Available online at www.sciencedirect.com

ScienceDirect

Information Sciences 178 (2008) 2621–2638

INFORMATION SCIENCES
AN INTERNATIONAL JOURNAL

www.elsevier.com/locate/ins

Risk-sensitive loss functions for sparse multi-category classification problems

S. Suresh *, N. Sundararajan, P. Saratchandran

School of Electrical and Electronics Engineering, Nanyang Technological University, 50 Nanyang Avenue, Singapore 638768, Singapore

Received 14 July 2007; received in revised form 1 February 2008; accepted 6 February 2008

Abstract

In this paper, we propose two risk-sensitive loss functions to solve the multi-category classification problems where the number of training samples is small and/or there is a high imbalance in the number of samples per class. Such problems are common in the bio-informatics/medical diagnosis areas. The most commonly used loss functions in the literature do not perform well in these problems as they minimize only the approximation error and neglect the estimation error due to imbalance in the training set. The proposed risk-sensitive loss functions minimize both the approximation and estimation error. We present an error analysis for the risk-sensitive loss functions along with other well known loss functions. Using a neural architecture, classifiers incorporating these risk-sensitive loss functions have been developed and their performance evaluated for two real world multi-class classification problems, viz., a satellite image classification problem and a micro-array gene expression based cancer classification problem. To study the effectiveness of the proposed loss functions, we have deliberately imbalanced the training samples in the satellite image problem and compared the performance of our neural classifiers with those developed using other well-known loss functions. The results indicate the superior performance of the neural classifier using the proposed loss functions both in terms of the overall and per class classification accuracy. Performance comparisons have also been carried out on a number of benchmark problems where the data is normal i.e., not sparse or imbalanced. Results indicate similar or better performance of the proposed loss functions compared to the well-known loss functions.
Published by Elsevier Inc.

Keywords: Multi-category classification; Neural network; Risk-sensitive loss function; Cross-entropy; Satellite imaging; Micro-array gene expression

1. Introduction

In classification problems, the goal is often to correctly identify the class label for an observed input pattern. For any classifier, its performance is dependent on the chosen loss function. For a given problem, selection of proper loss function $V(x)$ is often difficult [8,29]. Different classification techniques in machine learning

* Corresponding author. Tel.: +65 8157 4822.
E-mail address: sundaram.suresh@hotmail.com (S. Suresh).

0020-0255/$ - see front matter Published by Elsevier Inc.
doi:10.1016/j.ins.2008.02.009

employ different loss functions to get better classification accuracy. For example, classifiers developed using adaptive boosting algorithms (AdaBoost) employ exponential ($\exp^{-V(x)}$) loss function [5,9], support vector machines employ loss functions of the form $(1 - V(x), 0)$ [8,29] and neural network classifiers employ mean square error minimization [7,22] or cross-entropy [23]. Recently, much attention has been paid to solve multi-category classification problems using neural networks [7,13,22,23,25] using different loss functions.

It is well known, [7,13,22,23,25] that the classifiers using mean square error loss function produce an output that approximates the desired posterior probability. It has also been shown [7,13,22,23,25] that other loss functions such as modified mean square error, cross-entropy also approximate the posterior probability. The conditions on the above loss functions such that the outputs of the neural classifiers closely approximate the posterior probability are discussed in [13,25]. However, in most of these works only binary classification problems have been considered. The central issue in machine learning is the extension of the results presented for the binary classification problem to multi-category classification problems.

Usually, the multi-category classification problem is first converted into multiple two class problem (one-versus-all or one-versus-one) and solved independently [1,3,12,20]. The one-versus-all method is then combined with a winner-take-all strategy to get the class number for unknown pattern. Similarly, the one-versus-one method is combined with max-wins or pairwise coupling strategy [12]. In these approaches, the posterior probabilities for each class in a multi-category problem are estimated using the binary classification problem for each category. In recent years, researchers have proposed an "all-together" approach to solve multi-category classification problem in one step by considering all the classes together [15,31]. In [14], incremental class learning approach is presented. Here, the classifiers learn classes one by one. However, such approaches require more training time and training samples per class. Recently in [7,31], the posterior probability estimation results for binary classification problems are extended to multi-category classification problem. A complete overview on loss functions and their behavior on decision performance has been presented in [6].

In general, the performance of neural classifiers is measured by their expected misclassification instances. To achieve better performance, the classifier employs loss functions that minimize the misclassification for all classes. Here, the loss functions assign equal weightage to the samples in every class. For many real world classification problems, minimization of misclassification alone is not adequate. For example, in sensitive applications such as medical diagnosis, false alarm detection in radar systems, financial and critical control applications, misclassification of samples in different classes have different consequences. It is often important to measure the confidence level in the class label prediction and corresponding risk associated with the action behind every classifier decision. This risk associated with the decision should be taken into account for better performance of the classifier. One way to handle this problem is to integrate the risk factor in the loss function.

Another important issue in classification problems is the availability of small number of samples and a high imbalance in the samples per class (also known as sampling bias). The small number of samples and a high imbalance in class will incur additional difficulty to know the distribution completely. This can result in an increased error in classification. Also, the overlap between the classes adds an additional complexity to the problem and may cause in the deterioration of the classifier performance.

Detailed explanation of the above issues are provided in [11]. In [16], a new approach based on statistical learning theory with a fixed cost of misclassification is presented by assuming the existence of hyperplanes that separates the data completely. In [18,4], support vector machine based classifiers are developed using unequal cost of misclassification to handle the above issues. In all these approaches, the cost of misclassification is fixed a prior. Fixing the cost of misclassification for a given problem is often difficult. Since, the classifier formulation includes the cost of misclassification, selection of these values influences its performance significantly. For multi-category classification problems with strong overlap between classes and high imbalance in samples per class, fixing the cost a priori is nearly impossible.

In order to overcome these difficulties in multi-category classification problems, in this paper, we introduce two new risk-sensitive loss functions. In the proposed loss functions, a risk factor is integrated with the existing cross-entropy and modified hinge loss functions. During the learning process, the risk factor is adapted using the cost of misclassification and approximate posterior probability of appropriate class of the given training samples. The estimated risk factor is used further in the learning process to reduce the error in classification. Here, the risk factors are estimated such that they increase with an increase in misclassification. For example, if the sample belonging to class c_i is misclassified into class c_j then the corresponding risk factor (m_{ij}) is increased adaptively to achieve better performance. The paper also presents an error analysis for some of the well-known loss functions along with the proposed risk-sensitive loss functions to get some insight into their working.

For this study, we consider only a single hidden layer neural network based classifier. The performance of the proposed risk-sensitive loss functions are compared with the well-known loss functions, viz., least square, modified least square and cross-entropy. The performance of the classifiers using these loss functions have been evaluated for real world multi-category classification problems such as a satellite image classification problem [26] and a micro-array gene expression based cancer detection problem [21]. In the cancer detection problem, the number of training samples is small and also there is an imbalance in the number of training samples per class. For the satellite image classification problem, we have deliberately created the sparse and imbalanced training set to study the performance of the proposed loss functions in a detailed way. The study results indicate that inclusion of risk factor in the loss function improves considerably both the overall as well as per class classification accuracy.

Further, to evaluate how the proposed loss functions perform on normal (not sparse) data sets, results from benchmark problems from standard database with different ranges of data sets, number of classes and number of features are also compared. They indicate that the proposed loss functions produce similar/better performance than the classifier developed using other well known loss functions.

The organization of this paper is as follows: Section 2 describes the problem formulation. In Section 3, we present some of the well known loss functions for multi-category classification problems and their limitations. The proposed risk-sensitive loss functions and their characteristics are discussed in Section 4. Performance evaluation of the proposed loss functions on a number of real world and standard benchmark problems are carried out in Section 5. Section 6 summarizes the main conclusions from this study.

