Optimizing Classifiers for Hypothetical Scenarios

Reid A. Johnson[1], Troy Raeder[2], and Nitesh V. Chawla[1]

[1] University of Notre Dame, Notre Dame, IN 46556, USA
{rjohns15,nchawla}@nd.edu
[2] Dstillery, 470 Park Ave. S., 6th Floor, New York, NY 10016, USA
troy@dstillery.com

Abstract. The deployment of classification models is an integral component of many modern data mining and machine learning applications. A typical classification model is built with the tacit assumption that the deployment scenario by which it is evaluated is fixed and fully characterized. Yet, in the practical deployment of classification methods, important aspects of the application environment, such as the misclassification costs, may be uncertain during model building. Moreover, a single classification model may be applied in several different deployment scenarios. In this work, we propose a method to optimize a model for uncertain deployment scenarios. We begin by deriving a relationship between two evaluation measures, H measure and cost curves, that may be used to address uncertainty in classifier performance. We show that when uncertainty in classifier performance is modeled as a probabilistic belief that is a function of this underlying relationship, a natural definition of *risk* emerges for both classifiers and instances. We then leverage this notion of risk to develop a boosting-based algorithm—which we call *RiskBoost*—that directly mitigates classifier risk, and we demonstrate that it outperforms AdaBoost on a diverse selection of datasets.

1 Introduction

Many real-world problems necessitate the use of a classification model to assign items in a collection to target categories or classes. The chief objective of a classification model is to accurately predict the target class for each case in the data. Accordingly, when evaluating a classification model, one desires an accurate assessment of its performance on unseen data. Accurate model assessments are important because they permit candidate models to be meaningfully compared and allow one to determine whether a model will perform at an "acceptable" level. The notion of acceptable performance may be defined solely by internal concerns (e.g., the benefit of a model must outweigh its implementation cost) or by external factors (e.g., regulators may hesitate to approve a diagnostic test with a high false negative rate). No matter how it is applied, however, sound model assessment is a critical element of any classification task [10].

There are many ways to quantifiably assess the performance of a classifier. In this work, we quantify classifier performance via a *simple linear cost model*:

$$\ell = c_0\pi_0 e_0 + c_1\pi_1 e_1, \tag{1}$$

where c_i is the cost of misclassifying a class i instance, π_i is the proportion of class i instances in the data, and e_i is the error rate on class i instances. This cost model is

convenient and is commonly used in cost-sensitive learning. However, cost-sensitive methods generally assume that the parameters π_i and c_i are known and constant (e.g., [5,12,18,21]), an assumption that is often not borne out in practice [14]. Zadrozny and Elkan [20] provide a framework for estimating costs and probabilities when sample data are available, but for the purpose of scenario analysis (i.e., the process of evaluating possible future events for which such information is not readily available) [11].

In this work, we focus on developing classification models for hypothetical future deployment scenarios engendered by uncertain operating environments. We begin in Section 2 by connecting the current techniques for dealing with uncertain operating environments with a notion of cost. We then demonstrate in Section 3 that, as a result of this connection, there exists an underlying theoretical relationship between several of these methods that leads to a natural definition of the *risk* of an individual classifier and instance. Further, we find that this risk can be substantially mitigated via a boosting-based algorithm we call *RiskBoost*. In Section 4, we demonstrate that RiskBoost outperforms AdaBoost [8] over a diverse collection of datasets. Finally, we present our conclusions in Section 5.

2 Addressing Uncertain Cost in Classifier Performance

Consider a binary classification task where we have several cases or instances, each of which may be assigned to one of two categories or classes that are labeled 1 (positive) and 0 (negative). Further, assume that any classifier learned on the training data is capable of producing, for each input vector $\mathbf{x}$, a real-valued score $s(\mathbf{x})$ that is a monotonic function of $p(1|x)$, which is the probability that $\mathbf{x}$ belongs to class 1. These scores are mapped to binary classifications by choosing a threshold t such that an instance $\mathbf{x}$ is classified as class 0 (negative) if $s(x) < t$ and class 1 (positive) if $s(x) \geq t$.

