ROC and AUC with a Binary Predictor: a Potentially Misleading Metric

Abstract

In analysis of binary outcomes, the receiver operator characteristic (ROC) curve is heavily used to show the performance of a model or algorithm. The ROC curve is informative about the performance over a series of thresholds and can be summarized by the area under the curve (AUC), a single number. When a **predictor** is categorical, the ROC curve has only as many thresholds as the one less than number of categories; when the predictor is binary there is only one threshold. As the AUC may be used in decision-making processes on determining the best model, it important to discuss how it agrees with the intuition from the ROC curve. We discuss how the interpolation of the curve between thresholds with binary predictors can largely change the AUC. Overall, we believe a linear interpolation from the ROC curve with binary predictors, which is most commonly done in software, corresponding to the estimated AUC. We believe these ROC curves and AUC can lead to misleading results. We compare R, Python, Stata, and SAS software implementations.

Keywords: ROC, AUC, area under the curve

1 Introduction

In many applications, receiver operator characteristic (ROC) curves are used to show how a predictor compares to the true outcome. One of the large advantages of ROC analysis is that it is threshold-agnostic; performance of a predictor is estimated without a specific threshold and also gives a criteria to choose an optimal threshold based on a certain cost function or objective. Typically, an ROC analysis shows how sensitivity (true positive rate) changes with varying specificity (true negative rate or $1-$false positive rate). Analyses also typically weigh false positives and false negatives equally. In ROC analyses, the predictive capabilities of a variable is commonly summarized by the area under the curve (AUC), which can be found by integrating areas under the line segments. We will discuss how interpolation between these line segments affect the visualization of the ROC curve and corresponding AUC. Additionally, partial ROC (pROC) analysis keeps a specificity fixed and can summarize a predictor by the partial AUC (pAUC) or the optimal sensitivity at that fixed false positive rate.

Many predictors, especially medical tests, result in a binary decision; a value is higher than a pre-determined threshold or a substance is present. Similarly, some predictors are commonly collected as categorical or discrete such as low, normal, or high blood pressure while others are categorical by nature such as having a specific gene or not. These are useful indicators of presence a disease, which is a primary outcome of interest in medical settings, and are used heavily in analysis.

If one assumes the binary predictor is generated from a continuous distribution that has been thresholded, then the sensitivity of this thresholded predictor actually represents one point on the ROC curve for the underlying continuous value. Therefore the ROC curve of a binary predictor is not really appropriate, but should be represented by a single point on the curve. But alas, ROC and AUC analysis is done on binary predictors and used to inform if one variable is more predictive than the other (E et al. 2018; TV et al. 2017; Glaveckaite et al. 2011; Blumberg et al. 2016; Budwega et al. 2016; Mwipatayi et al. 2016; Xiong et al. 2018, Shterev et al. (2018); Kushnir et al. 2018; Snarr et al. 2017; Veltri et al. 2018). For example, these cases show that researchers use ROC curves and AUC to evaluate predictors, even when the predictors are categorical or binary. Although

there is nothing inherently wrong with this comparison, it can lead to drastically different predictors being selected based on these criteria if ties are treated slightly different ways. A more appropriate comparison of a continuous predictor and the binary predictor may be to compare the sensitivity and specificity (or overall accuracy) of the continuous predictor given the optimal threshold versus that of the binary predictor.

As categorical/binary predictors only have a relatively small number of categories, how ties are handled are distinctly relevant. Thus, many observations may have the same value/risk score. Fawcett (2006) describes the standard way of how ties are handled in a predictor: a probability of $\frac{1}{2}$ is given for the cases when the predictors are tied. When drawing the ROC curve, one can assume that all the ties do not correctly classify the outcome (Fawcett called the "pessimistic" approach) or that all the ties do correctly classify the outcome (called the "optimistic" approach), see Fig. 6 in (Fawcett 2006). But Fawcett notes (emphasis in original):

> Any mixed ordering of the instances will give a different set of step segments within the rectangle formed by these two extremes. However, the ROC curve should represent the *expected* performance of the classifier, which, lacking any other information, is the average of the pessimistic and optimistic segments.

This "expected" performance directly applies to the assignment of a half probability of success when the data are tied, which is implied by the "trapezoidal rule" from Hanley & McNeil (1982). Fawcett (2006) also states in the calculation of AUC that "trapezoids are used rather than rectangles in order to average the effect between points". This trapezoidal rule applies additional areas to the AUC based on ties of the predictor, giving a half a probability. This addition of half probability is linked to how ties are treated in the Wilcoxon rank sum test. As much of the theory of ROC curve testing, and therefore testing of differences in AUC, is based on the theory of the Wilcoxon rank-sum test, this treatment of ties is also relevant to statistical inference and not only AUC estimation.

