Journal of Cleaner Production 242 (2020) 118481

Contents lists available at ScienceDirect

Journal of Cleaner Production

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

Cost-imbalanced hyper parameter learning framework for quality classification

Yunchao Zhang [a], Yu Li [b], Zeyi Sun [b, *], Haoyi Xiong [c], Ruwen Qin [b], Chen Li [d]

[a] Department of Computer Science, Missouri University of Science and Technology, Rolla, MO, 65409, USA
[b] Department of Engineering Management and Systems Engineering, Missouri University of Science and Technology, Rolla, MO, 65409, USA
[c] Big Data Laboratory, Baidu INC, Beijing, 100085, China
[d] School of Economics and Management, Xinjiang University, Urumqi, Xinjiang, 830046, China

ARTICLE INFO

Article history:
Received 28 October 2018
Received in revised form
6 September 2019
Accepted 17 September 2019
Available online 20 September 2019

Handling editor: Yutao Wang

Keywords:
Quality classification
Cost-imbalanced
Hyper-parameter learning
Machine learning
Decision tree
Particle swarm optimization

ABSTRACT

A quality control system is an indispensable section in various manufacturing and service industries. It plays a critical role in reducing process flaws, optimizing process parameters, improving production quality and productivity, as well as enhancing customer satisfaction. In this paper, we propose an intelligent data-driven quality classification platform by leveraging a novel integrated hyper learning framework to further strengthen the cost-effectiveness in quality control by reducing the economic loss due to misclassification. The misclassification-dependent weights are proposed and used for training the classifier with an emphasis on cost-effectiveness. The proposed integrated hyper learning framework is used to optimally identify such weights. Specifically, the framework consists of two nested layers, where the inner-layer addresses the optimal classifier training with a given set of misclassification weights, while the out-layer updates such weights iteratively according to the performance in terms of the economic loss due to misclassification by the classifier identified by the inner-layer towards optimality. The case studies are implemented using five different datasets in different manufacturing and service industries, including food, auto, steel, and glass. The economic loss, as well as additional carbon emission due to misclassification when using the quality classifier identified through the proposed framework, is compared to three other algorithms under different settings of penalty costs due to misclassification. The results illustrate that the proposed intelligent data-driven quality classification platform outperforms the other ones in terms of the reduction of the economic loss due to misclassification and demonstrate the robustness of the performance with respect to various misclassification penalty costs. As for the carbon emission reduction, the proposed model can outperform, in most cases, the three other algorithms. While the consistency of this superiority cannot be guaranteed since the environmental concern is not modeled in the objective function.

1. Introduction

1.1. Research objective

In this paper, an intelligent data-driven quality classification platform using a hyper learning framework is proposed to optimize the cost-effectiveness in terms of the decision-making of quality classification for various industries. The hyper learning framework consists of two layers. The inner layer trains the optimal classifier given the labeled samples and a certain loss weight matrix to minimize the average economic loss incurred by the misclassification based on corresponding penalty cost. The outer layer searches for the optimal loss weights based on the outcome of the inner layer. The inner layer and outer layer are updated iteratively until the average economic loss converges. During this iterative process, it is expected that the misclassification with higher penalty cost could be learned more intensively by the classifier in the training process. The optimal loss weights can thus be identified and the average economic loss due to misclassification can be minimized as well. The proposed framework is implemented on multiple real-world datasets from different manufacturing and service industries.

* Corresponding author.
E-mail address: sunze@mst.edu (Z. Sun).

https://doi.org/10.1016/j.jclepro.2019.118481
0959-6526/

Nomenclature

Lower case

c_1 and c_2	two positive constants denoting the loss weights in particle swarm optimization
c_{lh}	the penalty loss due to misclassifying class l to class h
h	the column index of matrix **C** or **W**
i	the index of the particles in particle swarm optimization
j	the index of the data subsets
l	the row index of matrix **C** or **W**
m	the number of data subsets
p	the total number of the classes
r_1 and r_2	two random numbers between zero and one in particle swarm optimization
t	index of iteration in particle swarm optimization
w_{lh}	the weight used in training the target classifier when misclassifying class l to class h
x_j	the sample from data subset j
y_j	the vector storing the actual class of the sample x_j
z	the inertial weight in particle swarm optimization

Greek letter

θ	the parameters of classification model
θ^*	the optimal parameters of the classification model that can minimize $L(\bullet)$

Functions

$L(\bullet)$	financial loss function
$f(x_j\|\theta^*)$	the predicted class of sample x_j

Matrix

C	$p\times p$ matrix consisting of the penalty costs due to various misclassifications
W	the weight matrix storing the weights for different types of misclassifications that are used in training the classification platform
$\mathbf{W}^*$	the optimal weight matrix used in the training process that can lead to an optimal classifier
$\mathbf{V}_i^t$	the velocity of particle i at iteration t in particle swarm optimization
$\mathbf{Pb}_i^t$	the particle best location up to iteration t in particle swarm optimization
$\mathbf{Gb}^t$	the best location of the entire swarm up to iteration t in particle swarm optimization

1.2. Significance

Quality control is an important process in various industries. It plays a critical role in the improvement and optimization that may greatly contribute to reducing operational cost and carbon footprints. A well-designed quality control system ensures the customers receive high-standard products which satisfy their needs. While, defect products will likely jeopardize customers and potentially cause significant loss to manufacturers due to product recalls or loss of credibility. A great deal of research in the area of quality has been reported. For example, Liu et al. (2016) proposed an online quality control method for the remanufacturing assembly based on the state space model. El Khaled et al. (2017) conducted a review for cleaner quality control system using bioimpedance for fruits and vegetables industries. Li et al. (2019) investigated the performance of a pilot-scale aquaponics system for water quality control. Rangel et al. (2019) proposed a generalized quality control parameter for heterogenous recycled concrete aggregates.

1.3. State of the art of quality classification model

1.3.1. Classifier trained with equal loss weight

Smart data-driven classification techniques have been widely studied and used in modern manufacturing factories and service industries in order to accurately and promptly identify the appropriate quality of the products (Camejo et al., 2013; Rostami et al., 2015). A traditional quality classifier can be trained using pre-labeled data samples to classify unlabeled products into different quality levels based on their measured properties. In classical classification theory, the algorithms traditionally try to generate the classification model by minimizing the total misclassification error. Such a tradition is under the assumption of equal penalty cost disregarding different types of misclassifications. This simplified training configuration implies that the economic losses due to various misclassifications are the same, which is not necessarily the case when handling multi-class classification problems in the real world. In industry, different types of misclassifications often result in a various amount of loss, treating every type of misclassification error equally does not necessarily reduce the overall economic loss of the manufacturers.

Consider a binary classification model as an example, usually, type I (false alarm) and type II (miss detection) errors may result in different losses or penalties, depending on cases. For instance, in the classification of edibility of mushroom, classifying an edible mushroom to be poisonous will lead to a less serious result than judging a poisonous mushroom to be edible. Another example, diagnosing a patient to be healthy is much worse than diagnosing a healthy person to be sick. The consequent actions based on the false result of classification may be costly. Similarly, consider a multi-class classification problem, a typical decision-making in quality control, an overestimation that misclassifies the product with lower quality level to a higher one may lead to a more significant economic loss than an underestimation that misclassifies a product with a higher quality level to a lower one since the former one, in many cases, will involve customer complaints and compensations, and possibly cause credibility crisis.

1.3.2. Classifier trained with unequal loss weight

There, thus, needs a method that can train the classification model with various penalty weights for different types of misclassification errors. Intuitively, the misclassification error leading to a higher economic loss may require a larger misclassification loss weight to strengthen the learning effects during the training process, so that the number of such types of misclassification can be reduced as optimization progresses. However, it is hard to explicitly formulate the total economic loss due to misclassification as a function of the loss weight matrix used in the model training process. To bypass such a challenge, researchers have tried various strategies that can integrate the influence of different types of costs into the penalty weight matrix of the model. Such an integration strategy has been considered one of the most relevant directions of the research area in machine learning (Saitta, 2000). Various types of costs, e.g., test cost, teacher cost, intervention cost, misclassification penalty cost, etc., have been studied. For example, Turney (2000) created a taxonomy of different types of cost that are involved in machine learning. Chai et al. (2004)

proposed a test-cost sensitive classification approach using naive Bayes. Ling et al. (2004) proposed a method for building and testing decision trees that minimize the sum of the misclassification and test costs. Among these ones, the penalty cost due to misclassification is the one that obtains the most concentrations (Zhou and Liu, 2010).

In general, two major types of misclassification costs have been studied (Zhou and Liu, 2006). The first one is the class dependent cost, which assumes that the penalty cost is dependent on classes. In other words, each class has a unified misclassification penalty cost for different misclassifications that misclassify the products belong to this class to other different classes. Many studies based on this assumption have been reported. For example, Breiman et al. (1984) investigated the decision tree approach in classification learning. Elkan (2001) characterized the cost matrix for the problem of optimal learning and decision-making when different misclassification errors incur different penalties. Liu and Zhou (2006) revealed that class-imbalance often affects the performance of cost-sensitive classifiers through an empirical study using 38 datasets. Zhang and Zhou (2008) applied cost-sensitive learning in face recognition.

The second one is the example-dependent cost which assumes the penalty cost is dependent on each individual instance. The research based on this assumption has also been conducted. For example, Zadrozny and Elkan (2001) introduced a method to learn the cost and probability estimators when both are unknown in learning and decision-making. Brefeld et al. (2003) proposed a support vector machine with example dependent costs. Lozano and Abe (2008) derived a multi-class cost-sensitive boosting approach with p-norm loss functions.

In real applications, the class-dependent cost is more popularly used than the example-dependent cost since such a misclassification cost depending on the class can be obtained by expert opinions or existing experience without lots of difficulties. Rescaling is the most commonly used strategy that has been proposed to deal with class-dependent penalty cost to make cost-blind learning algorithm cost-sensitive. The principle of rescaling algorithm is to reassign the weight of samples for each class in training set according to the given misclassification cost, so that the influences of higher-cost classes will be larger than those of the lower-cost classes in the training process. Such a variation of influence is realized by the adjustment of sample size in training set for different classes so that the class with higher misclassification penalty cost can have a larger sample size.

The rescaling approach was first derived for binary classification. For example, Domingos (1999) proposed a general method for making classifiers cost-sensitive. Ting (2002) introduced an instance-weighting method to induce cost-sensitive trees. Elkan (2001) investigated the cost matrix for binary classification through rescaling to address the issue of imbalanced sample size in the training set. Later, the approach was further extended to multi-class classification (Zhou and Liu, 2010; Zhou and Liu, 2006). The weight obtained by the rescaling approach is based on the penalty cost of each class, which can be defined as the sum of the penalty cost due to misclassification of a given class into all other possible classes.

1.3.3. Limitation of existing unequal weight classifier

The limitation of the rescaling strategy for the class-dependent cost is that it averages the penalty costs of various types of misclassifications occurred to a certain class. It's not misclassification-dependent. In practice, the costs due to the various types of misclassifications from the same class are not necessarily the same. In addition, rescaling is considered an "off-line" strategy to adjust the influences during the training processes based on various misclassification costs, which cannot guarantee the minimized loss due to misclassification. For the example-dependent cost, costs of misclassifying one example into different classes are the same, which is not always the case in real world scenarios either.

Therefore, the existing "unequal weight" strategy proposed or used when training the classifier model still cannot address the concern that different types of misclassification may lead to different economic loss. In addition, although in many existing works of literature, costs are used directly and intuitively as the loss weight matrix when training the classification model, it may not necessarily be the optimal weight matrix for the problem. The complex interrelationships between the penalty weights and the corresponding economic loss are ignored.