Each classification threshold produces a unique classifier, the performance of which can be characterized by a confusion matrix of particular true positive (tp), false positive (fp), true negative (tn), and false negative (fn) values. Presently, the *de facto* standard method for evaluating classification models from a confusion matrix is the receiver operating characteristic, though alternatives such as the H measure and cost curves also exist. We first elaborate upon each of these below, after which we define a clear relationship between all three.

For the reader's convenience, a summary of the notations used in the this work is given as Table 1. For the remainder of the work, we use the term *classifier* to refer to a specific confusion matrix, whereas *classification algorithm* or *learning algorithm* is used to refer to a trained model for which a decision threshold has not been defined.

2.1 Addressing Cost with ROC Curves

The Receiver Operating Characteristic (ROC) curve [7,13] forms the basis for many of the techniques that we will discuss in the remainder of this work. An ROC curve is formed by varying the classification threshold t across all possible values. In a binary classification problem, each threshold produces a distinct confusion matrix that corresponds to a two-dimensional point (r_1, r_0) in ROC space, where $r_1 = fpr$ and $r_0 = tpr$.

Table 1. The notation used in this work.

Symbol	Description
ℓ	The total classification loss.
π_i	The proportion of class i instance in test data.
c_i	The cost of misclassifying a class i instance.
c	A normalized cost ratio, i.e., $c = c_0/(c_0 + c_1)$.
$u(c)$	The likelihood distribution over cost ratios.
e_i	The error rate on class i instances.
n_i	The number of class i test instances.
$\mathscr{L}_i$	The marginal cost of class i instances.
t	A classification threshold.
(r_{1_i}, r_{0_i})	The ith point on the ROC convex hull.
$f_i(x)$	The ith line segment on the lower envelope in cost space.
tp,fp	A true and false positive classification, respectively.
tn,fn	A true and false negative classification, respectively.
tpr,fpr	The true and false positive rate, respectively.
tnr,fnr	The true and false negative rate, respectively.

A point p_1 in ROC space is said to "dominate" a point p_2 in ROC space if p_1 is both above and to the left of p_2. It follows, then, that only classifiers on the convex hull of the ROC curve are potentially optimal for some value of c_i and π_i, as a point not on the convex hull will be dominated by a point that is on it [14]. As each point on the ROC convex hull represents classification performance at some threshold t, different thresholds will be optimal under different operating conditions c and π_i. For example, classifiers with lower false negative rates will be optimal at lower values of c, while classifiers with lower false positive rates will be optimal at higher values of c.

Now, let $p_i = (r_{1i}, r_{0i})$ and $p_{i+1} = (r_{1(i+1)}, r_{0(i+1)})$ be successive points on the ROC convex hull. Then p_{i+1} will produce superior classification performance to p_i if and only if the change in the false positive rate is offset by a corresponding change in the true positive rate. That is, if we set $\Delta x_i = r_{1(i+1)} - r_{1i}$ and $\Delta y_i = r_{0(i+1)} - r_{0i}$, then p_{i+1} is optimal if

$$c < \frac{\pi_1 \Delta y}{\pi_0 \Delta x + \pi_1 \Delta y}. \tag{2}$$

Similarly, given a fixed value for c, we can determine the optimal classifier at a given value of π_0. Then for p_{i+1} to outperform p_i, we require that

$$\pi_0 < \frac{(1-c)\Delta y}{c\Delta x + (1-c)\Delta y}. \tag{3}$$

Thus, the ROC convex hull can be used to select the optimal classification threshold (and classifier) under a variety of different operating conditions, a notion first articulated by Provost and Fawcett [14].

Relationship between ROC curves and cost. Each point in ROC space corresponds to a misclassification cost that can be specified via our simple linear cost model as

$$\ell = c_0\pi_0 r_1 + c_1\pi_1(1-r_0). \tag{4}$$

Note that only the ordinality (i.e., relative magnitude) of the cost is needed for ranking classifiers. Accordingly, if we assume that the cardinality (i.e, absolute magnitude) of the cost can be ignored, then, as $c = c_0/(c_0+c_1)$, we find that

$$\ell = c\pi_0 r_1 + (1-c)\pi_1(1-r_0). \tag{5}$$

This formulation will be used frequently throughout the remainder of this work.

