BRIEFINGS IN BIOINFORMATICS. VOL 13. NO 1. 83–97
Advance Access published on 21 March 2011
doi:10.1093/bib/bbr008

Caveats and pitfalls of ROC analysis in clinical microarray research (and how to avoid them)

Daniel Berrar and Peter Flach

Submitted: 2nd November 2010; Received (in revised form): 9th February 2011

Abstract

The receiver operating characteristic (ROC) has emerged as the gold standard for assessing and comparing the performance of classifiers in a wide range of disciplines including the life sciences. ROC curves are frequently summarized in a single scalar, the area under the curve (AUC). This article discusses the caveats and pitfalls of ROC analysis in clinical microarray research, particularly in relation to (i) the interpretation of AUC (especially a value close to 0.5); (ii) model comparisons based on AUC; (iii) the differences between ranking and classification; (iv) effects due to multiple hypotheses testing; (v) the importance of confidence intervals for AUC; and (vi) the choice of the appropriate performance metric. With a discussion of illustrative examples and concrete real-world studies, this article highlights critical misconceptions that can profoundly impact the conclusions about the observed performance.

Keywords: *receiver operating characteristic; area under the curve; model evaluation; multiple testing; microarrays*

INTRODUCTION

The emergence of high-throughput technologies in modern biology has led to numerous prognostic and predictive models, notably for cancer subtype classification, disease diagnosis and prognosis, patient risk group stratification, and prediction of response to chemotherapy. Predictive models for response to chemotherapy in breast cancer, for example, include a 92-gene predictor for response to docetaxel [1]; a 74-gene predictor of response to paclitaxel and fluorouracil, doxorubicin and cyclophosphamide [2]; an 85-gene predictor of response to docetaxel [3]; and a 44-gene classifier for response to tamoxifen [4]. Classifiers induced from DNA microarray data have also been used for a successful prediction of risk groups in cancer patients with different clinical outcomes [5–12]. Given that such genomic classifiers are developed for clinical applications, a thorough validation of their robustness and prognostic performance is of vital importance [13]. Specifically, the performance of the fully specified model on independent test sets determines the final assessment: is this really a significant model that warrants further, multi-center studies? Importantly, a genomic classifier is no different from any other diagnostic test in this respect and should therefore be evaluated with the same rigor. For classification tasks involving only two classes and models with a continuous output such as class posterior probabilities, the receiver operating characteristic (ROC) is generally considered the method of choice for performance assessment [14, 15].

Given that ROC analysis is now integral to modern biomarker research [15] and also increasingly employed by the biomedical text mining community [16], an account of the caveats and potential pitfalls is timely. Specifically, there are several misconceptions that can have a profound impact on the interpretation of performance comparisons. Generally,

Corresponding author. Daniel Berrar, Tokyo Institute of Technology, Interdisciplinary Graduate School of Science and Engineering, Suzukakedai Campus, G3-45, 4259 Nagatsuta, Midori-ku, Yokohama 226-8502, Japan. E-mail: berrar.d.aa@m.titech.ac.jp
Daniel Berrar is an Associate Professor at the Interdisciplinary Graduate School of Science and Engineering, Tokyo Institute of Technology. His main research interests are data mining and machine learning in the life sciences.
Peter Flach is a Professor of Artificial Intelligence at the Department of Computer Science, University of Bristol. His main research interests are data mining and learning from highly structured data.

genomic classifiers are built using a training set (based on some data resampling strategies such as cross-validation), and then validated on an independent test set [17]. Using simplified examples with relevance to real-world studies, we discuss several pitfalls of ROC analysis—and how to avoid them.

ROC AND AREA UNDER THE CURVE IN A NUTSHELL

ROC analysis was originally developed to analyze the trade-offs between hit rates and false alarm rates in signal detection [18, 19]. Today, this tool is widely used in a range of disciplines, notably in machine learning research. For a classifier with a continuous output (i.e. scores quantifying the degree of class membership), ROC curves depict the trade-offs between the false positive rate (or 1 minus specificity, depicted on the x-axis) and the true positive rate (or sensitivity, depicted on the y-axis). These trade-offs correspond to all possible binary classifications that any dichotomization of the continuous outputs would allow. Thus, ROC curves depict a classifier's performance over the range of thresholds for sensitivity and specificity.

ROC curves are frequently summarized in a single value, the area under the curve (AUC), which ranges from 0 to 1.0. To define AUC formally, we follow the notation by Hilden [63]. Let P be the probability that a randomly selected actual positive (+) case, $\mathbf{x}_+$, has a lower score, s_+, than an independently, randomly selected actual negative (−) case, $\mathbf{x}_-$. Here, a lower ranking score means that $\mathbf{x}_+$ is ranked before $\mathbf{x}_-$; $f(s_+)$ and $g(s_-)$ are the distribution functions of these scores.

$$\begin{aligned} P &= \Pr\{s_+ < s_- | \mathbf{x}_+ \text{ and } \mathbf{x}_-\} \\ &= \iint_{s_+ < s_-} f(s_+)\, \mathrm{d}s_+ \; g(s_-)\, \mathrm{d}s_- \\ &= \int F(s_-)\, \mathrm{d}G(s_-) \\ &= \int (\text{ROC ordinate})\, \mathrm{d}\,(\text{ROC abscissa}) \\ &= \text{AUC}. \end{aligned} \tag{1}$$

Thus, the AUC is the probability that if we pick a positive and a negative case at random, then the classifier will assign a lower ranking score to the positive case (hence, rank it before the negative case). This corresponds to a Wilcoxon rank-sum statistic [18].

To illustrate this summary statistic, assume that a classifier produces class posterior probabilities for the positive class as shown in Figure 1A. Under the assumption of equal misclassification costs for false positive and false negative classifications, $P = 0.5$ is here the optimal threshold, providing for the best trade-off between sensitivity and specificity of 0.80, respectively. Coincidentally, the resulting (shaded) AUC is also 0.80 in this example. What this means is that the classifier commits five ranking errors out of a possible total of 25: instance #3 (a negative) is ranked before #4, #5 and #8 (positives), and instances #6 and #7 (negatives) are ranked before #8 (a positive).

A 'perfect' classifier will have a ROC curve passing through (0, 1) and an AUC of 1.0. In contrast, if we randomly guessed the class labels by tossing a fair coin, then we would expect to correctly classify half of the positive and half of the negative cases, which corresponds to the point (0.5, 0.5) in ROC space. If we predicted 60% of the cases as members of the positive class using a biased coin, then we would expect to correctly predict 0.6 times the number of positive cases, i.e. $0.6 \times 5 = 3$, hence sensitivity would be $3/5 = 0.6$. Concomitantly, the specificity is expected to decrease to $2/5 = 0.4$, or equivalently, the false positive rate is expected to increase to 0.6, leading to the point (0.6, 0.6) in ROC space. Thus, the ROC curve of any random guesser is expected to be the diagonal $y = x$ (Figure 1B), and we can select any point on this line by changing the bias of the random guesser. The expected AUC of a random classifier is 0.5. An AUC of 0.5 is therefore commonly interpreted as an indicator for a useless model, whereas a value of 1.0 is associated with a perfect classifier [15, 21].

By multiplying the ROC ordinate by the prior probability (or relative frequency) of the class +, $P(+)$, and correspondingly the abscissa by $P(-)$, we obtain the frequency-scaled ROC (FROC) [63]. The slope of the FROC is interpretable as the posterior odds ±, given a specific threshold. In contrast, the slope of the ROC curve is interpretable as the likelihood ratio at a given threshold.