Others have discussed insights into binary predictors in addition to Fawcett (2006), but they are mentioned in small sections of the paper (Saito & Rehmsmeier 2015, Pepe et al. 2009). Other information regarding ties and binary data are blog posts or working papers such as `http://blog.revolutionanalytics.com/2016/11/calculating-auc.html`

or `https://www.epeter-stats.de/roc-curves-and-ties/`, which was written by the author of the **fbroc** (Peter 2016) package, which we will discuss below. Most notably, Hsu & Lieli (2014) is an extensive discussion of ties, but the paper was not published.

Although many discuss the properties of ROC and AUC analyses, we wish to first show the math and calculations of the AUC with a binary predictor and we then explore commonly-used statistical software for ROC curve creation and AUC calculation in a variety of packages and languages. Overall, we believe that AUC calculations alone may be misleading for binary or categorical predictors depending on the definition of the AUC. We propose to be explicit when reporting the AUC in terms of the approach to ties.

2 Mathematical Proof of AUC for Single Binary Predictor

First, we will show how the AUC is defined in terms of probability. This representation is helpful in discussing the connection between the stated interpretation of the AUC, the formal definition and calculation used in software, and how the treatment of ties is crucial when the data are discrete. Let us assume we have a binary predictor X and a binary outcome Y, such that X and Y only take the values 0 and 1, the number of replicates is not relevant here. Let X_i be the values of $X|Y = i$, where $i \in \{0, 1\}$.

Fawcett (2006) goes on to state:

> AUC of a classifier is equivalent to the probability that the classifier will rank a randomly chosen positive instance higher than a randomly chosen negative instance.

In other words, we could discern the definition $\text{AUC} = P(X_1 > X_0)$. Note, the definition here adds no value when the classifier is tied, this is a strict inequality. As there are only two outcomes for X, we can expand this probability using the law of total probability:

$$
\begin{aligned}
P(X_1 > X_0) &= P(X_1 > X_0 | X_1 = 1) P(X_1 = 1) \\
&\quad + P(X_1 > X_0 | X_1 = 0) P(X_1 = 0) \qquad (1) \\
&= P(X_1 > X_0 | X_1 = 1) P(X_1 = 1) \qquad (2)
\end{aligned}
$$

as $P(X_1 > X_0 | X_1 = 0) = 0$ because X_0 and X_1 are in $\{0, 1\}$. We see that $P(X_1 = 1)$ in equation (2) is the sensitivity:

$$
\begin{aligned}
P(X_1 = 1) &= P(X = 1 | Y = 1) \\
&= \frac{TP}{TP + FN} \\
&= \text{sensitivity}
\end{aligned}
$$

and that $P(X_1 > X_0 | X_1 = 1)$ in equation (2) is the specificity:

$$
\begin{aligned}
P(X_1 > X_0 | X_1 = 1) &= P(X_1 > X_0 | X_1 = 1, X_0 = 1) P(X_0 = 1) \\
&\quad + P(X_1 > X_0 | X_1 = 1, X_0 = 0) P(X_0 = 0) \\
&= P(X_1 > X_0 | X_1 = 1, X_0 = 0) P(X_0 = 0) \\
&= P(X_0 = 0) \\
&= P(X = 0 | Y = 0) \\
&= \frac{TN}{TN + FP} \\
&= \text{specificity}
\end{aligned}
$$

We combine these two to show that equation (2) reduces to:

$$
P(X_1 > X_0) = \text{specificity} \times \text{sensitivity}
$$

Thus, using the definition as $P(X_1 > X_0)$, the AUC of a binary predictor is simply the sensitivity times the specificity.

Let us change the definition of AUC slightly while accounting for ties, which we call $\text{AUC}_{\text{w/ties}}$, to:

$$\text{AUC}_{\text{w/ties}} = P(X_1 > X_0) + \frac{1}{2}P(X_1 = X_0)$$

which corresponds to the common definition of AUC. This AUC is the one reported by most software, as we will see below.

2.1 Simple Concrete Example

To give some intuition of this scenario, we will assume X and Y have the following joint distribution, where X is along the rows and Y is along the columns, as in Table 1.

Table 1: A simple 2x2 table of a binary predictor (rows) versus a binary outcome (columns)

	0	1
0	52	35
1	32	50

Therefore, the AUC should be equal to $\frac{50}{85} \times \frac{52}{84}$, which equals 0.364. This estimated AUC will be reported throughout the majority of this paper, so note the value.