2.2 Addressing Uncertain Cost with the H measure

An alternative to the ROC is the H measure, proposed by Hand [9] to address shortcomings of the ROC. Unlike the ROC, the H measure incorporates uncertainty in the cost ratio c by integrating directly over a hypothetical probability distribution of cost ratios. As the points on the ROC convex hull correspond to optimal misclassification cost over a contiguous set of cost ratios (see Equation 2), then, given known prior probabilities π_i, the average loss over all cost ratios can be calculated by integrating Equation 4 piecewise over the cost regions defined by the convex hull.

Relationship between the H measure and uncertain cost. To incorporate a hypothetical cost ratio distribution, we set $c = c_0/(c_0+c_1)$ and weight the integral by the cost distribution, denoted as $u(c)$. The final loss measure is then defined as:

$$\ell_H = \sum_{i=0}^{m} \int_{c^{(i)}}^{c^{(i+1)}} \big(c\pi_0 r_{1i} + (1-c)\pi_1(1-r_{0i})\big)u(c)dc. \tag{6}$$

The H measure is represented as a normalized scalar value between 0 and 1, whereby higher values correspond to better model performance.

2.3 Addressing Uncertain Cost with Cost Curves

Cost curves [6] provide another alternative to ROC curves for visualizing classifier performance. Instead of visualizing performance as a trade-off between false positives and true positives, they depict classification cost in the simple linear cost model against the unknowns π_i and c_i.

The marginal misclassification cost of class i can be written as $\mathscr{L}_i = \pi_i c_i$. This means that if the misclassification rate of class i instances increases by some amount Δe_i, then the total misclassification cost increases by $\mathscr{L}_i \Delta e_i$. The maximum possible cost of any classifier is $\ell_{max} = \mathscr{L}_0 + \mathscr{L}_1$, when both error rates are 1. Accordingly, we can define the normalized marginal cost (termed the probability cost by Drummond and Holte [6]) as $pc_i = \mathscr{L}_i/(\mathscr{L}_0+\mathscr{L}_1)$, and the normalized total misclassification cost as $\ell_{norm} = \ell/\ell_{max}$. Intuitively, the quantity pc_i can be thought of as the proportion of the total risk arising from class i instances, since we have $pc_0 + pc_1 = 1$, while ℓ_{norm} is the proportion of the maximum possible cost that the given classifier actually incurs.

Each ROC point (r_{1i}, r_{0i}) corresponds to a range of possible misclassification costs that depend on the marginal costs $\mathscr{L}_i$, as shown in Equation 4. We can rewrite Equation 4 as a function of pc_1 as follows:

$$\begin{aligned}\ell_{norm} &= (1-pc_1)r_{1i} + pc_1(1-r_{0i}) \\ &= pc_1(1-r_{0i}-r_{1i}) + r_{1i}.\end{aligned}$$

Thus any point in ROC space translates (i.e., can be transformed) into a line in cost space. Of particular interest are the lines corresponding to the ROC convex hull, as these lines represent classifiers with optimal misclassification cost. These lines enclose a convex region of cost space known as the lower envelope. The values of pc_1 for which a classifier is on the lower envelope provide scenarios under which the classifier is the optimal choice.

One can compute the area under the lower envelope to obtain a scalar estimate of misclassification cost. Here, we denote points on the convex hull by $(r_{1i}, r_{0i}), r_{00} < r_{01} < \ldots < r_{0m}$ in increasing order of x-coordinate, and we denote the corresponding cost lines as $f_i(x) = m_i x + b_i$, where m_i is the slope and b_i is the y-intercept of the ith cost line. The lower envelope is then composed of the intersection points of successive lines $f_i(x)$ and $f_{i+1}(x)$. We denote these points $p_i = (x_i, y_i)$, which can be calculated as

$$\begin{aligned}x_i &= \frac{r_{1(i+1)} - r_{1i}}{(r_{0(i+1)} - r_{0i}) + (r_{1(i+1)} - r_{1i})} \\ y_i &= \frac{r_{1i} - r_{1(i+1)}}{1 - r_{0(i+1)} - r_{1(i+1)}} + r_{1i}.\end{aligned}$$