ROC analysis enjoys a great deal of popularity in the machine learning community because it addresses a range of problems [18], including (i) determining the optimal operating parameter that minimizes the error rate or misclassification costs under specific class and cost distributions; (ii) identifying conditions under which two classifiers have unequal performance (i.e. a classifier A may be uniformly preferable to another classifier B, or A may dominate B in some

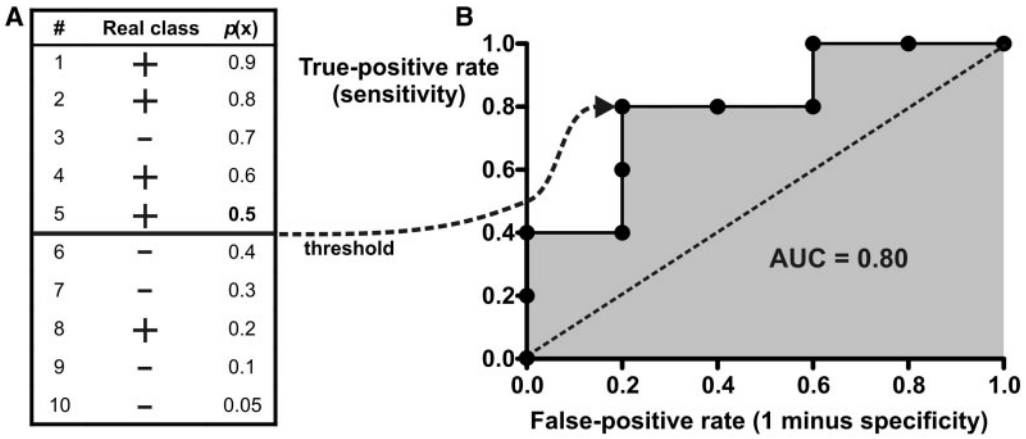

Figure 1: An example of a ROC curve. (**A**) Ten test cases are ranked in decreasing order based on the classification score (e.g. estimated class posterior probabilities). Each threshold on the score is associated with a specific false positive and true positive rate. For example, by thresholding the scores at 0.45 (or anywhere in the interval between 0.4 and 0.5), we misclassify one actual negative case (#3) and one actual positive case (#8). We may translate this into a classification rule: 'if $p(\boldsymbol{x}) \geq 0.45$, then the class is $+$, else the class is $-$'. The resulting false positive rate is 0.2, and the true positive rate is 0.8. The corresponding point in ROC space is (0.2, 0.8). (**B**) Proceeding analogously for all possible thresholds gives the ROC curve. The number of line segments corresponds to the number of cases (here, 10); the number of junction points corresponds to the number of intervals where we can set the thresholds (i.e. one plus the number of cases). The threshold associated with the point (0.2, 0.8) is the optimal operating threshold in this example, where optimality means that the sum of sensitivity and specificity is maximal. The shaded area shows the area under the curve, AUC = 0.80, which quantifies the classifier's performance over all possible thresholds. The dotted line $y = x$ is the expected ROC curve of a random guesser.

scenarios, whereas *B* may dominate *A* in others); (iii) identifying conditions under which a classifier performs no better than random by chance; and (iv) comparing classifiers regardless of units and scale. Thus, ROC curves convey all information necessary to assess the performance of a scoring classifier.

In contrast, common accuracy-based metrics (such as error rate, sensitivity and specificity) derived from the confusion matrix depend on a user-defined, single threshold value, which is often arbitrary. A major advantage of ROC curves is that they depict the performance of the classifiers over the complete range of threshold values. As the ROC curve is threshold-independent, so is the resulting AUC. Consequently, ROC and AUC are considered the most objective method for evaluating and comparing classification performances and recommended for the evaluation of binary classifiers [15].

A ROC curve is convex if any straight line interpolating between two points on the curve is never above the curve. Here, we follow standard machine learning terminology and use 'convex' to emphasise that the convex hull encloses the points on the ROC curve. In mathematical terms, the function defining the convex hull is concave (see also ref. [64], p. 109). The curve in Figure 1B is not convex but has two concavities: one between (0.0, 0.4) and (0.2, 0.8), and another between (0.2, 0.8) and (0.6, 1.0). Linear interpolation corresponds to making a random choice between the two extreme points. The simplest example of this is the ascending diagonal, which as we have seen corresponds to making a random choice between 'always predicting the positive class' [the point (1, 1)] and 'always predicting the negative class' [the point (0, 0)]. If a ROC curve has a concavity, it means that two thresholds exist between which the classification scores perform worse than random guessing. In Figure 1A, we have a negative before two positives between thresholds 0.8 and 0.5, and a positive after two negatives between thresholds 0.5 and 0.2. One obvious way to deal with this situation is to revert to random guessing between those thresholds, or—what practically works out as the same thing—fix the classifier's score to a constant in that interval. For example, we could set $P(\mathbf{x}) = 0.6$ for instances #3 to #5 and $P(\mathbf{x}) = 0.3$ for instances #6 to #8. If a positive and a negative receive the same score, then this counts as half a ranking error. So we have reduced five full ranking

errors to four half errors and one remaining full error (as negative #3 is still ranked before positive #8), and the AUC of the 'convexified' curve is 0.88. Note, however, that in this example the AUC of this convexified curve is larger than the AUC of the original curve (0.80). The process of convexifying introduces a positive bias of the estimated AUC relative to the true AUC for the large-sample setting. Thus, for the evaluation of classification performance, it is important to keep in mind that the AUC resulting from 'convexification' is an upward-biased statistic.

There is a well-defined algorithm for determining the minimum set of thresholds spanning all concavities in the curve. The result of interpolating between those thresholds is called the *ROC convex hull* [18]. Furthermore, if the slope of a convex hull segment is s (and the class distribution is uniform), then a calibrated probability is obtained by setting the classifier's score to $s/(1+s)$. In the example, we would predict $P(\mathbf{x}) = 1.0$ for instances #1 to #2, $P(\mathbf{x}) = 0.67$ for instances #3 to #5, $P(\mathbf{x}) = 0.33$ for instances #6 to #8, and $P(\mathrm{x}) = 0$ for instances #9 to #10. In general, for a class ratio $r = P(+)/P(-)$, we obtain calibrated probabilities as $sr/(1+sr)$.

CAVEATS AND PITFALLS OF ROC AND AUC

Despite these merits, ROC and AUC are not free from criticisms [23–25, 64]. Adams and Hand have shown that for calculating AUC, summarizing over all possible thresholds can be misleading [23]. In fact, as ROC curves and AUC are threshold-independent, they describe a classifier's performance over the entire operating range, which includes regions of no practical relevance such as the extreme right (high false positive rate) or extreme left side (low true positive rate).

Drummond and Holte argued that ROC analysis is inadequate for machine learning researchers in several important respects and proposed *cost curves* as an alternative [24]. Cost curves depict the normalized expected costs of classification as a function of the (predefined) misclassification costs and class distributions. Cost curves are conceptually similar to regret graphs [26] and convey the important information about binary classification performance, similar to ROC curves. However, compared with ROC curves, cost curves can be more easily interpreted when two classifiers have unequal performance for different operating points, i.e. when their ROC curves cross. To identify the conditions under which one classifier outperforms its competitor, we need to calculate and inspect iso-performance lines in ROC curves. An iso-performance line is a line on which points of equal performance are located [19]. Let FP and TP denote false positive and true positive, respectively. Then two points in ROC space, $(\mathrm{FP}_1, \mathrm{TP}_1)$ and $(\mathrm{FP}_2, \mathrm{TP}_2)$, have the same performance if $(\mathrm{TP}_2 - \mathrm{TP}_1)/(\mathrm{FP}_2 - \mathrm{FP}_1) = c(+, \mathbf{x}_-)P(\mathbf{x}_-)/c(-, \mathbf{x}_+)P(\mathbf{x}_+)$, where $c(+, \mathbf{x}_-)$ and $c(-, \mathbf{x}_+)$ indicate the costs for misclassifying an actual negative and an actual positive case, respectively; $P(\bullet)$ denotes prior probability. In contrast, the x-axis of cost curves directly shows the conditions under which one classifier is superior to another.

Lobo *et al.* [25] noted that the AUC can invite misleading interpretations. Although their study focused on geographical distribution models, their concerns are also relevant for clinical microarray research. To address some of these shortcomings, variants of the conventional AUC have been proposed, for example, the partial AUC [27]. Vanderlooy and Hüllermeier [28] investigated various variants of the AUC, but they concluded that these variants are ineffective for model evaluation and selection. Indeed, the conventional AUC still appears to be the metric of choice for evaluating and comparing binary classifiers [14, 15]. Genomic studies frequently involve the induction of hundreds of classifiers, in combination with various gene signatures, and a complete ROC analysis (with a visual inspection) is often impractical. Thus, the ultimate assessment is often solely based on the AUC. However, there are several misconceptions with respect to the interpretation of this statistic. To highlight these issues, we first revisit the differences between *ranking* and *classification*.