We will define this as the the strict definition of AUC, where ties are not taken into account and we are using strictly greater than in the probability and will call this value $\text{AUC}_{\text{definition}}$.

Note, if we reverse the labels, then the sensitivity and the specificity are estimated by 1 minus that measure, or $\frac{35}{85} \times \frac{32}{84}$, which is equal to 0.157. Thus, as this AUC is less than the original labeling, we would choose that with the original labeling.

If we used the calculation for $\text{AUC}_{\text{w/ties}}$ we see that we estimate AUC by $\text{AUC}_{\text{definition}} + \frac{1}{2}\left(\frac{50+52}{169}\right)$, which is equal to 0.604. We will show that most software report this AUC estimate.

2.1.1 Monte Carlo Estimation of AUC

We can also show that if we use simple Monte Carlo sampling, we can randomly choose X_0 and X_1. From these samples, we can estimate these AUC based on the definitions above.

Here, the function `est.auc` samples 10^6 random samples from X_1 and X_0, determines which is greater, or if they are tied, and then calculates $\widehat{\text{AUC}}_{\text{definition}}$ and $\widehat{\text{AUC}}_{\text{w/ties}}$:

```
R> est.auc = function(x, y, n = 1000000) {
R+   x1 = x[y == 1] # x | y = 1
R+   x0 = x[y == 0] # x | y = 0
R+   c1 = sample(x1, size = n, replace = TRUE)
R+   c0 = sample(x0, size = n, replace = TRUE)
R+   auc.defn = mean(c1 > c0) # strictly greater
R+   auc.wties = auc.defn + 1/2 * mean(c1 == c0) # half for ties
R+   return(c(auc.definition = auc.defn,
R+             auc.wties = auc.wties))
R+ }
R> sample.estauc = est.auc(x, y)
R> sample.estauc

auc.definition      auc.wties
      0.364232       0.604130
```

And thus we see these simulations agree with the values estimated above, with negligible Monte Carlo error.

2.1.2 Geometric Argument of AUC

We will present a geometric discussion of the ROC as well. In Figure 1, we show the ROC curve for the simple concrete example. In panel A, we show the point of sensitivity/specificity connected by the step function, and the associated AUC is represented in the shaded blue area, representing $\text{AUC}_{\text{definition}}$. In panel B, we show the additional shaded areas that are due to ties in orange and red; all shaded areas represent $\text{AUC}_{\text{w/ties}}$. We will show how to calculate these areas from $P(X_1 = X_0)$ in $\text{AUC}_{\text{w/ties}}$ such that:

$$
\begin{aligned}
P(X_1 = X_0) &= P(X_1 = 1, X_0 = 1) + P(X_1 = 0, X_0 = 0) \\
&= P(X_1 = 1)P(X_0 = 1) + P(X_1 = 0)P(X_0 = 0) \\
&= (\text{sensitivity} \times (1 - \text{specificity})) + ((1 - \text{sensitivity}) \times \text{specificity})
\end{aligned}
$$

so that combining this with (2) we have:

$$
\begin{aligned}
\text{AUC}_{\text{w/ties}} &= \text{specificity} \times \text{sensitivity} \\
&\quad + \frac{1}{2}(\text{sensitivity} \times (1 - \text{specificity})) \\
&\quad + \frac{1}{2}((1 - \text{sensitivity}) \times \text{specificity})
\end{aligned}
$$

Thus, we can see that geometrically from Figure 1:

$$\text{AUC}_{\text{w/ties}} =$$ 

where the order of the addition is the same respectively.

