Apples-to-Apples in Cross-Validation Studies: Pitfalls in Classifier Performance Measurement

George Forman
Hewlett-Packard Labs
1501 Page Mill Rd.
Palo Alto, CA 94304
ghforman@hpl.hp.com

Martin Scholz
Hewlett-Packard Labs
1501 Page Mill Rd.
Palo Alto, CA 94304
scholz@hp.com

ABSTRACT

Cross-validation is a mainstay for measuring performance and progress in machine learning. There are subtle differences in how exactly to compute accuracy, F-measure and Area Under the ROC Curve (AUC) in cross-validation studies. However, these details are not discussed in the literature, and incompatible methods are used by various papers and software packages. This leads to inconsistency across the research literature. Anomalies in performance calculations for particular folds and situations go undiscovered when they are buried in aggregated results over many folds and datasets, without ever a person looking at the intermediate performance measurements. This research note clarifies and illustrates the differences, and it provides guidance for how best to measure classification performance under cross-validation. In particular, there are several divergent methods used for computing F-measure, which is often recommended as a performance measure under class imbalance, e.g., for text classification domains and in one-vs.-all reductions of datasets having many classes. We show by experiment that all but one of these computation methods leads to biased measurements, especially under high class imbalance. This paper is of particular interest to those designing machine learning software libraries and researchers focused on high class imbalance.

1. INTRODUCTION

The field of machine learning has benefited from having a few standard performance metrics by which to judge our progress on benchmark classification datasets, such as the Reuters text dataset [4]. Many papers in the published literature have referenced each other's performance numbers in order to establish that a new method is an improvement or at least competitive with existing published methods. The importance of being able to cite others' performance figures increases over time. As methods and software systems become increasingly complex, it is more difficult for each researcher to meticulously reproduce each others' methods as baselines against which to compare one's own experiments. But the correctness of citing another's performance breaks down if the performance measures we use are incomparable. This clearly happens when one paper reports only F-measure and another reports only the Area Under the ROC Curve (AUC). But more insidiously, it can also catch us unawares when, say, the F-measure was measured in an *incompatible* way, or the AUC in one paper was measured in a way that inadvertently demands a consistently calibrated classifier as well.

F-measure and AUC are well-defined, mainstream performance metrics whose definitions can be found everywhere. Likewise, many publications describe the widely accepted practice of *cross-validation* for assessing and comparing the quality of classification schemes on a given labeled dataset.

But ironically, there is ambiguity and disagreement about how exactly to compute F-measure and AUC across the folds of a cross-validation study.[1] This was first brought to our attention by the number of questions we get from other researchers on how exactly to go about measuring these under cross-validation. Upon further investigation, we could not find the matter addressed in the literature. We informally surveyed dozens of articles and found that there is not just a little disagreement on the matter. Not only do different papers use different methods for computing F-measure or AUC, but most do not bother to specify how exactly they computed it under cross-validation—perhaps not realizing that there are choices. Heretofore it has not been highlighted in the literature, and certainly not illustrated why some common choices can lead to biased results. One of the anonymous reviewers of this article shared in their review that last year they had to deal with two instances of this problem, which caused experimental results to be positively biased. Finally, we have observed inconsistent and biased strategies available from different software libraries, as well as students' research software. Detecting such subtle inconsistencies is relatively difficult compared with bugs that make themselves known by halting execution.

Not only are the methods of computation different, but it also turns out that there can be significant disagreement in their outputs under some test conditions. This paper enumerates the different methods of calculation (Section 2), works through examples to illustrate that the differences can be large (Section 3), and demonstrates that a particular choice for computing F-measure is superior in terms of bias and variance (Section 4).

The method of calculation is particularly important when dealing with class imbalance. A dataset is imbalanced when the classes are not equally represented, i.e., the class of

[1]We invite the reader before proceeding to write down exactly how he or she typically computes these measures under cross-validation, for comparison with the discussion later.

interest is rare, which is a common situation in text datasets and is of growing research interest generally. High class imbalance also occurs when datasets having many classes are factored into a large number of one-vs.-all (OVA) subtasks.

2. PERFORMANCE MEASURES UNDER CROSS-VALIDATION

In this section we define and distinguish the different methods of calculating the performance scores. Given a labeled dataset and a classification algorithm, the question at hand is how to measure how well the classifier performs on the dataset.

2.1 Formal Notation Preliminaries

Let $\mathcal{X}$ denote our instance space, i.e., a set that covers all instances expressible in our representation. We assume a fixed but unknown distribution D underlying $\mathcal{X}$ that determines the probability or density to sample a specific example $x \in \mathcal{X}$. Each x is associated with a label from a finite set $\mathcal{Y}$.

A *hard classifier* is a function $c : \mathcal{X} \to \mathcal{Y}$. A *learning algorithm* is an algorithm that outputs a classifier c after reading a sequence $(x_1, y_1), \ldots, (x_t, y_t)$ of t labeled training examples, where each $x_i \in \mathcal{X}$ is an example from the instance space, and $y_i \in \mathcal{Y}$ the corresponding label of x_i.

We will refer to the sequence of examples as the *training set* and make the assumption that each labeled example in that set was sampled i.i.d. from D. The overall goal is to find learning algorithms that are likely to output classifiers with "good" behavior with respect to the same unknown underlying distribution D. As one important example, we might want a classifier c to have high *accuracy*, $P_{(x,y)\sim D}(c(x) = y)$.

In practice, we clearly have to rely on test sets to assess the performance of a classifier c with respect to D. A hold-out set or *test set* T sampled i.i.d. from the same D allows one to compute an *estimate* of various performance metrics. In this case, it is clearly desirable to use a method that gives unbiased and low variance estimates of the unknown ground truth performance value over the entire space D. Such estimates are based on counts. We focus on binary (hard) classification, where $\mathcal{Y}$ consists only of a "positive" and a "negative" label. Each classifier c segments the test set into four partitions, based on both the true label y_i and the predicted label $c(x_i)$ for each example $(x_i, y_i) \in T$. We will refer to the absolute number of true positives as TP, false positives as FP, false negatives as FN, and true negatives as TN. The test set accuracy is $(\mathrm{TP} + \mathrm{TN})/(\mathrm{TP} + \mathrm{TN} + \mathrm{FP} + \mathrm{FN})$, for example. We explicitly refer to the "ground truth accuracy" where we mean $P_{(x,y)\sim D}(c(x) = y)$ instead.

