



Dipartimento di Scienze Teoriche e Applicate
Università degli Studi dell'Insubria

Evaluating Binary Classifiers

what you should know about Performance Metrics
(and nobody told you)

Luigi Lavazza

Università degli Studi dell'Insubria, Varese, Italy

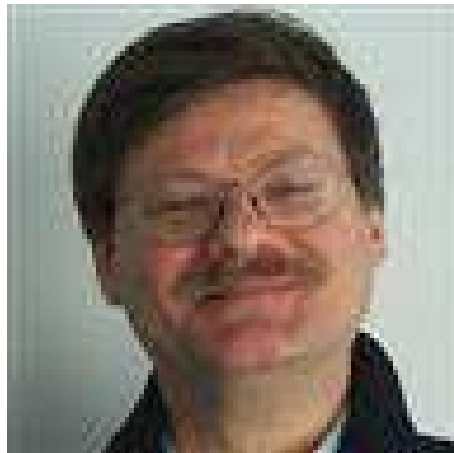
luigi.lavazza@uninsubria.it

The First International Conference on AI-based
Software Engineering for Digital Services
Nice, November 16 - 20, 2025



Digital Transformation
RESEARCH SOCIETY

The presenter



Professional experience

- Professor of Computer Science at the University of Insubria at Varese, Italy.
- Scientific consultant in digital innovation projects at CEFRIEL – Politecnico di Milano.

Scientific Activity

- Research: Empirical software engineering, software metrics and software quality evaluation; project management and effort estimation; Software process modeling, measurement and improvement; Open Source Software.
- Several international research projects
- Reviewer of EU funded projects.
- Co-author of over 200 scientific articles.
- PC member of several international Software Engineering conferences



Contents

- The problem
- Contingency tables (aka confusion matrices)
- Performance metrics
- The expressiveness of performance metrics
- The random classifier
- Properties of performance metrics
- Cost as a performance metric
- Effort-aware performance metrics
- Conclusions