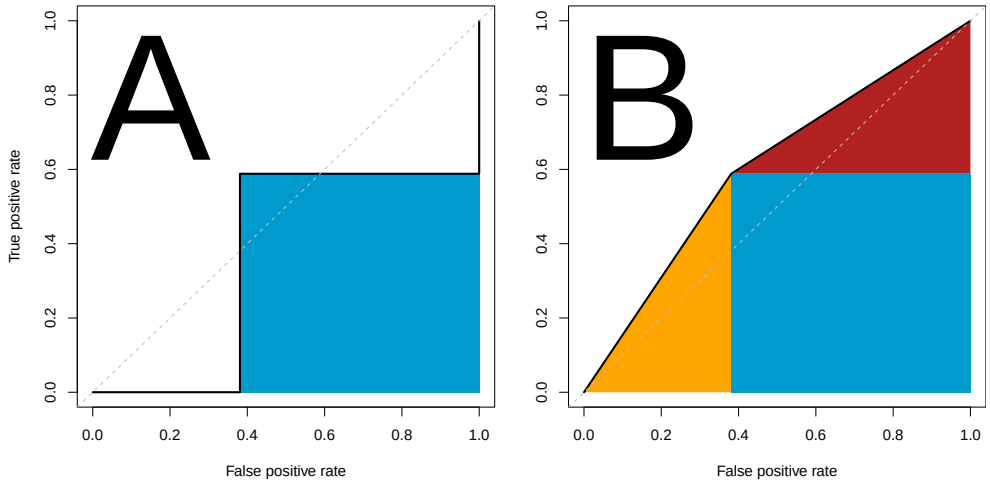


Figure 1: ROC curve of the data in the simple concrete example. Here we present a standard ROC curve, with the false positive rate or 1 − specificity on the x-axis and true positive rate or sensitivity on the y-axis. The dotted line represents the identity. The shaded area in panel represents the AUC for the strict definition. The additional shaded areas on panel B represent the AUC when accounting for ties.

2.2 AUC Calculation in Statistical Software

To determine how these calculations are done in practice, we will explore the estimated ROC curve and AUC from the implementations in the following `R` (R Core Team 2018) packages: **ROCR** (Sing et al. 2005), **caTools** (Tuszynski 2018), **pROC** (Robin et al. 2011), and **fbroc** (Peter 2016). We will also show these agree with the Python implementation in `sklearn.metrics` from **scikit-learn** (Pedregosa et al. 2011), the Stata functions `roctab` and `rocreg` (Bamber 1975, DeLong et al. 1988), and the SAS software functions `proc logistic` with `roc` and `roccontrast` . We note that the majority of these functions all count half the probability of ties, but differences exist in the calculation of confidence intervals of AUC and note some inconsistent behavior.

3 AUC Calculation: Current Implementations

This section will present code and results from commonly-used implementations of AUC estimation from R, Python, Stata, and SAS software. We will note agreement with the definitions of AUC above and any discrepancies. This section is not to be exhaustive, but give examples how to calculate AUC in these software and show that these definitions are consistently used in AUC analysis, primarily $\widehat{\text{AUC}}_{\text{w/ties}}$.

3.1 R

Here we will show the AUC calculation from the common `R` packages for ROC analysis. We will show that each report the value calculated in $\text{AUC}_{\text{w/ties}}$. The **caTools** (Tuszynski 2018) package calculates AUC using the `colAUC` function, taking in predictions as `x` and the binary ground truth labels as `y`:

```
R> library(caTools)
R> colAUC(x, y)

             [,1]
0 vs. 1 0.6036415
```

which reports $AUC_{w/ties}$.

In **ROCR** package (Sing et al. 2005), one must create a `prediction` object with the `prediction` function, which can calculate a series of measures. AUC is calculated from a `performance` function, giving a `performance` object, and giving the `"auc"` measure. We can then extract the AUC as follows:

```
R> library(ROCR)
R> pred = prediction(x, y)
R> auc.est = performance(pred, "auc")
R> auc.est@y.values[[1]]

[1] 0.6036415
```

which reports $AUC_{w/ties}$. We see this agrees with the plot from `ROCR` in Figure 2D.

The **pROC** (Robin et al. 2011) package calculates AUC using the `roc` function:

```
R> library(pROC)
R> pROC.roc = pROC::roc(predictor = x, response = y)
R> pROC.roc[["auc"]]

Area under the curve: 0.6036
```

which reports $AUC_{w/ties}$ and agrees with the plot from `pROC` in Figure 2E.

The **fbroc** package calculates the ROC using the `fbroc::boot.roc` and `fbroc::perf` functions. The package has 2 strategies for dealing with ties, which we will create 2 different objects `fbroc.default`, using the default strategy (strategy 2), and alternative strategy (strategy 1, `fbroc.alternative`):

```
R> library(fbroc)
R> fbroc.default = boot.roc(x, as.logical(y),
R+                          n.boot = 1000, tie.strategy = 2)
R> auc.def = perf(fbroc.default, "auc")
R> auc.def[["Observed.Performance"]]
```

```
[1] 0.6036415
```

```
R> fbroc.alternative = boot.roc(x, as.logical(y),
R+                              n.boot = 1000, tie.strategy = 1)
R> auc.alt = perf(fbroc.alternative, "auc")
R> auc.alt[["Observed.Performance"]]
```

```
[1] 0.6036415
```

which both report $\text{AUC}_{\text{definition}}$, identical results to above and agrees with the plot from `fbroc` in Figure 2F, which is for strategy 2.

