Regression Level Set Estimation Via Cost-Sensitive Classification

Clayton Scott, *Member, IEEE*, and
Mark Davenport, *Student Member, IEEE*

***Abstract*—Regression level set estimation is an important yet understudied learning task. It lies somewhere between regression function estimation and traditional binary classification, and in many cases is a more appropriate setting for questions posed in these more common frameworks. This note explains how estimating the level set of a regression function from training examples can be reduced to cost-sensitive classification. We discuss the theoretical and algorithmic benefits of this learning reduction, demonstrate several desirable properties of the associated risk, and report experimental results for histograms, support vector machines, and nearest neighbor rules on synthetic and real data.**

***Index Terms*—Cost-sensitive classification, learning reduction, regression level set estimation, supervised learning.**

I. Introduction

Consider a function $h : \mathbb{R}^d \rightarrow \mathbb{R}$ and a fixed value $\gamma \in \mathbb{R}$. The level set of h at level γ is the set

$$G^* = \{x : h(x) \geq \gamma\}.$$

In this paper, we consider the problem of estimating G^* from a training sample of noisy input/output pairs $(X_i, Y_i) \in \mathbb{R}^d \times \mathbb{R}, i = 1, \ldots, n$. Our only assumption on the training data is that they are realizations of (X, Y) such that h is the regression of Y on X, that is, $h(x) = E[Y \mid X = x]$.

The level set problem is relevant in a number of applications. Suppose for example that X represents demographic information of an individual and Y is income. While it may be instructive to estimate h, policy decisions often hinge on level sets such as those corresponding to the poverty line or certain tax brackets.

A second example is taken from medical decision making. Consider a cancer that is treated by either standard or aggressive chemotherapy, depending on a variable Y that characterizes the severity of the cancer. The choice of treatment is made by comparing Y to a threshold γ. This is the situation for osteosarcoma [1], where Y is the percent necrosis (cell death) in the tumor after an initial round of treatment, and $\gamma = 0.9$ by convention. The problem is that measuring Y involves an invasive biopsy. Suppose that X is a feature vector (whose acquisition is less invasive) collected from the patient, such as gene expression levels derived from an RNA microarray. Knowledge of the regression level set would allow for accurate treatment planning without a biopsy.

These two examples represent a much larger collection of potential applications. In a wide range of regression problems, if it is worthwhile to estimate the regression function h, it is also worthwhile to estimate certain level sets. Moreover, these level sets may be of ultimate importance. And in many classification problems, labels are obtained by thresholding a continuous variable. Thus, estimating regression level sets may be a more appropriate framework for addressing many problems that are currently envisioned in other ways.

Two naïve approaches to level set estimation are as follows. One is to use some method to estimate the regression function h and then threshold at γ. Another is to apply standard binary classification to the data $(X_i, \tilde{Y}_i)$, where $\tilde{Y}_i = I_{\{Y_i \geq \gamma\}} \in \{0, 1\}$. However, both approaches are unsatisfying. The first violates Vapnik's maxim: *When solving a given problem, try to avoid solving a more general problem as an intermediate step* [2]. The second approach ignores the information conveyed by the distance of the different response values from γ.

In this paper, we pose regression level sets estimation in terms of cost-sensitive classification. This approach lies somewhere between these two naïve approaches. It formulates the issue in terms of direct set estimation and thus bypasses the intermediate step of estimating h, while still accounting for response magnitudes. We argue that the cost-sensitive formulation provides a natural performance measure for the level set learning task.

Our approach can be described as a "learning reduction" from one supervised learning problem to another which is more fundamental or better understood. As discussed by Beygelzimer *et al.* [3], such reductions come with both algorithmic and theoretical benefits. From an algorithmic standpoint, we can estimate regression level sets using algorithms for cost-sensitive classification. Furthermore, as discussed below, cost-sensitive classification can be further reduced to conventional binary classification. Thus, standard methods such as support vector machines, decision trees, and nearest neighbors can be brought to bear on the problem. The ability to import and adapt existing classification algorithms is a principal advantage of our framework.

From a theoretical perspective, the analysis of algorithms for regression level set estimation can be deduced from well studied results for classification. For example, if we assume the regression function and noise are bounded, concentration inequalities like Hoeffding's can be applied as they are in the analysis of conventional classification algorithms [4], [5].

In previous work on regression level set estimation, Cavalier [6] demonstrated asymptotic minimax rates of convergence for piecewise polynomial estimators constructed with an excess mass criterion. Willett and Nowak [7], [8] also demonstrated minimax rates (for different smoothness classes) for estimators based on recursive dyadic partitions. A difference between these works and the present work is in the performance measure used to quantify the quality of an estimate. Our performance measure is the risk given by the expected misclassification cost, and its connection to the performance measures in the above cited works is spelled out in Section IV.

The paper is structured as follows. Section II reviews cost-sensitive classification and discusses the *costing* algorithm of [9]. Section III formally defines regression level set estimation and formulates a solution in terms of cost-sensitive classification. Section IV demonstrates several desirable properties of the risk proposed in Section III. Section V describes support vector and nearest neighbor algorithms for regression level set estimation. Section VI illustrates the proposed ideas with experiments on synthetic and real-world data. Conclusions and future work are discussed in Section VII.