Classification and ranking are two different concepts

For ROC analysis, it is no prerequisite that a classifier produces calibrated estimates such as posterior probabilities. It is sufficient that the classifier outputs *relative* scores for positive and negative instances [19]. A scoring classifier outputs a numeric score s for each test case $\mathbf{x}$, $s(+ \mid \mathbf{x})$. This score quantifies the degree of class membership to the positive class. Assuming that these scores are expressed on an additive scale and reach a maximum $s_{\max}$ in case of certainty, we can set $s(- \mid \mathbf{x}) = s_{\max} - s(+ \mid \mathbf{x})$. The ratio $f(\mathbf{x}) = s(+ \mid \mathbf{x})/s(- \mid \mathbf{x})$ can then be used to rank the instances from the most to the least likely positive.

Rankers can be turned into classifiers by setting a threshold on $f(\mathbf{x})$. The ROC curve visualizes the quality of a ranker without committing to a classification threshold. Analogously, the AUC quantifies the *ranking* ability of a model. Classifiers and rankers optimize a different loss function. Whereas classifiers aim at minimizing classification errors, rankers try to minimize the number of ranking errors.

Consider the problem of predicting the estrogen receptor (ER) status of breast cancer samples based on their gene expression profiles. We used a publicly available microarray data set of 119 breast cancer samples [9]. This data set contains 34 estrogen receptor negative (ER−) and 85 estrogen receptor positive (ER+) cases. Using stratified random sampling, we generated a training set of 69 samples (20 ER− and 49 ER+) and a test set of 50 samples (14 ER− and 36 ER+). Using Welch's *t*-test, we selected the top-30 genes (more precisely, these features are probe sets, referring to genes, ORFs or ESTs. For reasons of simplicity, we denote these here as *genes*), i.e. those genes with the highest discriminatory power for the ER status in the training set. Using the top-ranking genes and diagonal linear discriminant analysis [*dlda*, Equation (2)], we built a classifier (model #1). Then, we applied model #1 to the test set, as illustrated in Figure 2.

$$dlda(\mathbf{x}, c) = \sum_{i=1}^{p} \frac{(x_i - \bar{x}_{ci})^2}{s_i^2}. \quad (2)$$

Here, c denotes the class (i.e. either ER+ or ER-); p is the number of genes (here, $p = 30$), x_i is the value of the i-th gene, $\bar{x}_{ci}$ is the mean of all values of ith gene in class c, and ${s_i}^2$ is the pooled estimate of the variance of the ith gene. As a binary classifier, diagonal linear discriminant analysis assigns a case $\mathbf{x}$ to the class c that minimizes the sum in Equation (2).

For each case $\mathbf{x}$, we calculated the difference $d(\mathbf{x}) = dlda(\mathbf{x}, \text{ER}-) - dlda(\mathbf{x}, \text{ER}+)$ and used it as ranking score for ROC analysis. The sign of $d(\mathbf{x})$ was used for the classification decision, where sign $(d(\mathbf{x})) = -1$ means that $\mathbf{x}$ is classified as ER−. Loosely speaking, *dlda* yields a decision boundary (implicitly), and the classification result depends on which side of the boundary a case is located. For the test set, we obtained AUC = 0.96. One actual ER− test case was misclassified as ER+, and four actual ER+ test cases were misclassified as ER− (error rate = 0.10).

Next, we deliberately corrupted the training set by swapping the class label of a randomly selected ER+ case and a randomly selected ER− case: the ER+ case became ER−, and vice versa (Figure 2A, middle panel). Thus, the conditional distribution in the (corrupted) training set was no longer the same as in the test set and original training set. As *dlda* is a deterministic algorithm (unlike, for example, multilayer perceptrons), we expected to observe a weaker performance of model #2, compared with model #1, because the training set was corrupted—this was the only difference between the two models. Would the AUC of model #2 reflect this somehow inferior performance?

We selected the top-30 genes as described for model #1 and built model #2 using *dlda*. We applied this model to the same test set as before and performed ROC analysis. We repeated this experiment three times using different randomly selected ER+ and ER− cases. In all experiments, the AUC was equal to AUC = 0.96 of model #1. However, based on sign($d(\mathbf{x})$)—the side of the boundary that a case was located—model #2 misclassified one actual ER− test case as ER+ and six actual ER+ test cases as ER− (error rate = 0.14, in contrast to error rate = 0.10 of model #1). Why is the expected inferior performance reflected by the increased error rate, but not by the AUC?

Figure 3 gives a possible explanation. Here, the cases are intentionally arranged in a grid-like manner for illustration purposes only.

Model #1 (represented by the vertical line in the top panel of Figure 3) classifies cases on the left side as ER− cases and cases on the right side as ER+ cases. In the test set, model #1 misclassifies five cases (four ER+ and one ER−). By swapping the class label of two randomly selected cases (middle panel in Figure 3), we corrupted the training set. Given that the classes are imbalanced (a ratio of 20:49), the new decision boundary is expected to be shifted towards the majority class, as illustrated by the vertical line of model #2. For illustration purpose and reasons of simplicity, we assume here that the new boundary is shifted parallel to the boundary of model #1. Clearly, the effect of the label swap on the shape of the decision boundary depends on the specific machine learning algorithm. By swapping the class label of only two cases, as in this example, the separating hyperplane of a support vector machine would arguably not change. Different methods with different inductive biases have different resilience against corrupted data. However, the example can be easily extrapolated to more extreme scenarios with a higher

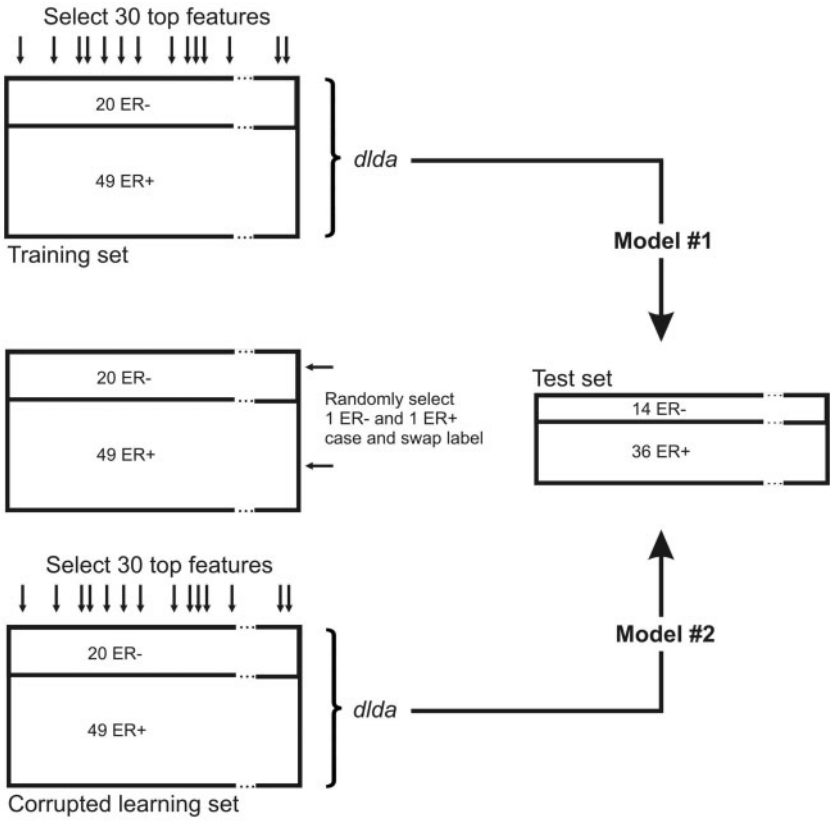

Figure 2: Experiments with deliberately corrupted microarray data. Model #1 is built from the original training set, whereas model #2 is built from deliberately corrupted training data. Both models are applied to the same test set.

level of noise that would affect even robust learners, too. It is also easy to contrive examples of 'corrupted' boundaries that are not shifted parallel to the 'good' boundary (for instance, by slightly slanting the boundary of model #2 in Figure 3). When we apply the 'corrupted' model to the test set, we make a total of seven misclassifications (i.e. six actual ER+ cases misclassified as ER−, and one actual ER− cases misclassified as ER+). Importantly, however, the *ranking* of the test cases relative to each other remains *unchanged*, irrespective of whichever model we use. Thus, the AUC of the two models is the same. The ROC curves, on the other hand, are not exactly the same. Their shape is the same, but they differ with respect to the scores $d(\mathbf{x})$ that yield the operating points. The identical AUC could now invite conclusions, such as 'both models #1 and #2 are equally good', 'both models have learnt the learning concept equally well', 'it does not matter which model we will use for future applications', and so on. However, here we know that these conclusions are not warranted.