2. Problem formulation

A multi-category classification problem can be stated in the following manner. Suppose we have the observation data, $\{(\mathbf{X}_i, \mathbf{Y}_i) : i = 1, 2, \ldots, N\}$, where $\mathbf{X}_i \in \Psi$ is a n-dimensional features of observation i and $\mathbf{Y}_i$ is its coded class label of dimension C, where C represents the total number of classes. If $\mathbf{X}_i$ is assigned to the class label c_k then kth element of $\mathbf{Y}_i$ is 1 and other elements are -1. Defining the jth element of vector $\mathbf{Y}_i$ as y_{ij} for convenience

$$y_{ij} = \begin{cases} 1 & \text{if } j = k \\ -1 & \text{otherwise}, \quad j = 1, 2, \ldots, C \end{cases} \tag{1}$$

The observation data are random variables and the observation $\mathbf{X}$ provides some useful information on the underlying probability distribution of the observation data to predict the corresponding class label with certain accuracy. Hence, the classification problem is to predict the coded class label $\mathbf{Y}$ of a new observation $\mathbf{X}$. This requires us to estimate a functional relationship between the class label and feature space from the known set of data

$$\mathbf{Y} = f(\mathbf{X}) \tag{2}$$

More generally, we call any function $f(\cdot)$ on the feature space a *classifier*. Its predicted value, (say) $\widehat{\mathbf{Y}}$ is a function of the data $\mathbf{X}$. Hence, we can write

$$\mathbf{Y} = \hat{f}(\mathbf{X}) + \epsilon_1 + \epsilon_2 \tag{3}$$

where $\hat{f}(\cdot)$ is an approximation to $f(\cdot)$ and it depends on the data $\{(\mathbf{X}_i, \mathbf{Y}_i) : i = 1, 2, \ldots, N\}$. Now, ϵ_1 is the approximation error and ϵ_2 is the estimation error and these are explained below.

The approximation error (ϵ_1) describes the closeness of $\hat{f}(\cdot)$ to the actual classifier $f(\cdot)$. The estimation error (ϵ_2) describes the closeness of the estimated input distribution to the underlying input distribution. The estimation error (ϵ_2) arises due to the finite nature of input data which makes it impossible to accurately represent the underlying input distribution. The influences of these errors on the performance of the classifier are discussed in detail in the later sections. The objective here is to find the best classifier function $f^*(\cdot)$ which minimizes both the approximation and estimation errors.

Since the input data and its coded class labels (training data) is available, we have considered a neural architecture for the classifier design. Neural network based classifiers are trained to find the joint probability distribution over the observation data for an accurate estimation of the desired coded class label **Y**. We assume here that the neural network can approximate the function and estimate the class label with the desired accuracy, (i.e., perfect model matching assumption). The optimal neural network classifier is defined as

$$\mathbf{Y} = N_f^*(\mathbf{X}, \mathbf{W}^*) \tag{4}$$

where $N_f^*(\cdot,\cdot)$ is the neural network approximation for the function $f(\cdot)$ and $\mathbf{W}^*$ is the optimal weight parameters.

Our goal is to find $N_f(\mathbf{X}, \mathbf{W})$, for different loss functions $L_f(N_f(\mathbf{X}, \mathbf{W}), \mathbf{Y})$ and study their performance for different real world problems. For a *finite* independently drawn training samples $(\mathbf{X}^s, \mathbf{Y}^s)$ and a given loss function, the empirical approximation error is given by

$$\epsilon_1 = \frac{1}{N}\sum_{i=1}^{N} L_f(N_f(\mathbf{X}_i^s, \mathbf{W}), \mathbf{Y}_i^s) \tag{5}$$

Finite training samples, a high imbalance in the number of samples per class and the population drift make it very difficult to describe the distribution completely. This results in an error called 'estimation error' (ϵ_2) given by

$$\epsilon_2 = E[N_f^*(\mathbf{X}, \mathbf{W}^*) - N_f^*(\mathbf{X}^s, \mathbf{W}^*)] \tag{6}$$

The purpose of training is to find the optimal functional map N_f^* and corresponding weight parameters $(\mathbf{W}^*)$ such that both these errors are minimum. In the case of non-convex loss functions, minimization of empirical classification error to find optimal weight matrices is a NP-hard problem. In literature, various convex loss functions are used to avoid the NP-hardness of the problem [24,30]. The basic idea behind using convex loss functions is to separate the classifier value for in-class ($\mathbf{Y}_i = 1$) and out-of-class ($\mathbf{Y}_i = -1$) as much as possible.

3. Some well-known loss functions

In this section, some well-known loss functions that are currently used in multi-category classification problems are analyzed and their limitations highlighted. For the sake of simplicity, the outputs of the classifier are taken to be $+1$ or -1.

- Least square (LS):

$$\mathrm{LS} = \frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{C}(\hat{y}_{ij} - y_{ij})^2 \tag{7}$$

- Modified least square (MLS):

$$\mathrm{MLS} = \begin{cases} 0 & \text{if } \hat{y}_{ij}y_{ij} > 1 \\ \frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{C}(\hat{y}_{ij} - y_{ij})^2 & \text{otherwise} \end{cases} \tag{8}$$

- Cross-entropy (CE):

$$\mathrm{CE} = \frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{C}\left[\frac{1+y_{ij}}{2}\log\frac{1+\hat{y}_{ij}}{2} + \frac{1-y_{ij}}{2}\log\frac{1-\hat{y}_{ij}}{2}\right] \tag{9}$$

All the above loss functions belong to the class of convex functions. Independent of the type of loss functions, the convexity assumption ensures that the minimization of error results in the optimal Bayesian classifier

[24,30]. The detailed discussion on the probabilistic bounds for a two-class problem using LS, MLS and CE loss functions are given in [30]. The study shows that the MLS loss function has better estimation capacity than the other loss functions.

It is well known that the minimum probability of error or the misclassification rate for a given problem is given by the optimal Bayes classifier. In general, the performance of any classifier is measured in comparison to a Bayesian classifier to see how close it is to the Bayesian classifier. In this context, for binary classification problem, closeness of a classifier to the Bayesian classifier using LS, MLS and hinge loss functions and the associated theoretical error bounds have been studied in [24,30]. Recently, in [28], necessary and sufficient conditions for the consistency of multi-category classification problems are presented for some well-known multi-category classification methods.

In the following section, a qualitative error analysis for the different loss functions is presented. It serves as a motivation for developing the risk-sensitive loss functions presented in this paper.

3.1. Error analysis of well-known loss functions

First, the basics of Bayesian classifier with appropriate mathematical notations are summarized as this sets the stage for our error analysis.

Let $\{c_1, c_2, \dots, c_C\}$ be the finite set of C distinct classes and $\mathbf{X}$ be the observed data of dimension n. Let $A = \{a_1, a_2, \dots, a_C\}$ be the finite set of possible actions taken by the classifier while assigning a class to the observed data. Finally, let $P(\mathbf{X}|c_j)$ be the class conditional probability density function of sample $\mathbf{X}$ belonging to class c_j and $P(c_j)$ be the prior probability that the data are drawn from the class c_j. Thus, the posterior probability $P(c_j|\mathbf{X})$ is given by

$$P(c_j|\mathbf{X}) = \frac{P(\mathbf{X}|c_j)P(c_j)}{P(\mathbf{X})} \tag{10}$$

where

$$P(\mathbf{X}) = \sum_{j=1}^{C} P(\mathbf{X}|c_j)P(c_j) \tag{11}$$

Let the sample originally belong to class c_j and suppose, we take an action a_i (deciding that the observed data belongs to class c_i), then the risk involved in taking action a_i is given by m_{ij}.