II. Review of Cost-Sensitive Classification

Cost-sensitive classification problems can be grouped into two kinds: those with *class-dependent* costs, and those with *example-dependent* costs. In binary classification with class-dependent costs, for

Manuscript received February 21, 2006; revised September 2, 2006. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Tulay Adali. This work was supported in part by the National Science Foundation under Grant No. 0240058.

C. Scott is with the Department of Electrical Engineering and Computer Science, University of Michigan, Ann Arbor, MI 48109 USA (e-mail: cscott-at-eecs-dot-umich-dot-edu).

M. Davenport is with the Department of Electrical and Computer Engineering, Rice University, Houston, TX 77005 USA (e-mail: md-at-rice-dot-edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TSP.2007.893758

1053-587X/$25.00 © 2007 IEEE

example, false positives and false negatives incur fixed costs c_0 and c_1. The goal is to learn, from training data, a classifier with low expected misclassification cost (Bayes risk). This framework is appropriate when false positives and false negatives carry different penalties and was the subject of most early research on cost-sensitive classification (see [10], [11], [12], and references therein).

The second variant, example-dependent cost-sensitive classification is a generalization of the class-dependent cost problem and is the relevant framework for this paper. Consider random variables $(X, Y, C) \in \mathbb{R}^d \times \{0, 1\} \times \mathbb{R}$, where X represents a pattern, Y a class label, and C is the cost associated with misclassifying X when the true label is Y. Cost-sensitive classification seeks to minimize the expected cost (or *risk*)

$$R(G) = E[CI_{\{G(X) \neq Y\}}].$$

Here, we overload the notation G to refer to both a subset of $\mathbb{R}^d$ and the classifier $G(x) = I_{\{x \in G\}}$. Clearly, this formulation of cost-sensitive classification is more general than simply assigning fixed costs c_0 and c_1 to errors from the respective classes.

By casting level set estimation as example-dependent cost-sensitive classification, we avoid the need for function estimation and may instead rely on algorithms for direct set estimation. Fortunately, there are many algorithmic strategies for example-dependent cost-sensitive classification. In some simple settings, such as the histogram classifier discussed in Section VI, direct empirical risk minimization is possible. Many other algorithms for conventional classification can be modified (in a manner specific to the algorithm) to include example-dependent costs. Support vector and nearest neighbor methods are described concretely in Section V.

Even when direct modification is not possible, Zadrozny *et al.* [9] provide a general "black-box" procedure for reducing cost-sensitive classification to conventional (cost-insensitive) classification. Their approach is based on the realization that minimizing the expected cost is equivalent to minimizing the probability of error for an appropriately reweighted distribution. The idea is implemented algorithmically by a strategy termed *costing*. The label for a test point is based on a majority vote over a finite number (determined by the user) of classifiers. Each of these classifiers is obtained by running a conventional classification algorithm on a data set obtained by resampling the original data set. Resampling is accomplished with cost-proportionate rejection sampling. The importance of costing for the present work is that it allows a reduction of regression level set estimation to conventional classification, the most fundamental and widely studied supervised learning problem.

III. Application to Regression Level Set Estimation

The regression level set estimation problem is stated formally as follows. Let $(X, Y) \in \mathbb{R}^d \times \mathbb{R}$ be random variables. Assume that for some (unknown) function $h : \mathbb{R}^d \to \mathbb{R}$ we have $Y \mid X = x \sim h(x) + \epsilon$, where ϵ is zero mean noise with Lebesgue density $f(\epsilon)$. Although it is not reflected in the notation, the distribution of ϵ may depend on x. Let $\gamma \in \mathbb{R}$ be fixed. The goal is to estimate the level set

$$G^* = \{x : h(x) \geq \gamma\}$$

using only a training sample $(X_i, Y_i) \in \mathbb{R}^d \times \mathbb{R}, i = 1, \ldots, n$ of realizations of (X, Y).

Our proposal is to estimate G^* by reducing to a cost-sensitive classification problem as follows. Define $\tilde{Y} = I_{\{Y \geq \gamma\}}$ and $C = |\gamma - Y|$, and define the risk $R(G)$ of a set G to be the expected cost for cost-sensitive classification based on $(X, \tilde{Y}, C)$, as follows:

$$R(G) = E[|\gamma - Y| I_{\{G(X) \neq \tilde{Y}\}}]. \quad (1)$$

We can now apply cost-sensitive learning algorithms to the training data $(X_i, \tilde{Y}_i, C_i)$ and the outcome will be an estimate of the regression level set G^*.

IV. Properties of the Risk

In this section, we give credence to the proposed reduction by demonstrating certain desirable properties of the risk and relating it to the classification risk and other metrics for the level set problem. Let $\bar{G}$ denote the complement of G, and let $G \Delta G' := (G \cap \bar{G}') \cup (G' \cap \bar{G})$ denote the symmetric difference of G and G'. Let F_X denote the distribution of X. The following is proven in the Appendix.

Proposition 1: The excess risk can be expressed as

$$R(G) - R(G^*) = \int_{G \Delta G^*} |h(x) - \gamma| dF_X. \quad (2)$$

Corollary 1: The risk $R(G)$ is minimized by the level set $G^* = \{x : h(x) \geq \gamma\}$.