This quandary is due to the *a priori* defined classification criterion: the sign of $d(\mathbf{x}) = dlda(\mathbf{x}, \text{ER}-) - dlda(\mathbf{x}, \text{ER}+)$, or the side of the boundary where a case happens to fall. Yet, this is common practice: the location of a case relative to the boundary (for example, the separating hyperplane of a support vector machine) determines the predicted class membership. Note that this example is not intended as a cautionary tale warning against *dlda*. The shifted decision boundary can be explained by the changed conditional class distributions due to the label swap. The example illustrates the conceptual difference between crisp classification and ranking. The example also shows that model comparisons based on AUC alone can be misleading.

ROC results should be interpreted with caution when the models were derived from different training sets. In practice, this can indeed be the case, specifically when performances are compared with results published in the literature. Here, the training sets are practically never identical due to different data samplings. It is often a tacit assumption that the

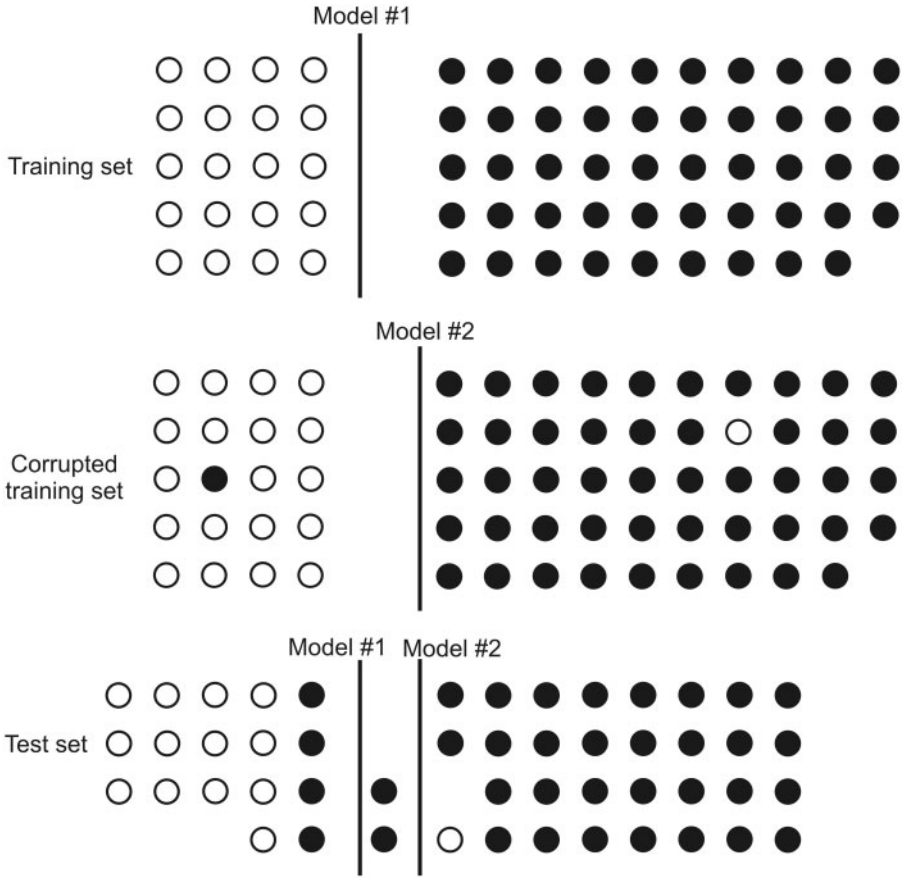

Figure 3: Experiments with deliberately corrupted microarray data. Circles represent cases of class ER–, full dots represent cases of class ER+. Model #1 results from the application of diagonal linear discriminant analysis to the original training set (upper panel). Model #2 results from the application of diagonal linear discriminant analysis to a corrupted training set (middle panel), where the class label of two randomly selected cases (1 ER+ and 1 ER−) were swapped. Both models are then applied to the same test set (bottom panel). The training and test cases are arranged in a grid-like manner for reasons of simplicity.

training set and the test set originate from the same distribution, but this is not necessarily the case in real-world applications [55].

Biases present a real threat to the validity of clinical microarray studies [30]. Specifically, biases due to mislabeled cases—as simulated in the previous example—can have a profound impact on the validity of classification results. Such inadvertent mislabelings indeed occur in real microarray studies [31]. For example, a recent study that aimed at deriving gene signatures for the prediction of response to the cancer drug doxorubicin inadvertently mislabeled 'sensitive' and 'resistant' cases [32]. Another example is a study that aimed at developing microarray-based predictors for response to cisplatin and pemetrexed [33]: class labels were accidentally corrupted, as shown in ref. [31]. Another study aimed at developing a genomic classifier for predicting response to primary platinum-based chemotherapy in ovarian cancer, and the classifier achieved an AUC of 0.875 [34]. However, Baggerly *et al.* [31, 35] pointed out severe problems due to mislabeled cases in this data set. A further study [36] that validated gene signatures for the prediction of response to neoadjuvant chemotherapy has been criticized for confounding treatment effects with run batch effects [31]. Despite this reported confounding, a high value of AUC (0.875) could be achieved [36].

Clearly, knowing the caveats and pitfalls of ROC analysis, specifically the difference between ranking and classification, is no safeguard against problems that are due to mislabelings or run batch effects. However, the example shown in Figure 3 offers a possible explanation why it can be possible to obtain relatively high values of AUC despite corrupted data.

The fallacy of the undistributed middle

Perhaps the most common misconception is that AUC = 0.5 necessarily implies that the classifier is no better than random guessing, but this is not necessarily the case, as discussed in ref. [18]. Indeed, if we randomly guess the class labels of the test cases (for example, by flipping a fair coin), then we expect to achieve an AUC of 0.5. It is a fallacy, however, to assume that *any* classifier that achieves AUC = 0.5 is no better than random guessing and has therefore no predictive ability. This common misconception is a 'fallacy of the undistributed middle': all random models score an AUC of 0.5, but not every model that scores an AUC of 0.5 is random. In fact, even a classifier that is perfect in some sense can achieve an AUC of 0.5.

Consider a classifier that is trained to predict either $y =$ 'disease' or $y =$ 'normal' for a case $\mathbf{x}$. Assume that a training set contains hundreds of variables, and assume that only variable i has predictive relevance, while all other variables represent pure noise or are otherwise irrelevant for the classification task. Let us assume that i exceeding a threshold t_1 indicates 'disease' and that i being below another threshold t_2 also indicates 'disease'. Thus, if $i \in [t_2, t_1]$, then the status is 'normal' and 'disease' otherwise. Manifold biomedical examples could be cited; for example, consider the variable 'systolic blood pressure' where values outside the normal range indicate 'disease', or the expression of a gene that should be in a normal range, i.e. neither over- nor under-expressed. Now assume that the classifier has correctly identified i as the only predictive variable and bases all predictions on its numerical values. Thus, assume that the classifier has learnt the decision function $c(y =$ 'normal' $\mid \mathbf{x},\ i \in [40; 70])$ and $c(y =$ 'disease' $\mid \mathbf{x},\ i \notin [40; 70])$ from the training set. The classifier then applies this function to the test set. Figure 4A shows the prediction results for a test set of 12 cases. This 'one-feature' classifier identified the only predictive variable and correctly classified all test cases. From an Occam's razor perspective, the classifier is both parsimonious and self-explanatory. The simple classification rule might also give insights into the pathophysiology. Hence, this classifier could rightly be considered perfect. However, consider now the ROC curve (Figure 4B) that results from the ranking of the test cases based on the attribute values.