In decision theoretic terminology, $R(a_i|\mathbf{X})$ is known as conditional risk and is defined as

$$R(a_i|\mathbf{X}) = \sum_{j=1}^{C} m_{ij} P(c_j|\mathbf{X}) \tag{12}$$

Only the optimal Bayesian classifier can minimize the conditional risk $R(a_i|\mathbf{X})$ completely. In general, a classifier is defined by $A(\mathbf{X}, c_i)$ that indicates that the sample $\mathbf{X}$ is assigned to class c_i. The conditional risk associated with the classifier $A(\mathbf{X}, c_i)$ is

$$R_A = \sum_{j=1, j\neq i}^{C} m_{ij} P(c_j|\mathbf{X}) \tag{13}$$

The overall risk is defined as the expected risk associated with the decision rule $A(\mathbf{X}, c_i)$ and is given by

$$E_{\mathbf{X}}(R_A) = E_{\mathbf{X}}\left(\sum_{j=1, j\neq i}^{C} m_{ij}\right) P(c_j|\mathbf{X}) \tag{14}$$

Without loss of generality, we assume that the risk involved in taking action a_i is the same for all classes, i.e., the risk ($\mathbf{m}$) is a vector of dimension $C \times 1$. If we set $m_k = 1\ \forall (k = 1, 2, \dots, C$ and $k \neq i)$ and $m_i = 0$, then we have the standard 1–0 formulation (this indicates that one is minimizing the misclassification).

If we know the true posterior probability $P(c_j|\mathbf{X})$ then the optimal Bayes classifier is defined as $A(\mathbf{X}, c_{j^*})$ where j^* is given by

$$j^* = \arg \max_{i \in \{1,2,\ldots,C\}} m_i P(c_i|\mathbf{X}) \tag{15}$$

The resulting minimal overall risk (l_β) is called the Bayes risk and indicates the best performance that can be achieved for a given classifier. The overall risk is given by

$$l_\beta = E_{\mathbf{X}} \left(\sum_{j=1, j \neq j^*}^{C} m_j P(c_j|\mathbf{X}) \right) \tag{16}$$

The above analysis assumes that the input data distribution is known which implies that the input data size is large. If the input data is small, knowledge of the distribution is not possible.

In this paper we are dealing with sparse and unbalanced data and one has to estimate the posterior probability using the neural classifier. Here, we assume that the training and testing samples are drawn from the same distribution. Let $N_f(\mathbf{X}^s, \mathbf{W})$ be the neural classifier and $\mathbf{W}$ is the weight parameters describing the input–output relationship. The objective of the neural classifier is to estimate the posterior probability with finite training samples ($\mathbf{X}^s$). In this context, the neural classifiers try to minimize a loss function

$$L_f(N_f(\mathbf{X}^s, \mathbf{W}), \mathbf{Y}^s) \tag{17}$$

where $\mathbf{Y}$ is the coded class label for a given sample $\mathbf{X}^s$ and is the same as c_j. Here, the L_f represents the deviation between the predicted class and the actual class. For a multi-category classification problem, the expected risk R_N of the neural classifier is defined as

$$R_N = L_f(N_f(\mathbf{X}^s, \mathbf{W}), \mathbf{Y}^s) \tag{18}$$

It is known that Bayesian classifier produces the minimal risk (optimal) that we can achieve. Hence, the expected risk of the neural classifiers can be written as

$$R_N \geqslant l_\beta + \epsilon_1 + \epsilon_2 \tag{19}$$

where l_β is the minimum risk obtained by the Bayesian classifier, ϵ_1 and ϵ_2 are some positive constants representing the approximation and estimation errors, respectively.

The approximation error (ϵ_1) describes the closeness of the best neural classifier to the Bayesian classifier. This error is generally influenced by a class of functions (F_n) which represent the classifier and method of estimating the best neural classifier. The estimation error (ϵ_2) arises due to the finite training sample size which makes it impossible to accurately represent the underlying input distribution. Also, it is affected by the imbalance in number of samples per class in the training set. A minimum estimation error implies that the classifier represents the underlying distribution more accurately and achieves minimum risk in misclassification. Further, even if one knows accurately the distribution the approximation error gives an indication of how close the classifier is to the optimal Bayesian classifier. Also, the presence of finite training samples and imbalance in class affects the description of underlying distribution thereby leading to further increase in the approximation error.

The objective of the neural classifier is therefore to minimize both these errors and it is done by minimizing

$$N_f^*(\mathbf{X}^s, \mathbf{W}^*) = \arg \min_{N_f \in F_n} \left[\frac{1}{N} \sum_{i=1}^{N} L_f(N_f(\mathbf{X}_i^s, \mathbf{W}), \mathbf{Y}_i^s) \right] \tag{20}$$

where F_n is a class of continuous functions. Here, the minimization of errors depend on how the loss function (L_f) is defined. In the following section, we discuss how well the well known loss functions given in Eqs. (7)–(9) minimize both the approximation and estimation errors.

3.2. Least square (LS) loss function

In this case, the loss function uses the square of deviation from the coded output (y_j). It is well known [7,23] that the neural classifier with LS formulation directly estimates the posterior probability. In the neural classifier, the sample $\mathbf{X}^s$ is assigned to the class having the highest posterior probability. Comparing the LS formulation with the Bayes rule (Eq. (15)), it can be seen that the risk term ($\mathbf{m}$) is not considered in the LS formulation. Also, the LS formulation does not consider the effect of finite training samples and imbalance in the training set. The above issues considerably affect the estimation error and hence the LS formulation only minimizes the approximation error and neglects the estimation error.

3.3. Modified least square (MLS) loss function

Similar to LS formulation, MLS also approximates the posterior probability. In LS formulation, the classifier restricts the output to be $+1$ or -1, where as in the MLS formulation, the classifier allows the output to grow. The truncated output of the classifier can approximate the posterior probability better than the LS models [30]. Similar to LS, MLS also neglects the estimation errors due to finite training samples and imbalances in training data.

3.4. Cross-entropy (CE) loss function

For a two-class problem, it has been shown in [6,23] that the classifier developed using CE closely approximates the posterior probability. However, the CE loss function has the following drawback. If the true posterior probability $P(Y = i|\mathbf{X}^s)$ is 1 then the activation values of output neurons should be very high and positive to achieve zero approximation error, i.e., $(1 - \mathrm{e}^{-\mathbf{W}\mathbf{X}_h})/(1 + \mathrm{e}^{-\mathbf{W}\mathbf{X}_h})$ is 1 if and only if $|\mathbf{W}\mathbf{X}_h|$ is a very high value. Since the output of hidden neuron $\mathbf{X}_h$ is always between $+1$ or -1, the output weights of the neural classifier should be very high to achieve a small error. Similarly, when the true posterior probability $P(Y = i|\mathbf{X}^s)$ is zero, then the activation values of output neurons should be very high and negative to achieve zero approximation error. The growth of weights affects the performance of the classifier. Such a classifier model is not well-behaved [10]. But in an overall sense the CE has a smaller approximation error than the LS and MLS. By comparing CE formulation and Bayes rule (Eq. (15)), we can see that the CE does not consider the risk term in classification. Also, the effect of finite samples and imbalance in training samples are not accounted in the formulation and hence it minimizes only the approximation error ϵ_1 and does not minimize the estimation error ϵ_2.

The above error analysis for some well known loss functions clearly shows that they minimize only the approximation error and neglect the estimation error. They also do not consider the risk factor in the formulation. These issues affect the performance of the classifier considerably. To obtain a better classifier, one has to define a proper loss function which minimizes both the approximation and estimation errors. Hence, in the next section, we propose two new loss functions which take into account these two errors by incorporating a risk factor in the formulation.

4. Risk-sensitive loss functions

Risk-sensitive loss functions are defined such that both the approximation and estimation errors are taken into account. This leads to considering both the learning error and the decision risk in the formulation.