THE PROBLEM

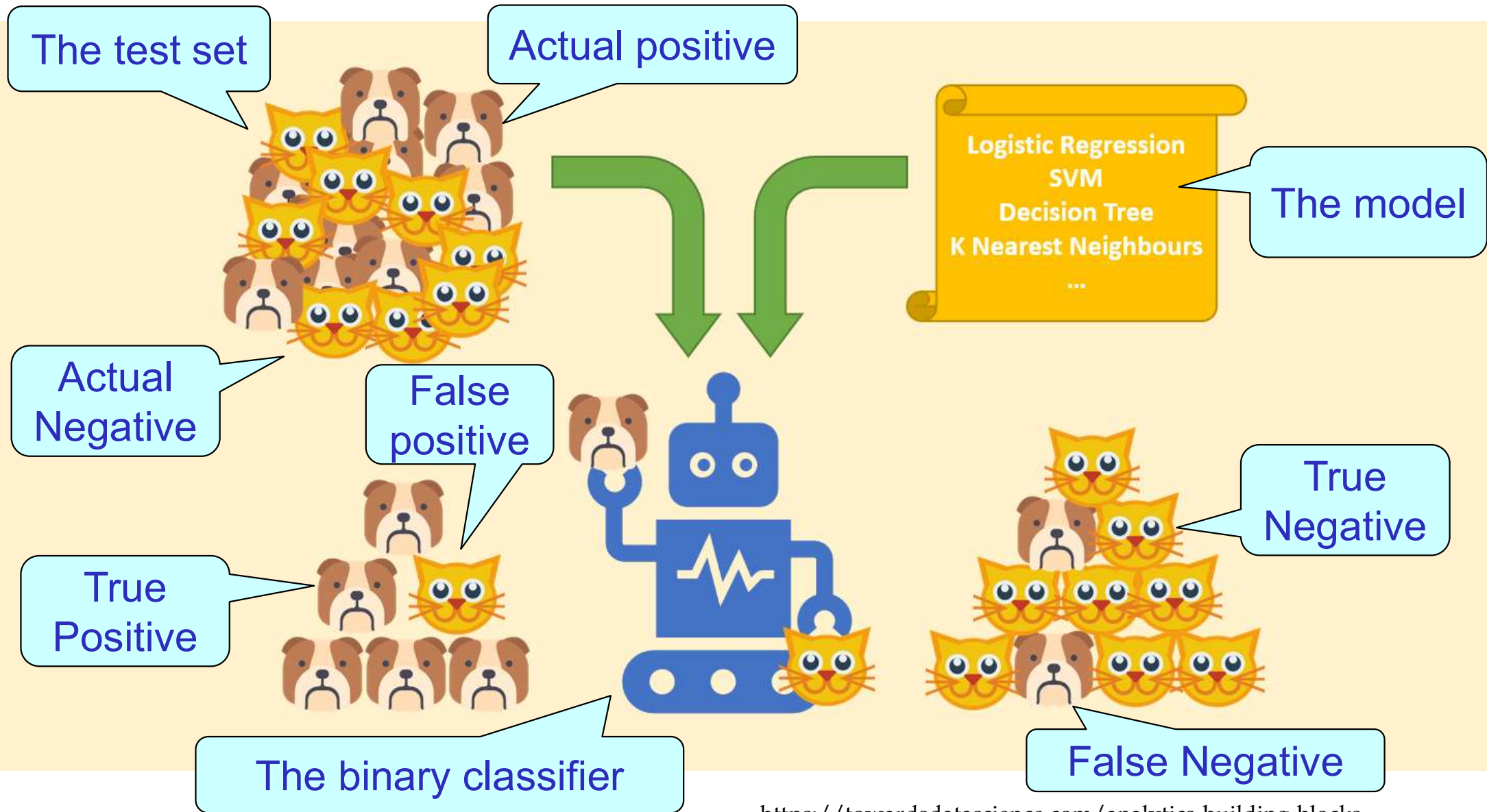


Binary classification

- Binary classification is the task of classifying the elements of a set into two groups on the basis of a classification rule.
 - ▶ Currently, classification is most often performed by AI models
- Typical binary classification problems include:
 - ▶ In software engineering:
 - predicting whether a given module is vulnerable;
 - predicting whether a given module is defective;
 - ▶ Medical diagnostic tests
 - determining if a patient has a certain disease or not;
 - ▶ In information retrieval
 - deciding whether a page should be in the result set of a search or not.
 - ▶ ...



Example: recognizing dogs (dog=positive, cat=negative)



<https://towardsdatascience.com/analytics-building-blocks-binary-classification-d205890314fc>
Himanshu Kulkarni



Classifiers are not perfect

- In general, the correct classification depends in a non-linear way from a huge number of factors, and possibly on chance.
- However, in practice
 - ▶ Not all factors are known
 - ▶ The relationships that links factors to outcomes is not completely and perfectly understood.
- The consequence is that some classifications are wrong
 - ▶ Almost always: exceptions are very rare.



Terminology

- True positive (TP)
 - ▶ An actually positive element is correctly classified positive
- True negative (TN)
 - ▶ An actually negative element is correctly classified negative
- False positive (FP)
 - ▶ An actually negative element is wrongly classified positive
 - ▶ That is, we have a false alarm. Aka Type I error
- False negative (FN)
 - ▶ An actually positive element is wrongly classified negative
 - ▶ We have a miss: we failed to recognize a really alarming situation. Aka Type II error



THE CONFUSION MATRIX



Confusion matrix

- Alias contingency table

		Actual		
		Negative	Positive	
Estimated	Negative	TN (True Negatives)	FN (False Negatives)	EN = TN + FN (Estimated Negatives)
	Positive	FP (False Positives)	TP (True Positives)	EP = FP + TP (Estimated Positives)
		AN = TN + FP (Actual Negatives)	AP = FN + TP (Actual Positives)	n = AN + AP = EN + EP

- The confusion matrix ***describes completely*** the performance of a classifier applied to a set of phenomena to be classified.



To be remembered

- The confusion matrix depends on
 - ▶ the classifier
 - ▶ the dataset to which the classifier was applied. !
- It does not make sense
 - ▶ to say “the confusion matrix of the model”
 - ▶ to compare models on the basis of confusion matrices (or any performance metric derived from the CM) without considering if the CMs were obtained using different datasets.



As a consequence ...

- Some properties of the CM depend on both the dataset and the classification model.
- Some properties of the CM depend only on the dataset to which the classification model was applied.
 - ▶ $FP+TN=AN$
 - ▶ $FN+TP=AP$
 - ▶ $AP/(AP+AN) = \rho = (FN+TP)/(FN+TP+FP+TN)$
 - ρ is the proportion of positive elements in the test set.