The example shown in Figure 4A is an instance of an XOR problem, which requires two decision boundaries for the separation of the test cases based on attribute i. Here, the AUC is 0.5 [95% confidence interval (CI), 0.10–0.90], which could lead to the conclusion that this classifier is no better than random guessing. This discrepancy can be explained by the fact that the AUC quantifies the ability of a classifier to *rank* cases relative to each other. Indeed, in this example, the ranking is no better than random by chance ($P = 1.0$; Wilcoxon rank-sum test). The ROC curve shows that this classifier is not a typical random guesser because the curve is different from the expected diagonal line $y = x$. However, genomic studies frequently involve the evaluation of hundreds of classifiers [39], and a visual inspection is therefore

A

#	True class	Attribute *i*	Predicted class
1	disease	100	disease
2	disease	90	disease
3	disease	80	disease
4	normal	70	normal
5	normal	60	normal
6	normal	50	normal
7	normal	55	normal
8	normal	45	normal
9	normal	40	normal
10	disease	30	disease
11	disease	20	disease
12	disease	10	disease

B

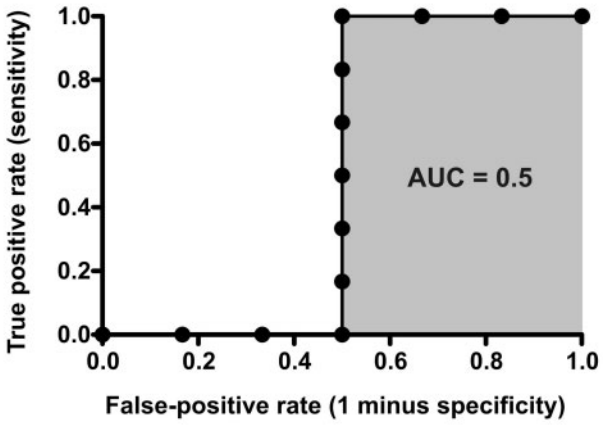

Figure 4: (**A**) A test set containing $n = 12$ cases of two classes that require two decision thresholds. (**B**) AUC of 0.5 does not necessarily indicate a useless model if the classification requires two thresholds (XOR problem).

not always feasible. In such scenarios, the AUC is then used as a single criterion, and models that score AUC = 0.5 are deemed useless [21] but possibly undeservingly so.

Note, that the classifier in this example is not a scoring classifier; the ranking of test instances was based on the 'raw' values of attribute *i*. The classification rule of this simple 'one-feature' classifier did not produce any ranking scores, such as posterior probabilities or other scores quantifying the degree of class membership. Therefore, we can object that the discrepancy in the previous example is due to inadequately chosen ranking scores, thus a violation of the ROC assumptions. This problem could be easily resolved in this example by considering posterior class probabilities, i.e. $P(y = \text{'normal'} \mid \mathbf{x}, i \in [40; 70]) = 1$ and $P(y = \text{'normal'} \mid \mathbf{x}, i \notin [40; 70]) = 0$. These scores would then lead to AUC = 1.0. It has been noted that ROC analysis can be performed for any variable with a continuous spectrum of results [38]; however, it is more precise to note that the values need to be proper ranking scores.

But even calibrated scores can lead to AUC = 0.5, although the classifier is not random. Consider the example in Figure 5A (adapted from ref. [68]). Here, the positive and negative instances (represented by squares and circles, respectively, with 1 unit length distance from each other) are not linearly separable but require two classification boundaries. The dotted line represents a separating hyperplane that defines the boundary between the positive and the negative class. Here, the signed distance (Figure 5B) to the boundary is the ranking score for ROC.

The hyperplane in Figure 5A is unable to solve the XOR problem, but it is not a random classifier, as the AUC might suggest. Linear separating hyperplanes are popular classifiers in genomic studies, for example, hyperplanes resulting from support vector machines. Commonly, the (signed) distances to the decision boundary are used as a surrogate for estimating the degree of class membership (often with probabilistic semantics) and used to perform ROC analysis [39]. Other classifiers, such as the family of linear discriminant models, produce class posterior probabilities, and the decision boundary is the set of points for which the log-odds are zero. If a classifier assigns the highest scores to exactly half of the instances of a particular class and the lowest scores to the other half as in the example shown in Figure 5B, then the AUC is exactly 0.5. Such a classifier performs excellently on some of the instances, but this performance is offset by the very poor performance on some other instances. However, such a classifier is clearly not equivalent to the flip of a coin. Thus, if we accept AUC = 0.5 as a binary indicator for a random classifier, then we might discard some models that could indeed deserve closer inspection of their ROC curves.

ROC analysis and the early retrieval problem

The early retrieval problem arises when we are interested in the top-ranked test cases only [15, 37], for example, in classification scenarios where the number of predicted objects is in the order of several hundreds or even thousands. An example is a drug discovery study that aims at ranking a multitude of chemical compounds based on their toxicological effect [40]. Here, it is typically possible to follow up on only a small number of predicted cases, thus,

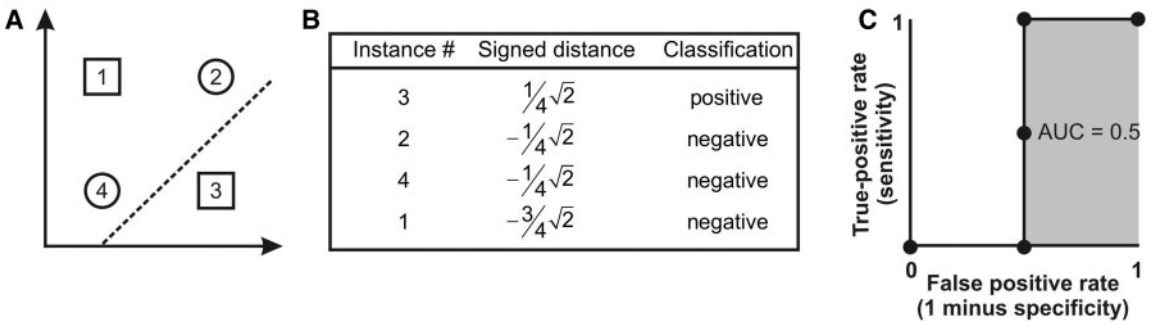

Figure 5: (**A**) XOR problem comprising two instances of the positive class (#1 and #3) and two instances of the negative class (#2 and #4). The instances are placed at a distance of 1 U length from each other. The separating hyperplane (dotted line) is used as decision boundary. (**B**) The instances are ranked based on the signed distance from the decision boundary. (**C**) ROC curve resulting from the scores of signed distances (example adapted from ref. [68]).

the top-ranked cases. In this scenario, conventional ROC curves (and AUC) are inadequate.

Consider an example where a classifier predicts the label of 1000 test cases, with 500 cases belonging to the positive and 500 cases belonging to the negative class. Assume that we have the resources to further investigate only 100 cases predicted as 'positive'. Ideally, the top-100 of the ranked list would then contain cases of the positive class only, whereas the ranking of the cases is irrelevant in the remainder of the list. Figure 6 shows the ROC curve for a simulated classification result, in which a model ranks 90 actual positive and 10 actual negative cases in the top-100 list. In the remainder of the list, 410 actual positive and 490 actual negative cases are randomly distributed. This model would be fit for our purpose.

Here, the steep increase in the early curve (dashed box in Figure 6) indicates that most of the top-ranked cases are indeed actual positive cases. But the good performance with respect to the 100 top-ranked cases is offset by the random performance with respect to the remaining 800 cases, which leads to an AUC of 0.58 only. But clearly, as we are not interested in the *entire* ROC curve, the AUC is meaningless.

Recently, Swamidass *et al*. [40] proposed the *concentrated ROC* (CROC) and the *concentrated AUC* (cAUC) to address the early recognition problem. In clinical microarray analysis, the number of predicted cases is typically much smaller than in the example shown in Figure 6. However, CROC is also suitable to distinguish whether a model with AUC = 0.5 is in fact a random classifier. Therefore, even for small sample sizes, CROC is an interesting alternative to ROC.