The predominant tool for computing estimates of learning algorithm performances is *k-fold cross-validation* (often 10-fold). It divides the available training data T into k disjoint subsets $T^{(1)}, \ldots, T^{(k)}$ of equal size. Each of the $T^{(i)}$ sets is used as a test set and is evaluated against a classifier trained from all the other data $T \setminus T^{(i)}$. Thus, we can get k different test set performances. Often we report the average of those as the overall estimate for the classifier on that dataset. This process aims to compute estimates that are close to the ground truth performance when running the learning algorithm on the complete set T. But we shall show in the following section that there is a problem with reporting the average F-measure in this way.

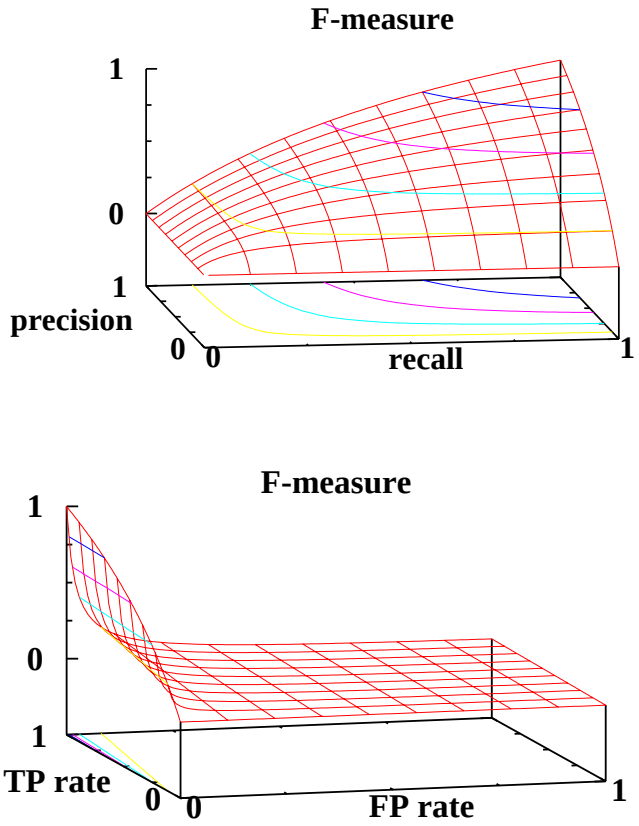

Figure 1: F-measure as a function of (a) precision and recall, or (b) true positive rate and false positive rate—shown assuming 1% positives.

We will use superscripts in this paper to refer to values that belong to specific cross-validation folds. For example, the number of true positives of fold i would be referred to as $\mathrm{TP}^{(i)}$, the precision of fold j as $\mathrm{Pr}^{(j)}$.

An option to the cross-validation approach discussed above is *stratified* cross-validation. The only difference is that it takes care that each subset $T^{(i)}$ contains the same number of examples from each class (± 1). This is common practice in the machine learning community, partly as a result of people using integrated learning toolboxes like WEKA [3] or RapidMiner [6] that provide stratification by default in cross-validation experiments. The main advantage of this procedure is that it reduces the experimental variance, which makes it easier to identify the best of the methods under consideration.

2.2 F-measure without Cross-Validation

While error rate or accuracy dominate much of the classification literature, F-measure is the most popular metric in the text classification and information retrieval communities. The reason is that typical text mining corpora have many classes and suffer from high class imbalance. Accuracy tends to undervalue how well classifiers are doing on smaller classes, whereas F-measure balances *precision* and *recall*.

Definition 1. The precision Pr and the recall Re of a classifier with TP true positive, FP false positives, and FN false negatives are

$$\begin{aligned} \mathrm{Pr} &:= \mathrm{TP}/(\mathrm{TP}+\mathrm{FP}) \quad \text{and} \\ \mathrm{Re} &:= \mathrm{TP}/(\mathrm{TP}+\mathrm{FN}) \end{aligned}$$

F-measure combines these two into a single number, which is useful for ranking or comparing methods. It can be thought of as an 'and' function: if either precision or recall are poor, then the resulting F-measure will be poor, shown graphically in Figure 1a. Formally, F-measure is the harmonic mean between precision and recall.

Definition 2. The F-measure of a classifier with precision Pr and recall Re is defined as

$$\mathrm{F} := 2 \cdot \frac{\mathrm{Pr} \cdot \mathrm{Re}}{\mathrm{Pr}+\mathrm{Re}} \tag{1}$$

Many research papers and software libraries simplify the definition of F-measure as follows:

$$\begin{aligned} \mathrm{F} &= 2 \cdot \frac{\mathrm{Pr} \cdot \mathrm{Re}}{\mathrm{Pr}+\mathrm{Re}} \\ &= 2 \cdot \frac{\left(\frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FP}}\right) \cdot \left(\frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FN}}\right)}{\left(\frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FP}}\right) + \left(\frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FN}}\right)} \\ &= (2 \cdot \mathrm{TP}) \,/\, (2 \cdot \mathrm{TP} + \mathrm{FP} + \mathrm{FN}) \end{aligned} \tag{2}$$

Thus, it computes F-measure in terms of the number of true positives and false positives. Figure 1b shows this view using the false positive rate and true positive rate on the x- and y-axes. The graph shown assumes 1% positives, resulting in the sharpness of the surface; when negatives abound, any substantial false positive rate will result in low precision and, therefore, low F-measure.