Proposition 1 establishes that the error associated with a misidentified point is proportional to the distance from the regression function at that point to the target level. This makes sense because points for which $|h(x) - \gamma|$ is large should be easier to classify, and any estimate that errs on such a point should be penalized more heavily than if it erred where $|h(x) - \gamma|$ is small. If we think of a classification problem where the labels are obtained by thresholding a continuous response variable, then points for which $|h(x) - \gamma|$ is small are "almost" in the other class anyway, so it is not as problematic to misclassify them.

The excess risk here is similar to the excess risk in conventional classification. Let $(X, \tilde{Y}) \in \mathbb{R}^d \times \{0, 1\}$ and denote $\eta(x) = E[\tilde{Y} | X = x]$. Recall that we identify subsets $G \subseteq \mathbb{R}^d$ with classifiers $G(x) = I_{\{x \in G\}}$. In conventional classification, the risk of a classifier is defined to be $\tilde{R}(G) = E[I_{\{G(X) \neq \tilde{Y}\}}]$. The Bayes classifier is a level set of η: $G^*(x) = I_{\{\eta(x) \geq 1/2\}}$. Furthermore, we have the formula [4]

$$\tilde{R}(G) - \tilde{R}(G^*) = 2 \int_{G \Delta G^*} |\eta(x) - 1/2| dF_X.$$

Conceptually, we may view conventional binary classification as a special regression level set estimation problem where the response variables have been "binarized" to 0 or 1. Conversely, from the discussion of the costing algorithm in Section II, we can think of regression level set estimation as a binary classification problem where the labels are obtained by thresholding the continuous responses and the probability mass of X has been reweighted in proportion to $|h(x) - \gamma|$.

Further theoretical guarantees are possible for certain cost-sensitive classification algorithms. For example, [9] relates the performance of their costing algorithm to the performance of the underlying conventional (cost-insensitive) classification algorithm. Translating their result to our setting gives the following.

Corollary 2: Let $B = E[|Y - \gamma|]$. Let S be a training sample drawn from (X, Y, C) and let S' be a sample derived from S by cost-proportionate rejection sampling as described in [9]. Let $\hat{G}$ be a classification algorithm based on S'. If the expected[1] probability of error of $\hat{G}$ is no more than ϵ, then the expected[2] value of $R(\hat{G})$ is no more than $B\epsilon$.

Let us now consider the connection between the excess risk $R(G) - R(G^*)$ and the performance measures studied in [6] and [7], [8]. We consider in particular the following two questions: 1) When does convergence to zero of one performance measure imply convergence of another? and 2) How useful from a practical standpoint are the various performance measures?

[1] This expectation is with respect to the random draw of $\boldsymbol{S'}$.

[2] This expectation is with respect to the random draw of $\boldsymbol{S}$.

The performance measures studied in [6] are i) the volume of the symmetric difference of the estimate and G^* and ii) the Hausdorff distance between the estimate and G^*. In general, it is not possible to answer question 1) without imposing some conditions on the underlying distribution. A complete characterization of such conditions is beyond the scope of this paper. Roughly speaking, however, let us suppose the distribution is reasonably well-behaved in the sense that a) the boundary of G^* is not too irregular, b) the distribution of X is mutually bounded by Lebesgue measure on some compact set, and c) h does not "flatten out" near the level γ. By a), if the Hausdorff distance tends to zero, so does the volume of the symmetric difference. By b) and equation (2), the volume of the symmetric difference tends to zero, so does our excess risk. Conversely, by c), if the excess risk tends to zero, then the volume of the symmetric difference also tends to zero. Also by c), it can be seen that convergence of the symmetric difference to zero implies convergence of the Hausdorff distance to zero. Thus, the different performance measures are asymptotically equivalent (under the stated assumptions) in the sense that an estimator that is consistent for one is consistent for the other.

With respect to question 2), our excess risk enjoys the clear advantage that it can be minimized without access to the true level set G^*, which is of course unknown in practice. Furthermore, $R(G)$ can be easily estimated given sufficient data. The two performance measures of [6], in contrast, cannot be easily estimated from data because of the dependence on G^*.

The performance measure employed in [7] and [8] is very similar to the cost-sensitive classification risk. In particular, they consider the risk (ignoring constants)

$$R_2(G) = E[|\gamma - Y| I_{\{G(X) \neq \tilde{Y}\}} - |\gamma - Y| I_{\{G(X) = \tilde{Y}\}}].$$

Conceptually, one may think of this risk as both penalizing errors and *rewarding* correct decisions in proportion to the distance to the regression function, whereas the cost-sensitive classification risk only penalizes errors. The two risks are related by

$$\begin{aligned} R_2(G) &= R(G) - R(\bar{G}) \\ &= E[|\gamma - Y| I_{\{G(X) \neq \tilde{Y}\}}] \\ &\quad - E[|\gamma - Y| I_{\{\bar{G}(X) \neq \tilde{Y}\}}] \\ &= E[|\gamma - Y| I_{\{G(X) \neq \tilde{Y}\}}] \\ &\quad - E[|\gamma - Y|(1 - I_{\{G(X) \neq \tilde{Y}\}})] \\ &= E[2|\gamma - Y| I_{\{G(X) \neq \tilde{Y}\}}] - E[|\gamma - Y|] \\ &= 2R(G) - E[|\gamma - Y|]. \end{aligned}$$

Note the last term does not depend on G. Consequently, the two risks are effectively the same. The advantage of our risk is the connection to cost-sensitive classification and the associated algorithmic benefits discussed earlier.