Independent columns

- The confusion matrix is made of two independent columns.
 - ▶ The column containing TN and FP tells how good was the model at classifying actual negatives.
 - ▶ The column containing TP and FN tells how good was the model at classifying actual positives.
- The columns are independent because a model classifies a given instance based only on that instance's characteristics.
- Let us consider two test sets $TS' = PS' \cup NS$ and $TS'' = PS'' \cup NS$ (where PS and NS represent the set of actual positive and actual negatives, respectively):
 - ▶ We get the same values of TN and FP for both datasets
 - ▶ We (likely) get different values of TP and FN for TS' and TS'' .
- One value from each column describes the CM entirely
 - ▶ In fact, $FP = AN - TN$, $TN = AN - FP$, etc.



From the confusion matrix to performance metrics

- Ideally, we want that
 - ▶ $EP=AP$, $EN=AN$,
 - ▶ $FP=FN=\emptyset$
- In practice, we will never get such outcome.
- Hence, we need to represent how good the classification is.
- Performance metrics were proposed to **summarize** a confusion matrix into a single number.
- All the proposed performance metrics are functions of the confusion matrix.



PERFORMANCE METRICS



Two most important PM

- TPR (true positive rate, alias recall or sensitivity) = TP/AP
 - ▶ It is the proportion of positive instances that were correctly classified
- TNR (true negative rate) = TN/AN
 - ▶ It is the proportion of negative instances that were correctly classified
 - ▶ Sometimes, FPR (False Positive Rate, alias specificity) = $FP/AN = 1 - TNR$ is used.
- Both TPR and TNR are derived by one column of the CM
 - ➔ TPR and TNR address independent and complementary characteristics of the performance



Performance indicators

- Many performance metrics were defined. Here is a sample:

Name	Definition	Aliases	Notes
PPV (positive proportion value)	$TP/EP = TP/(TP+FP)$	Precision	
FM	$2/(1/TPR+1/PPV)$	F-measure, F-score, F1, ...	Harmonic mean of TPR and PPV
Accuracy	$(TP+TN)/(AP+AN)$		
J	$TP/AP-FP/AN$		
Markedness	$TP/EP-FN/EN$		
Matthews Correlation Coefficient	$(TP*TN-FP*FN)/\sqrt{AP*EP*AN*EN}$	MCC, ϕ , Yule phi coefficient	$ MCC = \sqrt{\frac{\chi^2}{AP + AN}}$
Balanced accuracy	$(TPR+TNR)/2$		The arithmetic mean of TPR and TNR



More performance indicators

Jaccard	Jac	$\frac{TP}{n - TN}$
Ochiai-1	Oc1	$\frac{TP}{\sqrt{AP EP}}$
Ochiai-2	Oc2	$\frac{TP TN}{\sqrt{AP AN EP EN}}$
Tarantula	Tar	$\frac{AN TP}{AN TP + AP FP}$
g-mean (est.)	Ge	$\sqrt{\frac{TP TN}{EP EN}}$
g-mean (act.)	Ga	$\sqrt{\frac{TP TN}{AP AN}}$



How to read indicators

- Most indicators have values that range from 0 to 1, where 1 indicates perfect performance and 0 means worst possible performance
- ϕ ranges between -1 and 1
 - ▶ 1=perfect classification
 - ▶ 0=worst classification
 - ▶ -1=perfect inverse classification (positives are estimated negative and vice versa)



All PM can be defined in terms of TPR, TNR and ρ

Definition	Definition in terms of TPR , TNR , and ρ
$TPR = \frac{TP}{AP}$	
$TNR = \frac{TN}{AN}$	
$FPR = \frac{FP}{AN}$	$= 1 - TNR$
$PPV = \frac{TP}{EP}$	$= \frac{\rho TPR}{\rho TPR + (1 - \rho)(1 - TNR)}$
$Acc = \frac{TP + TN}{AP + AN}$	$= \rho TPR + (1 - \rho) TNR$
$BA = \frac{TPR + TNR}{2}$	$= \frac{TPR + TNR}{2}$
$F1 = \frac{2 PPV \cdot TPR}{PPV + TPR}$	$= \frac{2 \rho TPR}{\rho(1 + TPR) + (1 - \rho)(1 - TNR)}$
$Gmean = \sqrt{TPR \cdot TNR}$	$= \sqrt{TPR \cdot TNR}$
$MCC = \frac{TP \cdot TN - FP \cdot FN}{\sqrt{EN \cdot EP \cdot AN \cdot AP}}$	$= \frac{\sqrt{\rho(1 - \rho)}(TPR - FPR)}{\sqrt{(1 - (\rho TPR + (1 - \rho)FPR))(\rho TPR + (1 - \rho)FPR)}}$