Although the output is the same, these strategies for ties are different for the plotting for the ROC curve, which we see in Figure 3. The standard error calculation for both strategies use the second strategy (Fawcett's "pessimistic" approach), which is described in a blog post (`https://www.epeter-stats.de/roc-curves-and-ties/`) and can be seen in the shaded areas of the panels. Thus, we see that using either tie strategy results in the same estimate of AUC ($\text{AUC}_{\text{w/ties}}$), but using tie strategy 1 results in a plot which would reflect an AUC of $\text{AUC}_{\text{definition}}$ (Figure 3A), which seem to disagree. This result is particularly concerning because the plot should agree with the interpretation of AUC.

3.2 Python

In Python, we will use the implementation in `sklearn.metrics` from **scikit-learn** (Pedregosa et al. 2011). We will use the `R` package `reticulate` (Allaire et al. 2018), which will provide an Python interface to `R`. Here we use the `roc_curve` and `auc` functions from **scikit-learn** and output the estimated AUC:

```
R> # Adapted from https://qiita.com/bmj0114/items/460424c110a8ce22d945
R> library(reticulate)
R> sk = import("sklearn.metrics")
R> py.roc.curve = sk$roc_curve(y_score = x, y_true = y)
R> names(py.roc.curve) = c("fpr", "tpr", "thresholds")
R> py.roc.auc = sk$auc(py.roc.curve$fpr, py.roc.curve$tpr)
R> py.roc.auc
```

```
[1] 0.6036415
```

which reports $\text{AUC}_{\text{w/ties}}$. Although we have not exhaustively shown Python reports $\text{AUC}_{\text{w/ties}}$, `scikit-learn` is one of the most popular Python modules for machine learning and analysis. We can use `matplotlib` (Hunter 2007) to plot FPR and TPR from the `py.roc.curve` object, which we see in Figure 2B, which uses a linear interpolation by default and agrees with $\text{AUC}_{\text{w/ties}}$.

3.3 SAS Software

In SAS software (version 9.4 for Unix) (SAS & Version 2017), let us assume we have a data set named `roc` loaded with the variables/columns of `x` and `y` as above. The following commands will produce the ROC curve in Figure 2C:

```
R>   proc logistic data=roc;
R+       model y(event='1') = x;
R+       roc; roccontrast;
R+       run;
```

The resulting output reports $\text{AUC}_{\text{w/ties}}$, along with a confidence interval. The calculations can be seen in the SAS User Guide (`https://support.sas.com/documentation/cdl/en/statug/63033/HTML/default/viewer.htm#statug_logistic_sect040.htm`), which includes the addition of the probability of ties.

3.4 Stata

In Stata (StatCorp, College Station, TX, version 13) (Stata 2013), let us assume we have a data set with the variables/columns of `x` and `y` as above.

The function `roctab` is one common way to calculate an AUC:

```
R> roctab x y
```

```
 . roctab x y
```

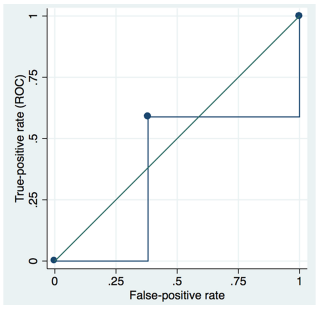
```
                    ROC                    -Asymptotic Normal--
          Obs       Area     Std. Err.      [95% Conf. Interval]
        -------------------------------------------------------
          169     0.6037       0.0379        0.52952     0.67793
```

which agrees with the calculation based on $\text{AUC}_{\text{w/ties}}$ and agrees with the estimates from above. One can also calculate the AUC using the `rocreg` function:

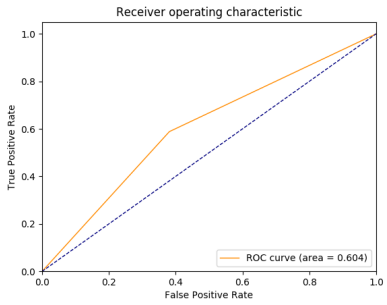
```
R> rocreg y x, nodots auc

 . rocreg y x, nodots auc

Bootstrap results                                 Number of obs      =       169
                                                  Replications       =      1000

Nonparametric ROC estimation

Control standardization: empirical
ROC method             : empirical

Area under the ROC curve

   Status    : y
   Classifier: x
------------------------------------------------------------------------------
             |    Observed               Bootstrap
         AUC |       Coef.       Bias    Std. Err.     [95% Conf. Interval]
-------------+----------------------------------------------------------------
             |    .3641457  -.0004513    .0451334     .2756857   .4526056  (N)
             |                                       .2771778    .452824  (P)
             |                                       .2769474   .4507576 (BC)
------------------------------------------------------------------------------
```

which agrees with the definition of $\text{AUC}_{\text{definition}}$ and is different from the output from `roctab`. The variance of the estimate is based on a bootstrap estimate, but the point estimate will remain the same regardless of using the bootstrap or not. This disagreement of estimates is concerning as the reported estimated AUC may be different depending on the command used in the estimation.