The area under the lower envelope can be calculated geometrically as the area of a convex polygon or analytically as a sum of integrals (the areas under the constituent line segments). For our purposes, it is convenient to express it as follows:

$$A(f_1 \ldots f_m) = \sum_{i=0}^{m} \int_{x_i}^{x_{i+1}} f_i(x)dx. \tag{7}$$

The function $A(\cdot)$ represents a loss measure, where higher values of A correspond to worse performance. This area represents the expected misclassification cost of the classifier, where all values of pc_1 are considered equally likely. In the next section, we discuss the implications of this loss measure.

3 Deriving and Optimizing on Risk from Uncertain Cost

In the previous section, we related several measures of classifier performance to a notion of cost. In this section, we elaborate on the consequences of these connections, from which we derive definitions of "risk" for classifiers and instances.

3.1 Relationship between Cost Curves and H measure

An interesting result emerges if we assume an accurate estimate of π_i, either from the training data or from some other source of background knowledge and replace the pair

(c_0, c_1) with $(c, 1-c)$. In this case, a hypothetical cost curve represents $\ell_c = c\pi_0 r_1 + (1-c)\pi_1(1-r_0)$ on the y-axis and c on the x-axis. We can rewrite this expression into the standard form of an equation for a line, which gives us $\ell_c = c(\pi_0 r_1 - \pi_1(1-r_0)) + (1-r_0)$.

The intersection points of successive lines, which would form the lower envelope, can similarly be derived as

$$x_i = \frac{\pi_1(r_{0i} - r_{0(i+1)})}{\pi_1(r_{0i} - r_{0(i+1)}) + \pi_0(r_{1i} - r_{1(i+1)})}. \tag{8}$$

Consequently, the area under the lower envelope can be expressed as:

$$A(f_1 \dots f_m) = \sum_{i=0}^{m} \int_{x_i}^{x_{i+1}} \big(c\pi_0 r_1 + (1-c)\pi_1(1-r_0)\big) dc. \tag{9}$$

As the endpoints x_i are the same as those used in the computation of the H measure (see Equation 2), it follows that the H measure is equivalent to the area under the lower envelope of the cost curve with uniform $u(c)$ and prior probabilities π_i known. Further, Hand has demonstrated that, for a particular choice of $u(c)$, the area under the ROC curve is equivalent to the H measure [9].

Thus, these three different techniques—ROC curves, H measure, and cost curves—are simply specific instances of the simple linear cost model. Rather than debating the relative merits of these specific measures, which is beyond the scope of this work (cf. [3,9] for such discussions), we instead focus on the powerful consequences of adhering to the more general model.

Intuitively, since the simple linear model underlies several measures of classifier performance, it also provides an avenue for interpreting model performance. In fact, we find that it provides an insight into model performance under hypothetical scenarios—that is, a notion of risk—that cannot be explicitly captured by these other measures. We elaborate on this below.

3.2 Interpreting Performance under Hypothetical Scenarios

As a consequence of the relationship between the H measure and cost curves, we can actually represent the H measure loss function in cost space. By representing different loss functions on a single set of axes, we form a series of scenario curves, each of which corresponds to a loss function.

Figure 1 depicts scenario curves for several different likelihood functions alongside a standard cost curve. Each curve quantifies the vulnerability of the classification algorithm over the set of all possible scenarios pc_1 for different probabilistic beliefs about the likelihood of different cost ratios. The likelihood distributions include: (1) the $Beta(2,2)$ distribution $u(c) = \frac{1}{6}c(1-c)$, as suggested by [9]; (2) a Beta distribution shifted so that the most likely cost ratio is proportional to the proportion of minority class instances (i.e., $c \propto \pi_0$); (3) a truncated Beta distribution where the probability of minority class instances is greater than the probability of majority class instances (i.e., $p(c_0 > c_1) = 0$), motivated by the observation that the minority class typically has the