Exceptions: Equation (1) is undefined in some situations. Precision is undefined if the classifier makes no positive predictions, $\mathrm{TP}+\mathrm{FP}=0$. This happens sometimes with small or class-imbalanced test sets, or with very conservative classifiers, such as those that learn in training to always vote for the majority class. Recall is undefined when there are no positives in the test set. This can happen in rare situations with highly imbalanced datasets if random sampling or unstratified cross-validation is used. Equation (2) smoothly extends the definition of F-measure to be well-defined (namely, zero) in most all situations. Even so, it still results in division-by-zero for the case that a particular test set has no positives ($\mathrm{TP}+\mathrm{FN}=0$) and the classifier agrees—that is, it makes no positive predictions ($\mathrm{TP}+\mathrm{FP}=0$). Test harness software that encounters any of these undefined situations above can do one of two things. It can substitute a zero for an otherwise undefined value, or, less commonly, it can leave out the occasional, troublesome test fold from the final computations. As we show later, these choices result in a negative or positive bias in the measurement of F-measure.

2.3 F-measure with Cross-Validation

In the previous two sections we separately discussed cross-validation and F-measure. Most researchers do not consider the combination of these two, the notion of *cross-validated F-measure*, to be ambiguous. In this section, we will give a description of three different combination strategies that are all actively used in the literature. Two of these allow for different ways of handling the undefined corner cases, so we end up with a total of five different aggregation strategies altogether. The number of strategies doubles to ten if we consider both unstratified and stratified cross-validation.

All subsequently discussed cases have in common that we train k classifiers, and that we evaluate the classifier $c^{(i)}$ (which we got in iteration i when training on $T \setminus T^{(i)}$) exclusively on the hold-out set $T^{(i)}$. The superscripted terms $\mathrm{TP}^{(i)}$ through $\mathrm{TN}^{(i)}$, $F^{(i)}$, $\mathrm{Pr}^{(i)}$, or $\mathrm{Re}^{(i)}$ refer to the test set performance of $c^{(i)}$ on $T^{(i)}$, as defined in Sections 2.1 and 2.2.

Using the precise notation and framework we have established, we are now in a position to define the three main ways that F-measure results are aggregated across the k folds of cross-validation.

1. We start with the case of simply averaging F-measure. In each fold, we record the F-measure $F^{(i)}$ and compute the final estimate as the mean of all folds:

$$F_{\mathbf{avg}} := \frac{1}{k} \cdot \sum_{i=1}^{k} F^{(i)}$$

2. Alternately, one can average precision and recall across the folds, using their final results to compute F-measure according to Equation 1:

$$\begin{aligned} \mathrm{Pr} &:= \frac{1}{k} \cdot \sum_{i=1}^{k} \mathrm{Pr}^{(i)} \\ \mathrm{Re} &:= \frac{1}{k} \cdot \sum_{i=1}^{k} \mathrm{Re}^{(i)} \\ F_{\mathbf{pr},\mathbf{re}} &:= 2 \cdot \frac{\mathrm{Pr} \cdot \mathrm{Re}}{\mathrm{Pr}+\mathrm{Re}} \end{aligned}$$

3. Instead, one can total the number of true positives and false positives over the folds, then compute F-measure according to either Equations 1 or 2:

$$\begin{aligned} \mathrm{TP} &:= \sum_{i=1}^{k} \mathrm{TP}^{(i)} \\ \mathrm{FP} &:= \sum_{i=1}^{k} \mathrm{FP}^{(i)} \\ \mathrm{FN} &:= \sum_{i=1}^{k} \mathrm{FN}^{(i)} \\ F_{\mathbf{tp},\mathbf{fp}} &:= (2 \cdot \mathrm{TP}) \,/\, (2 \cdot \mathrm{TP} + \mathrm{FP} + \mathrm{FN}) \end{aligned}$$

Exceptions: As discussed above, in some folds we might encounter the problem of undefined precision or recall. Let $V^{(i)} := 1$ if $\mathrm{Pr}^{(i)}$ and $\mathrm{Re}^{(i)}$ are *both* defined, and $V^{(i)} := 0$, otherwise. Precision will be undefined whenever a classifier $c^{(i)}$ does not predict any of the test examples in fold $T^{(i)}$ as positive. Recall can be undefined only if a fold does not contain any positives. This cannot happen with stratified cross-validation, unless the number of folds exceeds the number of positives, and it is considered rare for unstratified cross-validation.

One strategy for overcoming this problem is to substitute zero based on a reformulation of F-measure; see Equa-

tion (2). We will use this as the default interpretation throughout the paper, so $F^{(i)} := 0$ when $V^{(i)} = 0$.

An alternative is to declare any folds having undefined precision and recall as being invalid measurements and simply skip them. The folly of such a choice will be exposed in a later section. This might happen as an unintended consequence of the software throwing an exception. We will add a tilde to $F_{\mathbf{avg}}$ or $F_{\mathbf{pr,re}}$ whenever we refer to this latter computation. For example, the definition above then becomes

$$\tilde{F}_{\mathbf{avg}} := \frac{1}{\sum_{i=1}^{k} V^{(i)}} \cdot \sum_{i=1}^{k} F^{(i)}$$

2.4 Error Rate, Accuracy, and AUC

Accuracy and error rate do not have an equivalent problem under cross-validation: you get the same result whether you compute accuracy on each fold and then average, or if you tally the error count and then compute the accuracy rate just once at the end. Thus, the problem has not been a concern for many learning papers that have historically measured performance based only on error-rate or accuracy.

By contrast, AUC under cross-validation can be computed in two incompatible ways. The first is to sort the individual scores from all folds together into a single ROC curve and then compute the area of this curve, which we call $AUC_{\mathbf{merge}}$. The other is to compute the AUC for each fold separately and then average over the folds:

$$AUC_{\mathbf{avg}} := \frac{1}{k} \cdot \sum_{i=1}^{k} AUC^{(i)}$$

The problem with $AUC_{\mathbf{merge}}$ is that by sorting different folds together, it assumes that the classifier should produce well-calibrated probability estimates. Usually a researcher interested in measuring the quality of the probability estimates will use Brier score or such. By contrast, researchers who measure performance based on AUC typically are unconcerned with calibration or specific threshold values, being only concerned with the classifier's ability to rank positives ahead of negatives. So, $AUC_{\mathbf{merge}}$ adds a usually unintended requirement on the study: it will downgrade classifiers that rank well if they have poor calibration across folds, as we illustrate in Section 3.2.