V. Algorithms

In this section, we describe the algorithms that are later applied in Section VI. For each class of algorithms, we describe three variants: cost-insensitive classification based on binarized response values, direct cost-sensitive classification, and regression function estimation followed by thresholding.

A. Support Vector Machines

Support vector machines (SVMs) are among the most effective methods for learning classifiers from training data [13]. Conceptually, we construct the support vector classifier in a two-step process. In the first step, we transform the $\mathbf{x}_i \in \mathbb{R}^d$ via a mapping $\Phi : \mathbb{R}^d \to \mathcal{H}$ where $\mathcal{H}$ is a high (possibly infinite)-dimensional Hilbert space. The intuition is that we should be able to separate these classes more easily in $\mathcal{H}$ than in $\mathbb{R}^d$. For algorithmic reasons, we choose Φ so that we can compute inner products in $\mathcal{H}$ through the *kernel* operator $k(\mathbf{x}, \mathbf{x}') = \langle \Phi(\mathbf{x}), \Phi(\mathbf{x}') \rangle_{\mathcal{H}}$.

In the second step, we determine a hyperplane in the induced feature space according to the max-margin principle, which states that, in the case where we can separate the two classes by a hyperplane, we should pick the hyperplane that maximizes the *margin*—the distance between the decision boundary and the closest point to the boundary. This hyperplane is then our decision boundary. Thus, if $\mathbf{w} \in \mathcal{H}$ and $b \in \mathbb{R}$ are the normal vector and affine shift (or *bias*) defining the max-margin hyperplane, then the support vector classifier is given by $f_{\mathbf{w},b}(\mathbf{x}) = \operatorname{sgn}(\langle \mathbf{w}, \Phi(\mathbf{x}) \rangle_{\mathcal{H}} + b)$.

The original formulation of the SVM [14], which we shall call the cost-*insensitive* SVM, can be stated as the following quadratic program:

$$\begin{aligned} \min_{\mathbf{w},b,\boldsymbol{\xi}} \quad & \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{n} \xi_i \\ \text{subject to} \quad & y_i(k(\mathbf{w}, \mathbf{x}_i) + b) \geq 1 - \xi_i \quad \text{for } i = 1, \ldots, n \\ & \xi_i \geq 0 \quad \text{for } i = 1, \ldots, n \end{aligned}$$

where $C \geq 0$ is a parameter that controls overfitting.

A simple modification to this formulation leads to the cost-*sensitive* SVM

$$\begin{aligned} \min_{\mathbf{w},b,\boldsymbol{\xi}} \quad & \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{n} c_i \xi_i \\ \text{subject to} \quad & y_i(k(\mathbf{w}, \mathbf{x}_i) + b) \geq 1 - \xi_i \quad \text{for } i = 1, \ldots, n \\ & \xi_i \geq 0 \quad \text{for} \quad i = 1, \ldots, n \end{aligned}$$

where $C \geq 0$ is again a parameter that controls overfitting, and the c_i are weights depending on the individual sample.

Support vector regression (SVR) solves

$$\min_{\mathbf{w},b} \quad \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{n} |y_i - (k(\mathbf{w}, \mathbf{x}_i) + b)|_\epsilon$$

where

$$|y_i - (k(\mathbf{w}, \mathbf{x}_i) + b)|_\epsilon = \max\{0, |y_i - (k(\mathbf{w}, \mathbf{x}_i) + b)| - \epsilon\}$$

is the so-called *ϵ-insensitive loss function*. This optimization problem is solved by considering the equivalent formulation

$$\begin{aligned} \min_{\mathbf{w},b,\boldsymbol{\xi},\boldsymbol{\xi}^*} \quad & \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{n} (\xi_i + \xi_i^*) \\ \text{subject to} \quad & (k(\mathbf{w}, \mathbf{x}_i) + b) - y_i \leq \epsilon + \xi_i \quad \text{for } i = 1, \ldots, n \\ & y_i - (k(\mathbf{w}, \mathbf{x}_i) + b) \leq \epsilon + \xi_i^* \quad \text{for } i = 1, \ldots, n \\ & \xi_i, \xi_i^* \geq 0 \quad \text{for } i = 1, \ldots, n. \end{aligned}$$

See [13] for further details.