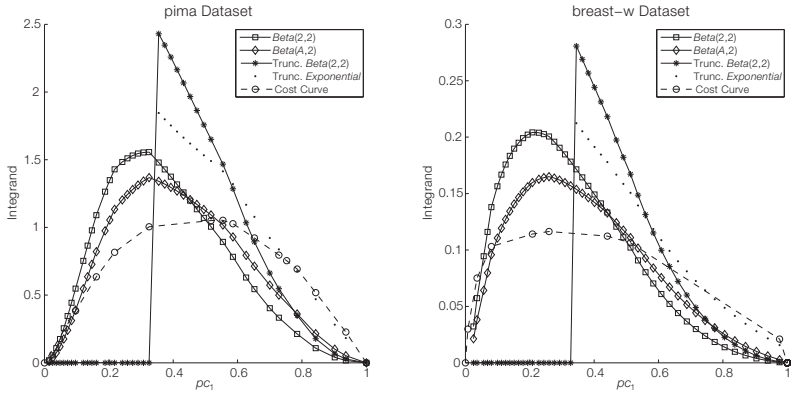

Fig. 1. Scenario curves for several different cost distributions $u(c)$ generated by a boosted decision tree model on the (a) `pima` and (b) `breast-w` datasets. The curves have been normalized such that (1) the area under each curve represents the value of the respective loss measure and (2) the maximum loss for the cost curve is 1.

highest misclassification cost; (4) a truncated exponential distribution where the parameter λ is set to ensure that the expectation of class i is inversely proportional to the proportion of that class in the data (i.e., $c_i \propto 1/\pi_i$); and (5) the cost curve, which assumes uniform distributions over probabilities and costs.

From the figure, it is clear that the choice of likelihood distribution can have a significant effect on both the absolute assessment of classifier performance (i.e., the area under the curve) and on which scenarios we believe will produce the greatest loss for the classifier. These curves also have intuitive meanings that may be useful when analyzing classifier performance. First, as the cost curve makes no *a priori* assumptions about the likelihood of different scenarios, it can present the performance of an algorithm over any given scenario. Second, if and when information about the likelihood of different scenarios becomes known, the cost curve presents the set of classifiers the pose the greatest risk (i.e., the components of the convex hull).

Both interpretations are important. On the one hand, an unweighted cost curve can be used to identify the set of scenarios over which a classifier performs acceptably for any domain-specific definition of reasonable performance. On the other hand, a weighted scenario curve can be used to identify where an algorithm should be improved in order to achieve the maximum benefit given the available information. From the second observation arises a natural notion of risk.

3.3 Defining Risk

Given a likelihood distribution over the cost ratio c, each classifier on the convex hull is optimal over some range of cost ratios (see Equation 2). From this, we can derive two

intuitive definitions: one for the risk associated with individual classifiers and one for the risk associated with individual instances.

Definition 1. *Assume that classifier $\mathscr{C}$ is optimal over the range of cost ratios $[c_1, c_2]$. Then the risk of classifier $\mathscr{C}$ is the expected cost of the classifier over the range for which it is optimal:*

$$risk(\mathscr{C}) = \int_{c_1}^{c_2} \ell_H(c)dc \tag{10}$$

Definition 2. *The risk of instance* **x** *is the aggregate risk over all classifiers that misclassify* **x**.

We discuss how these definitions may be applied to improve to classifier performance below.

3.4 RiskBoost: Optimizing Classification by Minimizing Risk

Since we can quantify the degree to which instances pose the greatest risk to our classification algorithm, it is natural to strengthen the algorithm by assigning greater importance to these "risky" instances.

Standard boosting algorithms such as AdaBoost combine functions based on the "hardness" of correctly classifying a particular instance [8]. Instead, we propose a novel boosting algorithm that reweights instances according to their relative risk, which we call *RiskBoost*. RiskBoost uses the expected misclassification loss ℓ to reweight instances that are misclassified by the most vulnerable classifier according to both classifier performance and the hypothetical cost ratio distribution. Pseudocode for RiskBoost is provided as Algorithm 1.