WEKA [3] as of version 3.6.1 uses the $AUC_{\mathbf{merge}}$ strategy in its Explorer GUI and in its `Evaluation` core class for cross-validation, but uses $AUC_{\mathbf{avg}}$ in its Experimenter interface.

Exceptions: Although traditionally not a problem, if there were any fold containing no positives, it would be impossible to compute AUC for that fold. Under stratified cross-validation, this can never be a problem. But without stratification—such as in a multi-label setting—and with great imbalance for some of the classes, this problem could arise. In this situation, some software libraries may fail altogether, others may silently substitute a zero or skip such folds.

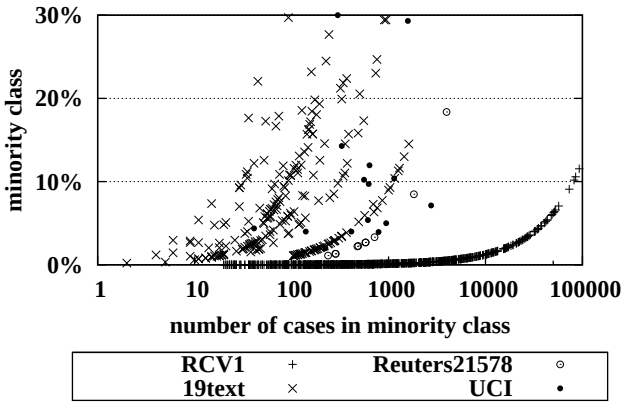

Figure 2: Class imbalance and minority class size for a variety of binary classification tasks in the literature [1,2,5,6].

3. ILLUSTRATION

Here we provide specific examples of cross-validation results that show wide disparity in performance, depending on the method of calculation. We begin with F-measure and follow with AUC. We use only four folds in order to simplify the exposition and reduce visual clutter; however, the disparity among the methods can be even more pronounced with normal 10-fold cross-validation or with higher numbers of folds. We use *stratified* cross-validation, although more extreme results could be demonstrated for unstratified situations where recall may sometimes be undefined. We chose examples that avoid all corner cases to be more convincing potentially (later we shall come back to the matter). The performance statistics are the actual results of a linear SVM (WEKA[3] SMO implementation with options -M -N 2 for Platt scaling) on binary text classification tasks drawn originally from Reuters (dataset re0 in [1]).

The examples here are demonstrated using highly imbalanced tasks in order to emphasize the disparity. The degree of imbalance we consider (1% positives and 2.5%) is not uncommon in text studies or in research that focuses on imbalance. Figure 2 shows the imbalance and the number of examples of the minority (positive) class for a set of binary tasks drawn from the old Reuters benchmark [4], the new Reuters RCV1 benchmark [5], 19 multiclass text datasets [1], and a collection of UCI and other datasets used in imbalance research [7].

3.1 F-measure

Table 1 shows the detailed numbers for each fold of a stratified cross-validation on a task having 1% positives out of 1504 data rows. This degree of class imbalance is considered challenging, especially for the small number of positives. Nonetheless, such small classes do appear among text and UCI benchmarks, and our purpose here is simply to illustrate a real example where the methods differ substantially.

In the table, we see the classifier made a relatively large number of false positive errors on the last two folds, leading to poor precision for those folds. Whenever precision or recall is low, then F-measure will also be low for those folds. Averaging the four per-fold F-measures, we get

Table 1: Example 4-fold stratified cross-validation shows F-measure can differ widely depending on how it is computed.

Fold	Negatives	Positives	TP	FP		Precision	Recall	F-measure	
1	373	3	3	0		100%	100%	100%	
2	372	4	4	1		80%	100%	89%	
3	372	4	4	13		24%	100%	38%	
4	372	4	3	5		38%	75%	50%	
Totals:	1489	15	14	19	Averages:	60%	94%	**69%**	$F_{\mathbf{avg}}$
			58%	$F_{\mathbf{tp,fp}}$		**73%**	$F_{\mathbf{pr,re}}$		

Table 2: A second example where the F-measure calculation methods disagree because the classifier predicted no positives on the second fold. Precision here(†) is set to zero to avoid division by zero; the metrics with a tilde instead skip this fold.

Fold	Negatives	Positives	TP	FP		Precision	Recall	F-measure	
1	372	4	2	0		100%	50%	67%	
2	372	4	0	0		0%†	0%	0%	
3	372	4	4	0		100%	100%	100%	
4	372	4	4	0		100%	100%	100%	
Totals:	1488	16	10	0	Averages:	75%	63%	**67%**	$F_{\mathbf{avg}}$
			77%	$F_{\mathbf{tp,fp}}$		**68%**	$F_{\mathbf{pr,re}}$	**89%**	$\tilde{F}_{\mathbf{avg}}$
						91%	$\tilde{F}_{\mathbf{pr,re}}$		

69% $F_{\mathbf{avg}}$. But if we instead average the precision and recall columns, then any especially low precision or recall value is smoothed over, rather than accentuated. Thus, even with the very poor 24% precision on one fold, the average precision and average recall are moderate, yielding 73% $F_{\mathbf{pr,re}} = 2 \times \frac{0.60 \times 0.94}{0.60 + 0.94}$, which is significantly higher than $F_{\mathbf{avg}}$. Finally, if we tally up the true positives and false positives across the folds (at lower left) and then compute F-measure from these, we get 58% $F_{\mathbf{tp,fp}} = \frac{2 \times 14}{2 \times 14 + 19 + 1}$, which is much lower than $F_{\mathbf{avg}}$. This illustrates that the difference can be large: $F_{\mathbf{pr,re}} = 1.26 \times F_{\mathbf{tp,fp}}$. In Section 4 we characterize the bias and variance of each, showing which is actually the better estimator.