B. Nearest Neighbors

The k nearest neighbor (k-NN) decision rule is a classic method for classification [15]. The rule assigns a label to a test point by taking a majority vote over the labels of the k training points that are closest according to a specified (usually Euclidean) metric. A cost-sensitive version is obtained by taking a "weighted" vote, where the weight assigned to a neighbor is the cost $c_i = |y_i - \gamma|$. Finally, k-NN regression assigns a response value to a test point by averaging the response values of the k closest training points. Note that for the nearest neighbor methodology,

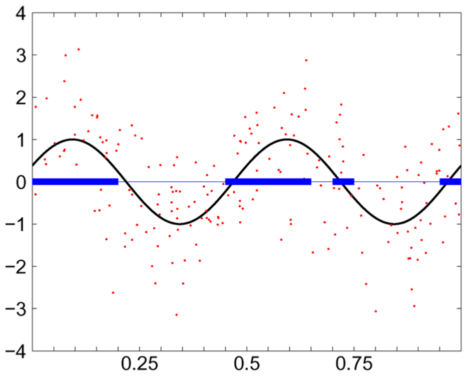

Fig. 1. Training sample of size 200 for the histogram experiment, together with the true regression function and a level set estimate.

the cost-sensitive classifier is identical to the thresholded regression estimate. Also note that all methods are equivalent when $k = 1$.

VI. Experiments

We consider first a simple, highly controlled simulation study using histograms, and second, experiments on real-world data using SVMs and nearest neighbor methods.

A. Simulation Study With Histograms

We compare three approaches for constructing histogram level set estimators: cost-sensitive classification, cost-insensitive classification, and regression function estimation followed by thresholding. Although histograms are very simple estimators, their simplicity does ensure a reasonably fair comparison. With more complicated models, confounding factors such as the selection of free parameters and differences in design criteria make it increasingly difficult to isolate the reasons for an algorithm's performance. Furthermore, synthetic data allows for very precise estimation of performance measures.

For this experiment, we are interested in the $\gamma = 0$ level set of $h(x) = \sin(4\pi x + \pi/8)$. The data are generated independently according to $X \sim \text{unif}[0,1]$ and $\epsilon \sim N(0,1)$. Fig. 1 depicts a typical realization, as well as the true regression function and a level set estimate.

The three estimates are based on a fixed partition of $[0,1]$ into 20 equally spaced bins. Each bin is assigned a label of 1 or 0 to indicate whether the bin does or does not belong to the estimate. For the cost-sensitive estimate (CS), the labels are determined by minimizing the empirical risk

$$\hat{R}_n(G) = \frac{1}{n}\sum_{i=1}^{n}(\gamma - Y_i)I_{\{X_i \in G, Y_i < \gamma\}} + (Y_i - \gamma)I_{\{X_i \notin G, Y_i \geq \gamma\}}.$$

For the cost-insensitive estimate (CI), the labels are assigned by minimizing the empirical probability of error. For the third method (REG), the regression function is estimated by a constant on each cell using an L^1 distortion. The constant is thus the median value of the response variables on the bin. An L^2 distortion was also considered, but this in fact leads to an estimate that is identical to CS, a coincidence stemming from the simplicity of the histogram estimate.

The experiment consisted of generating a training sample of size $n = 200$, computing the three estimates, and estimating their performance on a test set of size 10 000. The three performance measures are the cost-sensitive risk, the probability of error, and the Lebesgue measure of the symmetric difference with respect to the true level set. The results in Table I represent averages over 10 000 repetitions of the experiment and are accurate to four digits.

TABLE I

Results From the Histogram Simulation Study. The Reported Numbers Represent Averages Over 10 000 Repetitions of the Experiment and Are Accurate to Four Digits. The Methods Compared Are Cost-Sensitive (CS), Cost-Insensitive (CI), and $\boldsymbol{L}^1$ Regression Followed by Thresholding (L1)

	CS	CI	REG
cost-sensitive risk	0.2015	0.2149	0.2118
probability of error	0.2919	0.3017	0.2987
symmetric difference	0.1158	0.1368	0.1305

B. Real-World Data

We ran our algorithms on the benchmark data sets named "pyrim," "mpg," "housing," and "triazines." The data sets are available online with documentation.[3] They contain 74, 392, 506, and 186 examples each, with dimensionalities 27, 7, 14, and 60, respectively. We randomly permuted each data set 100 times. For each permutation, we used 70% for estimating the level set and the remaining 30% for testing the estimate's performance. For the SVM methods, 40% of the data were used for training, and 30% formed a holdout set for setting free parameters. The targeted level γ was taken to be the average of the response Y across the data set.

On these data sets, we compare eight methods in all. The four SVM methods are the direct cost-sensitive SVM (SVM-CS-DIRECT), the cost-sensitive SVM via costing (SVM-CS-COSTING), the cost-insensitive SVM (SVM-CI), and SVR followed by thresholding (SVM-REG). Similarly, the four nearest neighbor methods (using $k = 3$) are denoted 3-NN-CS-DIRECT, 3-NN-CS-COSTING, 3-NN-CI, and 3-NN-REG. Recall that 3-NN-CS-DIRECT and 3-NN-REG are equivalent. Other values of k were investigated, but they did not affect our conclusions. For costing, we vote over 25 resamples for the SVM and 100 for 3-NN.

To implement the support vector classifiers we used the SVM$^{\text{light}}$ package [16], while for support vector regression we employed the LIBSVM [17]. In all of our SVM experiments, we used a radial basis function (Gaussian) kernel and searched for the bandwidth parameter σ over a logarithmically spaced grid of 50 points from 10^{-4} to 10^4. We also searched for the regularization parameter C over a logarithmically spaced grid of 50 points from 10^{-3} to 10^3. In addition, for the SVR experiments, we searched for the width of the insensitive loss tube ϵ over a logarithmically spaced grid of 50 points from 10^{-3} to 0.5.