4.1. Risk-sensitive cross-entropy

First, we present a loss function based on entropy and misclassification risk. Consider an observation vector $\mathbf{X}^s$ that belongs to class c_k. The classifier model N_f predicts the class as c_j. The cost or risk involved in classifying the observation vector $\mathbf{X}^s$ into class c_j be m_{kj} and $m_{kj} \in \Re^+$. In a multi-class problem, the loss function should be defined such that the learning algorithm adapts the free parameters for minimizing both the approximation error and the risk in misclassification. Hence, we propose a new loss function called risk-sensitive cross-entropy (RSCE) which incorporates the effects of both the learning error and the risk factor and is given by

$$\mathrm{RSCE} = \frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{C} m_{kj}\left[\frac{1+y_{ij}}{2}\log\frac{1+\hat{y}_{ij}}{2} + \frac{1-y_{ij}}{2}\log\frac{1-\hat{y}_{ij}}{2}\right] \tag{21}$$

where m_{kj} is the risk factor and the index k is the true class label of observation feature vector $\mathbf{X}_i^s$. From the above equation, we can see that the loss function integrate the learning error and also the risk involved in misclassification. The RSCE loss function is effective in minimization of learning error and also improves the performance of the classifier model. This loss function directly estimates the posterior probability. Similar to cross-entropy, the RSCE also have weight saturation problem when the posterior probability is close to zero or one. The problem of weight saturation retards the search for a minimum in the error surface. Hence, we propose another loss function which overcomes the weight saturation problem and also have a minimum learning error.

4.2. Risk-sensitive hinge loss function

To overcome the above weight saturation problem, let us consider a neural network classifier with a linear activation function in the output layer. The neural classifier weights are adapted using the risk-sensitive hinge loss (RSHL) defined by

$$\mathrm{RSHL} = \begin{cases} -(m_{kj}+1)^2 y_{ij}\hat{y}_{ij}, & \text{if } y_{ij}\hat{y}_{ij} \leqslant -1 \\ (m_{kj} y_{ij}\hat{y}_{ij} - 1)^2, & \text{if } -1 \leqslant y_{ij}\hat{y}_{ij} \leqslant 0 \\ (y_{ij}\hat{y}_{ij} - 1)^2, & \text{if } 0 \leqslant y_{ij}\hat{y}_{ij} \leqslant 1 \\ 0 & \text{if } y_{ij}\hat{y}_{ij} \geqslant 1 \end{cases} \tag{22}$$

where m_{kj} is the risk factor and index k is the true class label of observation feature vector $\mathbf{X}_i^s$. Here, the hinge loss function presented in [30] is integrated with the risk factor to minimize the errors.

In the proposed RSHL function, if the predicted output and actual class label are different, then the RSHL values will be high due to the risk factor. The detailed explanation on the behavior of the RSHL is discussed in Section 4.4. The loss function allows the network output to grow beyond ± 1 and prevent the weight saturation. After the training process, the truncated outputs of the classifier approximate the posterior probability accurately. The truncated output is defined as

$$T(\hat{y}_{ij}) = \min(\max(\hat{y}_{ij}, -1), 1) \tag{23}$$

In general, for binary classification problems the class rule is obtained using the sign of real-valued decision function. In case of multi-category problems, we predict the class label (k) for a given sample $\mathbf{X}_i^s$ as

$$k = \arg \max_{j \in \{1,2,\ldots,C\}} \hat{y}_{ij} \tag{24}$$

In our formulation, the target vectors are coded between $+1$ or -1. Hence, the posterior probability of observation vector $\mathbf{X}_i^s$ is obtained as

$$\widehat{P}(c_k|\mathbf{X}_i^s) = \frac{T(\hat{y}_{ik}) + 1}{2}, \quad k = 1, 2, \ldots, C \tag{25}$$

The truncated output approximates the probability more effectively than the other loss functions [24,30].

4.3. Estimation of risk factor

The risk factor plays a vital role in minimization of error for classification problems by penalizing the misclassification heavily. The adaptation of risk factor in frequent intervals helps the classifier in minimizing the effect of imbalance in the training set. Now, we present the procedure to calculate the risk factor using estimated posterior probability.

Let us consider a multi-category classification problem with C classes. Let N_i be the number of training samples in class c_i and L be the number of epochs for developing the classifier model. The risk matrix is proportional to the number of misclassified patterns and cost of misclassification (β). During the training process, the risk matrix is adapted such that classifier model converges to the optimal Bayes classifier. The following steps are used to calculate the risk matrix:

1. Initialize the risk matrix to the cost of misclassification. Also, divide the training epochs into O blocks.
2. For a given training data set, adjust the weight matrix using the learning algorithm for L/O epochs. Now, test the network with the training data set and estimate the posterior probability for each pattern.

3. Compute the risk matrix

$$m_{kj} = \frac{\beta_j}{N_j} \sum_{i=1}^{N_j} \frac{\widehat{P}(c_k|\mathbf{X}_i^s) + \widehat{P}(c_j|\mathbf{X}_i^s)}{\widehat{P}(c_k|\mathbf{X}_i^s) + \epsilon}, \quad j = 1, 2, \ldots, C \tag{26}$$

where k is the actual class label and ϵ is a small positive real number.
4. Repeat the step 2 and 3 until training process is over.

4.4. Error analysis for risk-sensitive loss functions

Now, we present the error analysis of the proposed risk-sensitive loss functions.

4.4.1. Risk-sensitive cross-entropy

RSCE formulation (Eq. (21)) includes the effect of risk term in the formulation. In Bayes rule, the empirical risk term is assumed as a constant, whereas in the RSCE, the risk term is adapted based on the posterior probability during the training process. The adapted risk term helps in reducing the estimation error due to the imbalance in the training set as given below.

In RSCE formulation, the risk factor is adapted based on the estimated posterior probability as given in Eq. (26). Here, the cost of misclassification is initialized based on the number of samples in their respective classes. To understand the effect risk factor, we present two different scenarios, viz, (a) the sample $\mathbf{X}_i^s$ is classified correctly or (b) the sample $\mathbf{X}_i^s$ is misclassified.

Scenario (a): Here, the kth output of the neural classifier y_k is greater than or equal to one and all other output values are far less than y_k, i.e., the estimated probability for class c_k is close to one and for all other classes it is close to zero. From Eq. (26), we can see that the ratio $\frac{\widehat{P}(c_k|\mathbf{X}_i^s)+\widehat{P}(c_j|\mathbf{X}_i^s)}{\widehat{P}(c_k|\mathbf{X}_i^s)}$ is always close to one and hence, the risk factor remains the same. Since, the sample is classified correctly, the overall error is zero.

Scenario (b): Here, we assume that the neural classifier assigns the sample $\mathbf{X}_i^s$ to class c_j. Hence, the jth output of the neural classifier $\hat{y}_j$ is greater than or equal to one and all other outputs are far less than $\hat{y}_j$. The estimated probability for class c_j is close to one and for the other classes it is close to zero. In this situation, the ratio $\frac{\widehat{P}(c_k|\mathbf{X}_i^s)+\widehat{P}(c_j|\mathbf{X}_i^s)}{\widehat{P}(c_k|\mathbf{X}_i^s)}$ is a large number for class c_j and it is close to one for other classes, i.e., the risk factor is large for that particular class and it remains the same for other classes. Here, the adapted risk factor penalizes the misclassification heavily for that particular class.

The scenario (b) occurs frequently, if we have imbalance in the training set, i.e., fewer training samples in one or more classes. The presence of fewer samples in training data set does not describe the distribution completely. This leads to increase in the estimation error, i.e., increases the misclassification in that class. In the proposed loss function, the adaptation of risk factor during training process helps the classifier in reducing the estimation error due to imbalance in training samples per class.