For a different class (not shown) having exactly 4 positives in each of the four folds (1% positive), we found the classifier happened to make no positive predictions for one of the folds. This led to an undefined precision and penalized the classifier with zero F-measure for that fold, although generally the classifier performed well on the other folds. Finally there is the option to skip any folds that lead to undefined precision. These variants are marked with a tilde. Naturally, they assign better scores for having effectively removed a difficult fold from the test set. This naturally leads to a strong positive bias in the scoring function: $\tilde{F}_{\mathbf{pr,re}} = 1.34 \times F_{\mathbf{pr,re}}$.

3.2 AUC

Next we turn to the Area Under the ROC Curve. The primary issue in this case is that the soft score outputs from each of the fold classifiers are not necessarily calibrated with one another. For example, we conducted 4-fold stratified cross-validation of the same dataset for a different class dichotomy having 38 positives (2.5%). The AUC scores for each fold were 96%, 91%, 94% and 87%, which yield an average of 92% $AUC_{\mathbf{avg}}$. But these four classifiers were not calibrated with each other, as we illustrate in Figure 3. The left graph shows the false positive rate vs. the classifier score threshold and the right graph shows the same for true positive rate. Notably, only two of the folds happen to align; two other curves are greatly shifted horizontally. Thus, when the soft scores of all four folds are sorted together to form one ROC curve, its overall score is only 80% $AUC_{\mathbf{merge}}$. Unless the classifier is calibrated to output probabilities rather than just scores with some threshold, it is not meaningful to compare the scores from different folds. Note that this also applies for ranking metrics such as *Precision at 20* and *Mean Average Precision*; such metrics need to be computed separately for each fold and then averaged. If, on the other hand, the classifiers are *intended* to be calibrated and one wishes to penalize methods that produce inferior calibration, then one may sort all soft classifier outputs together and then compute the metric. Our purpose here is, again, simply to illustrate a substantial difference.

4. F-MEASURE BIAS AND VARIANCE

Here we address the following questions:

- Why do we expect cross-validated F-measure results to be biased?
- Do the different methods for estimating F-measure introduce different kinds of biases?
- Which method introduces the lowest bias in absolute terms and has the lowest variance?
- How do bias and variance change under class imbalance and changing target F-measures?

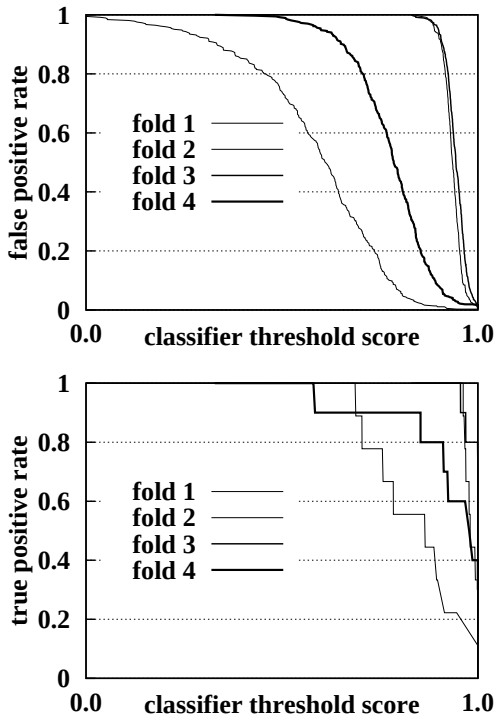

Figure 3: (a) Classifier false positive rate vs. output score. (b) true positive rate vs. output score.

4.1 Why We Expect Biased Results

Before stepping into the details, we want to discuss why F-measure is prone to biased estimates.

To this end, let us first study the behavior of accuracy. Accuracy tends to be "naturally" unbiased, because it can be expressed in terms of a binomial distribution: A "success" in the underlying Bernoulli trial would be defined as sampling an example for which a classifier under consideration makes the right prediction. By definition, the success probability is identical to the accuracy of the classifier. The i.i.d. assumption implies that each example of the test set is sampled independently, so the expected fraction of correctly classified samples is identical to the probability of seeing a success above. Averaging over multiple folds is identical to increasing the number of repetitions of the Binomial trial. This does not affect the posterior distribution of accuracy if the test sets are of equal size, or if we weight each estimate by the size of each test set.

In contrast, F-measure has the drawback that it cannot be broken down into F-measures of arbitrary example subsets. Referring to Equation (2), it can easily be seen that the impact of an individually sampled example on the overall estimate depends on which other examples are already part of the test set. This prohibits an exact computation of global F-measure in terms of the F-measures of each fold of a cross-validation. Having random variables in the denominator adds complexity, basically a form of "context dependencies." The averaged result will usually change whenever we swap examples between the test sets of folds, even when assuming we get the exact same classifier for all folds. Equation (2) illustrates that F-measure is concave in the number of true positives TP, and steepest near $TP = 0$. Especially under class imbalance, missing even a single true positive (compared to expectation based on the ground truth contingency table) might reduce the F-measure of a cross-validation fold substantially. In contrast, including an extra true positive has a much lower impact, so the overall bias is negative. Clearly, this is an unpleasant property under cross-validation.

Quantifying the bias for the methods considered in this paper analytically is a hard problem. Running simulations is comparably simple, and offers equally valuable insights into the problem.