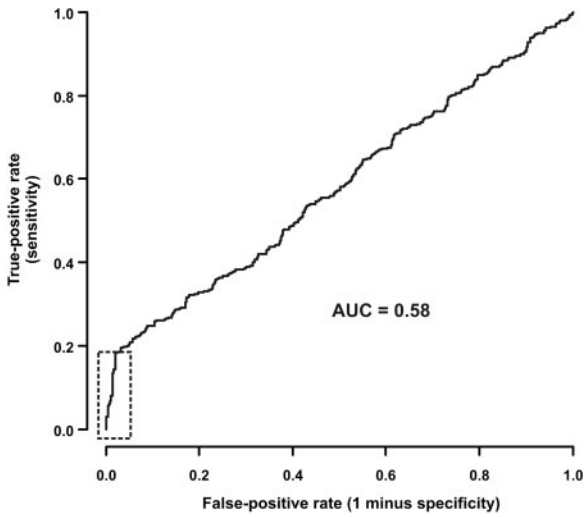

Figure 6: Illustration of the early retrieval problem for a simulated classification problem comprising 500 positive and 500 negative cases. The top-100 list of ranked cases contains 90 positives and 10 negatives; in the remaining list, the cases are randomly ranked.

The significance of the AUC in multiple testing scenarios

The AUC is frequently accompanied by a *P*-value, which reflects the probability of obtaining an AUC as high as or higher than the observed one when we base all predictions on random guessing. When we assess the performance of multiple models, for example, a single inductive learning algorithm applied to various gene signatures, we are effectively entering the arena of multiple hypotheses testing. Hence, the *P*-values for the *individual* AUC metrics must be adjusted to account for the multiplicity effects. The global (or family-wise) null hypothesis is that no classifier is a significant model. Suppose that a genomic study involves $n = 100$ classifiers that, in fact, are no better than random guessing. If we assess the AUC of each individual model at $\alpha_{\text{individual}} = 0.05$, then we expect to make five false positive decisions; hence, we erroneously conclude that the AUC of these five models is significantly >0.5. An approach to address this problem would be the Bonferroni's adjustment, which controls the family-wise error rate at the desired level $\alpha_{\text{family}} = 0.05$ by dividing $\alpha_{\text{individual}}$ by the total number of models (here, $\alpha_{\text{individual}} = 0.05/100$). This adjustment guarantees the control of the significance level, but it is also known to be very conservative and to lead to a high type II error. Holm's correction represents a more liberal approach and involves an ordering of the individual *P*-values in increasing order [41]; here, the smallest *P*-value is compared to $\alpha_{\text{individual}}/n$, the second smallest *P*-value is compared to $\alpha_{\text{individual}}/(n-1)$, and so on. There exists a plethora of methods addressing the multiplicity problem; for an overview, see for example [42]. Whereas the multiple testing problem arising in gene selection is now generally acknowledged [43], the problem of multiple models may not have received adequate attention in genomic studies yet. Technically, the same statistical methods could be applied. It is difficult to recommend a specific correcting method, though. The choice of the correcting method depends on how liberal or conservative the researcher wishes to be. In our experience, Holm's method works reasonably well for

controlling the family-wise false positive rate (see also [44]).

A central question, however, remains open: what represents an objectively 'good' value of AUC? This question is difficult to answer for a specific problem at hand, let alone for the general case. The multiplicity-adjusted *P*-value does not provide an answer to this question.

CIs are more meaningful than *P*-values

The *P*-value for the AUC of a specific model quantifies the extent to which it deviates from a model with AUC = 0.5. However, as we have shown with the fallacy of the undistributed middle, a model with AUC = 0.5 does not necessarily need to be a random guesser. Yet there is another, more fundamental problem. The *P*-value is not to be mistaken as the posterior probability that the null hypothesis is true. The *P*-value is the probability of observing an outcome as extreme as or more extreme than the actual one, given that the null hypothesis is true: *P*-value = Pr(observed or more extreme | H0 is true). Transposing the conditional would be another fallacy. A critically important point is that the *P*-value does not convey information about the effect size. For example, assume that two models are compared on a very large test set. A significance test for the difference between the AUC of two models may give a 'highly significant' result with $P < 0.0001$, but the actual, absolute difference (i.e. the effect size) between the areas could be too small to be of any practical relevance.

CIs, on the other hand, have long been advocated as a more informative alternative to hypothesis testing [45–48]. For the AUC, Hanley and McNeil [49] proposed a CI by assuming that the metric is an approximately normally distributed random variable. However, the normal assumption is violated, and the statistical underpinning therefore questionable. In contrast, methods that do not rely on the normality assumption often provide intervals that are too wide to be of any practical use [50]. Cortes and Mohri [50] proposed a distribution-independent alternative for deriving a CI for AUC by using a variance estimation over all classifications with a fixed error rate. The problem of this approach, however, is that it entangles AUC with a fixed error rate. In six benchmark data sets, the method by Cortes and Mohri [50] provided tight intervals, hence appears to be of practical use; but also the method by Hanley and McNeil performed well despite the violated assumptions.

Bootstrapping is an alternative for the construction of approximate CIs. In short, bootstrapping is a data resampling strategy that repeatedly, uniformly picks (with replacement) instances from a data set to generate bootstrap samples [51]. Each bootstrap sample contains the same number of cases as the original training set, but with duplicates. We could derive a bootstrapped CI for AUC as follows. First, we sample several hundred bootstrap test sets. Then, we calculate the AUC of the classifier on each set. Finally, we use the $\alpha/2$ and $1-\alpha/2$ percentiles of the distribution to obtain an empirical $(1-\alpha/2)$-level CI. Using the percentile bootstrap, we can also derive an approximate CI for the difference between the AUC of two classifiers. If this interval does not include zero, then we can be 95% confident that the two models have different performance; hence, we can be confident that the winning model is truly better for the specific task at hand.

In addition, the interval can tell us something about the effect size, i.e. whether this difference is of any practical relevance. In a multiple testing scenario where several such intervals are constructed, the individual significance levels need to be adjusted accordingly. Note, that we cannot compare the 95% CI of two different values of AUC (i.e. to check whether they overlap or not) to mimic a significance test of difference at the 5%-level.

Which metric matters for practical applications?

Figure 7 shows the most common performance metrics in a binary classification task.

In real-world applications, it is often difficult or even impossible to define the costs for false positive and false negative classifications. In such scenarios, metrics like the AUC are attractive because they average over the range of all possible thresholds, which are determined by the costs. Hence, AUC

	Real +	Real –
Predicted +	*a* (true positive)	*b* (false positive)
Predicted –	*c* (false negative)	*d* (true negative)
	a + *c*	*b* + *d*

Figure 7: Confusion matrix; common performance metrics are accuracy = $(a+d)/n$; error rate = $(b+c)/n$; sensitivity = $a/(a+c)$; specificity = $d/(b+d)$; positive predictive value = $a/(a+b)$; negative predictive value = $d/(c+d)$.

averages over the range of all possible costs. However, this also includes costs that, although theoretically possible, have no practical relevance, as shown in the early recognition problem.

Predictive genomic classifiers are models for the prediction of response to therapy based on gene expression data. Here, sensitivity refers to the detection of responsiveness among true responders. These models are often intentionally tailored for a high sensitivity and a high negative predictive value (NPV) [52, 53] because, ideally, we wish to identify at an earlier stage those patients who are likely not to respond, so that alternative treatment avenues could be pursued [36]. Recent examples of such models include a 30-gene classifier to predict response to neoadjuvant chemotherapy in breast cancer [39], and a 50-gene model to predict colon cancer recurrence [54]. ROC curves, however, depict sensitivity as a function of (1 minus) specificity. Thus, they do not readily reflect the desirable NPV. Therefore, an AUC-optimizing model may not be what we are looking for. Hand pointed out that in classification studies, there is frequently a mismatch between the criterion used to select and assess a model and the criterion that actually matters in the real application [55].

Open source software for ROC analysis

There exists a variety of both commercial and free software tools to perform ROC analysis. Here, we will focus on open source software. Freely available tools enjoy a great popularity in academic research because of the support from a generally large user community. This is particularly true for implementations based on the language and environment R [29], which is widely regarded as the *de facto* standard for statistical computing. For bioinformatics, the R *bioconductor* project [56] provides a variety of packages for data analysis and visualization. These packages include the library ROC that provides examples for ROC analysis in the context of microarray research. ROCR [57], another R library, offers arguably the most comprehensive collection of features, including advanced functions such as the ROC convex hull and functions to analyze results from resampled data sets. ROCR contains additional functions for analyzing and visualizing scoring classifiers, including cost curves [24]. The R library caTools [58] contains the function colAUC, which plots ROC curves and calculates AUC. A minor shortcoming of these packages is that they do not accompany the AUC with a *P*-value, in contrast to the function plot.roc of the library analog [59]. A commonly lacking (albeit important) feature of all these R functions is the calculation of a CI for AUC. As a workaround, we could combine the function boot.ci of the R library boot with the functions for ROC analysis. The function boot.ci provides five different bootstrapping methods to generate approximate CIs, including the percentile method described in the section 'CIs are more meaningful than *P*-values'.