Table II reports the estimated cost of each algorithm on each data set, averaged over all 100 permutations, along with standard deviations.

VII. Conclusion and Future Work

An interesting conclusion from the synthetic data study is that the cost-sensitive approach is superior to the naïve approaches with respect to all three metrics considered: expected misclassification cost, probability of error, and measure of the symmetric difference. Thus, for classification problems where the labels are obtained by thresholding a response variable, even if the design criterion is the probability of error, it may be advantageous to incorporate cost information.

Our experiments on real-world data suggest that costing is not an affective algorithm, at least for the sample sizes we considered. This may be partially explained by the fact that cost-proportionate rejection sampling leads to sample sizes that are only a fraction of the original sample size. Furthermore, if a few costs are substantially larger than all

[3] http://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/

TABLE II

EXPERIMENTAL RESULTS FOR FOUR REAL-WORLD DATA SETS. THE REPORTED NUMBERS REPRESENT AVERAGE ESTIMATED COSTS AND STANDARD DEVIATIONS OBTAINED FROM 100 PERMUTATIONS OF THE DATA INTO TRAINING AND TEST SETS

	pyrim	mpg	housing	triazines
SVM-CS-DIRECT	0.0388 ± 0.0124	0.2501 ± 0.2079	0.4668 ± 0.2651	0.0201 ± 0.0042
SVM-CS-COSTING	0.1186 ± 0.0104	2.4485 ± 0.2205	3.9922 ± 0.2855	0.1772 ± 0.0098
SVM-CI	0.0397 ± 0.0123	0.2360 ± 0.2060	0.4893 ± 0.2669	0.0204 ± 0.0042
SVM-REG	0.0399 ± 0.0127	0.2069 ± 0.1983	0.3791 ± 0.2483	0.0202 ± 0.0043
3-NN-CS-DIRECT	0.0158 ± 0.0094	0.2052 ± 0.1947	0.4735 ± 0.2699	0.0303 ± 0.0093
3-NN-CS-COSTING	0.0215 ± 0.0101	0.3047 ± 0.2223	0.5902 ± 0.2915	0.0392 ± 0.0098
3-NN-CI	0.0151 ± 0.0091	0.2009 ± 0.1951	0.5178 ± 0.2747	0.0312 ± 0.0092
3-NN-REG	0.0158 ± 0.0094	0.2052 ± 0.1947	0.4735 ± 0.2699	0.0303 ± 0.0093

others, then the fraction of points rejected can be quite high, leading to even smaller sample sizes.

As for the other methods, the 3-NN methods outperform the SVM methods on the first data set, while the reverse is true on the fourth data set. Within each methodology (3-NN or SVM), the three competitive approaches (direct cost-sensitive, cost-insensitive, and regression followed by thresholding) do not differ in a statistically significant way on any of the four data sets. If we look at the average cost (normalized by standard error) across the four data sets and across methodologies (3-NN and SVM), the results are 2.33, 2.38, and 2.27. Thus, the cost-sensitive approach appears to have a slight edge over the cost-insensitive method, which is to be expected since it does not throw away information. On the other hand, the regression/thresholding method seems to perform at least as well as the cost-sensitive approach.

Although the cost-sensitive and regression-based methods have comparable performance, there can be a significant difference in terms of computation time. In particular, the cost-sensitive SVM has two free parameters while SVR has three. When conducting a grid search over parameter values and using a holdout or cross-validation error estimate, the increased computational complexity of SVR is substantial. This observation may extend to other algorithmic frameworks because regression is a harder problem and will often require the specification of more free parameters than classification.

An interesting problem for future work is to demonstrate a general algorithmic framework for estimating *multiple* level sets (of the same regression function) simultaneously. One approach is to reduce the problem to multiclass cost-sensitive classification, but it would be important to constrain the estimated sets to be *nested* [8].

APPENDIX I
PROOF OF PROPOSITION 1

Observe

$$\begin{aligned} R(G) &= E[(\gamma - Y)I_{\{X\in G, Y\le\gamma\}} + (Y-\gamma)I_{\{X\notin G, Y\ge\gamma\}}] \\ &= \int_G \left(\int_{-\infty}^{\gamma} (\gamma - y) f(y|x)dy\right) dF_X \\ &\quad + \int_{\bar{G}} \left(\int_{\gamma}^{\infty} (y-\gamma) f(y|x)dy\right) dF_X \\ &= \int_G \psi_0(x)dF_X + \int_{\bar{G}} \psi_1(x)dF_X \end{aligned}$$

where

$$\begin{aligned} \psi_0(x) &= \int_{-\infty}^{\gamma} (\gamma - y) f(y|x)dy \\ \psi_1(x) &= \int_{\gamma}^{\infty} (y-\gamma) f(y|x)dy. \end{aligned}$$