4.2 Details of the Simulation

We repeatedly simulated 10-fold cross-validation over a dataset with 1000 cases: 900 training and 100 testing for each fold. The performance of the binary classifier was simulated such that it had controlled ground-truth F-measure, with its precision exactly equal to its recall. Thus, we can postulate a classifier with 80% F-measure that exhibits 80% precision and 80% recall in ground-truth. For generating our simulated test set results, we first allocate the positives and negatives to the folds, either stratified or randomly for unstratified. Then within each fold we sample from the binomial distribution to determine the number of its positives that become true positives and the number of its negatives that become false positives. There is no expensive learning step required. By repeating the simulation a million times, we were able to determine the distribution of scores generated for each of the five methods of computing F-measure. This experiment methodology simplifies matters for two reasons. First, it gives us a notion of ground truth, as we know the correct outcome beforehand (the ground-truth F-measure). We clearly want a validation method that reports the ground truth with no bias or very little bias as well as low variance. Second, under the i.i.d. assumption and given the "ground truth" contingency table of our classifiers, we can assess the bias and variance of each method.

In our simulations, we evaluated scenarios with 1% to 25% of the cases being positive. Since there are only 1000 cases, at 1% there are just 10 positives in the dataset. This extreme case is intentional in order to bring out the exceptional behavior when no positives are predicted in some folds occasionally. Clearly most researchers would avoid drawing any conclusions with so few positives in their dataset. But there are two major exceptions. First, in the medical domain, conclusions about classifiers are often drawn on datasets having very few cases; for example, the heavily studied Leukemia dataset by Golub et al. [2] has just 74 examples divided unevenly in four classes. Second, some machine learning research that focuses on learning under class imbalance draws conclusions from studies on many different datasets or classification tasks having a small number of positives each. It is hoped that when aggregated over many imbalanced tasks, the superior classifiers will become known. In order for these conclusions to be accurate and comparable across the literature, it would be important to measure F-measure correctly even under what some might call extreme situations. And, of course, when writing software we cannot control all the test situations to which it may later be put.

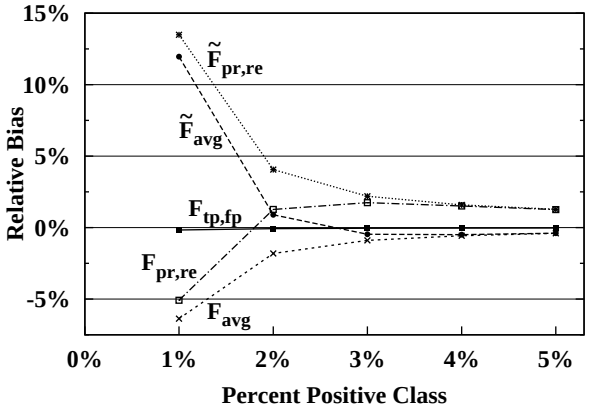

Figure 4: Bias under stratified 10-fold cross-validation.

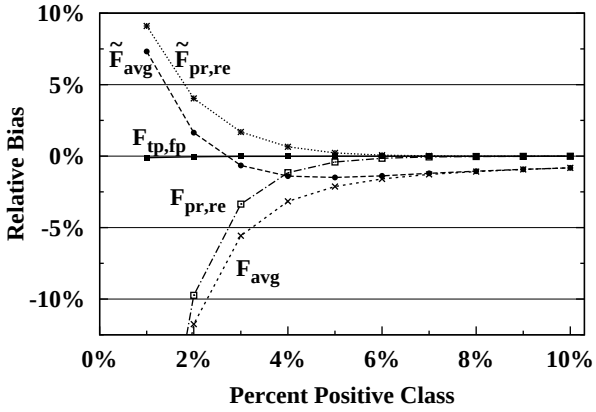

Figure 5: Bias under *unstratified* 10-fold cross-validation.

4.3 Simulation Results

Figure 5 shows the relative bias of each method under 10-fold stratified cross-validation with a classifier having exactly 80% F-measure in ground-truth. Only one method is almost perfectly unbiased, $F_{\mathbf{tp,fp}}$, and therefore it is the recommended way to compute F-measure. This is the fundamental result of this analysis. We go on to offer intuition for the biases of the other methods. The x-axis varies the class prior from 1% to 5% positives in order to illustrate different effects. As we move to the left, a greater proportion of test folds have undefined precision: the two methods that in these situations substitute zero (the minimum possible F-measure) have a negative bias, $F_{\mathbf{avg}}$ and $F_{\mathbf{pr,re}}$; whereas the two methods that instead skip such folds have a positive bias, $\tilde{F}_{\mathbf{avg}}$ and $\tilde{F}_{\mathbf{pr,re}}$. Recall that substituting zeros is not an arbitrary decision: The function converges to 0 as we approach any point that has an undefined precision or recall. So 0 is the correct value here, and the negative bias might be a bit surprising at first. The reason for this lies in the concave shape of the F-measure function, see Section 4.1.

As we move to the right, folds with undefined precision occur less often, and so the distinction disappears between like pairs of lines. At the right, the $F_{\mathbf{pr,re}}$ method has a relative bias $>+1\%$, and the $F_{\mathbf{avg}}$ method has a smaller negative bias. Why? Since F-measure operates like an *and*-function between precision and recall, any fold having by random variation especially low precision *or* low recall will receive a low $F^{(i)}$ score. Given 10-folds, there are ten chances to get an especially low $F^{(i)}$ score by chance, bringing $F_{\mathbf{avg}}$ down on average; in contrast, averaging the precision and recall over the ten folds generally results in less extreme values from which their harmonic mean $F_{\mathbf{pr,re}}$ is computed. Thus, $F_{\mathbf{pr,re}}$ is far less likely to have an especially low precision or recall score, and it shows a substantial positive bias.

Next we examine how the bias depends on the ground-truth F-measure, which we vary from 60% to 95%. The three panels in Figure 6 show the results of 10-fold stratified cross-validation for datasets having 1%, 5%, and 25% positives. For each dataset, as the ground-truth F-measure declines, the bias of each method generally becomes more extreme. Figure 7 shows the same for *unstratified* 10-fold cross-validation. The y-axis is held the same, except for the leftmost dataset where the range of bias is greatly increased (note its y-axis). Without stratification, undefined precision and, rarely, undefined recall can affect the measurements, as described previously. Already with the 5% positive dataset we see the zero-substitution methods $F_{\mathbf{avg}}$ and $F_{\mathbf{pr,re}}$ have substantial negative bias. (In the rightmost graph with 25% positives, $F_{\mathbf{pr,re}}$ and $\tilde{F}_{\mathbf{pr,re}}$ are not visible as they are overlaid atop $F_{\mathbf{tp,fp}}$.) To cover all these situations, $F_{\mathbf{tp,fp}}$ is clearly the preferred method.