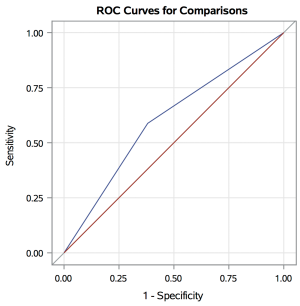
Using `rocregplot` after running this estimation, we see can create an ROC curve, which is shown in Figure 2A. We see that the estimated ROC curve coincides with the estimated AUC from `rocreg` ($\text{AUC}_{\text{definition}}$) and the blue rectangle in Figure 1. Thus, `roctab` is one of the most common ways in Stata to estimate AUC, but does not agree with the common way to plot ROC curves.



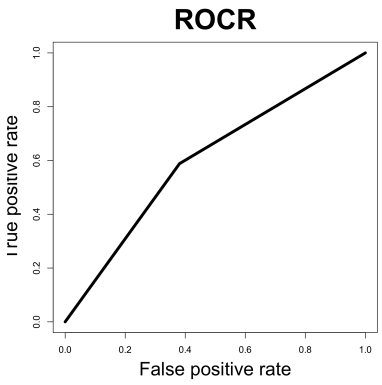
(A) Stata ROC Plot



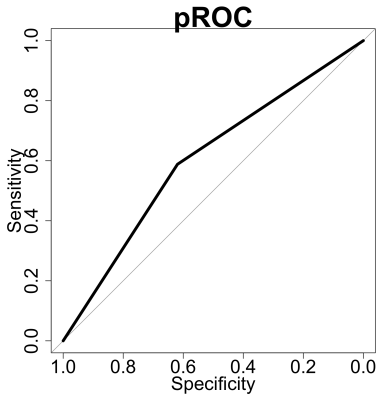
(B) Python ROC Plot from scikit-learn



(C) SAS ROC Plot



(D) ROCR ROC plot



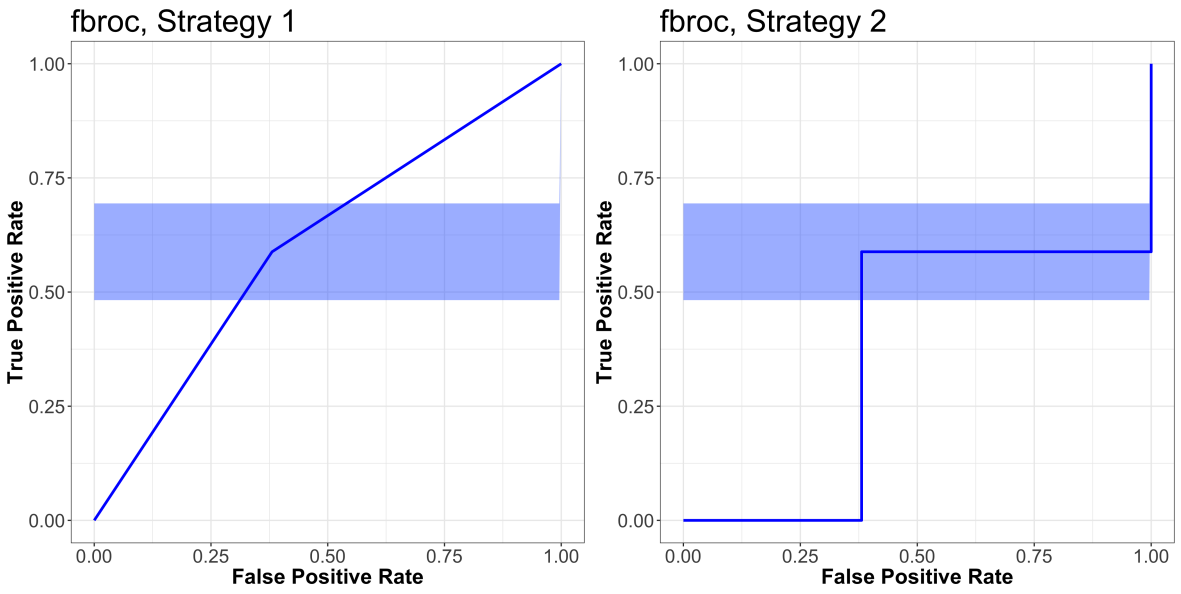
(E) pROC ROC plot

(F) fbroc Strategy 2 (default)

Figure 2: Comparison of different ROC curves for different `R` packages, `scikit-learn` from `Python`, `SAS`, and `Stata`. Each line represents the ROC curve, which corresponds to an according area under the curve (AUC). The blue shading represents the confidence interval for the ROC curve in the `fbroc` package. Also, each software represents the curve as the false positive rate versus the true positive rate, though the `pROC` package calls it sensitivity and specificity (with flipped axes). Some put the identity line where others do not. Overall the difference of note as to whether the ROC curve is represented by a step or a linear function. Using the first tie strategy for ties (non-default) in `fbroc` gives the same confidence interval but an ROC curve using linear interpolation.

