.

4.5 Other Cost-Sensitive Classifiers

While MetaCost and decision trees are the most popular approaches to cost-sensitive classification, plethora of other methods have also been adapted to work with varying misclassification costs [10, 15, 43]. Below we will shortly discuss the most important cost-sensitive versions of popular classification models, namely SVMs (Sect. 4.5.1), ANNs (Sect. 4.5.2), and NN classifiers (Sect. 4.5.3).

4.5.1 Support Vector Machines

SVMs can be easily adjusted to work with cost-sensitive setting [11, 24]. One of the main reasons behind their sensitivity to skewed datasets lies in soft margin objective function assigning identical costs (parameter C) for both positive and negative class. Different Error Cost (DEC) approach [61] uses the provided (or estimated) cost matrix to assign separate misclassification costs to each of classes. Therefore, for positive class we use parameter $C^+ = C(1,0)$ and for negative we use $C^- = C(0,1)$. This modifies the calculation of soft margin objective function to:

$$\begin{aligned} &\min \left(\frac{1}{2} w \cdot w + C^+ \sum_{i|y_i=+1}^{l} \xi_i + C^- \sum_{i|y_i=-1}^{l} \xi_i \right) \\ &\text{subject to } \underset{i=1,\cdots,l}{\forall} \underset{\xi_i \geq 0}{\forall} y_i \left(w \cdot \Phi(x_i) + b \right) \geq 1 - \xi_i, \end{aligned} \tag{4.15}$$

The dual Lagrangian optimization problem can be then represented as follows:

$$\begin{aligned} &\max_{\alpha_i} \left(\sum_{i=1}^{l} \alpha_i - \frac{1}{2} \sum_{i=1}^{l} \sum_{j=1}^{l} \alpha_i \alpha_j y_i y_j K(x_i, x_j) \right) \\ &\text{subject to } \underset{i=1,\cdots,l}{\forall} \underset{0 \leq \alpha_i^+ \leq C^+}{\forall} \underset{0 \leq \alpha_i^- \leq C^-}{\forall} \sum_{i=1}^{l} y_i \alpha_i = 0, \end{aligned} \tag{4.16}$$

where α_i^+ and α_i^- stand for Lagrangian multipliers for positive and negative classes. When misclassification costs are unknown DEC uses the IR to initialize C^+ and C^-.

4.5.2 Artificial Neural Networks

Cost-sensitive modifications of ANNs [67] involve alternation of weight update functions [7], sampling solutions [2], and moving threshold approaches. The latter one is a post-processing meta-learning solution to cost-sensitive learning. We denote the real-valued output of an ANN if a form of support for instance x belonging to i-th class as $O_i(x)$(for$i \in M$), where $\sum_{i=1}^{M} O_i(x) = 1$ and $0 \leq O_i(x) \leq 1$. Standard ANNs use Winner-Take-All approach for determining the final predicted class: $\text{argmax}_i\, O_i(x)$. However, moving threshold approach modifies the values of outputs by including the misclassification cost:

$$O_i^*(x) = \eta \sum_{j=1}^{M} O_i(x) C(i, j), \tag{4.17}$$

where eta is a normalization term in order to scale cost-sensitive outputs to $\sum_{i=1}^{M} O_i^* = 1$ and $0 \leq O_i^* \leq 1$. A cost-sensitive ANN with moving threshold makes its predictions based on $\text{argmax}_i\, O_i^*(x)$. Threshold-moving approaches for ANNs have been overlooked for a long time and are not as popular as sampling-based methods for class imbalance. However, some studies report its high usefulness for dealing with datasets with skewed distributions [30, 44, 67]. Other works report that simply changing the data distribution without considering the imbalance effect on the classification threshold (and thus adjusting it properly) may be misleading [50].