Finally we want to discuss the bias of $F_{\mathbf{tp,fp}}$. The same argument of F-measure being concave applies here, and explains a (very small) negative bias. We repeatedly sample from a ground truth contingency table (our simulation) and then average the biases. Underestimating the fraction of true positives has a higher impact than overestimating it, especially near 0. The main difference between $F_{\mathbf{tp,fp}}$ and the methods that average cross-validation folds is that the former avoids the highly non-linear regions of the F-measure functions near 0 by considering aggregates. This reduced the bias by two orders of magnitude in our experiments.

Having analyzed the bias, we now turn to variance. Figure 8 shows the standard deviation relative to the ground-truth F-measure. At 5% positives and more we see that $F_{\mathbf{tp,fp}}$ shows least variance. Although it does not always show the least variance at 1%, the other methods here are unacceptably biased.

5. DISCUSSION AND CONCLUSIONS

The upshot of the empirical analysis is that (a) $F_{\mathbf{tp,fp}}$ is the by far most unbiased method and should be used for computing F-measure, and (b) this distinction becomes important for greater degrees of class imbalance as well as for less accurate classifiers. The $F_{\mathbf{avg}}$ method, which is in common use, penalizes methods that may occasionally predict zero positives for some test folds. This causes an unintentional and undesired bias in some research literature to prefer methods that err on the side of producing more false positives. This is naturally of greater concern for researchers who are focused on studying class imbalance. But it should also be of concern to software programmers, whose software may someday be used in class imbalanced situations, and to researchers studying large numbers of

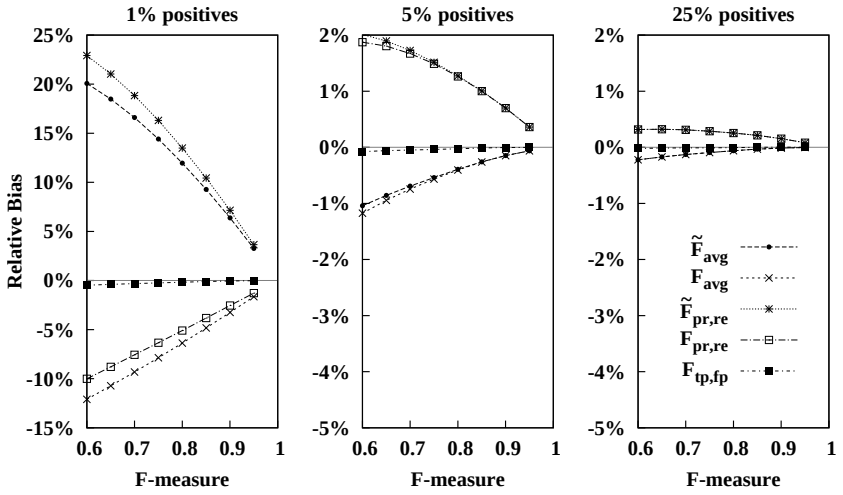

Figure 6: Relative bias under stratified 10-fold cross-validation.

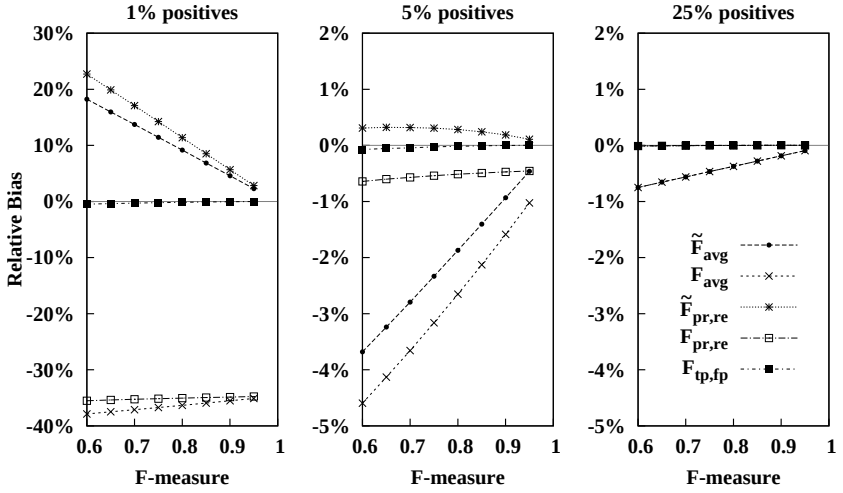

Figure 7: Relative bias under *unstratified* 10-fold cross-validation.

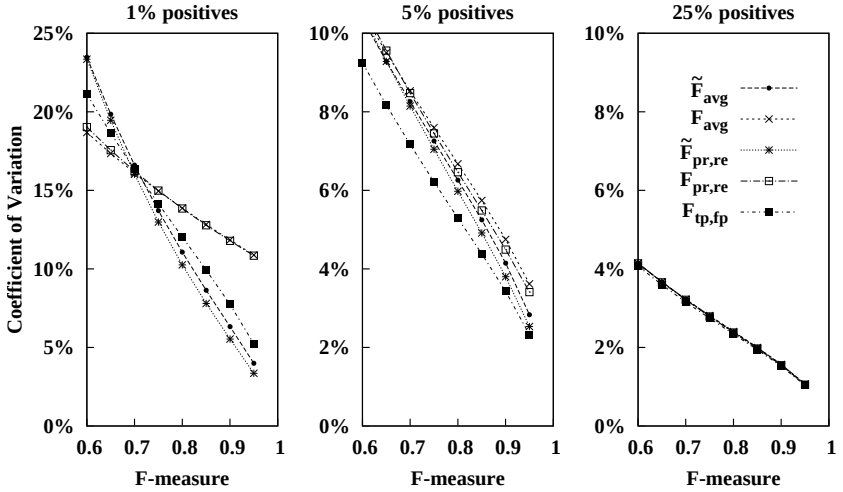

Figure 8: Relative Standard Deviation under stratified 10-fold cross-validation.