Thus, we see in Figure 2 that all ROC curves are interpolated with a linear interpolation, which coincides with the calculation based on $\text{AUC}_{\text{w/ties}}$, except for the Stata ROC curve, which interpolates using a step function and coincides with $\text{AUC}_{\text{definition}}$. The confidence interval estimate of the ROC curve for `fbroc`, which is shaded in blue in Figure 2F,

corresponds to variability based on $\text{AUC}_{\text{definition}}$, though it shows the ROC curve based on $\text{AUC}_{\text{w/ties}}$.



(A) fbroc Strategy 1

(B) fbroc Strategy 2 (default)

Figure 3: Comparison of different strategies for ties in the `fbroc` package. The blue shading represents the confidence interval for the ROC curve. Overall the difference of note as to whether the ROC curve is represented by a step or a linear function. Using the first tie strategy for ties (non-default) in `fbroc` gives the same confidence interval as the second strategy but an ROC curve using linear interpolation, which may give an inconsistent combination of estimate and confidence interval.

Figure 3 shows that using the different tie strategies gives a linear (strategy 2, default, panel (B), duplicated) or step function/constant (strategy 1, panel (A)) interpolation. In each tie strategy, however, the AUC is estimated to be the same. Therefore, tie strategy 1 may give an inconsistent combination of AUC estimate and ROC representation.

4 Conclusion

We have shown how the ROC curve is plotted and AUC is estimated in common statistical software when using a unviariate binary predictor. There are inconsistencies across software

platforms, such as `R` and Stata, and even within some packages, such as `fbroc`. We believe these calculations do not reflect the discreteness of the data. We agree that using a binary predictor in an ROC analysis may not be appropriate, but we note that researchers and users will perform this. Therefore, we believe additional options for different calculations accounting for ties should be possible or warnings for discrete data may be presented to the user. Overall, we hope that indicating how ties are handled would become more common, especially for discrete data in practice.

References

Allaire, J., Ushey, K. & Tang, Y. (2018), *reticulate: Interface to 'Python'*. R package version 1.10.0.9001.
URL: *https://github.com/rstudio/reticulate*

Bamber, D. (1975), 'The area above the ordinal dominance graph and the area below the receiver operating characteristic graph', *Journal of mathematical psychology* **12**(4), 387–415.

Blumberg, D. M., De Moraes, C. G., Liebmann, J. M., Garg, R., Chen, C., Theventhiran, A. & Hood, D. C. (2016), 'Technology and the glaucoma suspect', *Investigative ophthalmology & visual science* **57**(9), OCT80–OCT85.

Budwega, J., Sprengerb, T., De Vere-Tyndalld, A., Hagenkordd, A., Stippichd, C. & Bergera, C. T. (2016), 'Factors associated with significant MRI findings in medical walk-in patients with acute headache', *Swiss Med Wkly* **146**, w14349.

DeLong, E. R., DeLong, D. M. & Clarke-Pearson, D. L. (1988), 'Comparing the areas under two or more correlated receiver operating characteristic curves: a nonparametric approach', *Biometrics* pp. 837–845.

E, M., C, M., K, S. & et al (2018), 'Diagnostic criteria of ulcerative pyoderma gangrenosum: A delphi consensus of international experts', *JAMA Dermatology* **154**(4), 461–466.
URL: *+ http://dx.doi.org/10.1001/jamadermatol.2017.5980*

Fawcett, T. (2006), 'An introduction to roc analysis', *Pattern recognition letters* **27**(8), 861–874.

Glaveckaite, S., Valeviciene, N., Palionis, D., Skorniakov, V., Celutkiene, J., Tamosiunas, A., Uzdavinys, G. & Laucevicius, A. (2011), 'Value of scar imaging and inotropic reserve combination for the prediction of segmental and global left ventricular functional recovery after revascularisation', *Journal of Cardiovascular Magnetic Resonance* **13**(1), 35.