Algorithm 1 RiskBoost

Require: A base learning algorithm $\mathscr{W}$, the number of boosting iterations n, and m training instances $\mathbf{x}_1 \ldots \mathbf{x}_m$.

Ensure: A weighted ensemble classifier.

Initialize a weight distribution D over the instances such that $D_1(\mathbf{x}_i) = 1/m$.
for $j = 1$ to n **do**
  Train a new instance $\mathscr{W}_j$ of the base learner $\mathscr{W}$ with weight distribution D_j.
  Compute the loss ℓ of the learner on the training data via Equation 6.
  Set $\beta_j = \frac{1-0.5*\ell}{0.5*\ell}$.
  Compute the risk of each classifier on the ROC convex hull via Equation 10.
  for each instance **x** misclassified by the classifier of greatest risk **do**
    Set $D_{j+1}(\mathbf{x}) = \beta_j \cdot D_j(\mathbf{x})$.
  end for
  Otherwise set $D_{j+1}(\mathbf{x}) = D_j(\mathbf{x})$.
  Normalize such that $\sum_i D_{j+1}(\mathbf{x}_i) = 1$.
end for
return The final learner predicting $p(1|\mathbf{x}) = z\sum_j p_j(1|\mathbf{x})\beta_j$, where z is chosen such that the probabilities sum to 1.

4 Experiments

To evaluate the performance of RiskBoost, we compare it with AdaBoost on 19 classification datasets from the UCI Machine Learning Repository [1]. We employ RiskBoost by setting its risk calculation (i.e., Equation 10) as $u(c) = Beta(2,2)$, as suggested by [9]. AdaBoost is employed with the AdaBoost.M1 variant [8]. For both algorithms, we use 100 boosting iterations of unpruned the C4.5 decision trees, which previous work has shown benefit substantially from AdaBoost [15].

In order to compare the classifiers, we use 10-fold cross-validation. In 10-fold cross-validation, each dataset is partitioned into 10 disjoint subsets or folds such that each fold has (roughly) the same number of instances. A single fold is retained as the validation data for evaluating the model, while the remaining 9 folds are used for model building. This process is then repeated 10 times, with each of the 10 folds used exactly once as the validation data. As the cross-validation process can exhibit a significant degree of variability [16], we average the performance results from 100 repetitions of 10-fold cross-validation to generate reliable estimates of classifier performance. Performance is reported as AUROC (area under the Receiver Operating Characteristic).

4.1 Statistical Tests

Previous literature has suggested the comparison of classifier performance across multiple datasets based on ranks. Following the strategy outlined in [4], we first rank the performance of each classifier by its average AUROC. The Friedman test is then used to determine if there is a statistically significant difference between the rankings of the classifiers (i.e., that the rankings are not merely randomly distributed), after which the Bonferroni-Dunn post-hoc test is applied to control for multiple comparisons.

4.2 Results

From Table 2, we observe that RiskBoost performs better than AdaBoost in 14 of the 19 datasets evaluated, with 1 tie. Further, we find that RiskBoost performs statistically significantly better than AdaBoost at a 95% confidence level over the collection of evaluated datasets. The 95% critical distance of the Bonferroni-Dunn procedure for 19 datasets and 2 classifiers is 0.45; consequently, an average rank lower than 1.275 is statistically significant, which RiskBoost achieves with an average rank of 1.21. Similar results were achieved for 10 repetitions of 10-fold cross-validation (where RiskBoost's average rank was 1.11), 50 repetitions (1.26), and 500 repetitions (1.21).

4.3 Discussion

For a better understanding of the general intuition behind RiskBoost, Figure 2 shows the progression for AdaBoost and RiskBoost when optimizing the H measure with the $Beta(2,2)$ cost distribution. At each iteration, the RiskBoost ensemble directly boosts the classifier of greatest risk, which is represented by the global maximum in the figure. Successive iterations of RiskBoost lead to direct cost reductions for this classifier,

Table 2. AUROC performance of AdaBoost and RiskBoost on several classification datasets. Bold values indicate the best performance for a dataset. Checkmarks indicate the model performs statistically significantly better at the confidence level $1 - \alpha$.