THE EXPRESSIVENESS OF PERFORMANCE METRICS



The expressiveness of performance metrics

- We want to know the performance of a given model applied to a given dataset.
- The confusion matrix tells you **everything**.
- TPR and TNR and ρ tell you **everything** as well (in relative terms).
- Each performance metric tells you **something**.
 - ▶ Each performance metric concentrates on a specific aspect of the performance.



The limited expressiveness of performance metrics: the case of the F-measure

- An example of limited representativeness is given by the F-measure
 - ▶ One of the most used PM
- F-measure was conceived in the information retrieval domain.
 - ▶ Suitable when TN is very large and sometimes not even known, i.e., when ρ is very small
 - ▶ For example, a Google search yields TP, FP and FN in the order of 10-100, and TN in the order of 10^{13} .
- T-measure is the harmonic mean of TPR ($TP/(TP+FN)$) and PPV ($TP/(TP+FP)$)
 - ▶ It depends on TP, FP, FN.
- When used with datasets with ρ not very small, it may yield biased indications.



A baseline is needed

- In most cases, performance metrics yield values in the $[0,1]$ range.
 - ▶ 0 indicates perfect misclassification
 - ▶ 1 indicates perfect classification.
- How good is a given value X of the PM?
- As a minimum, it should be better than the value obtained by a “baseline” model.
 - ▶ The most basic baseline model is the random model.



THE RANDOM CLASSIFIER



Random classifiers

- You could perform the classification by tossing a coin.
 - ▶ Every instance from the test set is classified positive with probability p , depending on the “coin” used.
 - ▶ Note that the individual characteristics of an instance are not taken into account to perform the classification.
- Like with any other classification, you get a confusion matrix
- Of course, you may be lucky and get a good classification or unlucky and get a very bad classification.
 - We are interested in the average performance of the random model.



Random classifiers

- You could perform the classification by tossing a coin.
 - ▶ Like with any other classification, you get a confusion matrix
- If you perform the random classification several times, you can compute the average TP, FP, TN and FN
 - ▶ That is, you get a confusion matrix representing the average performance of random estimation



The confusion matrix of the random classifier

		Actual		
		Negative	Positive	
Estimated	Negative	$(1-p) AN$	$(1-p) AP$	$EN = (1-p) n$
	Positive	$p AN$	$p AP$	$EP = p n$
		AN	AP	$n = AN + AP = EN + EP$

- This confusion matrix supports the computation of various performance indicators
 - ▶ $TPR = TP/AP = pAP/AP = p$
 - ▶ $TNR = TN/AN = (1-p)AN/AN = 1-p$
 - ▶ ...



Random classifiers

- When a classifier is evaluated, we should always check if it performs better than random classification.
 - ▶ Unfortunately, this is not always done, also in published papers.

A note on random classification

- Suppose that you want to make predictions concerning a situation characterized by $AP \ll AN$.
 - ▶ For instance, you wish to predict software modules defectiveness, and you expect (e.g., based on previous projects' data) that the rate of defective modules (AP/n) is 5%.
- In these case, you should use a “coin” that has 0.05 probability of classifying a module positive.

- ▶ E.g., a dice like this



- In what follows we assume that random classification classifies elements positive with probability AP/n



PROPERTIES OF PERFORMANCE METRICS



Property

- If both TPR and TNR are better than random
 - ▶ i.e., $TPR > \rho$ and $TNR > 1 - \rho$
- then all performance metrics are better than random.

- Given two models A and B, both evaluated on the same dataset,
- $TPR_A > TPR_B$ and $TNR_A > TNR_B$ implies that PM_A better than PM_B , for all PM.



A source of confusion

- What if model A performs better with respect to positives and model B performs better with respect to negatives?
 - ▶ i.e., $TPR_A > TPR_B$ and $TNR_A < TNR_B$
 - In this cases, it is possible that A appears better than B according to a performance metrics PM' and worse than B according to a performance metric PM''
 - In general, if you read in a paper that the author's model (evaluated using a given dataset) outperforms all the other published models evaluated using performance metric P , you cannot be sure that the model is better than the others in classifying both positives and negatives!
- ➔ If you want to prove that your model is really better than others you must compare it via both TPR and FPR.