4.4.2. Risk-sensitive hinge loss function

For a two-class problem, it has been proved [30] that the truncated output of the classifier developed using hinge loss function can approximate the posterior probability accurately and the bound on approximation error is less than that of other loss functions. The RSHL is similar to the class of hinge loss functions, but incorporates the risk factor explicitly in the formulation. The defined RSHL function is given by

$$\mathrm{RSHL} = \begin{cases} 0 & \text{if } \hat{y}_j y_j \geqslant 1 \quad \text{(a)} \\ (y_j\hat{y}_j - 1)^2 & \text{if } 0 \leqslant y_j\hat{y}_j \leqslant 1 \quad \text{(b)} \\ (m_{kj}y_j\hat{y}_j - 1)^2 & \text{if } -1 \leqslant y_j\hat{y}_j \leqslant 0 \quad \text{(c)} \\ -(m_{kj}+1)^2 y_j\hat{y}_j & \text{if } y_j\hat{y}_j \leqslant -1 \quad \text{(d)} \end{cases} \tag{27}$$

The RSHL has different effects for different regions of error. Here, the posterior probability is obtained using Eq. (25) and class number is predicted using Eq. (24). On analyzing RSHL, it can be seen that

Table 1
Loss functions and the errors they minimize

Loss function	Approximation error (ϵ_1)	Estimation error (ϵ_2)
LS	$\surd$	–
MLS	$\surd$	–
CE	$\surd$	–
RSCE	$\surd$	$\surd$
RSHL	$\surd$	$\surd$

a. approximation error is zero when the predicted network output ($\hat{y}_j$) is greater than or equal to the coded output (y_j);
b. approximation error is equal to the square of deviation when the predicted output is not equal to the coded output but both have the same sign;
c. when the sign of the predicted output and coded output are different, then the risk factor is included to take care of the risk in misclassification;
d. when $\hat{y}_j y_j < -1$ the classifier penalizes the misclassified samples linearly.

In our formulation, the risk factor is different for different classes and the loss function penalizes the misclassified samples heavily. Furthermore, in RSHL, the risk factor is included only when the sample is misclassified whereas in RSCE risk factor is considered in all situations. Also, the adaptation of risk factor in RSHL minimizes the estimation error due to imbalance in training data. Thus, RSHL minimizes both errors ϵ_1 and ϵ_2. In summary, Table 1 shows at a glance all the loss functions and the errors they minimize.

In the next section, the performance of the proposed risk-sensitive loss functions are evaluated on a number of benchmark and real world problems.

5. Experimental results

In this section, we compare the performance of the proposed risk-sensitive loss functions with other general loss functions widely used in classification problems. First, we study the effect of newly developed risk-sensitive loss functions for multi-class problems with sparse data conditions. For this purpose, we consider the practical satellite image classification problem [26]. The satellite image classification problem has large number of samples for model development but it has only 6 input features. Since, the classes have strong overlap with other classes and difficulty in obtaining the ground truth (original class number) increases the complexity in model development. Using this satellite data set, we artificially create the imbalance in the training set and study the performance of the loss functions. Next, we consider a micro-array gene expression based cancer classification problem [21]. Here, fewer number of samples available for model development and also have high imbalance in the training/testing data set. The micro-array gene expression based cancer classification problem has 16063 features, but the number samples available for development of classifier model is only 199. The presence of very few training samples and large number of input features affects the performance of the classifier model. Finally, we also compare the performance of the loss functions for different real world classification problems from UCI Machine Learning Repository (MLR) [2]. In our experimental study, the data set is divided into training and testing sets. For each data set, the optimal number of hidden neurons required to minimize the loss functions are calculated using the constructive and destructive procedure presented in [27]. The performance of the multi-layer perceptron based classifier model developed using new loss functions are also compared with support vector machines (SVM).

5.1. Effect of imbalance in training data on classification – satellite image classification problem

In order to study the effect of imbalance in the training set, first we look at a classification problem with balanced training data and then artificially create the imbalance in the training set and study the performance of proposed risk-sensitive loss functions. For this purpose, we have taken a satellite image classification problem. For this, we consider a portion of high resolution, multi-spectral Landsat 7 Thematic-Mapper images acquired form southern region of India [26]. The wavelength range for the thematic mapper sensor is from the visible, through the mid-IR, into the thermal-IR portion of the electromagnetic spectrum. The portion of Landsat image is $15 \times 15.75\ \text{km}^2$ (500×525 pixels) and has 30 m spatial resolution, which corresponds to pixel spacing of 30 m. The satellite image is radio-metrically and geometrically corrected using the satellite model and platform/ephemeris information. The image is further rotated and aligned to a user-defined map projection, using ground control points to improve the satellite model. The color composite image of the study area and corresponding ground truth are given in Fig. 1. The satellite image was taken when there is no cloud. Even though the image was taken during no cloud day, small and minor cloud may present but not visible in the image. The presence of small and minor cloud may affect the performance of the classifier models.

The aim of the study to develop neural network classifier to distinguish the nine classes described in the Fig. 1. The nine classes are defined based on the Level-II standards and the map obtained using this approach can be directly used for practical applications. The ground truth is prepared for the Landsat image by visual inspection, auxiliary data (maps and aerial photography) and field work. For development of supervised neural classifiers, it is necessary to define the classes and select some of the region in the image for training of each class.

The training data for each class are selected from the region where the field work has been carried out [26]. The ground truth obtained for 10,000 pixels from field work are used to develop neural classifier models. The classifier models are validated using the remaining pixels. The learning rate and number of epochs used in this study are 0.003 and 10,000, respectively. In this problem, the cost of misclassification is different for different classes. Since, the first three classes belongs to forest group, the cost of misclassification is set as 10. Similarly, the agriculture, grass and bare-ground class belongs to farms group. Hence, the cost of misclassification is set as 15. Since, fewer number of samples is available for shadow class and also has strong overlap with urban and bare-ground classes; the cost of misclassification is set as 20. For other classes, the cost of misclassification is set as 5.

The classification/confusion matrix (Q) is used to obtain the statistical measures for both the class-level and global performances of the classifier. Class-level performance is indicated by the percentage classification which tells us how many samples belonging to a particular class have been correctly classified.

The percentage classification η_i for class c_i is

$$\eta_i = \left(\frac{q_{ii}}{N_i^{\text{Te}}}\right) \times 100 \tag{28}$$

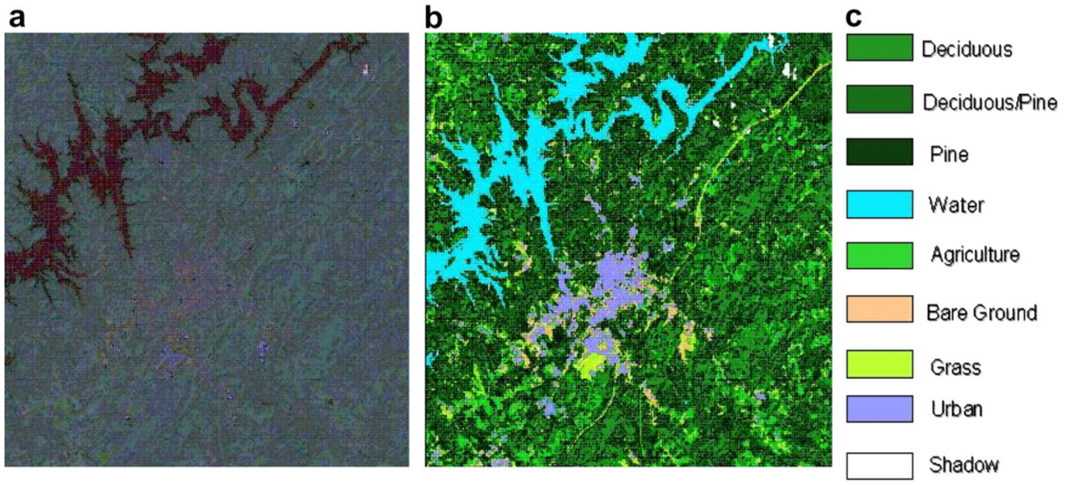


Fig. 1. Typical satellite image and its ground truth: (a) satellite image, (b) ground truth and (c) false color for each class.

where q_{ii} is the number of correctly classified samples and N_i^{Te} is the number of samples for the class c_i in the testing data set. The global performance measures are the average (η_{a}) and overall (η_{o}) classification efficiency [17], which are defined as

$$\eta_{\mathrm{a}} = \frac{1}{C}\sum_{i=1}^{C}\eta_i \tag{29}$$

$$\eta_{\mathrm{o}} = \frac{1}{N^{\mathrm{Te}}}\sum_{i=1}^{C}q_{ii} \tag{30}$$

where N^{Te} is the number of samples in the testing set.