1.4. Contribution

Therefore, motivated by the status_quo, we propose an intelligent data-driven quality classification platform using a hyper learning framework designed to optimize the cost-effectiveness. The contribution of this paper can be summarized from three folds as follows: (1) A nested hyper learning framework is proposed where various optimization methods and machine-learning models can be selected and used for the out-layer and inner-layer of the framework in respective with different objectives representing different concerns or interests to form a quality classification platform. (2) A misclassification-dependent classifier can be trained through identifying an optimal loss weight matrix, which addresses the limitation that loss weight for classifier training is not specific to different misclassification and implicitly reveals the complex mechanisms between the monetary loss and corresponding penalty weight. (3) From the theoretical point of view, the contribution is still non-trivial. The loss matrix for misclassification depends on the output of predictors in many typical machine learning algorithms that have been used (Khosravi et al., 2018; De Clercq et al., 2019), while the output of the predictor would be changed per update during the training procedure. In this way, it might be challenging to stabilize the loss estimation of predictors. To tackle this problem, we define a surrogated loss weight matrix, where each entry refers to a pseudo loss of misclassification. We thus make the loss of misclassification independent of the predictor output. A nested learning procedure with two-layer optimization framework, that optimizes the loss weight matrix and the parameters of predictors simultaneously, so as to improve the overall performance of learning with imbalanced loss.

The rest of the paper is organized as follows. Section 2 introduces the proposed framework. Section 3 illustrates the evaluation framework consisting of an optimization algorithm, datasets used for evaluation, and experimental results to validate the effectiveness of the proposed framework. Section 4 discusses the advantages and limitations of the proposed framework. Section 5 concludes the paper and proposes future research directions.

2. Integrated hyper learning framework

In this section, the integrated hyper learning framework is introduced. The objective of the proposed framework is to find the optimal loss weight matrix used in training the proposed quality classification model that can minimize the average economic loss due to the misclassification from m testing sets in m-cross validation. Mathematically, it can be formulated as

$$\underset{\mathbf{W}\in R^{p^2}}{\operatorname{argmin}}\left\{\frac{1}{m}\sum_{j=1}^{m}L\left(y_j,f\left(x_j|\theta^*\right)|\ \mathbf{C}\right)\right. \tag{1}$$

where $L(\cdot)$ is the financial loss function due to the misclassification.

$j = 1, 2, \ldots, m$ is the index of the data subsets. y_j is the vector storing the actual class of the sample x_j in subset j. θ is the parameters of the classification model, and θ^* is the optimal parameters of the classification model that can minimize $L(\cdot)$. $f(x_j|\theta^*)$ is the predicted class of sample x_j in subset j.

C is a $p \times p$ matrix consisting of the penalty costs due to various misclassifications. Here p is the total number of the classes. We use l and h to denote the row and column indexes of this matrix, respectively. Specifically, the element c_{lh} in **C** denotes the penalty loss due to misclassifying class l to class h.

W is the weight matrix storing the weights for different types of misclassifications that are used in training the classification platform. Similarly, the element w_{lh} in **W** denotes the weight that will be used in training the target classification model when misclassifying class l to class h. $\mathbf{W}^*$ is the optimal weight matrix used in the training process that can lead to an optimal classifier.

The details of the procedure to solve the objective (1) is shown in Algorithms 1 and 2. Algorithm 1 takes the given weight w, dataset D, and the misclassification cost matrix **C** as inputs and gives the average loss $\overline{\ell_t}$ as an output. The line 5 in Algorithm 2 then uses Algorithm 1 to calculate the average loss with given weight matrix w_t. Using all the obtained loss up to iteration t and the weight w of the current iteration, Algorithm 2 updates w until an optimal weight matrix w^* that can minimize the misclassification loss is obtained. The details of the update method we use will be further discussed in section 3.

Algorithm 1 Average Loss

```
1:  procedure AVERAGELOSS(w, D, C)
2:      ℓ_t ← 0;
3:      for j = 1, 2, 3, ..., m do
4:          Randomly split D into D_Train and D_Test
5:          according to fixed ratio r;
6:          θ* ← arg min_{θ∈Θ} L(θ|w, D_Train, C);
7:          ℓ_t^j ← L(θ*|w, D_Test, C);
8:          ℓ_t ← ℓ_t + ℓ_t^j;
9:      ℓ̄_t ← ℓ_t / m;
10:     return ℓ̄_t;
```

Algorithm 2 Minimize Average Loss

```
1:  procedure OPTIMIZE(D, C)
2:      t ← 1;
3:      Initialize w_1;
4:      do
5:          ℓ̄_t ← AverageLoss(w_t, D, C);
6:          if ℓ̄_t has not converged then
7:              w_{t+1} ← Update(w_t, ℓ̄_t, ..., ℓ̄_1)
8:              t ← t + 1
9:      while ℓ̄_t has not converged
10:     w* ← w_t;
11:     return w*;
```

3. Framework evaluation

In this section, we implement the proposed integrated hyper learning framework using different datasets. We first introduce the optimization algorithm used in the proposed framework to update **W** towards its optimality $\mathbf{W}^*$. Then, we introduce the experimental design for our evaluation. Finally, the experimental results are demonstrated and the performance is compared to the baseline setting with equal weights as well as other modeling methods.

3.1. Optimization algorithm

Without knowing the details with respect to the differentiability and convexity of the loss function $L(\cdot)$ in the search space of **W**, for generality, we propose to use Particle Swarm Optimization (PSO), a typical meta-heuristic method without requiring the prerequisites of differentiability and convexity, as the optimization modeling tool located at the out-layer framework, to update **W** as listed in line 7 of proposed Algorithm 2 until a near optimal solution of $\mathbf{W}^*$ that can minimize the loss function is obtained.

PSO is a typical population-based meta-heuristic algorithm inspired and characterized by foraging behaviors of animal swarms (Kennedy et al., 2001). In the framework of PSO, the population dynamics simulate the behavior of a bird flock, where social sharing of information takes place and individuals profit from the discoveries and previous experience of all other peers during the search for food. Each particle in the swarm is assumed to 'fly' over the search space looking for promising regions (optimal solutions) on the landscape. PSO has been extensively studied and widely used when solving some high-dimension complex optimization problems (Jerald et al., 2005; Wang and Li, 2014). Some researchers have recently explored the theoretical connection between such heuristic algorithms and some existing analytical models such as Hamiltonian systems and Nesterov's method (Freidlin and Hu, 2011; Hu and Li, 2017).

When using PSO in the out-layer of the proposed framework, each candidate solution (weight matrix) is encoded as a particle. The particle's fitness is quantified by the value of the loss function resulted from the identified classification model using the given weight matrix. The fitness function can be formulated as

$$f\left(\mathbf{W}_i^t\right) = \frac{1}{m}\sum_{j=1}^{m} L\left(y_j, f\left(x_j|\theta_i^t\right)|\ \mathbf{C}\right)\Big) \tag{2}$$

where $\mathbf{W}_i^t$ is the location of (i.e., the weight matrix denoted by) the particle i ($i = 1, \ldots, I$) at iteration t ($t = 1, \ldots, T$). θ_i^t is the parameter of the classification model trained at iteration t using $\mathbf{W}_i^t$. The fitness is defined as the average loss from m classification models after m cross-validation using $\mathbf{W}_i^t$. The value of $\mathbf{W}_i^t$ is updated at each iteration by

$$\mathbf{W}_i^{t+1} = \mathbf{W}_i^t + \mathbf{V}_i^t \tag{3}$$

where $\mathbf{V}_i^t$ is the velocity of particle i at iteration t. It is determined by the particle's previous velocity, the distance between the best previous location of the particle itself and its current location, as well as the distance between the swarm's best experienced location and the current location. $\mathbf{V}_i^t$ can be updated as follows

$$\mathbf{V}_i^{t+1} = z\mathbf{V}_i^t + c_1 r_1\left(\mathbf{Pb}_i^t - \mathbf{W}_i^t\right) + c_2 r_2\left(\mathbf{Gb}^t - \mathbf{W}_i^t\right) \tag{4}$$

where $\mathbf{Pb}_i^t$ is the particle best location up to iteration t. $\mathbf{Gb}^t$ is the best location of the entire swarm up to iteration t. z is the inertial weight. c_1 and c_2 are two positive constants denoting the loss weights. r_1 and r_2 are two random numbers between zero and one.

When the PSO is launched with a given swarm size and iteration number, the location of each particle is generated randomly within a predetermined range. m-cross validation is conducted using each particle (i.e., the weight matrix) to identify the classification model and the corresponding loss. The particle that can lead to the minimal loss value is recorded as the best particle in the swarm. After that, the location of each particle is updated as shown in Algorithm 3 so that a set of new weight matrices can be generated for training the new classification model at next iteration through m-cross validation. The updated particles will compare their current locations to the historical ones according to the fitness functions to find their respective best locations up to current iteration. The best particle of the entire swarm will also be updated if necessary. This

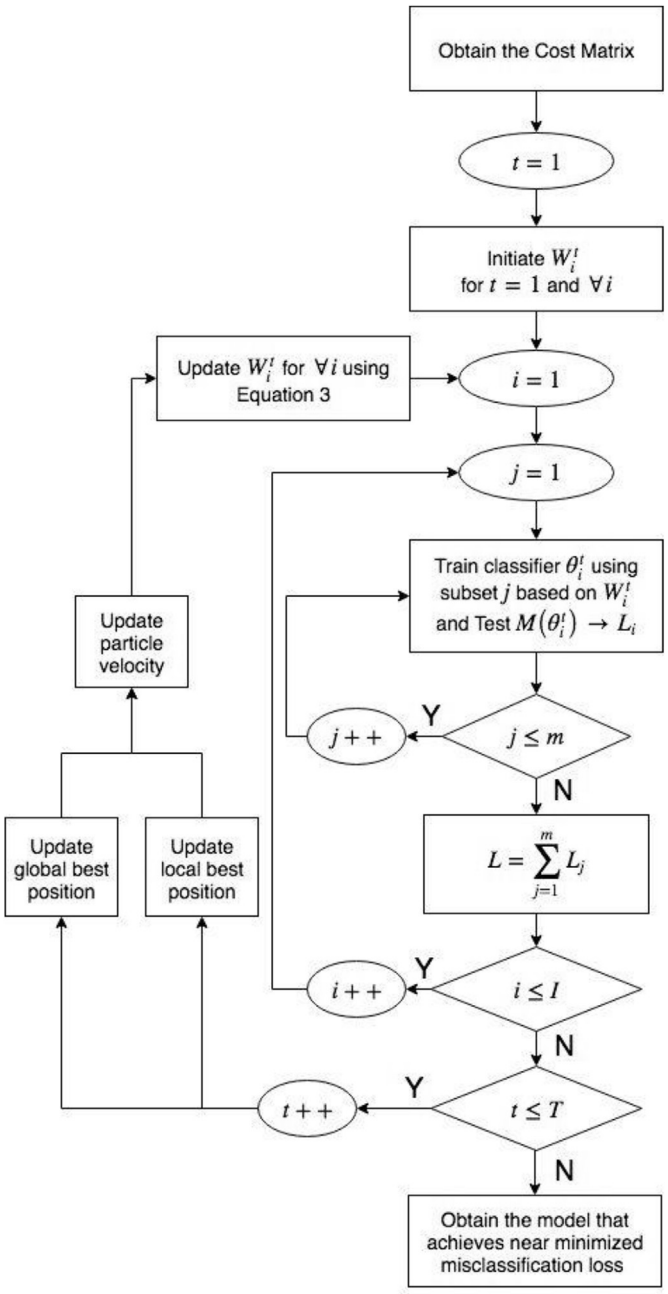

Fig. 1. Flowchart of the proposed framework.

process will be repeated until the predetermined iteration number is reached. The best particle location and the corresponding optimal parameters of the classification model that can lead to a minimal loss when using the classification model can be obtained.