COST



What about cost?

- In general, classification errors cause costs.
- With software,
 - ▶ A false negative implies that a defect or a vulnerability go undetected.
 - The consequences can be quite expensive!
 - ▶ A false positive implies that some time is spent to treat a piece of code that did not need any treatment.
 - The consequence usually is a relatively models amount of effort wasted.
 - But many false positives could decrease the confidence of developers in the classification model.



Cost as a performance metric

- If individual costs are known, a very relevant performance metric is given by costs incurred:

$$Cost = TP \times C_{debug} + FN \times C_{failure} + FP \times C_{inspect}$$

- ▶ where

- C_{debug} is the cost of debugging an actually defective module
- $C_{failure}$ is the cost of a failure due to a defect
- $C_{inspect}$ is the cost of uselessly inspecting a defect-free module

- ▶ Note: a better definition of cost should consider the possibility that treating false positives introduces new defects, or that the treatment of true positives does not actually remove defects, etc.

- Note: the “misclassification cost” has been proposed as a PM
 - ▶ It does not account for the treatment cost of true positives



THE (SO CALLED) EFFORT-AWARE PERFORMANCE METRICS



Notes & warnings

- Effort-aware metrics (EAMs) apply when model yield the *probability* that an instance is positive.

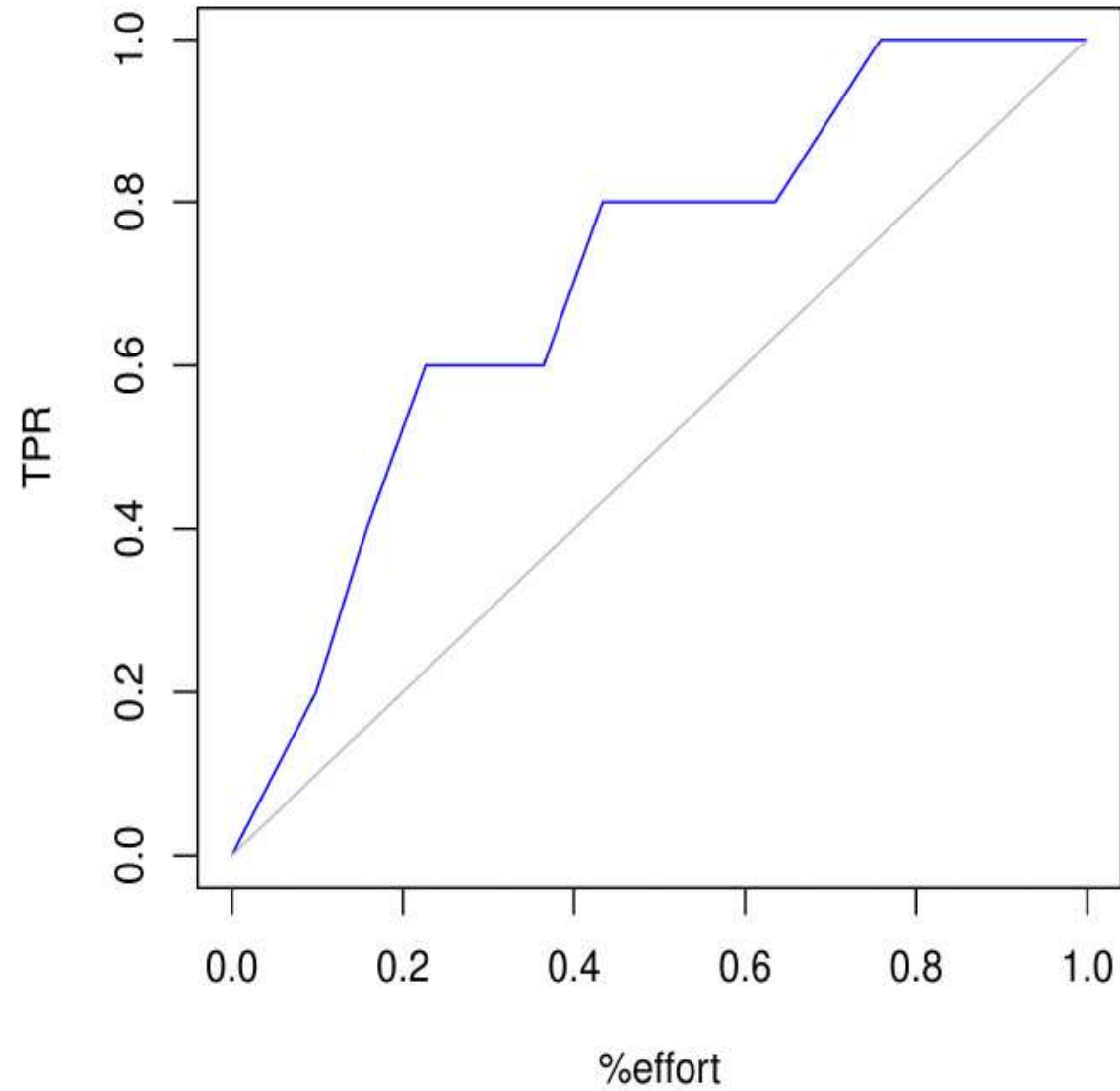


Effort-aware performance metrics

- Some authors criticized traditional performance metrics (i.e., those computed in the basis of the CM) because
 - ▶ they do not account for the effort needed to inspect the modules that were classified defective
 - ▶ they do not consider that the effort available to inspect the modules that were classified defective might be limited.
- They proposed metrics that are based on cost efficiency curves, i.e., curves that describe TPR as a function of the effort spent.
 - ▶ The underlying assumption is that
 - Models compute for each module its probability of being defective
 - Modules are analysed in order of decreasing probability
 - The effort spent to analyse each module depends on the size measured in LOC
 - EP depend on the available effort



