2024 18th International Conference on Ubiquitous Information Management and Communication (IMCOM) | 979-8-3503-3101-1/24/$31.00 ©2024 IEEE | DOI: 10.1109/IMCOM60618.2024.10418419

NoSimple: Data Bias Evaluation Metrics

Sylwan Rahardja and Pasi Fränti
University of Eastern Finland, Joensuu, Finland
sylwanrahardja@ieee.org, franti@cs.uef.fi

***Abstract*— *Simple objects* are defined as objects invariably correctly classified by all outlier detectors. Its presence impairs performance of binary classifiers such as ROC or F1 score. A large number of simple objects falsely improve performance of binary classifiers when evaluated by ROC or F1 score. This impairs reliability of classifier evaluation. This manuscript proposes evaluation without simple objects (NoSimple). NoSimple preprocesses data to factor in simple objects by removing the simple objects for the evaluation phase. Experiments with 30 real-world datasets demonstrate that NoSimple significantly reduced the average ROC of all classifiers by 0.04 ~ 0.06. NoSimple is most effective when the percentage of simple objects exceeds 30%. By introducing a new method to reliably evaluate outlier classifiers, NoSimple has the potential to revolutionize evaluation metrics and has a multitude of applications in data science research.**

***Index Terms*— evaluation metric, ROC, F1 score, NoSimple, evaluation without simple objects, outlier detection, CTR.**

I. Introduction

Evaluation metrics play an important role in model comparison and model selection. Evaluation metrics analyze predicted and ground truth labels to provide a tangible outcome in terms of a real number. Evaluation metrics rely on thresholds to assign predicted labels for data objects when measuring score-based binary classifiers.

Typical evaluation metrics for score-based binary classifiers can be categorized into three groups based on applying single, multiple, or all thresholds [15] to the prediction results of binary classifiers as shown in Fig. 1. Single-threshold-dependent evaluation metrics include the balanced accuracy (BA) [1], the geometric mean (GM) [1, 2], F1 score [1, 3], and Matthews' Correlation Coefficient (MCC) [4]. Multiple-thresholds-dependent evaluation metrics include the partial area under curve (pAUC) [5, 6], the standardized partial area (sPA) [7] and the concordant partial area under curve (pAUCc) [8]. All-thresholds-dependent evaluation metrics include average precision (AP) [9], area under curve plotted with receiver operating characteristic (ROC) [10], predictive ROC curve [11], the positive tradeoff curve (PTC) [12], H measure [13], and area under the cost curve (AUCC) [14].

The evaluation metrics each have their limitations. Single-threshold-dependent evaluation metrics are too specific [15]. Hence, these metrics lack information when nearby threshold leads to rapid performance change [7, 16]. In contrast, all-threshold-dependent evaluation metrics are too general [15]. They include performance from thresholds that would not be used in practice [17, 18] and do not provide any information about the distribution of performance along various thresholds [17]. Multiple-threshold-dependent evaluation metrics rely on threshold values chosen. They achieve a middle ground in terms of specificity.

All the above-mentioned evaluation metrics measure the performance of classifiers using all the prediction results without considering the distribution of data objects. We categorize the objects in data into two groups. Objects which can never be misclassified by any classifiers are called *simple objects*, other objects are called *non-simple objects*. The distribution of simple objects and non-simple objects will affect the result of the evaluation metrics. As shown in Fig. 1, two classifiers measured by ROC [10] have 0.78 and 0.34 ROC originally. Upon increasing the number of the simple objects in data, the difference in performance approaches zero and the ROC approaches 1.00, approaching perfection. This phenomenon should not be disregarded especially in large data applications, which have a relatively higher probability to contain more simple objects. Hence, it is clear that evaluation metrics should consider simple objects.

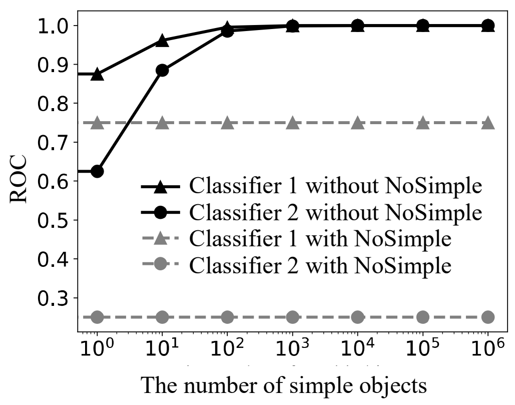

Fig. 1. Prediction results of Classifier 1 = {1, 2, 110, 6, 120}, Prediction results of Classifier 2 = {100, 150, 2, 130, 3}, label = {0, 0, 0, 1, 1}. Different numbers of zeroes serving as simple objects are added to the prediction results of both classifiers. The ROCs with and without NoSimple are plotted. Upon increasing the number simple objects, NoSimple preserves the distinction between classifiers.