Algorithm 3 Update Weight

procedure PSO UPDATE($\mathbf{W}_t, \bar{L}_t, \ldots, \bar{L}_1$)
  for all particle w_t in $\mathbf{W}_t$ **do**
    $v_t \leftarrow v_t + c_1 r_1(\mathbf{Pb}_t - w_t) + c_2 r_2(\mathbf{Gb} - w_t)$;
    $w_t \leftarrow w_t + v_t$;
  return $\mathbf{W}_t$;

The overall workflow of the proposed integrated hyper learning framework is illustrated in Fig. 1.

3.2. Dataset

In this subsection, we introduce the datasets used for the evaluation. To evaluate the proposed framework, we use five datasets, including red wine grade classification, white wine grade classification, 2nd-hand car quality evaluation, steel plate faults classification, and glass type identification. For each dataset, we divide the total data into two groups. The first group is for building the model with identified optimal weight matrix. The second group represents the new incoming data for validating the effectiveness of the model obtained. The first data group is further divided into ten subsets for a ten-cross validation. At each round of validation, nine subsets are used to train a classification model using a given weight matrix. The remaining subset is then used to find the loss value using the model parameters obtained by the model trained.

The details of red and white wine quality cases can be found in (Cortez et al., 2009). In this reference, two datasets were created for red and white wine samples with sizes of 1599 and 4898, respectively. The inputs include various test results (e.g. PH values, color intensity, total phenols, ash, etc.) (Cortez et al., 2009) and the output is the grade of the tested wine based on sensory data (median of at least 3 evaluations made by wine experts). Each expert graded the wine quality between 0 (very bad) and 10 (very excellent).

The details of 2nd-hand car quality evaluation case can be found in (Bohanec and Rajkovic, 1988). The sample size is 1728. Six attributes (i.e. buying price, maintenance price, number of doors, the capacity of holding passengers, size of luggage boot, and safety) (Bohanec and Rajkovic, 1988) are used as inputs. The output is described by four classes, unacceptable, acceptable, good, and very good.

The details of steel plate faults classification case can be found in (Buscema et al., 2010). The data has 1941 instances. The inputs include twenty-seven attributes (e.g., minimum fault width, maximum fault width, minimum fault length, maximum fault length, pixel area, steel plate thickness, etc.) (Buscema et al., 2010)

$$\begin{pmatrix} 0 & 10 & \cdots & 90 & 100 \\ 8 & 0 & & 80 & 90 \\ \vdots & & \ddots & & \vdots \\ 72 & 64 & & 0 & 10 \\ 80 & 72 & \cdots & 8 & 0 \end{pmatrix}$$

Fig. 2. Example of a Cost matrix 1.

Table 1
Different configurations of penalty cost due to misclassification.

Cost Type	Description
Cost matrix 1	Overestimation is poorer than underestimation, linear relationship between misclassification cost and misclassification extent
Cost matrix 2	Overestimation is poorer than underestimation, nonlinear relationship between misclassification cost and misclassification extent
Cost matrix 3	Underestimation is poorer than overestimation, linear relationship between misclassification cost and misclassification extent
Cost matrix 4	Underestimation is poorer than overestimation, nonlinear relationship between misclassification cost and misclassification extent

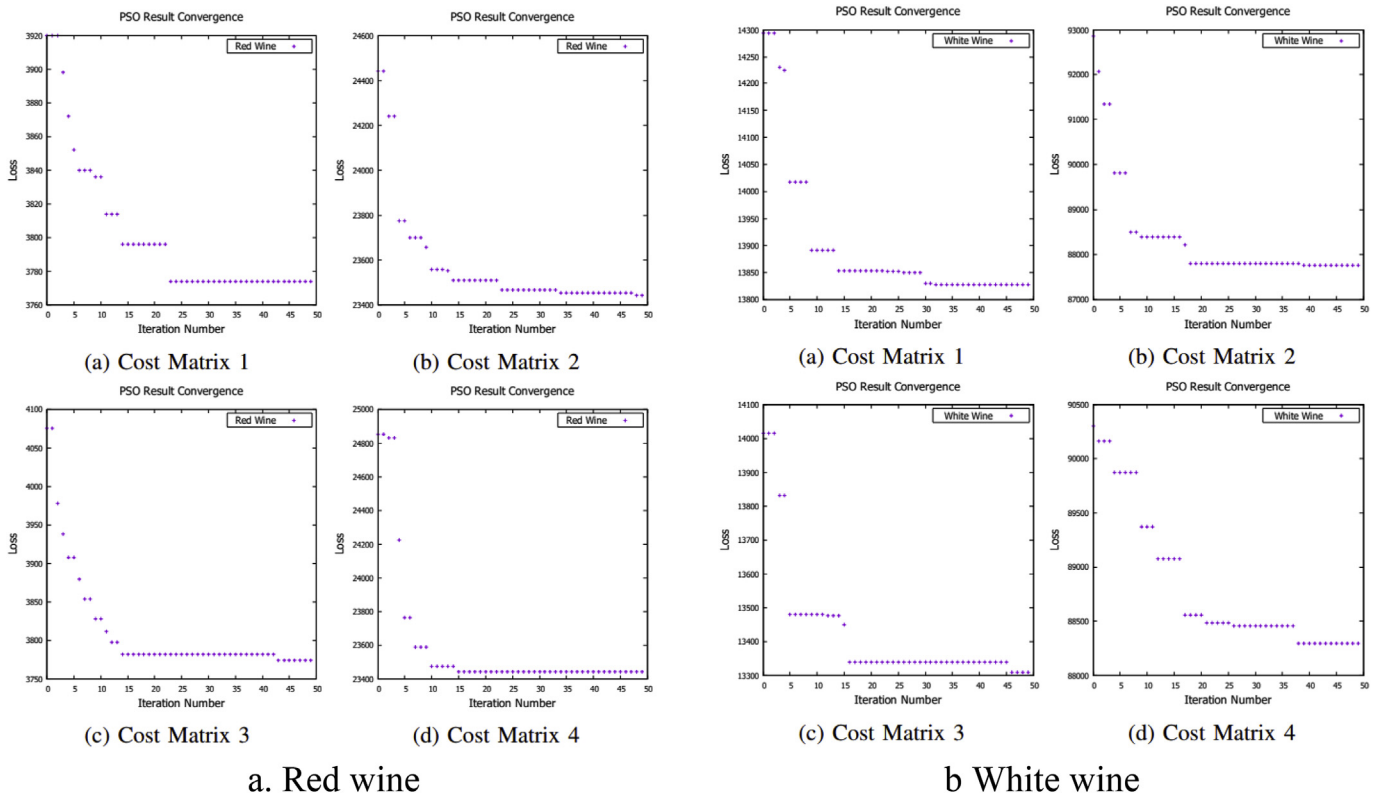

Fig. 3. Loss convergence process in PSO for red & white wine classification cases. (For interpretation of the references to color in this figure legend, the reader is referred to the Web version of this article.)

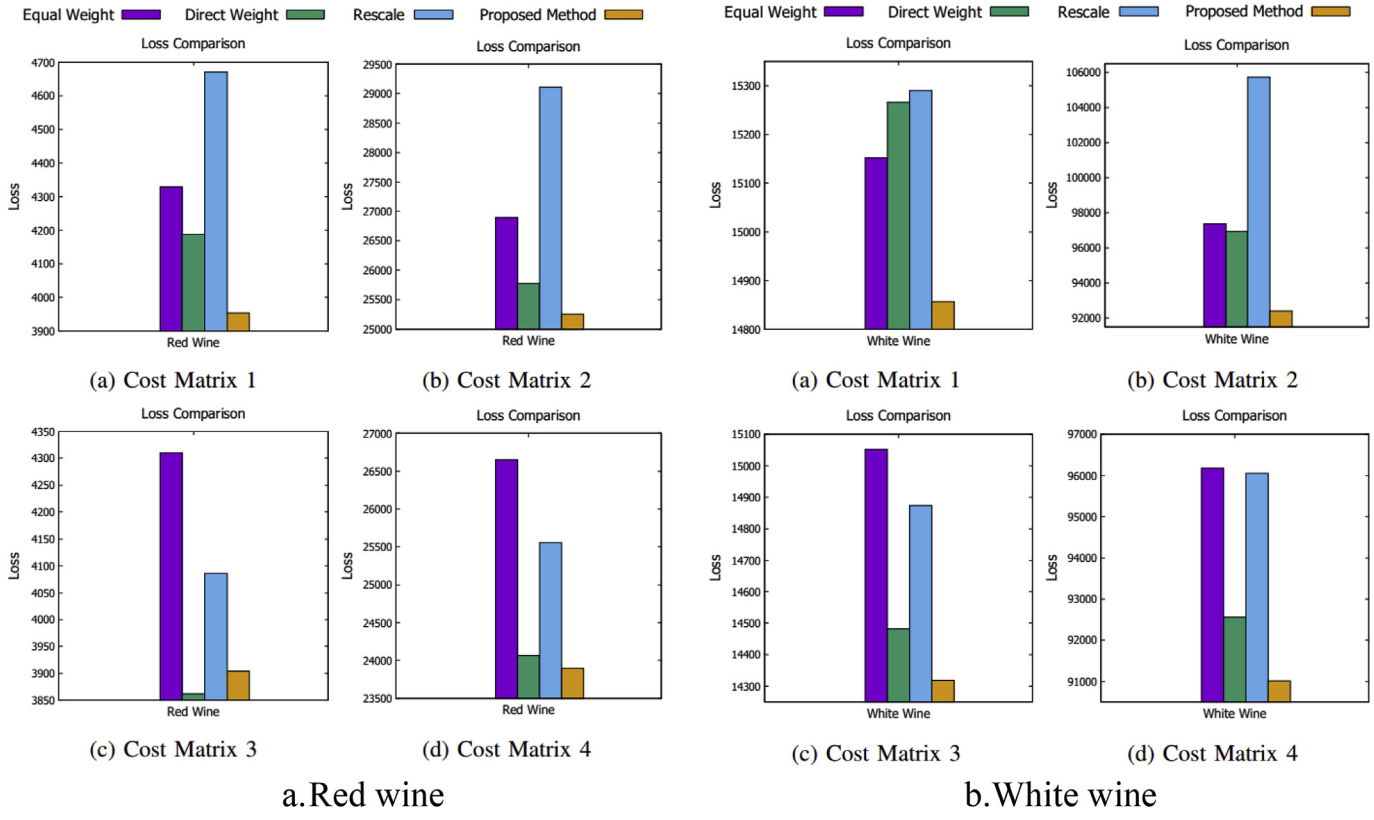

Fig. 4. Loss comparison between proposed framework and three other weight algorithms in red & white wine quality cases. (For interpretation of the references to color in this figure legend, the reader is referred to the Web version of this article.)

Table 2
Comparison of loss reduction using different cost matrices for red & white wine quality case.