Among the open source data mining suites, Weka is arguably the most widely used in bioinformatics research [60]. Weka provides functions for ROC analysis, but does not readily give CIs for AUC. StAR (Statistical comparison of ROC curves, [61]) is a software specifically designed for the pair-wise comparison of AUC of different classifiers. This unique feature of StAR is frequently missing in other software packages, including commercial tools such as GraphPad Prism [62]. Prism, however, is one of the few tools that accompany the AUC with a CI by default.

DISCUSSION

The ROC has emerged as the method of choice for assessing and visualizing the performance of binary genomic classifiers. ROC analysis offers advantages over conventional scalar metrics as it disentangles performance evaluation from misclassification costs, class imbalances and single thresholds. To avoid potentially misleading interpretations, it is important to acknowledge the fundamental difference between classification and ranking. ROC measures the latter. Here, we not only discussed the merits of ROC but also paid attention to possible pitfalls in its interpretation, particularly with respect to its scalar summary statistic, the AUC. In real-world genomic studies, it is not uncommon to evaluate thousands of classifiers. As it is impractical to visually inspect all resulting ROC curves, summary statistics are the *de facto* standard method to compare and select models. Also, introductory texts sometimes suggest that the rationale for plotting ROC curves is that they allow the calculation of the AUC [63].

Although the AUC is certainly a more meaningful performance measure than accuracy [20], it has attracted criticisms [23–25, 64]. Recently, Hand showed that a particular interpretation of AUC as expected minimum loss is incoherent in that the

expectation is taken over classifier-dependent score distributions [64]. Hand proposed the new H-measure as an alternative to AUC (an implementation in R is available at ref. [22]).

Another problem is that microarray data are typically characterized by the small-*n*-large-*p* problem: the data set contains many genes (*p*) but only very few cases (*n*) [65]. ROC curves derived from such small-sample data sets may not reliably reflect a classifier's true performance [66]. Hanczar *et al*. [66] cautioned against the use of ROC analysis unless the sample size is 'very large', yet it remains an open question how large a sample size is needed for a reliable ROC analysis.

Performance metrics in a clinical setting should take pretest probabilities into account [67]—a feature that AUC cannot provide. As a (one-dimensional) scalar, the AUC cannot paint the full picture of a classifier's performance, in contrast to the two-dimensional ROC curves. Judgments based on only point estimates for AUC should therefore be interpreted with caution.

In a clinical setting, a central question is how closely AUC measures clinical gain. Consider the hypothetical scenario in which two patients with a suspected life-threatening condition require an urgent treatment. However, only one patient really has this condition, whereas the other patient does not. Suppose that the hospital has the equipment to treat only one patient. Here, AUC is interpretable as the chance of choosing the right patient to treat, and (1 – AUC) is the number of missed preventions of avoidable deaths per triage situation. In this (admittedly extreme) scenario, the clinical gain is manifest. In other, perhaps more realistic scenarios, the clinical gain is less direct, but it is actually the outcome of real interest. From this perspective, we may regard AUC as one (of many) inputs for a decision process that aims at maximizing some predefined clinical gain. For the general case, we cannot derive any summary statistics from ROC to directly measure clinical benefit or loss [67]. Hilden suggested utility models of pseudo-regret as a more direct approach, for example, models based on the quadratic (Brier) pseudo-regret function that rank classifiers by the expected pretest-posttest difference in utility [63, 67].

One may argue that the criticisms against AUC do not pertain to ROC analysis *per se*. Indeed, ROC curves *are* a valuable tool for evaluating and comparing classifiers, and we do encourage the use of ROC analysis. However, it is easy to read too much in a simple summary statistic such as AUC. We hope that this article contributes to a more careful use and interpretation of AUC and hence of ROC analysis.

Key Points

- ROC analysis measures a model's ability to rank positive and negative cases relative to each other.
- Not all classifiers with an AUC of 0.5 are random guessers.
- The AUC cannot directly measure clinical gain (or loss), which is the outcome that really matters in a clinical setting.
- AUC should always be accompanied by a confidence interval.
- For predictive genomic classifiers, the negative predictive value may be more important than AUC.

Acknowledgements

The authors thank the anonymous reviewers for their valuable feedback that greatly helped improve the article. Specifically, the authors are grateful to one of the anonymous reviewers for the valuable comments on the clinical gain.

References

1. Chang JC, Wooten EC, Tsimelzon A, *et al*. Gene expression profiling for the prediction of therapeutic response to docetaxel in patients with breast cancer. *Lancet* 2003;**362**(9381): 362–9.
2. Ayers M, Symmans WF, Stec J, *et al*. Gene expression profiles predict complete pathologic response to neoadjuvant paclitaxel and fluorouracil, doxorubicin, and cyclophosphamide chemotherapy in breast cancer. *J Clin Oncol* 2004; **22**(12):2284–93.
3. Iwao-Koizumi K, Matoba R, Ueno N, *et al*. Prediction of docetaxel response in human breast cancer by gene expression profiling. *J Clin Oncol* 2005;**23**(3):422–31.
4. Jansen MP, Foekens JA, van Staveren IL, *et al*. Molecular classification of tamoxifen-resistant breast carcinomas by gene expression profiling. *J Clin Oncol* 2005;**23**(4):732–40.
5. van't Veer LJ, Dai H, van de Vijver MJ, *et al*. Gene expression profiling predicts clinical outcome of breast cancer. *Nature* 2002;**415**:530–6.
6. Bueno-de-Mesquita JM, van Harten WH, Retel VP, *et al*. Use of 70-gene signature to predict prognosis of patients with node-negative breast cancer: a prospective community-based feasibility study (RASTER). *Lancet Oncol* 2007;**8**(12):1079–87.
7. Chang HY, Nuyten DS, Sneddon JB, *et al*. Robustness, scalability, and integration of a wound-response gene expression signature in predicting breast cancer survival. *Proc Natl Acad Sci USA* 2005;**102**(10):3738–43.
8. Sørlie T, Perou CM, Tibshirani R, *et al*. Gene expression patterns of breast carcinomas distinguish tumor subclasses with clinical implications. *Proc Natl Acad Sci USA* 2001; **98**(19):10869–74.
9. Sotiriou C, Wirapati P, Loi S, *et al*. Gene expression profiling in breast cancer: understanding the molecular basis of

histologic grade to improve prognosis. *J Natl Cancer Inst* 2006;**98**(4):262–72.