4.5.3 Nearest Neighbors

The popular k-NN classifier also finds its usage in cost-sensitive learning [22, 51]. It uses the minimization of conditional risk, similarly to decision tree approaches, as cost parameter modifies the probabilities of assigning new instance x to each of classes. However, k-NN utilizes the proportion of k-NN of x belonging to i-th class to estimate $P(i|x)$. Cost-sensitive k-NN rescales the cost matrix parameters, so that for a two-class problems we have $C(1, 0) + C(0, 1) = 1$. Then the classification rule assigns x to class 1 if $k_1/k > C(0, 1)$ and to class 0 otherwise, where k_1 is the number of instances from class 1 among k-NN of x.

4.6 Hybrid Cost-Sensitive Approaches

While sampling methods seem attractive due to their requirements of modifying (rebalance) only training set (and thus a flexibility of applying any type of classifier afterwards), they may suffer from problems related to loss of information after

removing some instances, increasing noise or overlapping with introduction of new instances, or even causing a data set shift (more details on data-level approaches can be found at Chap. 5). Cost-sensitive methods may suffer from incorrectly estimated parameters of cost matrix. This tend to happen when the search procedure get stuck in a local minimum, or the search space is too big to efficiently find a (sub)optimal solution.

A potentially attractive solution lies in using a hybrid approach that will combine advantages of its components, while alleviating their drawbacks. One idea lies in conducting a small-scale sampling of the training set before learning a cost-sensitive classifier. This will reduce the IR, leading to a reduction of misclassification costs for minority instances and thus making the search process less biased. As we introduce/remove lower number of instances, we reduce the risk of deleting important information or introducing noise. Finally, cost-sensitive classifiers are usually faster than sampling approaches. Thus a hybrid methodology allows for a computational speed-up when compared with a scenario that uses only sampling.

Abkani et al. [1] proposed SMOTE with Different Costs (SDC) algorithm that works in three steps. Firstly, no undersampling of the majority class is conducted, thus not allowing for any loss of information. Secondly, a SVM with different misclassification costs is being trained on supplied dataset in order to reduce the bias towards the majority class. Finally, SMOTE is applied on minority distances in order to improve the definition of the learned class boundary. Similar idea was proposed by Wang et al. [62].

Chawla et al. [9] developed a wrapper approach that automatically learns sampling ratios individually for each dataset via evaluation function optimization. One can plug-in misclassification cost into this evaluator. An internal cross-validation on training data is used to establish undersampling and oversampling ratios, apply it to all instances and train a classifier on the modified data set.

Peng [49] proposed an adaptive undersampling and oversampling, where a pool of classifiers is being trained using different sampling ratios and estimated misclassification costs, and a weighted combination function is used for fusion of base classifiers.

4.7 Summarizing Comments

This chapter discussed the idea of cost-sensitive learning in the context of varying misclassification costs and class imbalance. Taxonomy of cost-sensitive approaches was presented and most representative algorithms were described in details, with a special emphasis on meta-learning solutions and decision trees. Despite over two decades of progress in this field, there are many directions to be pursued by researchers in this domain. Let us conclude this chapter by discussing the most important open issues and future challenges that cost-sensitive learning from imbalanced data must face in years to come.

- **Cost-sensitive solutions for data-level difficulties**: cost-sensitive solutions used so far associate on the level of classes. Yet, instances within the minority class may pose different levels of difficulty and mistakes on some of them should be penalized stronger than on the others. Developing methods that could induce cost-sensitive classifiers that take into account intrinsic data characteristics seems as a promising direction.
- **Hybrid cost-sensitive learning**: combining cost-sensitive approaches with sampling (please refer to Chap. 5), or potentially other algorithm-level solutions, is a worthwhile, yet larger unexplored area. It is necessary to gain a deeper insight into scenarios in which one of these approaches outperforms the other, in order to be able to create a more versatile compound algorithm.
- **Cost matrix estimation from non-stationary data**: learning misclassification costs is challenging on its own, but is even more challenging when conducted on non-stationary data streams [28] (see also Chap. 11). There is a need to develop new online cost-sensitive approaches that can combine advantages of ROC-based analysis with low computational complexity and capabilities of tackling concept drift.
- **Cost-sensitive approaches for other learning paradigms**: cost-sensitive learning should be expanded to other learning domains where class imbalance is present, such as multi-label/multi-instance problems [63], regression [54], or time-series analysis [55]. A complete description of these areas can be found at Chap. 12.