Therefore

$$\begin{aligned} &R(G) - R(G^*) \\ &\quad = \int_G \psi_0(x)dF_X + \int_{\bar{G}} \psi_1(x)dF_X \\ &\qquad - \int_{G^*} \psi_0(x)dF_X - \int_{\bar{G^*}} \psi_1(x)dF_X \\ &\quad = \int_{G\cap\bar{G^*}} \psi_0(x)dF_X - \int_{G^*\cap\bar{G}} \psi_0(x)dF_X \\ &\qquad + \int_{G^*\cap\bar{G}} \psi_1(x)dF_X - \int_{G\cap\bar{G^*}} \psi_1(x)dF_X \\ &\quad = \int_{G\cap\bar{G^*}} (\psi_0(x) - \psi_1(x))dF_X \\ &\qquad + \int_{G^*\cap\bar{G}} (\psi_1(x) - \psi_0(x))dF_X. \end{aligned}$$

Now

$$\begin{aligned} &\psi_1(x) - \psi_0(x) \\ &\quad = \int_{-\infty}^{\infty} (y-\gamma) f(y|x)dy \\ &\quad = \int_{-\infty}^{\infty} (h(x) - \gamma + \epsilon) f(\epsilon)d\epsilon \\ &\quad = h(x) - \gamma \end{aligned}$$

because ϵ is zero mean. Since $x \in G^* \iff h(x) - \gamma \ge 0$, we have

$$R(G) - R(G^*) = \int_{G\Delta G^*} |h(x) - \gamma| dF_X$$

as desired.

REFERENCES

[1] T.-K. Man, M. Chintagumpala, J. Visvanathan, J. Shen, L. Perlaky, J. H. M. Johnson, N. Davino, J. Murray, L. Helman, W. Meyer, T. Triche, K.-K. Wong, and C. C. Laus, "Expression profiles of osteosarcoma that can predict response to chemotherapy," *Cancer Res.*, vol. 65, pp. 8142–8150, Sep. 2005.

[2] V. Vapnik, *The Nature of Statistical Learning Theory*. New York: Springer-Verlag, 1995.

[3] A. Beygelzimer, V. Dani, T. Hayes, J. Langford, and B. Zadrozny, "Error-limiting reductions between classification tasks," in *Proc. 22nd Int. Machine Learning Conf. (ICML)*, L. D. Raedt and S. Wrobel, Eds. New York: ACM Press, 2005.

[4] L. Devroye, L. Gyorfi, and G. Lugosi, *A Probabilistic Theory of Pattern Recognition*. New York: Springer, 1996.

[5] A. B. Tsybakov, "Optimal aggregation of classifiers in statistical learning," *Ann. Stat.*, vol. 32, no. 1, pp. 135–166, 2004.

[6] L. Cavalier, "Nonparametric estimation of regression level sets," *Statistics*, vol. 29, pp. 131–160, 1997.

[7] R. Willett and R. Nowak, "Minimax optimal level set estimation," in *Proc. SPIE, Wavelets XI*, San Diego, CA, Jul. 31–Aug. 4 2005, vol. 5914.

[8] R. Willett and R. Nowak, "Minimax optimal level set estimation," *IEEE Trans. Image Process.* 2006 [Online]. Available: http://www.ee.duke.edu/~willett/, submitted for publication

[9] B. Zadrozny, J. Langford, and N. Abe, "Cost sensitive learning by cost-proportionate example weighting," in *Proc. 3rd Int. Conf. Data Mining*, Melbourne, FL, 2003, IEEE Computer Society Press.

[10] P. Domingos, "MetaCost: A general method for making classifiers cost sensitive," in *Proc. 5th Int. Conf. Knowledge Discovery Data Mining*, San Diego, CA, 1999, pp. 155–164, ACM Press.

[11] C. Elkan, "The foundations of cost-sensitive learning," in *Proc. 17th Int. Joint Conf. Artificial Intelligence*, Seattle, WA, 2001, pp. 973–978.

[12] D. Margineantu, "Class probability estimation and cost-sensitive classification decisions," in *Proc. 13th Eur. Conf. Machine Learning*, Helsinki, Finland, 2002, pp. 270–281.

[13] B. Scholkopf and A. J. Smola, *Learning with Kernels*. Cambridge, MA: MIT Press, 2002.

[14] C. Cortes and V. Vapnik, "Support-vector networks," *Mach. Learn.*, vol. 20, no. 3, pp. 273–297, 1995.

[15] T. M. Cover and P. E. Hart, "Nearest neighbor pattern classification," *IEEE Trans. Inf. Theory*, vol. IT-13, no. 1, pp. 21–27, 1967.

[16] T. Joachims, "Making large-scale SVM learning practical," in *Advances in Kernel Methods—Support Vector Learning*. Cambridge, MA: MIT Press, 1999.

[17] C. C. Chang and C. J. Lin, "LIBSVM: A Library for Support Vector Machines," 2001 [Online]. Available: http://www.csie.ntu.edu.tw/~cjlin/libsvm

Comments on "Detection of Distributed Sources Using Sensor Arrays"

Sridhar Ramakrishnan, *Student Member, IEEE*, and
Satish Udpa, *Fellow, IEEE*

***Abstract*—In the above correspondence (Y. Jin and B. Friedlander, "Detection of distributed sources using sensor arrays," *IEEE Trans. Signal Process.*, vol. 52, no. 6, pp. 1537–1548, June 2004), Jin and Friedlander develop a GLR-based detector for detecting a random spatially distributed signal source using an array of sensors. We show that the expression for required SNR (RSNR) has been incorrectly derived, which has led the authors to draw incorrect conclusions in their work. In this correspondence, we correct this particular error and a few other typographical errors, and provide appropriate conclusions to the original work.**