Dataset	AdaBoost.M1	RiskBoost
breast-w	0.9829	**0.9899**
bupa	0.7218	0.7218
credit-a	0.8973	**0.9187**
crx	0.8970	**0.9191**
heart-c	0.8643	**0.8919**
heart-h	0.8531	**0.8723**
horse-colic	**0.8501**	0.8295
ion	**0.9753**	0.9744
krkp	0.9985	**0.9996**
ncaaf	0.8658	**0.9144**
pima	0.7803	**0.7872**
promoters	**0.9611**	0.8863
ringnorm	0.9793	**0.9849**
sonar	0.9281	**0.9344**
threenorm	0.9094	**0.9210**
tictactoe	**0.9994**	0.9986
twonorm	0.9834	**0.9885**
vote	0.9733	**0.9856**
vote1	0.9338	**0.9543**
Average Rank	1.79	1.21
$\alpha = 0.05$		✓

resulting in a gradual but consistent reduction from peak risk. By contrast, AdaBoost establishes an arbitrary threshold for "incorrect" instances. As a result, AdaBoost does not always focus on the instances that contribute greatest to the overall misclassification cost, which ultimately results in the erratic behavior demonstrated by AdaBoost's scenario curves.

Though RiskBoost offers promising performance over a diverse array of classification datasets, we note that there is an expansive literature on cost-sensitive boosting (e.g., [12,18,19]) and boosting with imbalanced data (e.g., [2,17,18]) that can be used to tackle similar problems. A critical feature that sets our work apart from prior efforts, however, is that previous work tacitly assumes that misclassification costs are known, whereas RiskBoost can expressly optimize misclassification costs that are unknown and uncertain. Further, we demonstrate that this strategy for risk mitigation actually arises naturally from the framework of scenario analysis. We leave further empirical evaluation of RiskBoost with cost-sensitive boosting algorithms as future work.

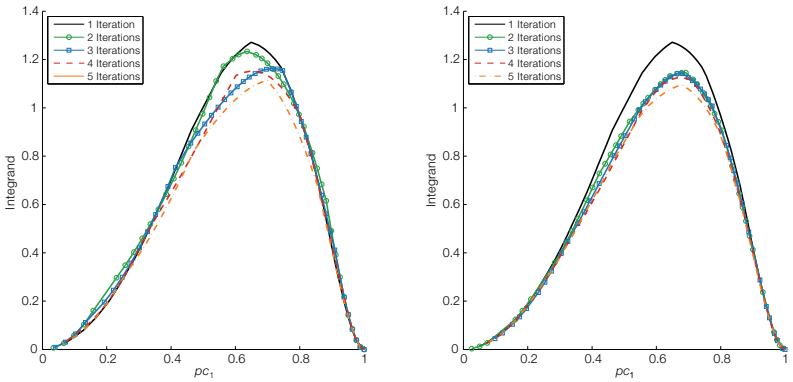

Fig. 2. Scenario curves for successive iterations of (a) AdaBoost and (b) RiskBoost ensembles on the `ncaaf` dataset.

5 Conclusion

Classification models are an integral tool for modern data mining and machine learning applications. When developing a classification model, one desires a model that will perform well on unseen data, often according to some hypothetical future deployment scenario. In doing so, two critical questions arise: First, how does one estimate performance so that the best-performing model can be selected? Second, how can one build a classifier that is optimized for these hypothetical scenarios?

Our work focuses on addressing these questions. By examining the current approaches for evaluating classifier performance in uncertain deployment scenarios, we derived a relationship between H measure and cost curves, two well-known techniques. As a consequence of this relationship, we found that ROC curves, H measure, and cost curves can be represented as specific instances of a simple linear cost model. We found that by defining scenarios as probabilistic expressions of belief in this simple linear cost model, intuitive definitions emerge for the risk of an individual classifier and the risk of an individual instance. These observations suggest a new boosting-based algorithm—RiskBoost—that directly mitigates the greatest component of classification risk, and which we find to outperform AdaBoost on a diverse selection of classification datasets.