Cost matrix	Loss reduction compared to equal weight algorithm	Loss reduction compared to direct weight algorithm	Loss reduction compared to rescale algorithm
Red Wine			
Cost matrix 1	9%	6%	15%
Cost matrix 2	6%	2%	13%
Cost matrix 3	9%	1%	4%
Cost matrix 4	10%	1%	6%
White Wine			
Cost matrix 1	2%	3%	3%
Cost matrix 2	5%	5%	13%
Cost matrix 3	5%	1%	4%
Cost matrix 4	5%	2%	5%

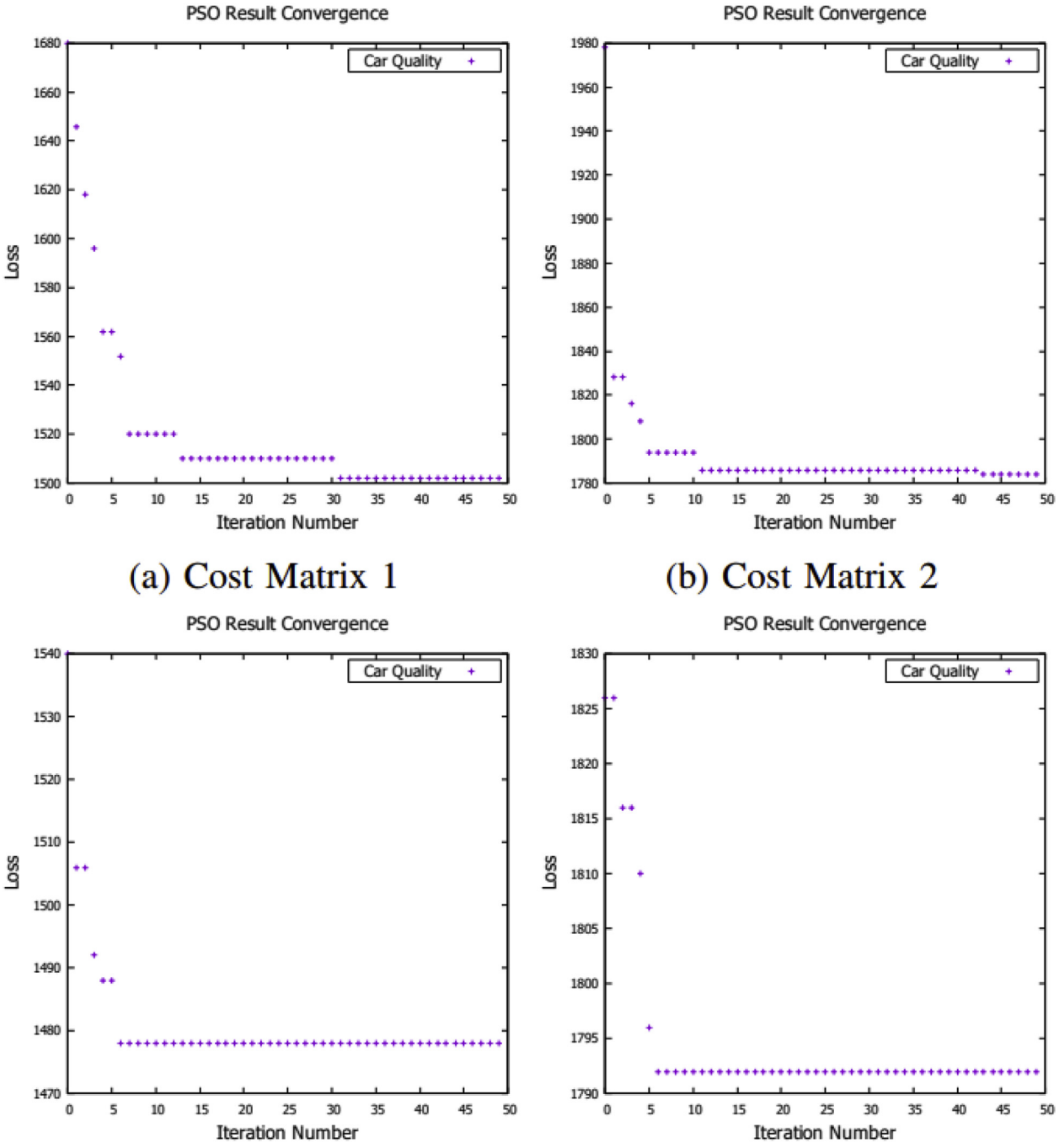

Fig. 5. Loss convergence process in PSO for 2nd-hand car quality evaluation case.

that describe the geometric shape of the fault. The output is described by seven types of faults, pastry, Z-scratch, K-scratch, stains, dirtiness, bumps, and other faults.

The details of glass identification case can be found in (Evett and Spiehler, 1987). This dataset contains 214 instances. The inputs include the contents of nine oxides (i.e. Na, Fe, K, etc.) (Evett and Spiehler, 1987) in the glass. The output is described by six types of glasses.

3.3. Experimental results

3.3.1. Red wine and white wine quality classification

The proposed integrated hyper learning framework is implemented for the red and white wine quality classification cases using four different penalty costs due to misclassification based on different assumptions, as summarized in Table 1.

For example, cost matrix 1 is built as shown in Fig. 2 which is

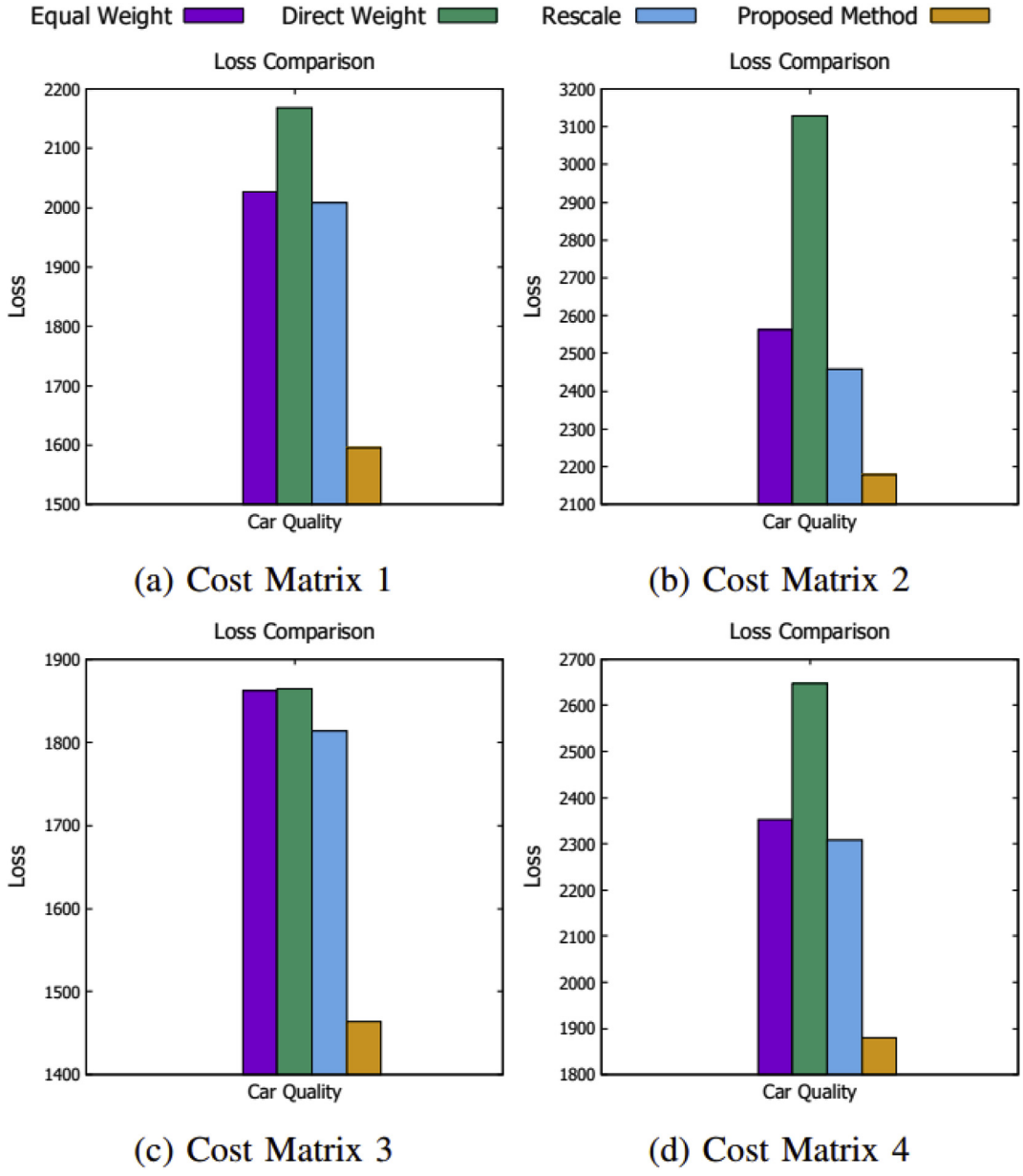

Fig. 6. Loss comparison between the proposed framework and three other weight algorithms for 2nd-hand car case.

Table 3
Comparison of loss reduction using different cost matrices for 2nd-hand car case.

Cost matrix	Loss reduction compared to equal weight algorithm	Loss reduction compared to direct weight algorithm	Loss reduction compared to rescaling algorithm
Cost matrix 1	21%	26%	21%
Cost matrix 2	15%	30%	11%
Cost matrix 3	21%	21%	19%
Cost matrix 4	20%	29%	19%

based on the assumptions that (1) the overestimation will lead to a higher financial loss compared to the underestimation of the same extent, (2) the misclassification costs between any two adjacent levels are the same, and (3) the misclassification cost is linearly proportional to the extents of misclassification, i.e., the distance between the actual class and the identified class.

In Fig. 2, the rows denote the actual class, and the columns denote the predicted class by the classifier. The larger the index, the better the quality level is represented. The upper-right half of the matrix stores the penalty costs due to different misclassifications of overestimation, while the lower-left half of the matrix stores the penalty costs due to different misclassifications of underestimation.

$$\begin{pmatrix} 0 & 10 & \cdots & 90 & 100 \\ 8 & 0 & & 80 & 90 \\ \vdots & & \ddots & & \vdots \\ 72 & 64 & & 0 & 10 \\ 80 & 72 & \cdots & 8 & 0 \end{pmatrix}$$

In this experiment, Random Decision Tree (Breiman, 2017) is selected as the inner-layer classification modeling tool. Random

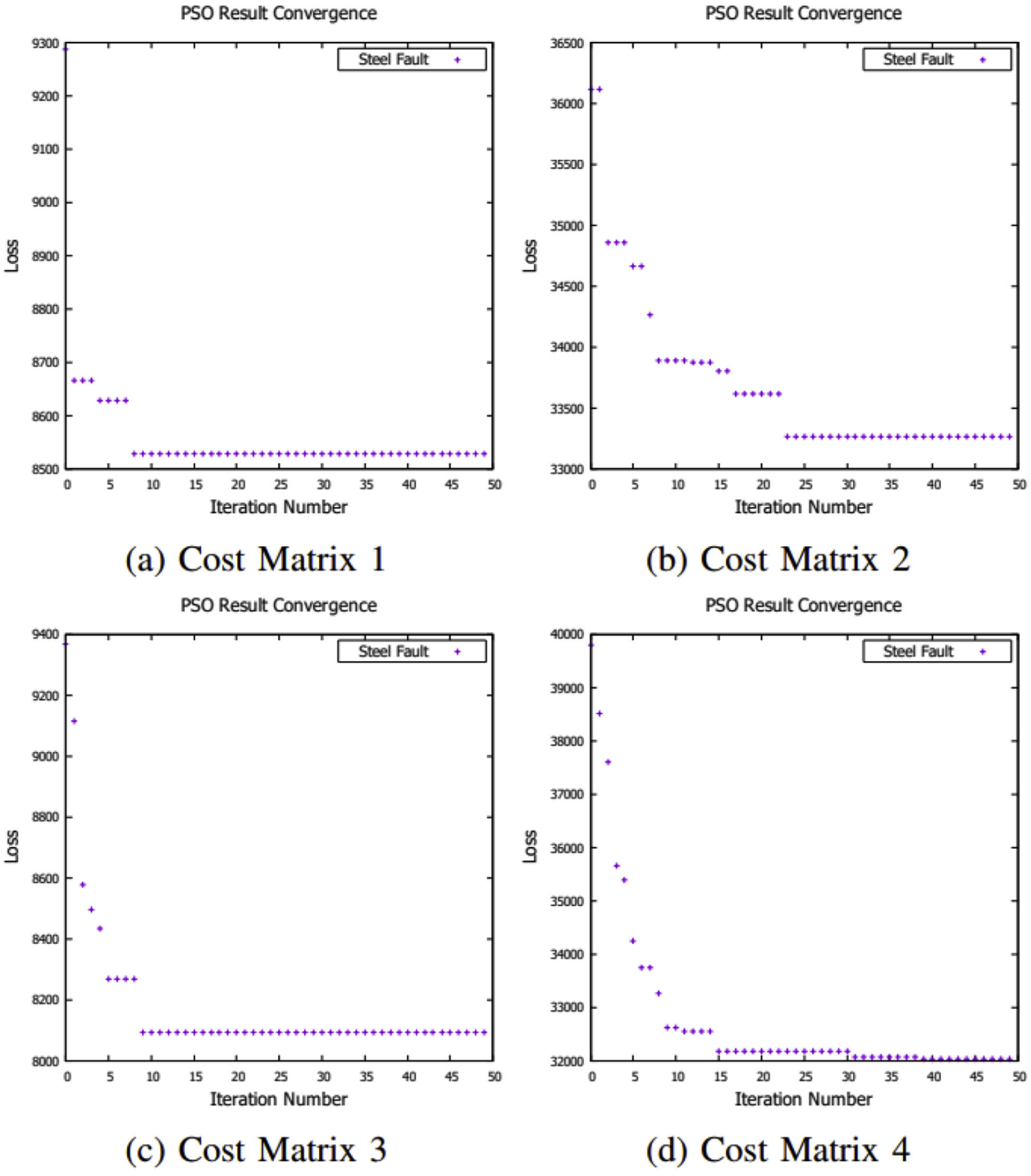

Fig. 7. Loss convergence process in PSO for steel fault classification case.