A cost-efficiency curve





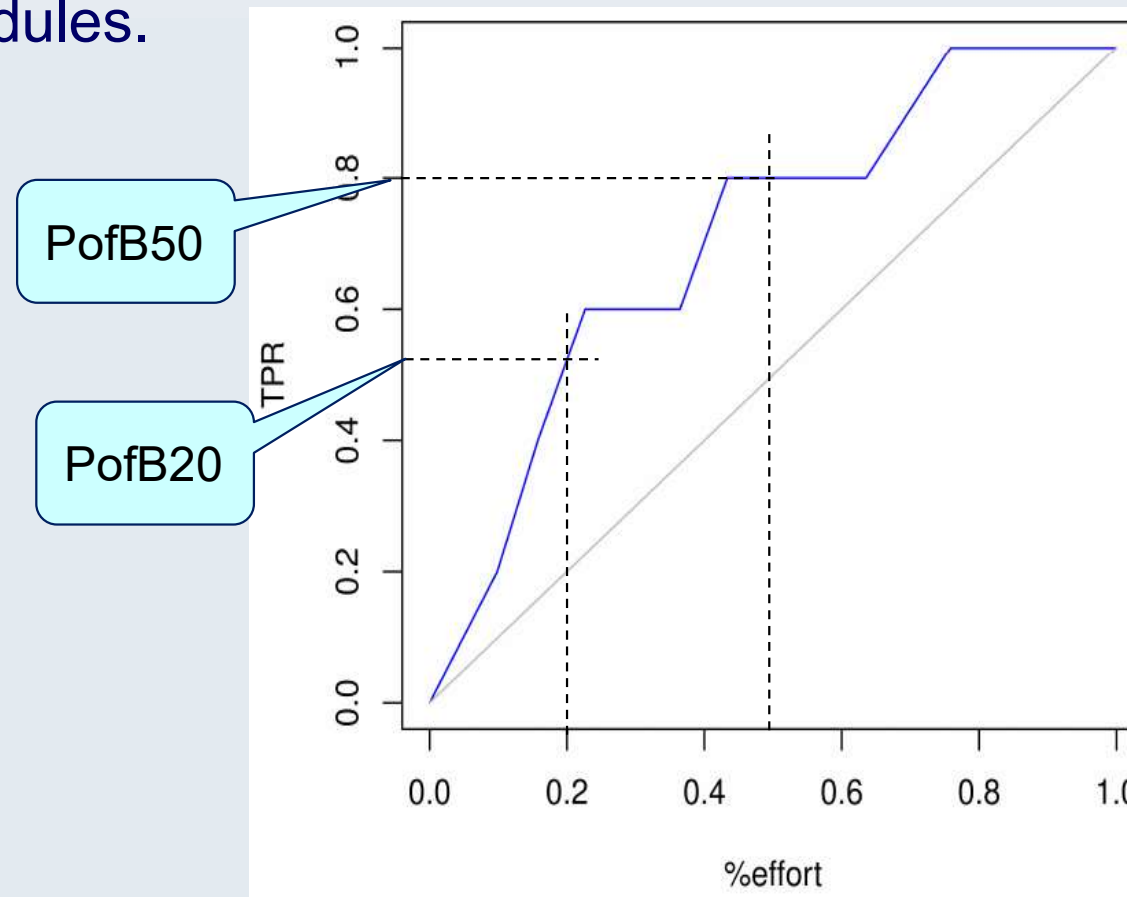
Note

- Sometimes cost-efficiency curves plot the proportion of bugs found, rather than the proportion of defective modules found, as a function of effort.
- The performance metrics based on cost-efficiency curve change accordingly.