Hanley, J. A. & McNeil, B. J. (1982), 'The meaning and use of the area under a receiver operating characteristic (ROC) curve.', *Radiology* **143**(1), 29–36.

Hsu, Y.-C. & Lieli, R. (2014), 'Inference for ROC curves based on estimated predictive indices: A note on testing AUC = 0.5', *Unpublished manuscript* .

Hunter, J. D. (2007), 'Matplotlib: A 2D graphics environment', *Computing In Science & Engineering* **9**(3), 90–95.

Kushnir, V. A., Darmon, S. K., Barad, D. H. & Gleicher, N. (2018), 'Degree of mosaicism in trophectoderm does not predict pregnancy potential: a corrected analysis of pregnancy outcomes following transfer of mosaic embryos', *Reproductive Biology and Endocrinology* **16**(1), 6.

Mwipatayi, B. P., Sharma, S., Daneshmand, A., Thomas, S. D., Vijayan, V., Altaf, N., Garbowski, M., Jackson, M., Benveniste, G., Denton, M. et al. (2016), 'Durability of the balloon-expandable covered versus bare-metal stents in the covered versus balloon expandable stent trial (COBEST) for the treatment of aortoiliac occlusive disease', *Journal of vascular surgery* **64**(1), 83–94.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M. & Duchesnay, E. (2011), 'Scikit-learn: Machine learning in Python', *Journal of Machine Learning Research* **12**, 2825–2830.

Pepe, M., Longton, G. & Janes, H. (2009), 'Estimation and comparison of receiver operating characteristic curves', *The Stata Journal* **9**(1), 1.

Peter, E. (2016), *fbroc: Fast Algorithms to Bootstrap Receiver Operating Characteristics Curves*. R package version 0.4.0.
URL: *https://CRAN.R-project.org/package=fbroc*

R Core Team (2018), *R: A Language and Environment for Statistical Computing*, R Foundation for Statistical Computing, Vienna, Austria.
URL: *https://www.R-project.org/*

Robin, X., Turck, N., Hainard, A., Tiberti, N., Lisacek, F., Sanchez, J.-C. & MÃijller, M. (2011), 'pROC: an open-source package for R and S+ to analyze and compare ROC curves', *BMC Bioinformatics* **12**, 77.

Saito, T. & Rehmsmeier, M. (2015), 'The precision-recall plot is more informative than the ROC plot when evaluating binary classifiers on imbalanced datasets', *PloS one* **10**(3), e0118432.

SAS, S. & Version, S. (2017), '9.4 [computer program]', *Cary, NC: SAS Institute* .

Shterev, I. D., Dunson, D. B., Chan, C. & Sempowski, G. D. (2018), 'Bayesian multi-plate high-throughput screening of compounds', *Scientific Reports* **8**(1), 9551.

Sing, T., Sander, O., Beerenwinkel, N. & Lengauer, T. (2005), 'ROCR: visualizing classifier performance in R', *Bioinformatics* **21**(20), 7881.
URL: *http://rocr.bioinf.mpi-sb.mpg.de*

Snarr, B. S., Liu, M. Y., Zuckerberg, J. C., Falkensammer, C. B., Nadaraj, S., Burstein, D., Ho, D., Gardner, M. A., Butto, A., Ewing, S. G. et al. (2017), 'The parasternal short-axis view improves diagnostic accuracy for inferior sinus venosus type of atrial septal defects by transthoracic echocardiography', *Journal of the American Society of Echocardiography* **30**(3), 209–215.

Stata, S. (2013), 'Release 13. statistical software', *StataCorp LP, College Station, TX* .

Tuszynski, J. (2018), *caTools: Tools: moving window statistics, GIF, Base64, ROC AUC, etc.* R package version 1.17.1.1.
URL: *https://CRAN.R-project.org/package=caTools*

TV, L., GH, B., JA, C., S, S., K, K. & GY, O. (2017), 'A revised approach for the detection of sight-threatening diabetic macular edema', *JAMA Ophthalmology* **135**(1), 62–68. **URL:** *+ http://dx.doi.org/10.1001/jamaophthalmol.2016.4772*

Veltri, D., Kamath, U. & Shehu, A. (2018), 'Deep learning improves antimicrobial peptide recognition', *Bioinformatics* **1**, 8.

Xiong, X., Li, Q., Yang, W.-S., Wei, X., Hu, X., Wang, X.-C., Zhu, D., Li, R., Cao, D. & Xie, P. (2018), 'Comparison of swirl sign and black hole sign in predicting early hematoma growth in patients with spontaneous intracerebral hemorrhage', *Medical science monitor: international medical journal of experimental and clinical research* **24**, 567.