Decision Tree is a tree structure statistical model that can be used for regression and classification. A decision tree classifier is capable of learning decision rules based on data features and represent feature conjunctions with branches. Incoming samples will be classified into leaves of the decision tree, which represent class labels.

PSO is used as the out-layer optimization tool to identify a near optimal weight set that can lead to a minimized financial loss due to misclassification. The particle size is set as 100. The iteration number is set as 50. The learning rates of PSO are set as 0.9 and 0.1 for particle best and global best, respectively. The evolution of the loss value along with the iteration in red wine and white wine cases is illustrated in Fig. 3.

The loss value due to misclassifications when using proposed framework is compared to three other weight algorithms, i.e., equal weight for various misclassifications in model training, direct

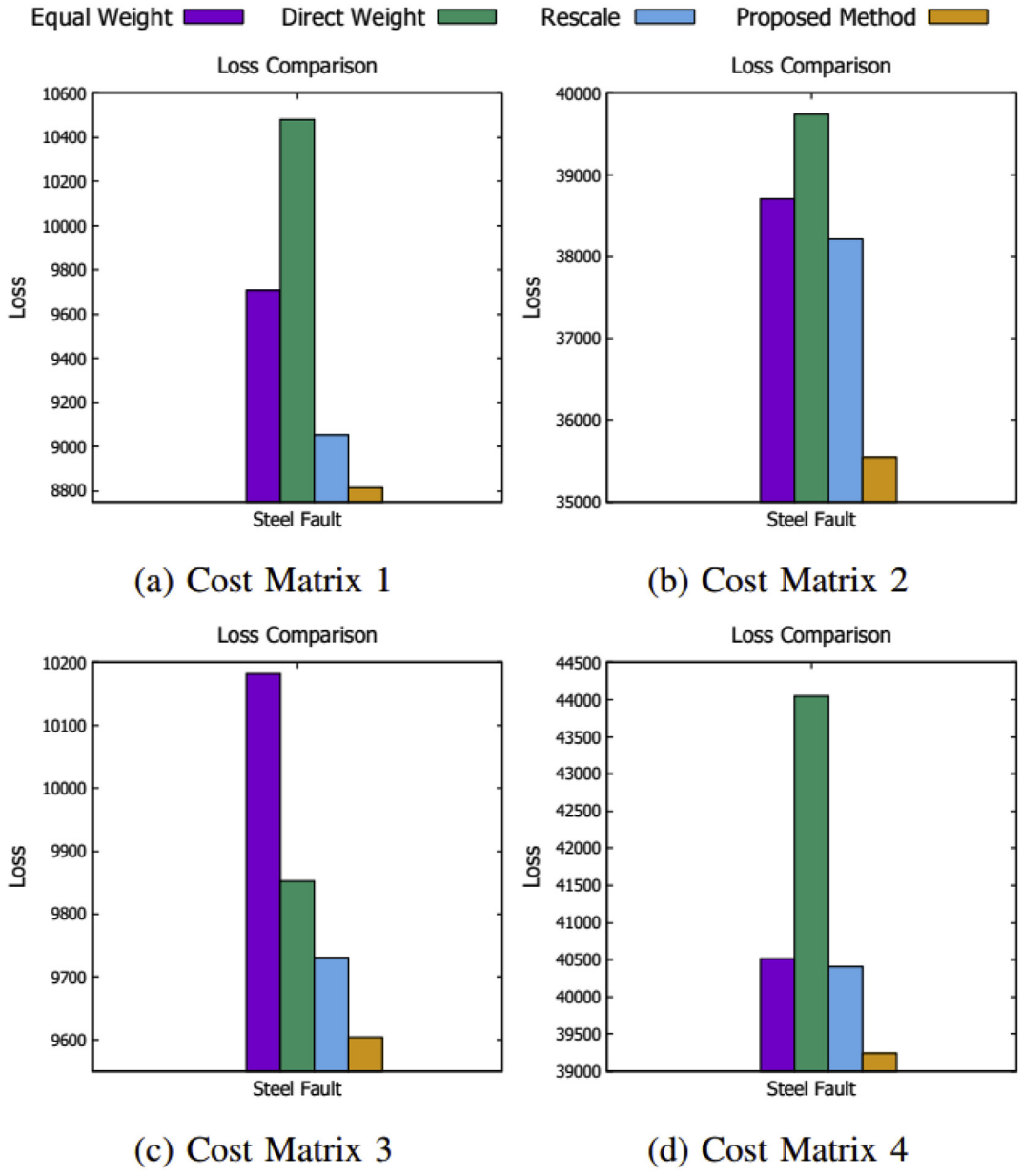

Fig. 8. Loss comparison between the proposed framework and three other weight algorithms for steel fault classification case.

Table 4
Comparison of loss reduction using different cost matrices for steel fault case.

Cost matrix	Loss reduction compared to equal weight algorithm	Loss reduction compared to direct weight algorithm	Loss reduction compared to rescaling algorithm
Cost matrix 1	9%	16%	3%
Cost matrix 2	8%	11%	7%
Cost matrix 3	6%	3%	1%
Cost matrix 4	3%	11%	3%

weight (i.e., the cost matrix after normalization is used as the loss weight matrix) in model training, and class size rescaling (i.e. adjust the number of instances in each class according to the cost ratios). The comparison is illustrated in Fig. 4 for the red wine and white wine cases. It can be seen that the optimal weight obtained by the proposed framework can lead to the lowest economic loss due to misclassification compared to all three other algorithms, while the direct weight and rescale algorithms perform poorer than the classical classification tool using equal weight.

Specifically, the loss due to the misclassification for red wine can be successfully reduced using the optimal weight generated by proposed framework by 9%, 6%, and 15% compared with the equal weight algorithm, direct weight algorithm, and rescaling algorithm, respectively. The loss due to the misclassification for white wine can be successfully reduced by 2%, 3% and 3% using the proposed framework compared with the equal weight algorithm, direct weight algorithm, and rescaling algorithm, respectively. The loss reduction in wine quality classification case using four different cost matrices are illustrated in Table 2.

It can be seen that for all four penalty cost assumptions, the

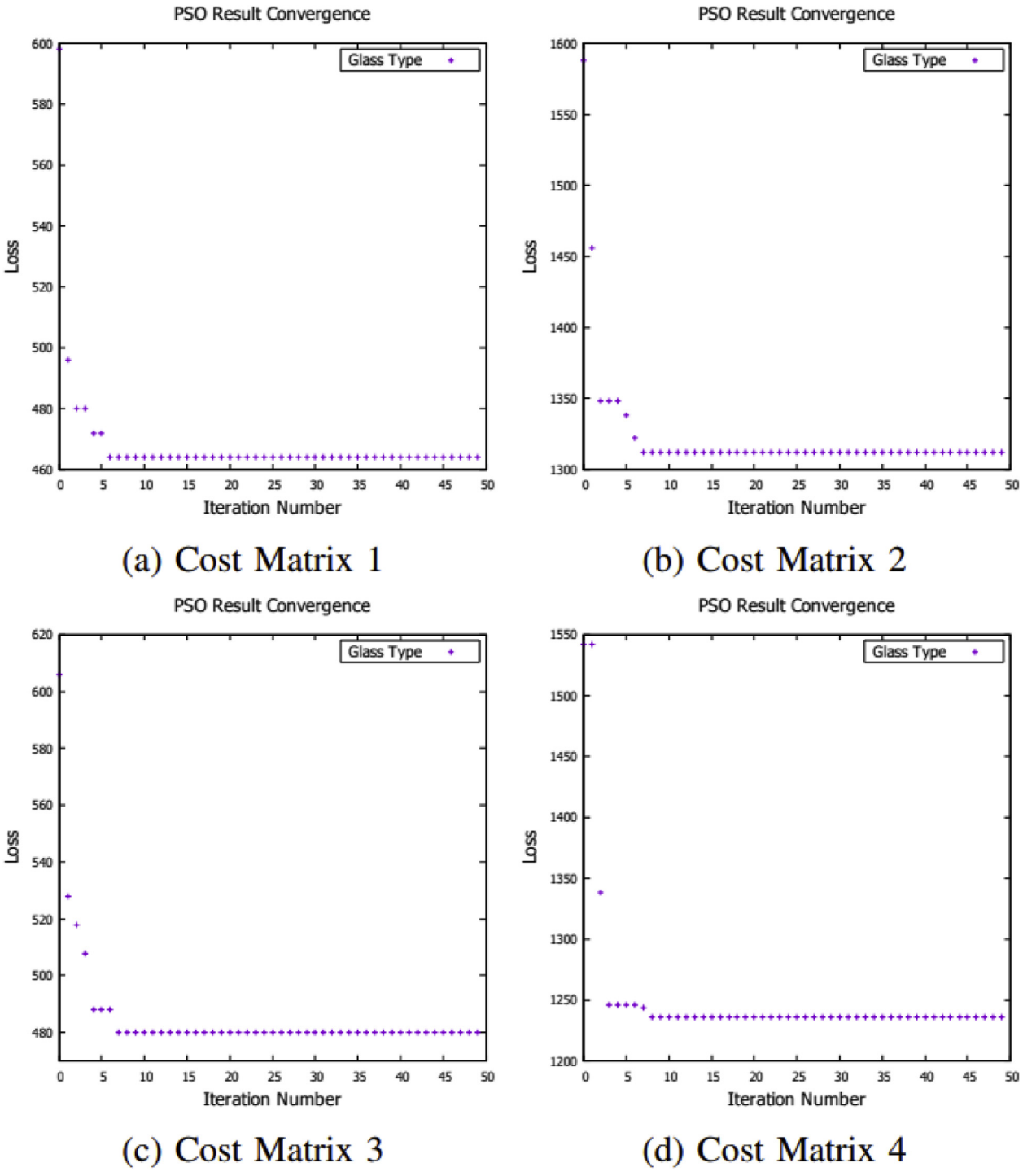

Fig. 9. Loss convergence process in PSO for glass type identification case.

proposed framework can outperform the other three methods with a lower economic loss due to misclassification. The results illustrate that the proposed algorithm is robust in different misclassification penalty cost settings so that it is potentially applicable in various business modes.

3.3.2. 2nd-hand car quality classification

In the 2nd-hand car evaluation case, the particle number and iteration number of PSO remain the same, 100 and 50, respectively. The global learning rate and particle learning rate are carefully tuned to be 2 and 0.9, respectively. Loss convergence using PSO can be observed in Fig. 5. The comparisons of loss due to misclassification among the proposed framework and three other algorithms following the 1st assumption of penalty cost matrix are illustrated in Fig. 6. The loss reductions using four different cost matrices for 2nd-hand car case are illustrated in Table 3.