Effort-aware performance metrics

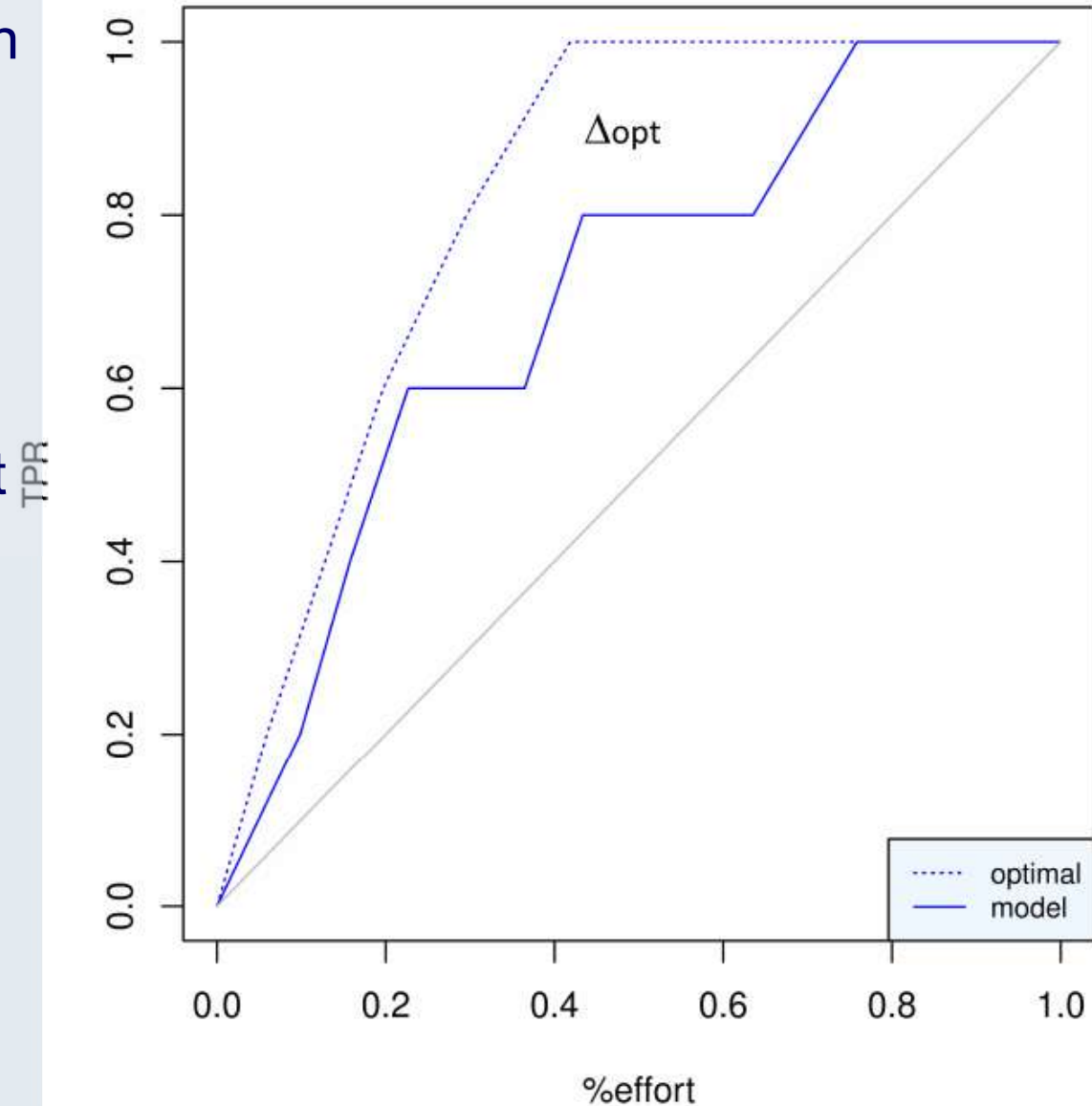
- PofB20 and PofB50
- The value of TPR when the effort spent is 20% or 50%, respectively
 - ▶ The percentage refers to the amount of effort needed to analyse all the system's modules.