6 Acknowledgements

This work is supported by the National Science Foundation (NSF) Grant OCI-1029584.

References

1. Bache, K., Lichman, M.: UCI machine learning repository (2013), `http://archive.ics.uci.edu/ml`

2. Chawla, N.V., Lazarevic, A., Hall, L.O., Bowyer, K.W.: SMOTEBoost: Improving prediction of the minority class in boosting. In: Proceedings of the 7th European Conference on Principles and Practice of Knowledge Discovery in Databases (ECML PKDD). pp. 107–119. Springer (2003)
3. Davis, J., Goadrich, M.: The relationship between Precision-Recall and ROC curves. In: Proceedings of the 23rd International Conference on Machine Learning (ICML). pp. 233–240. ACM (2006)
4. Demšar, J.: Statistical comparisons of classifiers over multiple data sets. Journal of Machine Learning Research (JMLR) 7, 1–30 (2006)
5. Domingos, P.: Metacost: A general method for making classifiers cost-sensitive. In: Proceedings of the 5th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD). pp. 155–164. ACM (1999)
6. Drummond, C., Holte, R.C.: Cost curves: An improved method for visualizing classifier performance. Machine Learning 65(1), 95–130 (2006)
7. Fawcett, T.: An introduction to ROC analysis. Pattern Recognition Letters 27(8), 861–874 (2006)
8. Freund, Y., Schapire, R.E.: Experiments with a new boosting algorithm. In: Proceedings of the 13th International Conference on Machine Learning (ICML). pp. 148–156 (1996)
9. Hand, D.J.: Measuring classifier performance: A coherent alternative to the area under the ROC curve. Machine Learning 77(1), 103–123 (2009)
10. Hastie, T., Tibshirani, R., Friedman, J.: The Elements of Statistical Learning, vol. 2 (2009)
11. Lempert, R.J., Popper, S.W., Bankes, S.C.: Shaping the Next One Hundred Years: New Methods for Quantitative, Long-Term Policy Analysis. Rand Corp (2003)
12. Masnadi-Shirazi, H., Vasconcelos, N.: Cost-sensitive boosting. IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI) 33(2), 294–309 (2011)
13. Provost, F., Fawcett, T.: Analysis and visualization of classifier performance: Comparison under imprecise class and cost distributions. In: Proceedings of the 3rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD). pp. 43–48. AAAI (1997)
14. Provost, F., Fawcett, T.: Robust classification for imprecise environments. Machine Learning 42(3), 203–231 (2001)
15. Quinlan, J.R.: Bagging, boosting, and C4.5. In: Proceedings of the 13th National Conference on Artificial Intelligence (AAAI). pp. 725–730 (1996)
16. Raeder, T., Hoens, T.R., Chawla, N.V.: Consequences of variability in classifier performance estimates. In: Proceedings of the 10th IEEE International Conference on Data Mining (ICDM). pp. 421–430. IEEE (2010)
17. Seiffert, C., Khoshgoftaar, T.M., Hulse, J.V., Napolitano, A.: RUSBoost: Improving classification performance when training data is skewed. In: Proceedings of the 19th International Conference on Pattern Recognition (ICPR). pp. 1–4. IEEE (2009)
18. Sun, Y., Kamel, M.S., Wong, A.K.C., Wang, Y.: Cost-sensitive boosting for classification of imbalanced data. Pattern Recognition 40(12), 3358–3378 (2007)
19. Ting, K.M.: A comparative study of cost-sensitive boosting algorithms. In: Proceedings of the 17th International Conference on Machine Learning (ICML). pp. 983–990
20. Zadrozny, B., Elkan, C.: Learning and making decisions when costs and probabilities are both unknown. In: Proceedings of the 7th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD). pp. 204–213. ACM (2001)
21. Zadrozny, B., Langford, J., Abe, N.: Cost-sensitive learning by cost-proportionate example weighting. In: Proceedings of the 3rd IEEE International Conference on Data Mining (ICDM). pp. 435–442. IEEE (2003)