The number of hidden neurons and global performance measures for different classifier models are given in Table 2. The neural network classifier results are also compared with the SVM. In this paper, we use multi-class SVM with Gaussian kernels. The optimal values of cost (c) and kernel width (γ) are obtained using 5-fold cross-validation. The number of support vectors and global performance measures are also given in Table 2. From the table, we can see that the neural network classifier developed using the proposed loss function performs slightly better than the SVM and the other classifiers.

5.1.1. Imbalance in training data

To study the effect of training data imbalance in the classifier performance, we define the percentage imbalance factor (IF) in class c_i as

$$IF_i = \left(1 - \frac{N_i^{\mathrm{Tr}}}{\max\limits_{\forall j}\left(N_j^{\mathrm{Tr}}\right)}\right) \times 100 \tag{31}$$

In the previous study, the number of training samples selected form each class was balanced, i.e., 1000 samples per class. To study the effectiveness of the loss functions for imbalance in the training set, the percentage imbalance factor of 10%, 20%, 30%, 40%, 50%, 60% and 70% is introduced deliberately in the training sample for class-II (c_2). The class-level and global performances measures are recorded for different classifiers with imbalanced data sets. Fig. 2 shows the average classification efficiency using different loss functions with respect to the reduction in training samples for class-II. It should be noted that in this figure and also in all the following figures, the imbalance (x-axis) increases from right to left.

From Fig. 2, it can be seen that RSCE and RSHL produce better classifier performance compared to the other loss functions, especially when the imbalance is high. Also, the average classification efficiency for RSHL and RSCE are almost constant from 0% to 60% imbalance factor whereas for other loss functions it starts decreasing as the number of samples are reduced from 10% itself. Fig. 3 shows the overall efficiency and this trend is clearly seen in this figure. These figures clearly indicate that the proposed risk-sensitive loss functions perform better when there is an imbalance in the training data.

Although, the average and overall classification efficiency are global indicators of the classifier performances, the class-level classification performance provide more insight to understand the effect of imbalance in the training data. For example, 1% increase in average classification is due to 9% (9% because we have nine

Table 2
Results for satellite image classification problem – no imbalance

Method	Number of hidden neurons	Average efficiency (η_{a})	Overall efficiency (η_{o})
LS	32	90.2695	88.6173
MLS	28	91.1901	90.2363
CE	26	92.5042	90.7761
RSCE	26	92.1796	90.3633
RSHL	28	93.3481	92.9111
SVM	1298[a]	93.3201	92.5501

[a] Number of support vectors.

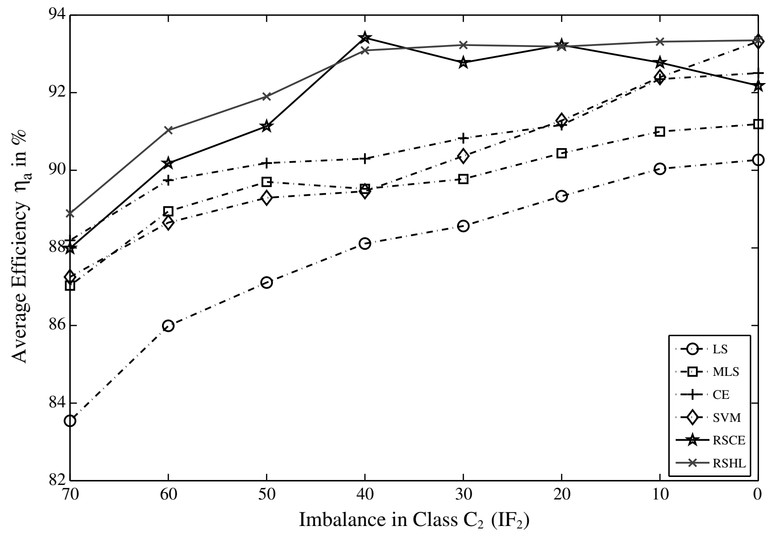


Fig. 2. Average efficiency for different training data with imbalance in c_2.

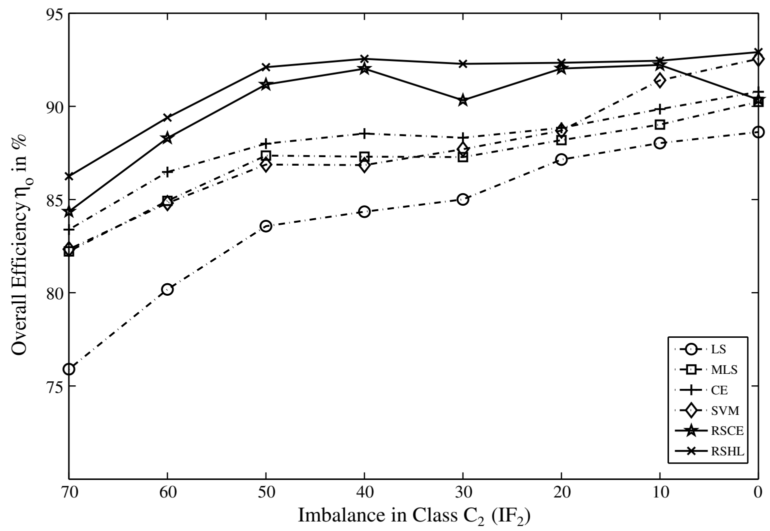
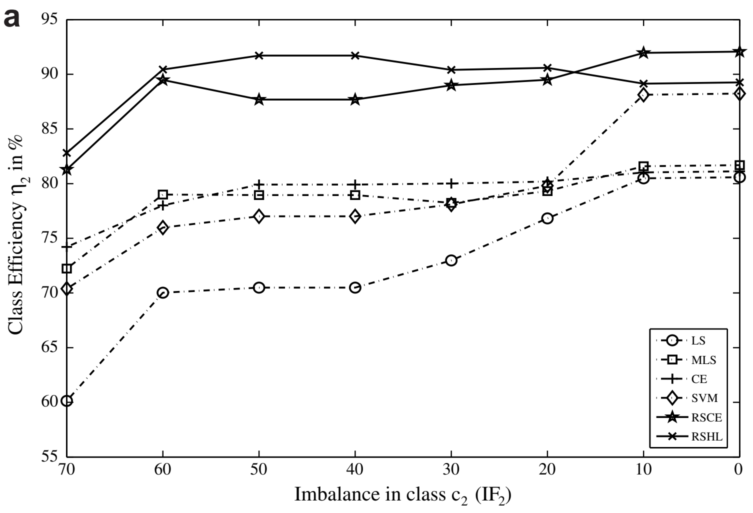


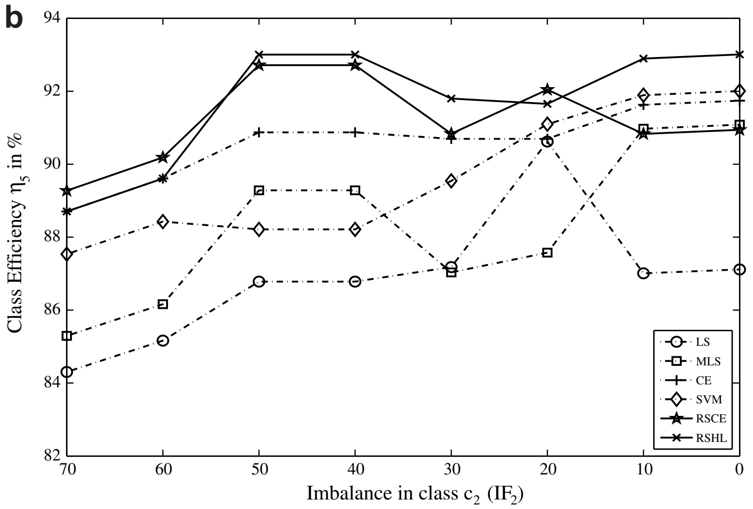
Fig. 3. Overall efficiency for different training data with imbalance in c_2.

classes in this problem) increase in class-level efficiency. Hence, we present in Fig. 4a–c the effect of training data imbalance on class-level performances.