Another effort-aware PM: Δ_{opt}

- Δ_{opt} measure the area between the cost-efficiency curve of the optimal model and the curve of the considered model.
- The optimal model is the one that considers positive model first, in order of increasing effort required for analysis.
 - ▶ In this way, it achieves the highest possible TPR for any value of %effort.





Why EAMs can be useful

- It has been shown that classification model that appear reasonably accurate according to traditional performance metrics are actually worse than random when analysis effort is taken into account.
- This supports the idea that cost-based evaluation is useful.
 - ▶ Though not necessarily carried out as done by EAMs.



Problems with effort-aware performance metrics

- EAMs assume that analysis effort is proportional to the size measured in LOC. This effort model is not realistic.
 - ▶ Different effort models can lead to completely different conclusions.
- Cost efficiency curves consider only TPR as a function of effort. They ignore TNR (or FPR).
 - ▶ Accordingly, metrics like PofB or Δ_{opt} provide a partial evaluation of performances.
- EAMs are actually aware only of the analysis effort. They do not consider other types of effort.
 - ▶ E.g., the effort needed to correct a module that has been recognized as faulty after being released are not considered, even though such corrections can (and usually do) require a relevant amount of effort.
- ...



CONCLUSIONS



Conclusions

- Evaluating binary classifications is not easy.
- Many researchers are rather ignorant of the issues related to the evaluation of classifiers.
 - ▶ They use a given metric just because it was used in previous papers
 - ▶ So, evaluation errors are repeated...



Suggestions

- Always provide confusion matrices
 - ▶ or, alternatively, TPR, TNR and ρ
- Remember that comparisons should be based on the same test set
- Remember that a model should perform better than the random model
- Whenever possible, consider the costs resulting from the activities that use the classifiers.
 - ▶ To this end, use realistic cost models.



What else?

- When a model yield the probability that an instance is positive, specific techniques are used to evaluate the models' performance.
- The best-known is the ROC curve.
- Unfortunately, also the ROC curve and the associated metrics have a number of problems
 - ▶ They are also not well understood: many researchers wrote that ROC curves are “threshold independent” when actually they account for all possible thresholds, while they are independent on any specific threshold.



References to my papers

- Morasca, Sandro, and Luigi Lavazza. "On the assessment of software defect prediction models via ROC curves." *Empirical Software Engineering* 25.5 (2020): 3977-4019.
- Lavazza, Luigi, and Sandro Morasca. "Comparing ϕ and the F-measure as performance metrics for software-related classifications." *Empirical Software Engineering* 27.7 (2022): 185.
- Lavazza, Luigi, and Sandro Morasca. "Considerations on the region of interest in the ROC space." *Statistical Methods in Medical Research* 31.3 (2022): 419-437.
- Lavazza, Luigi, Sandro Morasca, and Gabriele Rotoloni. "On the reliability of the area under the ROC curve in empirical software engineering." *Proceedings of the 27th International Conference on Evaluation and Assessment in Software Engineering*. 2023.
- Lavazza, Luigi, Sandro Morasca, and Gabriele Rotoloni. "An Experience in the Evaluation of Fault Prediction." *International Conference on Product-Focused Software Process Improvement*. Cham: Springer Nature Switzerland, 2023.
- Lavazza, Luigi, and Sandro Morasca. "Common problems with the usage of f-measure and accuracy metrics in medical research." *IEEE Access* 11 (2023): 51515-51526.
- Lavazza, Luigi, Gabriele Rotoloni, and Sandro Morasca. "Critical Considerations on Effort-aware Software Defect Prediction Metrics." *arXiv preprint arXiv:2504.19181* (2025).
- Lavazza, Luigi, Sandro Morasca, and Gabriele Rotoloni. "Software Defect Prediction evaluation: New metrics based on the ROC curve." *Information and Software Technology* (2025): 107865.



**THANKS FOR YOUR ATTENTION!
QUESTIONS?**