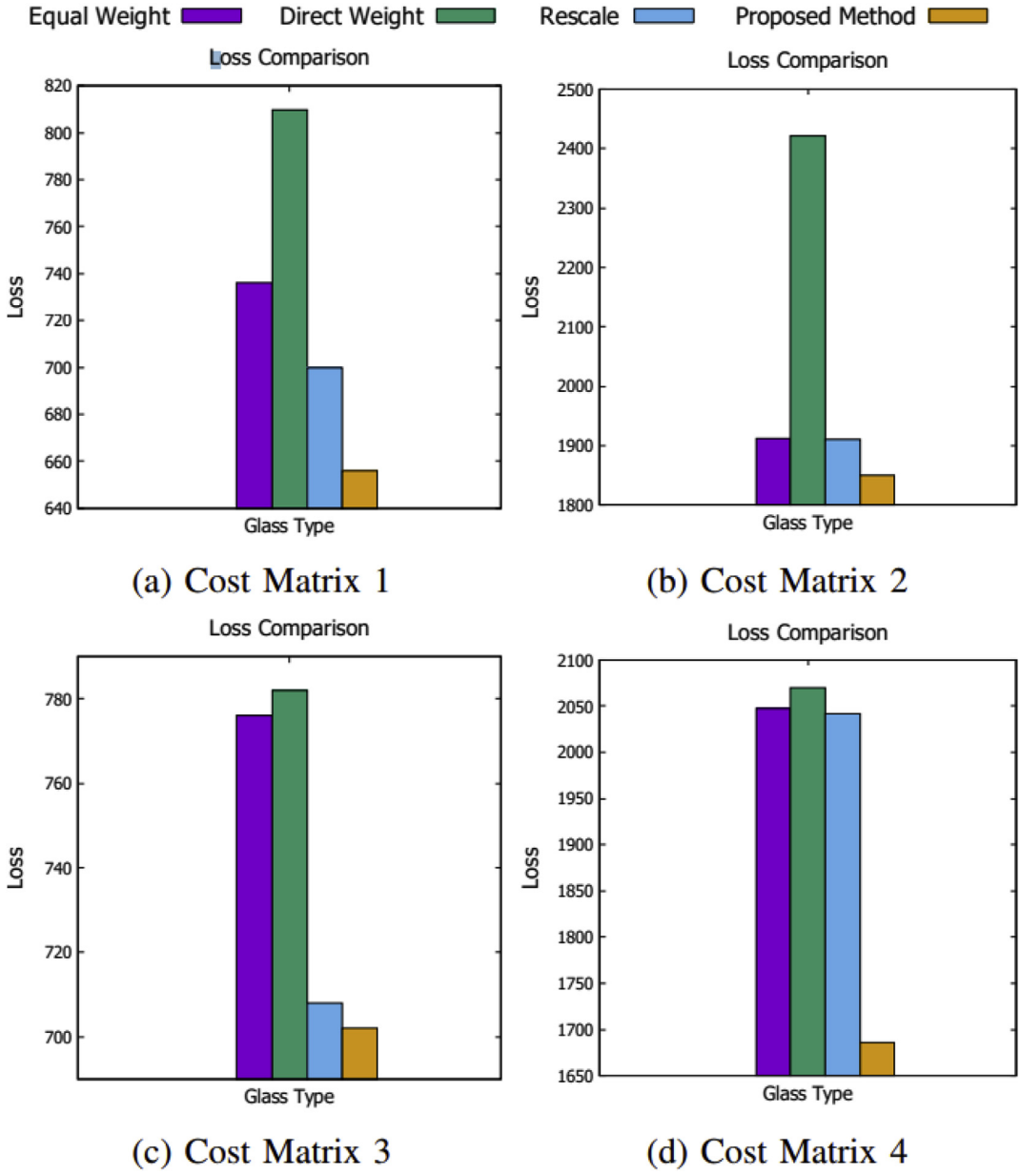

Fig. 10. Loss comparison between the proposed framework and three other weight algorithms for glass type identification case.

Table 5
Comparison of loss reduction using different cost matrices for glass type case.

Cost matrix	Loss reduction compared to equal weight algorithm	Loss reduction compared to direct weight algorithm	Loss reduction compared to rescaling algorithm
Cost matrix 1	11%	19%	6%
Cost matrix 2	3%	24%	3%
Cost matrix 3	10%	10%	1%
Cost matrix 4	18%	19%	17%

3.3.3. Fault classification in steel quality control

In the case of steel fault classification, the particle number and iteration number of PSO remain the same, 100 and 50, respectively. The global learning rate is tuned to be 0.95 and the particle learning rate is tuned to be 0.1. The loss convergence of PSO is demonstrated in Fig. 7. Fig. 8 below shows the comparison of loss due to misclassification between the proposed framework and three other algorithms following the 1st assumption of penalty cost matrix. The loss reductions using four different cost matrices for steel fault classification case are illustrated in Table 4.

3.3.4. Type classification in quality control of glass manufacturing

In the glass type identification case, the particle number and iteration number of PSO remain the same, 100 and 50, respectively. The global learning rate is set to be 0.9 and the particle learning rate is set to be 0.1 for PSO. The loss convergence process is demonstrated in Fig. 9. It is shown in Fig. 10 the comparison of loss due to misclassification among the proposed framework and three other algorithms following the 1st assumption of cost matrix setting. The loss reductions using four different cost matrices for glass type identification case are illustrated in Table 5.

Tables 2–5 reveal that the proposed framework performs consistently better than equal weight algorithm, direct weight algorithm, and rescaling algorithm on all tested datasets under all four different assumptions of the penalty cost due to misclassification. Direct weight algorithm performs better than equal weight algorithm in the wine quality classification case, but not in the other cases. Rescaling reduces the loss more than equal weight in steel fault classification, car quality evaluation, and glass type identification, but not in the wine case. Neither direct weight algorithm nor rescaling performs stably under different type of misclassification cost settings. Thus, compared to the direct weight algorithm and rescaling algorithm, the proposed framework is more robust and can more significantly reduce the economic loss due to misclassification under various business modes.

3.4. Model performance in terms of environmental sustainability

In addition to the performance comparison from the perspective of economic viability as intended by the model formulation, the resultant performance from the perspective of environmental sustainability of the proposed framework is also examined through comparing with equal weight, direct weight, and rescale models. It is assumed when quality overestimation happens, the manufacturer or service provider needs to offer a makeup, i.e., a correctly categorized product needs to be delivered to the customer to replace the overestimated one, which leads to the emissions from additional production or transportation for processing the makeup.

3.4.1. Red wine and white wine

Specifically, for the case of wine, it is assumed that the additional emission is mainly from the production process for the makeup wine. It has been reported that the approximate carbon footprint of a bottle of wine is around 1.28 kg (Francis, 2017). The resultant comparisons for four different cost matrices using red wine and white wine datasets are illustrated in Fig. 11. It can be seen that the proposed method can result in the lowest amount of carbon emission using all types of cost matrices.

3.4.2. Steel

For the case of steel, it is assumed that the additional emission is mainly from steel production. It is reported that for every kg of steel

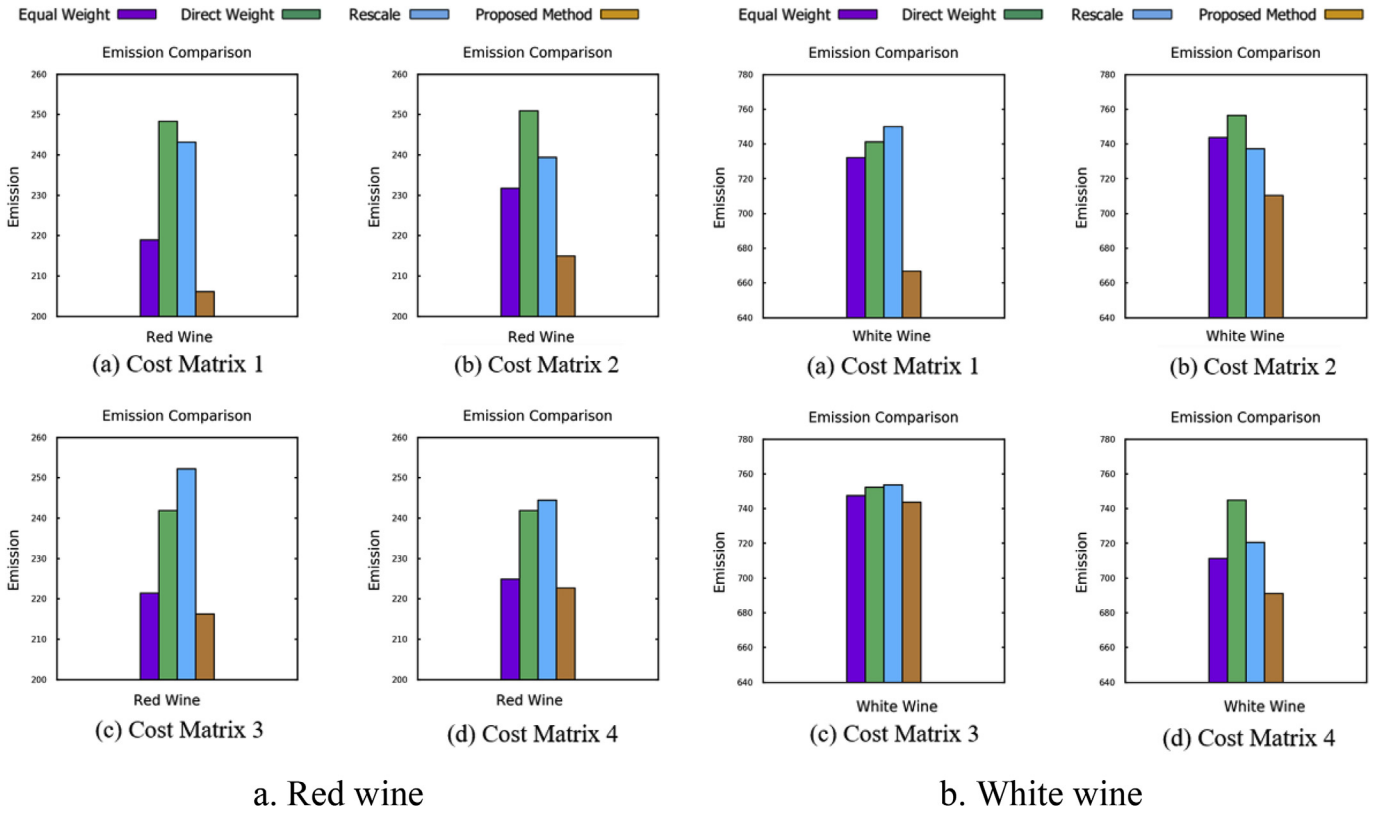

Fig. 11. Carbon emission comparison between the proposed framework and three other weight algorithms for red & white wine cases. (For interpretation of the references to color in this figure legend, the reader is referred to the Web version of this article.)

produced, 1.83 kg of CO_2 is emitted (World Steel Association, 2014). A steel plate with a size of $1 \times 1 \times 0.03$ (m) weighs in average 235.5 kg. Thus, the production of such a steel plate will result in about 431 kg CO_2 emissions. The resultant comparison for four different cost matrices is illustrated in Fig. 12. It can be seen that the proposed method can lead to a lower carbon emission compared to the other three models using all types of cost matrices.