We envision that next decade will bring significant developments in this area, as many contemporary real-world applications call for existence of such machine learning methods.

References

1. Akbani, R., Kwek, S., Japkowicz, N.: Applying support vector machines to imbalanced datasets. In: Machine Learning: ECML 2004, 15th European Conference on Machine Learning, Pisa, Proceedings, pp. 39–50, 20–24 Sept 2004
2. Alejo, R., Toribio, P., Sotoca, J.M., Valdovinos, R.M., Gasca, E.: Resampling methods versus cost functions for training an MLP in the class imbalance context. In: Advances in Neural Networks – ISNN 2011 – 8th International Symposium on Neural Networks, ISNN 2011, Guilin, Proceedings, Part II, pp. 19–26, 29 May–1 June 2011
3. Anastasio, M.A., Kupinski, M.A., Nishikawa, R.M.: Optimization and FROC analysis of rule-based detection schemes using a multiobjective approach. IEEE Trans. Med. Imaging **17**(6), 1089–1093 (1998)
4. Bernard, S., Chatelain, C., Adam, S., Sabourin, R.: The multiclass ROC front method for cost-sensitive classification. Pattern Recogn. **52**, 46–60 (2016)
5. Bourke, C., Deng, K., Scott, S.D., Schapire, R.E., Vinodchandran, N.V.: On reoptimizing multi-class classifiers. Mach. Learn. **71**(2–3), 219–242 (2008)
6. Cao, P., Zhao, D., Zaïane, O.R.: An optimized cost-sensitive SVM for imbalanced data learning. In: Advances in Knowledge Discovery and Data Mining, 17th Pacific-Asia Conference, PAKDD 2013, Gold Coast, Proceedings, Part II, pp. 280–292, 14–17 Apr 2013