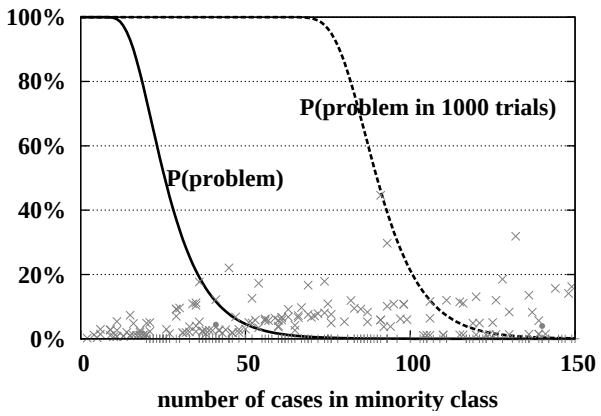

Figure 9: The probability of having at least one fold with no positives in 10-fold unstratified cross-validation, which results in undefined recall. The second curve shows this probability increasing given many independent trials: testing many different classes, many datasets to study, or random splits of the same dataset.

datasets in aggregate without careful scrutiny, especially datasets with many classes or multi-label settings.

Normally the stratification option is used to reduce experimental variance, but in some studies it is omitted. Without stratification, we run some risk of having zero positives in one or more of the folds, leading to undefined recall and undefined AUC. This risk grows greatly if there is a small number of positives available in the dataset. Figure 9 shows the probability of this problem occurring for 10-fold unstratified cross-validation, varying the number of positives available. The grey data points reflect the actual number of positives available for some of the binary classification tasks shown previously in Figure 2. Given that every research effort deals with many repeated trials, and/or multiple classes being studied within each dataset, and/or multiple datasets, the right-hand curve shows the probability that the problem occurs in 1000 independent trials. The point is that when studying datasets that have, say, less than 100 examples for some class, it is fairly probable that some of unstratified experiments will encounter some folds with no positives to test. This leaves AUC and possibly F-measure undefined.

Now, the straightforward answer is simply to always use stratification to avoid this potential problem. But stratification can only be used for single-label datasets. In multi-label settings it is infeasible to ensure that each and every class is (equally) represented in every fold. Thus, the risk of encountering undefined recall and AUC values is mainly a concern for multi-label settings—an area of growing research interest.

In conclusion, we urge the research community to consistently use $F_{\mathbf{tp,fp}}$ and $AUC_{\mathbf{avg}}$. Be careful when using software frameworks; as useful as they are for getting experiments done efficiently and consistently, they can also hide imporant details—details that matter in some situations. For example, as of version 3.6.1, WEKA's `Explorer` GUI produces $F_{\mathbf{tp,fp}}$ and $AUC_{\mathbf{merge}}$ by default, whereas its `Experimenter` produces $F_{\mathbf{avg}}$ and $AUC_{\mathbf{avg}}$—as do many other software frameworks. In order to obtain both $F_{\mathbf{tp,fp}}$ and $AUC_{\mathbf{avg}}$, one needs to be explicit in one's programming. To obtain $F_{\mathbf{tp,fp}}$, use a single call to `Evaluation.fMeasure()` on a confusion matrix that has been loaded with all cross-validation folds. But to obtain $AUC_{\mathbf{avg}}$, call `Evaluation.areaUnderROC()` separately for each fold and then average.

To keep things in perspective, there are a variety of known pitfalls that are more frequently a problem than the subtle issues of computation raised in this paper: using only a single, often weakly chosen, baseline method; not making sure the baselines have reasonable options and tuning; and unintentionally leaking information from the test set, sometimes as a result of *twinning* in datasets containing near duplicate cases in training and testing. Altogether, our diverse research community needs to continue to make progress and generally adopt best practices for high quality machine learning research.

6. REFERENCES

[1] G. Forman. BNS feature scaling: an improved representation over TF-IDF for SVM text classification. In *Proceedings of the 17th ACM Conference on Information and Knowledge Management (CIKM)*, pages 263–270, New York, NY, 2008. ACM.

[2] T. Golub, D. Slonim, P. Tamayo, C. Huard, M. Caasenbeek, J. Mesirov, H. Coller, M. Loh, J. Downing, M. Caligiuri, C. Bloomfield, and E. Lander. Molecular classification of cancer: class discovery and class prediction by gene expression monitoring. *Science*, 286:531–537, 1999.

[3] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten. The WEKA data mining software: An update. *SIGKDD Explorations*, 11(1), 2009.

[4] D. D. Lewis and M. Ringuette. A comparison of two learning algorithms for text categorization. In *Symposium on Document Analysis and Information Retrieval*, pages 81–93, Las Vegas, NV, Apr. 1994. ISRI; Univ. of Nevada, Las Vegas.

[5] D. D. Lewis, Y. Yang, T. Rose, and F. Li. RCV1: A new benchmark collection for text categorization research. volume 5, pages 361–397, 2004. http://www.jmlr.org/papers/volume5/lewis04a/lewis04a.pdf.

[6] I. Mierswa, M. Wurst, R. Klinkenberg, M. Scholz, and T. Euler. Yale: Rapid prototyping for complex data mining tasks. In L. Ungar, M. Craven, D. Gunopulos, and T. Eliassi-Rad, editors, *KDD '06: Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 935–940, New York, NY, USA, August 2006. ACM.

[7] T. Raeder, G. Forman, and N. V. Chawla. *Data Mining: Foundations and Intelligent Paradigms*, chapter Learning with Imbalanced Data: Evaluation Matters. Intelligent Systems Reference Library. Springer Verlag, 2010.