***Index Terms*—Distributed source, sensor array, signal detection.**

In the above correspondence, [1, eq. (50)] expresses the required SNR (RSNR) incorrectly as

$$\rho = \frac{P}{\sum_{j=1}^{r} \lambda_j^2} \frac{\left(\sum_{j=1}^{r} \frac{\rho\lambda_j}{\rho\lambda_j+1} \right)^2}{\sum_{j=1}^{r} \left(\frac{\rho\lambda_j}{\rho\lambda_j+1} \right)^2} \frac{\psi_{M_1}^{-1}(1-PFA)}{\psi_{M_0}^{-1}(1-PD)}. \tag{1}$$

The expression for RSNR when derived correctly should read as

$$\rho = \frac{P}{\sum_{j=1}^{r} \lambda_j^2} \frac{\sum_{j=1}^{r} \left(\frac{\rho\lambda_j}{\rho\lambda_j+1} \right)^2}{\sum_{j=1}^{r} \frac{\rho\lambda_j}{\rho\lambda_j+1}} \frac{\psi_{M_0}^{-1}(1-PFA)}{\psi_{M_1}^{-1}(1-PD)}. \tag{2}$$

Manuscript received March 10, 2006; revised July 15, 2006. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Fulvio Gini.

The authors are with the Electrical and Computer Engineering Department, Michigan State University, East Lansing, MI 48824 USA (e-mail: rsridhar@egr.msu.edu; udpa@egr.msu.edu).

Digital Object Identifier 10.1109/TSP.2007.893740

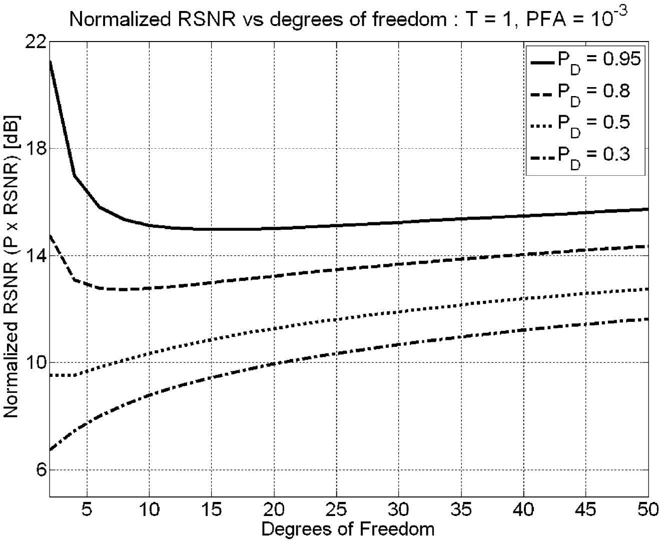

Fig. 1. Normalized RSNR versus degrees of freedom ($\boldsymbol{r} \in [1, 25]$, $\boldsymbol{T} = 1$) for different target $\boldsymbol{P}_D = [0.95, 0.8, 0.5, 0.3]$ and $\boldsymbol{PFA} = 0.001$.

A simplified form of the above expression is obtained when we consider the case where all the principal eigenvalues of $\mathbf{R}_s$ are approximately equal, i.e., $\lambda_i \approx P/r$, $i = 1, \ldots, r$. Defining $v\,(= 2Tr = M_0 = M_1)$ as the degrees of freedom, we, thus, obtain

$$\text{RSNR} \approx \frac{v}{2TP} \left[\frac{\psi_v^{-1}(1-PFA)}{\psi_v^{-1}(1-PD)} - 1 \right] \tag{3}$$

instead of

$$\text{RSNR} \approx \frac{v}{2P} \left[\frac{\psi_v^{-1}(1-PFA)}{\psi_v^{-1}(1-PD)} - 1 \right] \tag{4}$$

as expressed in [1]. Consequently, the expression for output SNR defined as RSNR × SNRG becomes

$$\text{RSNR} \times \text{SNRG} \approx \left[\frac{\psi_v^{-1}(1-PFA)}{\psi_v^{-1}(1-PD)} - 1 \right] \tag{5}$$

as opposed to

$$\text{RSNR} \times \text{SNRG} \approx T \left[\frac{\psi_v^{-1}(1-PFA)}{\psi_v^{-1}(1-PD)} - 1 \right] \tag{6}$$

mentioned as [1, eq. (51)] in the original work by Jin and Friedlander.

As a result of the incorrect expression in (4), Fig. 8 in the original correspondence, i.e., the plot of RSNR versus degrees of freedom v for different P_D, fails to capture the variation in the RSNR performance for changing T (number of time snapshots) and changing r (effective rank of $\mathbf{R}_s$, which is a measure of the angular spread of the signal), independently. The figure would be correct only under a special case of $T = 1$ snapshot, and not in general for all T. The number of degrees of freedom v contains information of both T and r, but the effect of increasing r on RSNR (T, being held constant at different values) is markedly different from the effect of increasing T (r, being held constant at different values) on RSNR. Figs. 1–4 in this correspondence depict this variation in the RSNR performance for four different cases.