7. Castro, C.L., de Pádua Braga, A.: Novel cost-sensitive approach to improve the multilayer perceptron performance on imbalanced data. IEEE Trans. Neural Netw. Learn. Syst. **24**(6), 888–899 (2013)
8. Chatelain, C., Adam, S., Lecourtier, Y., Heutte, L., Paquet, T.: A multi-model selection framework for unknown and/or evolutive misclassification cost problems. Pattern Recogn. **43**(3), 815–823 (2010)
9. Chawla, N.V., Cieslak, D.A., Hall, L.O., Joshi, A.: Automatically countering imbalance and its empirical relationship to cost. Data Min. Knowl. Discov. **17**(2), 225–252 (2008)
10. Cheng, F., Zhang, J., Wen, C., Liu, Z., Li, Z.: Large cost-sensitive margin distribution machine for imbalanced data classification. Neurocomputing **224**, 45–57 (2017)
11. Datta, S., Das, S.: Near-Bayesian support vector machines for imbalanced data classification with equal or unequal misclassification costs. Neural Netw. **70**, 39–52 (2015)
12. Domingos, P.M.: Metacost: a general method for making classifiers cost-sensitive. In: Proceedings of the Fifth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Diego, pp. 155–164, 15–18 Aug 1999
13. Drummond, C., Holte, R.C.: C4.5, class imbalance, and cost sensitivity: why under-sampling beats over-sampling. In: ICML Workshop on Learning from Imbalanced Datasets II, Washington, DC, pp. 1–8 (2003)
14. Du, J., Cai, Z., Ling, C.X.: Cost-sensitive decision trees with pre-pruning. In: Advances in Artificial Intelligence, 20th Conference of the Canadian Society for Computational Studies of Intelligence, Canadian AI 2007, Montreal, Proceedings, pp. 171–179, 28–30 May 2007
15. Ducange, P., Lazzerini, B., Marcelloni, F.: Multi-objective genetic fuzzy classifiers for imbalanced and cost-sensitive datasets. Soft Comput. **14**(7), 713–728 (2010)
16. Everson, R.M., Fieldsend, J.E.: Multi-class ROC analysis from a multi-objective optimisation perspective. Pattern Recogn. Lett. **27**(8), 918–927 (2006)
17. Fawcett, T.: Roc graphs: notes and practical considerations for researchers. Technical report (2004)
18. Fawcett, T.: An introduction to ROC analysis. Pattern Recogn. Lett. **27**(8), 861–874 (2006)
19. Ferri, C., Flach, P.A., Hernández-Orallo, J.: Learning decision trees using the area under the ROC curve. In: Proceedings of the Nineteenth International Conference on Machine Learning (ICML 2002), University of New South Wales, Sydney, pp. 139–146, 8–12 July 2002
20. Garcia, L.P.F., Lorena, A.C., Matwin, S., de Leon Ferreira de Carvalho, A.C.P.: Ensembles of label noise filters: a ranking approach. Data Min. Knowl. Discov. **30**(5), 1192–1216 (2016)
21. Hand, D.J., Till, R.J.: A simple generalisation of the area under the ROC curve for multiple class classification problems. Mach. Learn. **45**(2), 171–186 (2001)
22. Hand, D.J., Vinciotti, V.: Choosing k for two-class nearest neighbour classifiers with unbalanced classes. Pattern Recogn. Lett. **24**(9–10), 1555–1562 (2003)
23. Jackowski, K., Krawczyk, B., Wozniak, M.: Cost-sensitive splitting and selection method for medical decision support system. In: 13th International Conference on Intelligent Data Engineering and Automated Learning – IDEAL 2012, Natal, Proceedings, pp. 850–857, 29–31 Aug 2012
24. Katsumata, S., Takeda, A.: Robust cost sensitive support vector machine. In: Proceedings of the Eighteenth International Conference on Artificial Intelligence and Statistics, AISTATS 2015, San Diego, 9–12 May 2015 (2015)
25. Knoll, U., Nakhaeizadeh, G., Tausend, B.: Cost-sensitive pruning of decision trees. In: Machine Learning: ECML-94, European Conference on Machine Learning, Catania, Proceedings, pp. 383–386, 6–8 Apr 1994
26. Krawczyk, B.: Cost-sensitive one-vs-one ensemble for multi-class imbalanced data. In: 2016 International Joint Conference on Neural Networks, IJCNN 2016, Vancouver, pp. 2447–2452, 24–29 July 2016
27. Krawczyk, B.: Learning from imbalanced data: open challenges and future directions. Prog. AI **5**(4), 221–232 (2016)