10. Wang Y, Klijn JG, Zhang Y, *et al*. Gene-expression profiles to predict distant metastasis of lymph-node-negative primary breast cancer. *Lancet* 2005;**365**(9460):671–9.
11. Liu R, Wang X, Chen GY, *et al*. The prognostic role of a gene signature from tumorigenic breast-cancer cells. *N Engl J Med* 2007;**356**(3):217–26.
12. Chi JT, Wang Z, Nuyten DS, *et al*. Gene expression programs in response to hypoxia: cell type specificity and prognostic significance in human cancers. *PLoS Med* 2006;**3**(3): e47.
13. Michiels S, Koscielny S, Hill C. Prediction of cancer outcome with microarrays: a multiple random validation strategy. *Lancet* 2005;**365**(9458):488–92.
14. Ahmed AA, Brenton JD. Microarrays and breast cancer clinical studies: forgetting what we have not yet learnt. *Br Cancer Res* 2005;**7**(3):96–9.
15. Søreide K. Receiver-operating characteristic (ROC) curve analysis in diagnostic, prognostic and predictive biomarker research. *J Clin Pathol* 2009;**62**:1–5.
16. Wong AKS, Lee JWT, Yeung DS. Improving text classifier performance based on AUC. *Proceedngs of Eighteenth International Conference on Pattern Classification*, Hong Kong, 2006, 268–71.
17. Simon R. A roadmap for developing and validating therapeutically relevant genomic classifiers. *J Clin Oncol* 2005; **23**(29):7332–41.
18. Flach P. ROC analysis. In: Sammut C, Webb GI, (eds). *Encyclopedia of Machine Learning*. Berlin/Heidelberg: Springer, 2010;869–74.
19. Fawcett T. ROC Graphs: Notes and practical considerations for researchers. *Technical Report HPL-2003-4*. HP Laboratories, 2004.
20. Ling CX, Huang J, Zhang H. AUC: A statistically consistent and more discriminating measure than accuracy. *Proc 18th Intl Conf Art Int* 2003;**18**:519–26.
21. Obuchowski NA, Lieber ML, Wians FH, Jr. ROC curves in clinical chemistry: Uses, misuses, and possible solutions. *Clin Chem* 2004;**50**:1118–25.
22. R code for calculating H-measure. http://www2.imperial.ac.uk/~djhand/. (31 January 2011, date last accessed).
23. Adams NM, Hand DJ. (1999) Comparing classifiers when the misallocation costs are uncertain. *Pattern Recog* 1999; **32**(7):1139–47.
24. Drummond C, Holte RC. Cost curves: An improved method for visualizing classifier performance. *Mach Learn* 2006;**65**:95–130.
25. Lobo JM, Jiménez-Valverde A, Real R. AUC: a misleading measure of the performance of predictive distribution models. *Global Ecol Biogeogr* 2008;**17**:145–51.
26. Hilden J, Glasziou P. Regret graphs, diagnostic uncertainty and Youden's index. *Stat Med* 1996;**15**:969–86.
27. McClish DK. Analyzing a portion of the ROC curve. *Med Decis Making* 1989;**9**(3):190–5.
28. Vanderlooy S, Hüllermeier E. A critical analysis of variants of AUC. *Mach Learn* 2008;**72**:247–62.
29. R Development Core Team. R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria, 2009. http://www.R-project.org.
30. Ransohoff DF. Bias as a threat to the validity of cancer molecular-marker research. *Nat Rev Cancer* 2005;**5**(2): 142–9.
31. Baggerly KA, Coombes KR. Deriving chemosensitivity from cell lines: forensic bioinformatics and reproducible research in high-throughput biology. *Ann Appl Stat* 2009; **3**(4):1309–34.
32. Potti A, Dressman HK, Bild A, *et al*. Genomic signatures to guide the use of chemotherapeutics. *Nat Med* 2006;**12**(11): 1294–300.
33. Hsu DS, Balakumaran BS, Acharya CR, *et al*. Pharmacogenomic strategies provide a rational approach to the treatment of cisplatin-resistant patients with advanced cancer. *J Clin Oncol* 2007;**25**(28):4350–7.
34. Dressman HK, Berchuck A, Chan G, *et al*. An integrated genomic-based approach to individualized treatment of patients with advanced-stage ovarian cancer. *J Clin Oncol* 2007;**25**(5):517–25.
35. Baggerly KA, Coombes KR, Neeley ES. Run batch effects potentially compromise the usefulness of genomic signatures for ovarian cancer. *J Clin Oncol* 2008;**25**:1186–7.
36. Bonnefoi H, Potti A, Delorenzi M, *et al*. Validation of gene signatures that predict the response of breast cancer to neoadjuvant chemotherapy: a substudy of the EORTC 10994/BIG 00-01 clinical trial. *Lancet Oncol* 2007;**8**(12): 1071–8.
37. Clark RD, Webster-Clark DJ. Managing bias in ROC curves. *J Comput Aided Mol Des* 2008;**22**:141–6.
38. Søreide K, Korner H, Søreide JA. Diagnostic accuracy and receiver-operating characteristics curve analysis in surgical research and decision making. *Ann Surg* 2011;**253**(1): 27–34.
39. Hess KR, Anderson K, Symmans WF, *et al*. Pharmacogenomic predictor of sensitivity to preoperative chemotherapy with paclitaxel and fluorouracil, doxorubicin, and cyclophosphamide in breast cancer. *J Clin Oncol* 2006;**24**(26):4236–44.
40. Swamidass SJ, Azencott CA, Daily K, Baldi P. A CROC stronger than ROC: measuring, visualizing and optimizing early retrieval. *Bioinformatics* 2010;**26**(10):1348–56.
41. Holm S. A simple sequentially rejective multiple test procedure. *Scand J Stat* 1979;**6**:65–70.
42. Manly KF, Nettleton D, Hwang JT. Genomics, prior probability, and statistical tests of multiple hypotheses. *Genome Res* 2004;**14**(6):997–1001.
43. George SL. Statistical issues in translational cancer research. *Clin Cancer Res* 2008;**14**(19):5954–8.
44. Holland B. On the application of three modified Bonferroni procedures to pairwise multiple comparisons in balanced repeated measures designs. *Comp Stat Quarterly* 1991;**6**: 219–31.
45. Rothman J. Writing for Epidemiology. *Epidemiology* 1998; **9**(3):333–7.
46. Rothman J. A show of confidence. *N Engl J Med* 1978;**299**: 1362–3.
47. Gardner MJ, Altman DG. Confidence intervals rather than P values: estimation rather than hypothesis testing. *Brit Med J* 1986;**292**:746–50.
48. Poole C. Low *p*-values or narrow confidence intervals: which are more durable? *Epidemiology* 2001;**12**(3): 291–4.

49. Hanley JA, McNeil BJ. The meaning and use of the area under a receiver operating characteristic (ROC) curve. *Radiology* 1984;**143**(1):29–36.
50. Cortes C, Mohri M. Confidence intervals for the area under the ROC curve. *Adv Neural Inf Process Syst* 2005;**17**:305–12.
51. Efron B. Better bootstrap confidence intervals. *J Am Stat Assoc* 1987;**82**:171–200.
52. Salter KH, Acharya CR, Walters KS, *et al*. An integrated approach to the prediction of chemotherapeutic response in patients with breast cancer. *PLoS ONE* 2008;**3**(4):e1908.
53. Karlsson E, Delle U, Danielsson A, *et al*. Gene expression variation to predict 10-year survival in lymph-node-negative breast cancer. *BMC Cancer* 2008;**8**:254.
54. Garman KS, Acharya CR, Edelman E, *et al*. A genomic approach to colon cancer risk stratification yields biologic insights into therapeutic opportunities. *Proc Natl Acad Sci USA* 2008;**105**(49):19432–7.
55. Hand DJ. Classifier technology and the illusion of progress. *Stat Sci* 2006;**21**(1):1–15.
56. R Bioconductor project. http://www.bioconductor.org (31 January 2011, date last accessed).
57. Sing T, Sander O, Beerenwinkel N, Lengauer T. ROCR: visualizing classifier performance in R. *Bioinformatics* 2005; **21**(20):3940–1.
58. Tuszynski J. colAUC {caTools}, R library. http://cran.r-project.org/web/packages/caTools/(6 March 2011, date last accessed).
59. Simpson G. plot.roc {analogue}, R library. http://cran.r-project.org/web/packages/analogue/(6 March 2011, date last accessed).
60. Hall M, Frank E, Holmes G, *et al*. The WEKA data mining software: an update. *SIGKDD Expl* 2009;**11**(1):10–8.
61. Vergara IA, Norambuena T, Ferrada E, *et al*. StAR: a simple tool for the statistical comparison of ROC curves. *BMC Bioinformatics* 2008;**9**:265.
62. GraphPad Prism. http://www.graphpad.com/prism/ (6 March 2011, date last accessed).
63. Hilden J. The area under the ROC curve and its competitors. *Med Decis Making* 1991;**11**(2):95–101.
64. Hand DJ. Measuring classifier performance: a coherent alternative to the area under the ROC curve. *Mach Learn* 2009;**77**:103–23.
65. Simon R. Supervised analysis when the number of candidate features (*p*) greatly exceeds the number of cases (*n*). *ACM SIGKDD Expl Newsletter* 2003;**5**(2):31–7.
66. Hanczar B, Hua J, Sima C, *et al*. Small-sample precision of ROC-related estimates. *Bioinformatics* 2010;**26**(6):822–30.
67. Hilden J. What properties should an overall measure of test performance possess? *Clin Chem* 2005;**51**(2):471.
68. Flach P. The many faces of ROC analysis in machine learning. *Tutorial Notes of the 21st International Conference on Machine Learning, 2004 (ICML04)* 2004. http://www.cs.bris.ac.uk/~flach/ICML04tutorial/index.html (31 January 2011, date last accessed).