3.4.3. Glass

For the case of glass, it is assumed that the additional emission is mainly from glass production. It is reported that about 8.4 kg of CO_2 is emitted for one kg of general glass production (Green Ration Book, 2010). The resultant comparison for four different cost matrices is illustrated in Fig. 13. It can be seen that the proposed method results in the lowest amount of carbon emission for cost matrices 1 and 3. While for cost matrix 2, the additional emission due to overestimation is higher than the emission from "equal weight" method, it is resulted from the fact that the number of overestimation from equal weight algorithm is lower than that from the proposed algorithm. The possible reason can be that the overestimation with a large extent is less preferred by the proposed algorithm than the one with a small extent since the cost matrix is

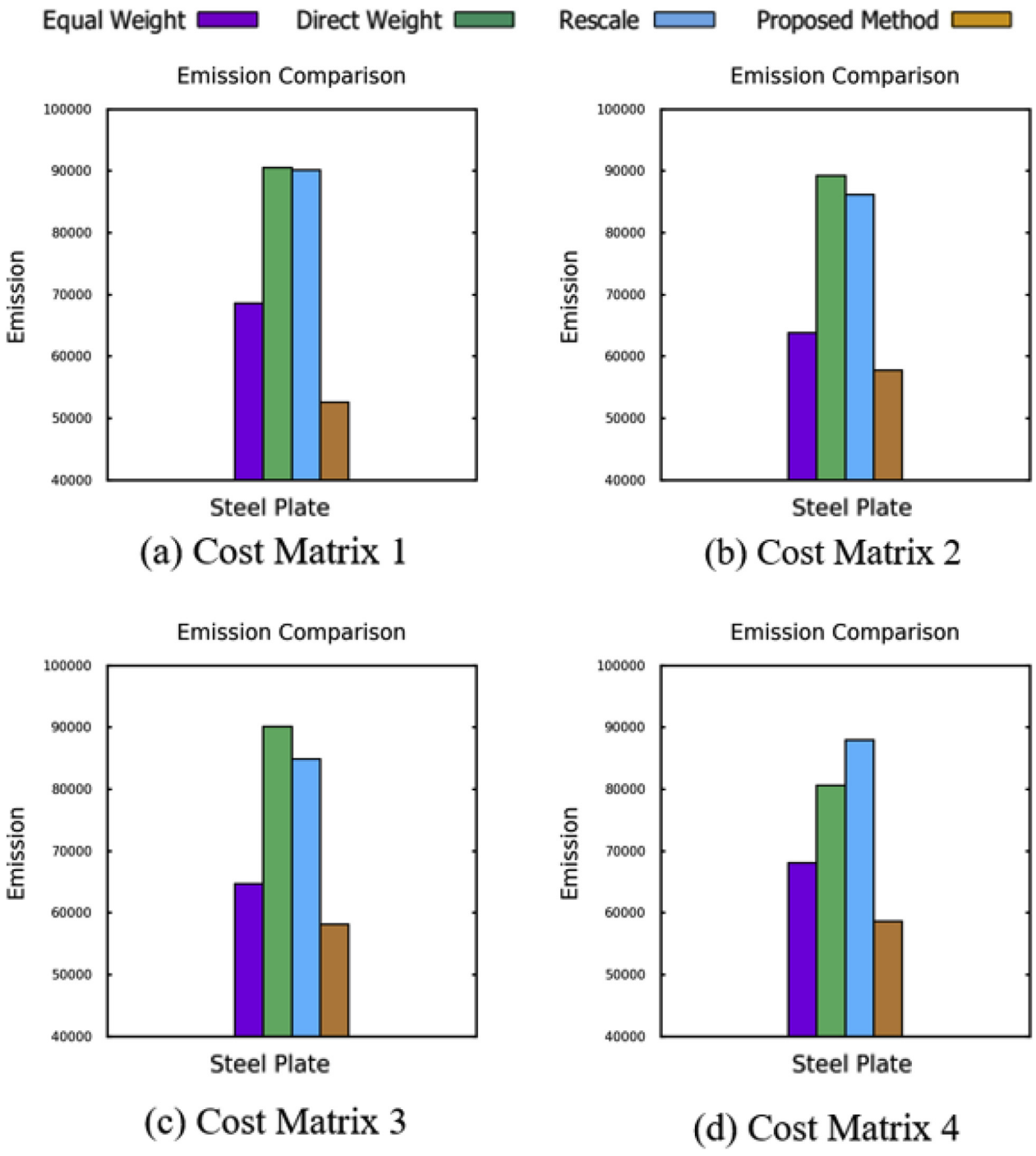

Fig. 12. Carbon emission comparison between the proposed framework and three other weight algorithms for steel plate case.

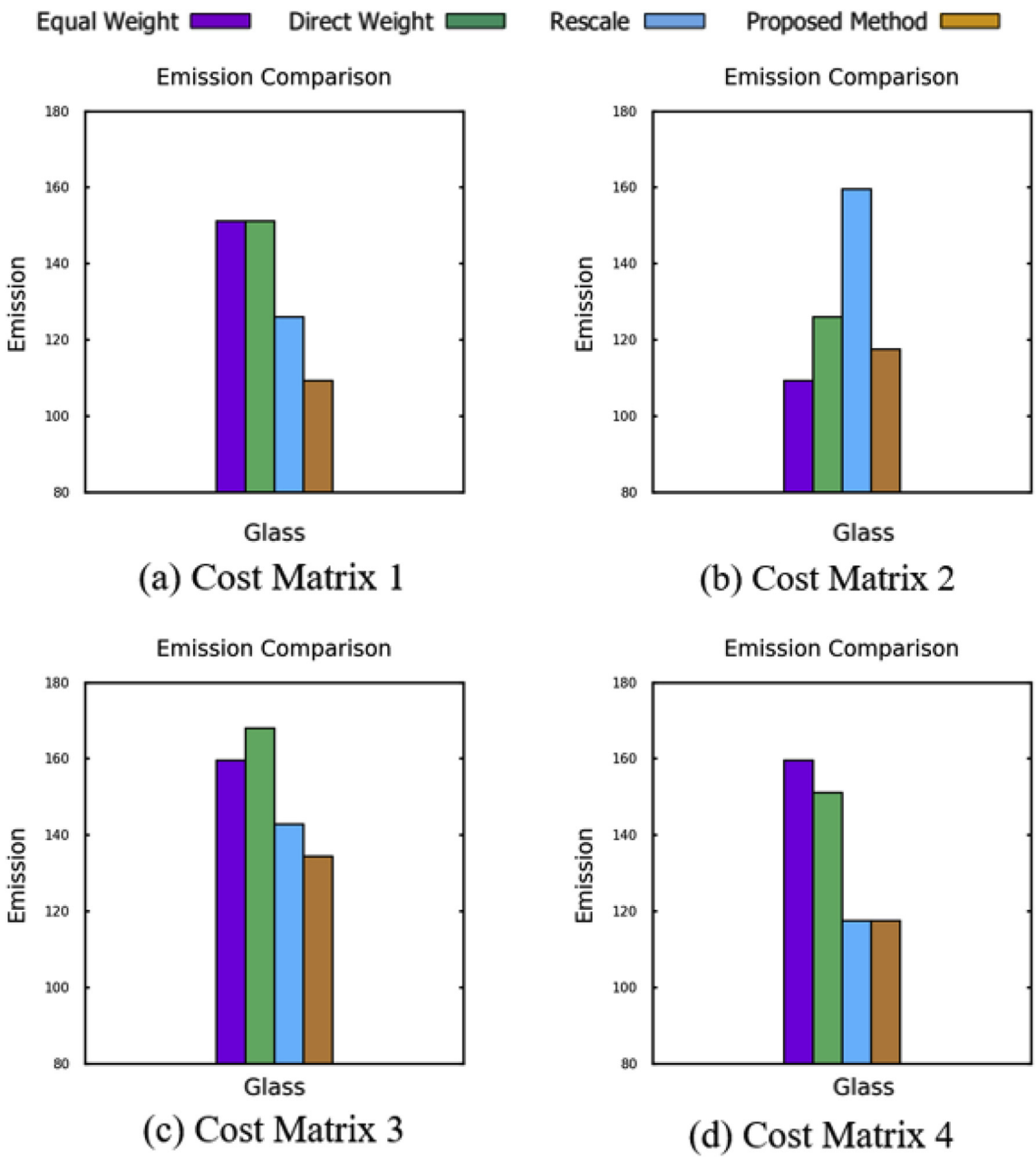

Fig. 13. Carbon emission comparison between the proposed framework and three other weight algorithms for the glass case.

nonlinear with respect to the extent of overestimation (i.e., the larger the overestimation extent, the larger the increase of the penalty cost). The overestimation cases may consist of a lot of counts with a small extent and thus the total number of overestimation can be larger. For cost matrix 4, the additional emission due to overestimation is equal to the emission from "rescale" method, which is resulted from the fact that the number of overestimation from two algorithms are the same. The possible reason may be the fact that underestimation is poorer than overestimation in cost matrix 4. Thus, the proposed algorithm prefers to reduce the number of underestimation, and so the number of overestimation may be a little bit higher.

3.4.4. 2nd hand car

For the case of 2nd hand car, it is assumed that the emission is mainly from the transportation of 2nd hand cars to dealer stores. There is no report on the average travelling distance of 2nd hand cars. We talked to several 2nd hand car dealers and found that 100 mile is a reasonable assumption. The average miles per gallon for heavy trucks is about 6.47 (Geotab, 2017). The CO_2 emission due to diesel consumption is 22.38 pounds per gallon (U.S. Energy Information Administration, 2014). Thus, the carbon emission due to each case of overestimation is 345.9 kg. The resultant comparison for four different cost matrices is illustrated in Fig. 14. It can be seen that the proposed method can result in the lowest amount of carbon emission using all types of cost matrices.

It can be seen from Figs. 11–14, in most cases, the additional emissions due to overestimation obtained by the proposed algorithm can outperform the ones from three other algorithms considering four different cost matrices. However, this superiority cannot be consistently guaranteed since the emission is not explicitly included in the objective function.