28. Krawczyk, B., Skryjomski, P.: Cost-sensitive perceptron decision trees for imbalanced drifting data streams. In: European Conference on Machine Learning and Knowledge Discovery in Databases, ECML PKDD 2017, Skopje, Proceedings, Part II, pp. 512–527, 18–22 Sept 2017
29. Krawczyk, B., Woźniak, M.: Designing cost-sensitive ensemble – genetic approach. In: 3rd International Conference on Image Processing and Communications Challenges 3, IP&C 2011, Proceedings, pp. 227–234 (2011)
30. Krawczyk, B., Woźniak, M.: Cost-sensitive neural network with ROC-based moving threshold for imbalanced classification. In: 16th International Conference on Intelligent Data Engineering and Automated Learning – IDEAL 2015, Wroclaw, Proceedings, pp. 45–52, 14–16 Oct 2015
31. Krawczyk, B., Woźniak, M., Schaefer, G.: Cost-sensitive decision tree ensembles for effective imbalanced classification. Appl. Soft Comput. **14**, 554–562 (2014)
32. Krawczyk, B., Schaefer, G., Woźniak, M.: A hybrid cost-sensitive ensemble for imbalanced breast thermogram classification. Artif. Intell. Med. **65**(3), 219–227 (2015)
33. Landgrebe, T., Duin, R.P.W.: Approximating the multiclass ROC by pairwise analysis. Pattern Recogn. Lett. **28**(13), 1747–1758 (2007)
34. Landgrebe, T., Duin, R.P.W.: Efficient multiclass ROC approximation by decomposition via confusion matrix perturbation analysis. IEEE Trans. Pattern Anal. Mach. Intell. **30**(5), 810–822 (2008)
35. Landgrebe, T., Paclík, P.: The ROC skeleton for multiclass ROC estimation. Pattern Recogn. Lett. **31**(9), 949–958 (2010)
36. Landgrebe, T., Paclík, P., Tax, D.M.J., Verzakov, S., Duin, R.P.W.: Cost-based classifier evaluation for imbalanced problems. In: Structural, Syntactic, and Statistical Pattern Recognition, Joint IAPR International Workshops, SSPR 2004 and SPR 2004, Lisbon, Proceedings, pp. 762–770, 18–20 Aug 2004
37. Li, F., Zhang, X., Zhang, X., Du, C., Xu, Y., Tian, Y.: Cost-sensitive and hybrid-attribute measure multi-decision tree over imbalanced data sets. Inf. Sci. **422**, 242–256 (2018)
38. Ling, C.X., Yang, Q., Wang, J., Zhang, S.: Decision trees with minimal costs. In: Proceedings of the Twenty-first International Conference on Machine Learning (ICML 2004), Banff, 4–8 July 2004
39. Liu, X., Zhou, Z.: The influence of class imbalance on cost-sensitive learning: an empirical study. In: Proceedings of the 6th IEEE International Conference on Data Mining (ICDM 2006), Hong Kong, pp. 970–974, 18–22 Dec 2006
40. Liu, M., Xu, C., Luo, Y., Xu, C., Wen, Y., Tao, D.: Cost-sensitive feature selection by optimizing f-measures. IEEE Trans. Image Process. **27**(3), 1323–1335 (2018)
41. Liu, Z., Ma, C., Gao, C., Yang, H., Lan, R., Luo, X.: Cost-sensitive collaborative representation based classification via probability estimation with addressing the class imbalance. Multimed. Tools Appl. **77**(9), 10835–10851 (2018)
42. López, V., Fernández, A., Moreno-Torres, J.G., Herrera, F.: Analysis of preprocessing vs. cost-sensitive learning for imbalanced classification. Open problems on intrinsic data characteristics. Expert Syst. Appl. **39**(7), 6585–6608 (2012)
43. López, V., del Río, S., Benítez, J.M., Herrera, F.: Cost-sensitive linguistic fuzzy rule based classification systems under the mapreduce framework for imbalanced big data. Fuzzy Sets Syst. **258**, 5–38 (2015)
44. Maloof, M.A.: Learning when data sets are Imbalanced and when costs are unequal and unknown. In: International Conference on Machine Learning, Washington, DC (2003)
45. Moepya, S.O., Akhoury, S.S., Nelwamondo, F.V.: Applying cost-sensitive classification for financial fraud detection under high class-imbalance. In: 2014 IEEE International Conference on Data Mining Workshops, ICDM Workshops 2014, Shenzhen, pp. 183–192, 14 Dec 2014
46. Narasimhan, H., Agarwal, S.: Support vector algorithms for optimizing the partial area under the ROC curve. Neural Comput. **29**(7), 1919–1963 (2017)
47. Núñez, M.: The use of background knowledge in decision tree induction. Mach. Learn. **6**, 231–250 (1991)