In this problem, the reflectance characteristics of bands (feature vector) belonging to deciduous pine class (class-II) has strong similarity with the deciduous (class-I), pine (class-III), agriculture (class-V) and grass (class-VII). The other remaining classes are insensitive to the imbalance in training data (reduction of data in class-II). In Fig. 4 we have shown the effect of imbalance in training set (reduction of data in class-II) in class-level performance only for classes V and VII for illustration.

Fig. 4a shows the percentage classification for class-II when the training data is reduced for class-II. Similarly, the percentage classification for class-V and class-VII are shown in Fig. 4b–c, respectively. From Fig. 4a, we can observe that the performance of the classifier using RSHL and RSCE are almost constant with respect to imbalance in training data. Also, the class-level performance is approximately 10% more than the other loss functions. From Fig. 4b–c, we can observe that the classifiers using RSHL loss function have higher





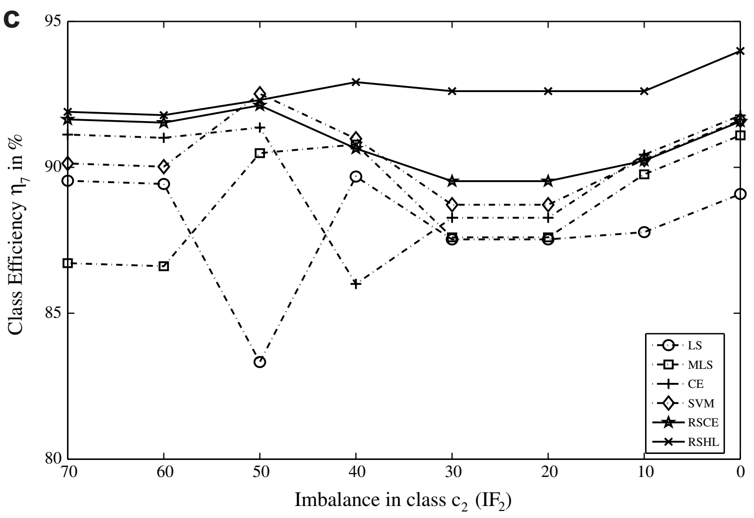


Fig. 4. Effect of imbalance in training data for class-level performance of classes overlapping with the class-II (a) class-II (b) class-V and (c) class-VII.

classification accuracy for class-V and class-VII. Similar behavior is observed in class-I and class-III. As mentioned earlier, the class-level performances indicate more insight into the performance of classifier. From the class-level performance, we can see that the efficiency for risk-sensitive loss functions is better than other models and approximately 10% increase in efficiency for class-II reflected in 1% increase global parameters. The simulation results clearly show that the risk-sensitive loss functions develop better classifier under imbalance in the training data.

5.2. Micro-array gene expression based cancer classification

Cancer detection and classification using standard clinical is a difficult process. Hence, in recent years biomedical research activities are focused towards identification of human cancer and molecular classification of cancer. Recently in [21], gene expression data is used to classify the human cancer types using support vector machines. Gene expression signatures database is a collection of micro-array data for human tumor and normal tissue specimens. The database is collected from 6 medical institutions for 14 different tumors. Each tissue specimens consist of 16,063 genes and the database has 198 primary tumor samples for 14 common tumor types. This classification problem is a typical example for large feature space and fewer number of samples for model development. Hence, in [19], recursive feature elimination method is used to identify most significant genes. Using the genes as inputs to the neural network, the classifier models are developed for different number of most significant genes. The simulation results clearly show that the performance increases with increase in number of significant genes as input to the network.

Hence, in our study, we use the complete gene features as input to the classifier models. Out of 198 patterns, 144 patterns are randomly selected for training and 54 patterns are used for validation. From the training data one can easily observe that the percentage imbalance vary from 10% to 25%. The imbalance in the training data affects the performances of classifier model considerably. The imbalance in the training data and fewer training samples are quite common in the bio-informatics and medical diagnosis. For such problems, the proposed risk-sensitive loss functions can develop a better classifier models than the general loss functions. For this purpose, we develop the classifier models using different loss functions. The simulation studies are repeated for 20 times. The global performance parameters and percentage variance in overall efficiency for different models are given in Table 3. From the table, we can see that the multi-layer perceptron classifier model developed using RSHL has better performance than the other classifier models. As mentioned [21], the support vector machine based one-verses-all (OVA) method gives overall accuracy of 78% where as classifier models based on RSCE and RSHL functions provides better accuracy. Also, the percentage variance in overall accuracy for the RSHL is less than the other loss functions. This reflects the ability of the risk-sensitive loss functions to generate better model under imbalance in training data and fewer training patterns.

5.3. Classification problems from machine learning repository (MLR)

The problems considered from the repository include image segmentation, character recognition, medical and financial applications. The details of each database selected in our study are given in Table 4. The performance of the classifier model developed using new risk-sensitive loss functions are compared with well known loss functions.

Table 3
Results for micro-array gene expression based cancer classification problem – imbalanced data

Method	Number of hidden neurons	Average efficiency (η_a)	Overall efficiency (η_o)	Percentage variance
LS	500	73.271	72.413	7.231
MLS	550	77.092	75.184	4.561
CE	450	79.910	76.213	3.921
RSCE	550	83.392	81.213	4.727
RSHL	600	85.216	83.125	3.521
SVM in [21]	–	–	78	–

Table 4
Database selected for studying the effect of loss functions

Sl. No.	Database name	No. of features	No. of patterns	No. of classes
1	Fisher's Iris (FI)	4	150	3
2	Thyroid gland (TG)	5	215	3
3	Wine recognition (WR)	13	178	3
4	Optical digit (OD)	64	5620	10
5	Image segment (IS)	19	2310	7
6	Multiple feature (MF)	649	2000	10

Table 5
Comparative performance of six data sets selected from MLR – no imbalance

Database		General loss function			Risk-sensitive functions	
		LS	MLS	CE	RSCE	RSHL
FI	H	16	12	10	12	16
	η_a	96.92	97.12	97.89	98.23	98.12
	η_o	96.27	96.18	97.12	97.88	97.81
TG	H	30	28	16	21	24
	η_a	88.23	91.74	93.61	97.89	98.88
	η_o	87.38	90.27	92.56	96.72	98.05
WR	H	44	32	36	36	38
	η_a	91.72	93.27	94.23	95.99	97.12
	η_o	90.01	92.09	93.98	95.42	96.15
OD	H	110	96	100	90	94
	η_a	92.23	94.51	96.67	96.78	95.99
	η_o	91.54	93.23	95.21	95.32	94.87
IS	H	55	75	45	50	60
	η_a	79.23	93.01	84.78	88.92	90.12
	η_o	77.38	80.27	82.56	86.72	88.05
MF	H	1370	1610	910	1210	1080
	η_a	97.01	98.41	98.32	98.44	98.44
	η_o	96.78	97.23	97.45	97.37	97.46

Now, we consider the six databases selected from the MLR, where the equal number of samples available for each class in the training set. The classifier models are generated by splitting the database into training and testing set. The learning rate (η) and maximum number of epochs selected for our simulation studies are 0.03 and 10,000, respectively.