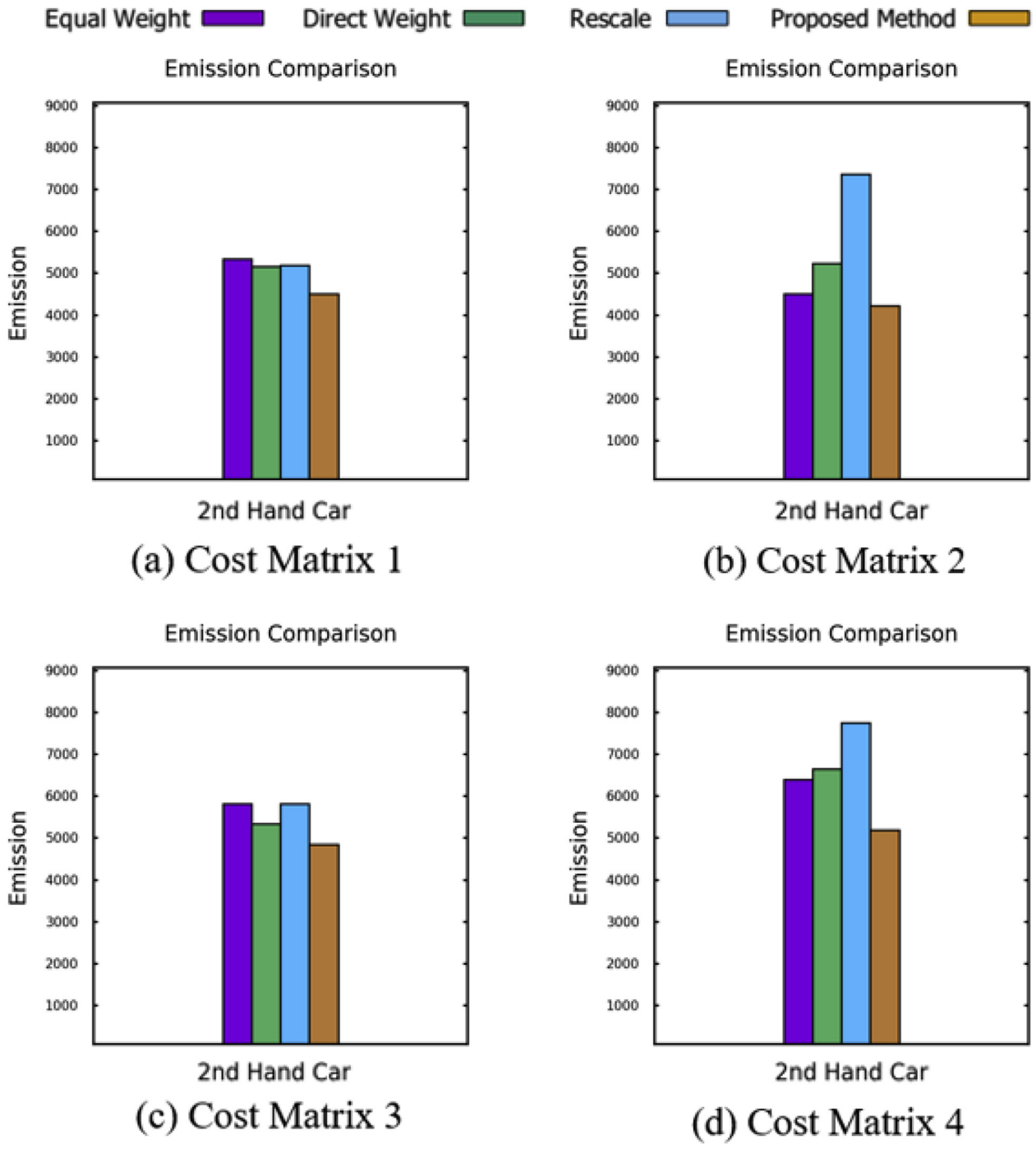

Fig. 14. Carbon emission comparison between the proposed framework and three other weight algorithms for 2nd-hand car case.

4. Discussion

The advantage and contribution of the proposed framework can be discussed from two perspectives as follows. First, the proposed framework is a generalized framework that offers a platform where various optimization methods for hyper parameter identification and different machine learning models with respective objectives for various concerns such as economic viability and environmental sustainability can be combined. Second, in the proposed framework, a surrogated loss weight matrix where each entry refers to a pseudo loss of misclassification is defined. The loss of misclassification is independent of the predictor output and thus the estimation of the loss matrix can be stabilized by excluding the disturbances from the update of predictor output per each iteration in model training.

The limitation of the proposed framework is that only economic performance in quality classification is considered and modeled as the primary objective, while the concerns in terms of emission, waste reduction, etc., are not explicitly integrated into the model. Although the resultant performance in the case study illustrates that the proposed algorithm can, in most cases, outperform the other algorithms in terms of the concern of environmental

sustainability, the consistency of this superiority cannot be guaranteed. In addition, through the performance comparison between the proposed model and "direct weight" algorithm, it reveals that the use of penalty costs of different misclassifications as the loss weight matrix in model training is less cost-effective. However, the relationship between the penalty cost and loss weight for each type of misclassification has not been explicitly revealed in this paper.

5. Conclusions and future work

In this paper, an integrated hyper learning framework for cost-imbalanced quality classification is proposed. A nested learning model with two-layer optimization algorithms that optimize the loss weight matrix and the parameters of predictors simultaneously, so as to improve the overall performance of learning with imbalanced loss. Case studies based on the datasets from different manufacturing and service providers are implemented. The cost-effectiveness of the proposed framework is validated through the comparison with the other three models considering four different penalty cost matrices. Meanwhile, the superiority of environmental-related performance measure (i.e., CO_2 emission) of the proposed framework is also demonstrated in most cases through the comparison with three other algorithms considering four different cost matrices.

For future research, four aspects can be considered to further extend the research contribution delivered by this paper. First, various concerns, especially from the perspective of cleaner production, can be used as the objectives in the proposed framework. Multi-objective formulation needs to be developed to involve the concerns of both economic viability and environmental sustainability to explore the balance between economic and environmental interests. Second, the interrelationship between the penalty cost and loss weight, depending on each misclassification needs to be investigated. Third, the effectiveness of the proposed algorithm on other datasets with different characteristics, such as the data with high dimensions, low sample sizes, and unbalanced class distributions can be analyzed. Fourth, other classifiers rather than random decision tree can be tested in the inner-layer of the framework, and, other meta-heuristic or analytical optimization methods rather than PSO can be examined and explored as the optimization tool for the out-layer of the framework to enhance the generalizability of the proposed framework.

References

Bohanec, M., Rajkovic, V., 1988. Knowledge acquisition and explanation for multi-attribute decision making. In: 8th Intl Workshop on Expert Systems and Their Applications, Avignon, France, pp. 59–78.

Brefeld, U., Geibel, P., Wysotzki, F., 2003. Support vector machines with example dependent costs. In: Proceedings of the 14th European Conference on Machine Learning. Cavtat-Dubrovnik, Croatia, pp. 23–34.

Breiman, L., Friedman, J.H., Olsen, R.A., Stone, C.J., 1984. Classification and Regression Trees. Wadsworth, Belmont, CA.

Breiman, L., 2017. Classification and Regression Trees. Routledge.

Buscema, M., Terzi, S., Tastle, W., 2010. A new meta-classifier. In: Proceedings of NAFIPS-2010, Toronto, Canada, pp. 1–7.

Camejo, J., Pacheco, O., Guevara, M., 2013. Classifier for drinking water quality in real time. In: Proceedings of 2013 International Conference on Computer Applications Technology (ICCAT).

Chai, X., Deng, L., Yang, Q., Ling, C.X., 2004. Test-cost sensitive naive Bayes classification. In: Proceeding of the 4th IEEE International Conference on Data Mining. Brighton, UK, pp. 51–58.

Cortez, P., Cerdeira, A., Almeida, F., Matos, T., Reis, J., 2009. Modeling wine preferences by data mining from physicochemical properties. Decis. Support Syst. 47 (4), 547–553.

De Clercq, D., Jalota, D., Shang, R., Ni, K., Zhang, Z., Khan, A., Wen, Z., Caicedo, L., Yuan, K., 2019. Machine learning powered software for accurate prediction of biogas production: a case study on industrial-scale Chinese production data. J. Clean. Prod. 218, 390–399.

Domingos, P., 1999. MetaCost: a general method for making classifiers cost-sensitive. In: Proceedings of the 5th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Diego, CA, pp. 155–164.

El Khaled, D., Castellano, N.N., Gazquez, J.A., García Salvador, R.M., Manzano-Agugliaro, F., 2017. Cleaner quality control system using bioimpedance methods: a review for fruits and vegetables. J. Clean. Prod. 140 (3), 1749–1762.

Elkan, C., 2001. The foundations of cost-sensitive learning. In: Proceedings of the 17th International Joint Conference on Artificial Intelligence. Seattle, WA, pp. 973–978.

Evett, Ian W., Spiehler, Ernest J., 1987. Rule Induction in Forensic Science. Central Research Establishment. Home Office Forensic Science Service, Aldermaston, Reading, Berkshire RG7 4PN.

Francis, L., 2017. The carbon footprint of a bottle of wine. https://www.sestrasystems.com/carbon-footprint-of-a-bottle-of-wine/.

Freidlin, M., Hu, W., 2011. On perturbations of generalized landau-lifshitz dynamics. J. Stat. Phys. 144, 978.

Geotab, 2017. The state of fuel economy in trucking. https://www.geotab.com/truck-mpg-benchmark/.

Green Ration Book, 2010. The cost of everyday living. http://www.greenrationbook.org.uk/resources/footprints-glass/.

Hu, W., Li, J., 2017. On the fast convergence of random perturbations of the gradient flow. arXiv:1706.00837, Retrieved from. https://arxiv.org/abs/1706.00837.

Jerald, J., Asokan, P., Prabaharan, G., Saravanan, R., 2005. Scheduling optimisation of flexible manufacturing systems using particle swarm optimisation algorithm. Int. J. Adv. Manuf. Technol. 25 (9), 964–971.

Kennedy, J., Eberhart, R.C., Shi, Y., 2001. Swarm Intelligence (Morgan Kaufmann: CA).

Khosravi, A., Koury, R.N.N., Machado, L., Pabon, J.J.G., 2018. Prediction of hourly solar radiation in Abu Musa Island using machine learning algorithms. J. Clean. Prod. 176, 63–75.

Li, C., Zhang, B., Luo, P., Shi, H., Wu, W.M., 2019. Performance of a pilot-scale aquaponics system using hydroponics and immobilized biofilm treatment for water quality control. J. Clean. Prod. 208, 274–284.

Ling, C.X., Yang, Q., Wang, J., Zhang, S., 2004. Decision trees with minimal costs. In: Proceedings of the 21st International Conference on Machine Learning. Banff, Canada, pp. 69–76.

Liu, M., Liu, C., Ge, M., Zhang, Y., Liu, Z., 2016. The online quality control method for reassembly based on state space model. J. Clean. Prod. 137, 644–651.

Liu, X.Y., Zhou, Z.H., 2006. The influence of class imbalance on cost-sensitive learning: an empirical study. In: Proceedings of the 6th IEEE International Conference on Data Mining. Hong Kong, China, pp. 970–974.

Lozano, A.C., Abe, N., 2008. Multi-class cost-sensitive boosting with p-norm loss functions. In: Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. Las Vegas, NV, pp. 506–514.

Rangel, C.S., Filho, R.D.T., Amario, M., Pepe, M., Puente de Andrade, G., 2019. Generalized quality control parameter for heterogenous recycled concrete aggregates: a pilot scale case study. J. Clean. Prod. 208, 589–601.

Rostami, H., Dantan, J.-V., Homri, L., 2015. Review of data mining applications for quality assessment in manufacturing industry: support Vector Machines. Int. J. Metrol. Qual. Eng. 6 (4), 59p.

Saitta, L. (Ed.), 2000. Machine Learning - a Technological Roadmap. University of Amsterdam, Amsterdam, the Netherlands.

Ting, K.M., 2002. An instance-weighting method to induce cost-sensitive trees. IEEE Trans. Knowl. Data Eng. 14 (3), 659–665.

Turney, P.D., 2000. Types of cost in inductive concept learning. In: Proceedings of the ICML-2000 Workshop on Cost-Sensitive Learning. Stanford, CA, pp. 15–21.

U.S. Energy Information Administration, 2014. How much carbon dioxide is produced by burning gasoline and diesel fuel? http://www.patagoniaalliance.org/wp-content/uploads/2014/08/How-much-carbon-dioxide-is-produced-by-burning-gasoline-and-diesel-fuel-FAQ-U.S.-Energy-Information-Administration-EIA.pdf.

Wang, Y., Li, L., 2014. A PSO algorithm for constrained redundancy allocation in multi-state systems with bridge topology. Comput. Ind. Eng. 68, 13–22.

World Steel Association, 2014. Steel's contribution to a low carbon future. World steel position paper. Retrieved from. http://www.worldsteel.org/dms/internetDocumentList/bookshop/Steel-s-contribution-to-a-Low-Carbon-Future-2014/document/Steel_s%20contribution%20to%20a%20Low%20Carbon%20Future, 202014.

Zadrozny, B., Elkan, C., 2001. Learning and making decisions when costs and probabilities are both unknown. In: Proceedings of the 7th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 204–213.

Zhang, Y., Zhou, Z.H., 2008. Cost-sensitive face recognition. In: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition. Anchorage, AK, vol.32, pp. 1758–1769, 10.

Zhou, Z.H., Liu, X.Y., 2006. On multi-class cost-sensitive learning. In: Proceeding of the 21st National Conference on Artificial Intelligence. Boston, MA, pp. 567–572.

Zhou, Z.H., Liu, X.Y., 2010. On Multi-class Cost-Sensitive Learning. Comput. Intell. 26, 232–257.