48. Penar, W., Woźniak, M.: Cost-sensitive methods of constructing hierarchical classifiers. Expert. Syst. **27**(3), 146–155 (2010)
49. Peng, Y.: Adaptive sampling with optimal cost for class-imbalance learning. In: Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence, Austin, pp. 2921–2927, 25–30 Jan 2015
50. Provost, F.: Machine learning from imbalanced data sets 101. In: Proceedings of the AAAI 2000 Workshop on Imbalanced Data Sets, pp. 1–3 (2000)
51. Qin, Z., Wang, A.T., Zhang, C., Zhang, S.: Cost-sensitive classification with k-nearest neighbors. In: 6th International Conference on Knowledge Science, Engineering and Management, KSEM 2013, Dalian, Proceedings, pp. 112–131, 10–12 Aug 2013
52. Qiu, C., Jiang, L., Li, C.: Randomly selected decision tree for test-cost sensitive learning. Appl. Soft Comput. **53**, 27–33 (2017)
53. Radtke, P.V.W., Granger, E., Sabourin, R., Gorodnichy, D.O.: Skew-sensitive boolean combination for adaptive ensembles – an application to face recognition in video surveillance. Inf. Fusion **20**, 31–48 (2014)
54. Riccardi, A., Fernández-Navarro, F., Carloni, S.: Cost-sensitive adaboost algorithm for ordinal regression based on extreme learning machine. IEEE Trans. Cybern. **44**(10), 1898–1909 (2014)
55. Roychoudhury, S., Ghalwash, M.F., Obradovic, Z.: Cost sensitive time-series classification. In: European Conference on Machine Learning and Knowledge Discovery in Databases, ECML PKDD 2017, Skopje, Proceedings, Part II, pp. 495–511, 18–22 Sept 2017
56. Sáez, J.A., Krawczyk, B., Woźniak, M.: Analyzing the oversampling of different classes and types of examples in multi-class imbalanced datasets. Pattern Recogn. **57**, 164–178 (2016)
57. Sheng, S., Ling, C.X.: Hybrid cost-sensitive decision tree. In: Knowledge Discovery in Databases: PKDD 2005, 9th European Conference on Principles and Practice of Knowledge Discovery in Databases, Porto, Proceedings, pp. 274–284, 3–7 Oct 2005
58. Skryjomski, P., Krawczyk, B.: Influence of minority class instance types on SMOTE imbalanced data oversampling. In: First International Workshop on Learning with Imbalanced Domains: Theory and Applications, LIDTA@PKDD/ECML 2017, Skopje, pp. 7–21, 22 Sept 2017
59. Ting, K.M.: An instance-weighting method to induce cost-sensitive trees. IEEE Trans. Knowl. Data Eng. **14**(3), 659–665 (2002)
60. Turney, P.D.: Cost-sensitive classification: empirical evaluation of a hybrid genetic decision tree induction algorithm. J. Artif. Intell. Res. **2**, 369–409 (1995)
61. Veropoulos, K., Campbell, C., Cristianini, N.: Controlling the sensitivity of support vector machines. In: Proceedings of the International Joint Conference on AI, Stockholm, pp. 55–60 (1999)
62. Wang, S., Li, Z., Chao, W., Cao, Q.: Applying adaptive over-sampling technique based on data density and cost-sensitive SVM to imbalanced learning. In: The 2012 International Joint Conference on Neural Networks (IJCNN), Brisbane, pp. 1–8, 10–15 June 2012
63. Wang, X., Liu, X., Japkowicz, N., Matwin, S.: Resampling and cost-sensitive methods for imbalanced multi-instance learning. In: 13th IEEE International Conference on Data Mining Workshops, ICDM Workshops, Dallas, pp. 808–816, 7–10 Dec 2013
64. Zhang, X., Hu, B.: A new strategy of cost-free learning in the class imbalance problem. IEEE Trans. Knowl. Data Eng. **26**(12), 2872–2885 (2014)
65. Zhang, S., Qin, Z., Ling, C.X., Sheng, S.: "Missing is useful": missing values in cost-sensitive decision trees. IEEE Trans. Knowl. Data Eng. **17**(12), 1689–1693 (2005)
66. Zhao, H., Li, X., Xu, Z., Zhu, W.: Cost-sensitive decision tree with probabilistic pruning mechanism. In: 2015 International Conference on Machine Learning and Cybernetics, ICMLC 2015, Guangzhou, pp. 81–87, 12–15 July 2015
67. Zhou, Z., Liu, X.: Training cost-sensitive neural networks with methods addressing the class imbalance problem. IEEE Trans. Knowl. Data Eng. **18**(1), 63–77 (2006)
68. Zhou, Q., Zhou, H., Li, T.: Cost-sensitive feature selection using random forest: selecting low-cost subsets of informative features. Knowl. Based Syst. **95**, 1–11 (2016)