For the first three standard classification problems, the cost of misclassification (β_i) for each class is selected as 5. For OD and MF data sets, the cost of misclassification for each class is 10 and it is 15 for IS data set. Initial minimal network configuration is selected using thumb rule, i.e., initial number of hidden neurons is equal to sum of input and output neurons. The optimal number of hidden nodes and testing accuracy are calculated. The global performance parameters and optimal number of hidden neurons (H) results are presented in Table 5. The simulation results show that the risk-sensitive loss function performs better than the other models in case of image segmentation problem. From the table, we can see that the classifier models developed using the proposed risk-sensitive loss functions performs slightly better than the other loss functions for other data sets.

6. Conclusions

In this paper, two new risk-sensitive loss functions have been proposed for multi-category classification problems where the training data is sparse and/or highly imbalanced. Using these risk-sensitive loss functions, neural network based classifiers have been developed. The adaptive risk factor in the risk-sensitive loss functions helps to find the best classifier (which minimizes both the approximation and estimation errors) closer to the optimal Bayesian classifier. The performances of the risk-sensitive loss functions are evaluated using two real world problems where there is a high imbalance and sparsity in the training data. The results clearly indicate that the proposed risk-sensitive loss functions perform better than the well known loss functions in the literature. Also, the study has been carried out on bench-mark problems where there is no imbalance or sparsity. The results indicate that the performance for the classifier developed using risk-sensitive loss functions is similar or better than that of the most commonly used loss functions. In summary, these results clearly show that the proposed risk-sensitive loss functions can handle problems with fewer training data and imbalance.

Acknowledgement

The authors would like to thank the reviewers for their valuable comments and suggestions, which have improved the paper.

References

[1] E.L. Allwein, R.E. Schapire, Y. Singer, Reducing multiclass to binary: a unifying approach for margin classifiers, Journal of Machine Learning Research 1 (2000) 113–141.
[2] A. Asuncion, D.J. Newman, UCI Machine Learning Repository, 2007. URL <http://www.ics.uci.edu/~mlearn/MLRepository.html>.
[3] E. Bauer, R. Kohavi, An empirical comparison of voting classification algorithms: bagging, boosting and variants, Machine Learning 36 (1–2) (1999) 105–142.
[4] M.P.S. Brown, W.N. Grundy, D. Lin, N. Cristianini, C.W. Sugnet, T.S. Furey, M. Ares, D. Haussler, Knowledge-based analysis of microarray gene expression data by using support vector machines, Proceedings of the National Academy of Sciences 97 (1) (2000) 262–267.
[5] A. Christmann, I. Steinwart, On robustness properties of convex risk minimization methods for pattern recognition, Journal of Machine Learning Research 5 (2004) 1007–1034.
[6] A. Cichocki, R. Unbenhauen, Neural Networks for Optimization and Control, Wiley, Baffins Lane, UK, 1993.
[7] J. Cid-Sueiro, J. Arribas, S. Urban-Munoz, A.R. Figueiras-Vidal, Cost functions to estimate a posteriori probabilities in multi-class problems, IEEE Transactions on Neural Networks 10 (3) (1999) 645–656.
[8] N. Cristianini, J.S. Taylor, An Introduction to Support Vector Machines, Cambridge University Press, Cambridge, UK, 2000.
[9] Y. Freund, R.E. Schapire, A decision theoretic generalization of on-line learning and an application to boosting, Journal of Computer and System Sciences 55 (1) (1997) 119–139.
[10] S. Geman, E. Bienenstock, Neural networks and the bias/variance dilemma, Neural Computation 4 (1) (1992) 1–58.
[11] D.J. Hand, Construction and Assessment of Classification Rules, John Wiley and Sons, Chichester, England, 1997.
[12] T. Hastie, R. Tibshirani, Classification by pairwise coupling, Annals of Statistics 26 (2) (1998) 451–471.
[13] R.G.J.W. Miller, P. Smyth, On loss functions which minimize to conditional expected values and posterior probabilities, IEEE Transactions on Information Theory 39 (4) (1993) 1404–1408.
[14] M. Jacek, L. Shastri, Incremental class learning approach and its application to handwritten digit recognition, Information Sciences 141 (3–4) (2002) 193–217.
[15] C. Jiang, J. Wu, Y. Liang, Multi-category classification by least squares support vector regression, Lecture Notes in Computer Science 3496 (1) (2005) 863–868.
[16] G. Karakoulas, J. Shawe-Taylor, Optimizing classifiers for imbalanced training sets, in: Proceedings in Advances in Neural Information Processing Systems II, 1999.
[17] J.K. Kishore, L.M. Patnaik, V. Mani, V.K. Agrawal, Application of genetic programming for multi-category pattern classification, IEEE Transactions on Evolutionary Computation 4 (3) (2000) 242–258.
[18] Y. Lin, Y. Lee, G. Wahba, Support vector machines for classification in nonstandard situations, Machine Learning 46 (1–3) (2002) 191–202.
[19] R. Linder, D. Dew, H. Sudhoff, D. Theegarten, K. Remberger, S.J. Poppl, W. Wagner, The subsequent artificial neural network approach might bring more classificatory power to ANN-based DNA micro-array analyses, Bioinformatics 20 (18) (2004) 3544–3552.
[20] P. Lingras, C. Butz, Rough set based 1-v-1 and 1-v-r approaches to support vector machine multi-classification, Information Sciences 177 (8) (2007) 3782–3798.
[21] S. Ramaswamy, P. Tamayo, R. Rifkin, S. Mukherjee, C. Yeang, M. Angelo, C. Ladd, M. Reich, E. Latulippe, P.J. Mesirov, T. Poggio, W. Gerald, M. Loda, S.E. Lander, Multi-class cancer diagnosis using tumor gene expression signatures, Proceedings of the National Academic of Sciences 98 (26) (2002) 15149–15154.
[22] M.D. Richard, R.P. Lippmann, Neural network classifiers estimate Bayesian a posteriori probabilities, Neural Computation 3 (4) (1991) 461–483.
[23] R. Rojas, A short proof of the posterior probability property of classifier neural networks, Neural Computation 8 (1) (1996) 41–43.

[24] L. Rosasco, E. De Vito, A. Caponnetto, M. Piana, Are loss functions all the same? Neural Computation 16 (5) (2004) 1063–1076.
[25] M. Saerens, Building cost functions minimizing to some summary statistics, IEEE Transactions on Neural Networks 11 (6) (2000) 1263–1271.
[26] S. Suresh, V. Mani, S.N. Omkar, N. Sundararajan, Multi-spectral satellite image classification using an evolving neural networks approach, Journal Aerospace Science and Technology 58 (4) (2006) 287–304.
[27] S. Suresh, S.N. Omkar, V. Mani, T.N. Guruprakash, Lift coefficient prediction at high angle of attack using recurrent neural networks, Aerospace Science and Technology 7 (8) (2003) 595–602.
[28] A. Tewari, P.L. Bartlett, On the consistency of multiclass classification methods, Journal of Machine Learning Research 8 (2007) 1007–1025.
[29] V. Vapnik, Statistical Learning Theory, Wiley, New York, 1998.
[30] T. Zhang, Statistical behavior and consistency of classification methods based on convex risk minimization, Annals of Statistics 32 (1) (2003) 56–85.
[31] T. Zhang, Statistical analysis of some multi-category large margin classification methods, Journal of Machine Learning Research 5 (2004) 1225–1251.