To reduce the effect of simple objects on evaluation metrics, we propose a framework called *evaluation without simple objects* (NoSimple). NoSimple works in two steps. Firstly, it removes the prediction results of common simple objects of classifiers pending evaluation. Secondly, it employs existing evaluation metrics with remaining prediction results. The results of the evaluation metrics will be used as the performance indicator for classifiers. As shown in Fig. 2, NoSimple can be applied to all existing evaluation metrics simply by changing the input data from all prediction results to partial results. We hypothesize that the proposed new versatile framework will improve all existing evaluation metric.

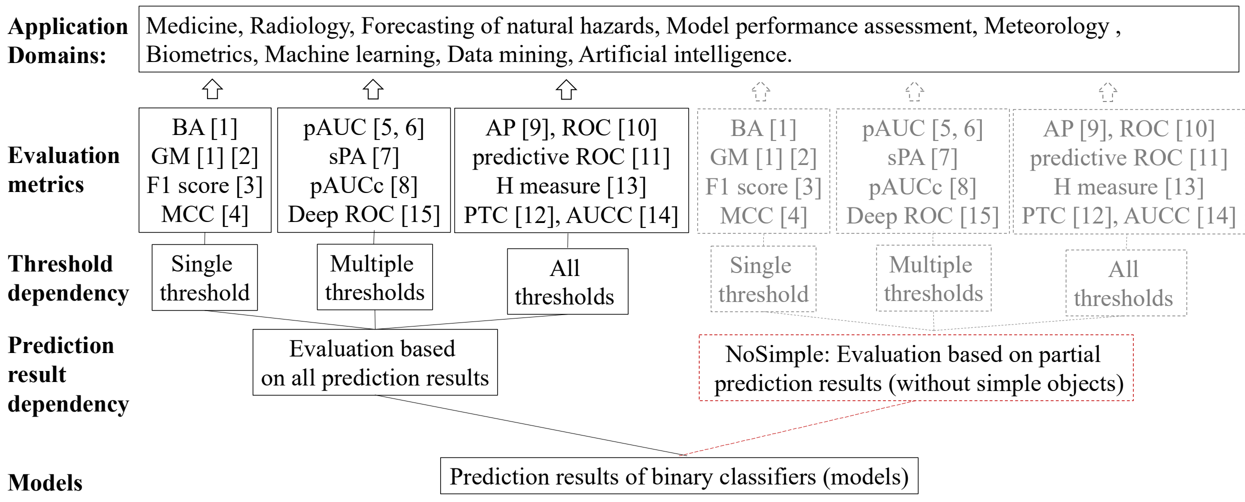
Application Domains:
Medicine, Radiology, Forecasting of natural hazards, Model performance assessment, Meteorology, Biometrics, Machine learning, Data mining, Artificial intelligence.
Evaluation metrics
BA [1] GM [1] [2] F1 score [3] MCC [4]
pAUC [5, 6] sPA [7] pAUCc [8] Deep ROC [15]
AP [9], ROC [10] predictive ROC [11] H measure [13] PTC [12], AUCC [14]
Threshold dependency
Single threshold
Multiple thresholds
All thresholds
Prediction result dependency
Evaluation based on all prediction results
NoSimple: Evaluation based on partial prediction results (without simple objects)
Models
Prediction results of binary classifiers (models)

Fig. 2. Methods of evaluating the prediction results of binary classifiers based on all results and the proposed partial results. NoSimple may influence all the existing evaluation metrics, hence affecting various application domains.

II. Simple Objects and Their Effects

A. Simple objects

Given a data set of objects X for a binary classification task, the objects that will always be correctly classified by any classifiers are defined as simple objects, while the remainder are *non-simple objects*. Within the simple objects, they are further divided into 2 subsets, namely *simple positive objects* which are classified as positive and *simple negative objects* which are classified as negative.

Logically, to identify all simple objects, one must test all classifiers. Realistically, this is operationally impossible. To tackle this dilemma, we focus on one classifier and define its simple objects as *local simple objects*. When repeated with multiple classifiers, there will be multiple groups of local simple objects, providing an approximation of all simple objects. We define local simple objects by relying exclusively on one classifier. Given a classifier d, it has a prediction score S^d for each object in X. An object is classified as a local simple object if its prediction score is smaller than the score of any positive objects in S^d, or larger than the score of any negative objects in S^d. A set of local simple objects φ^d can be expressed as follows:

$$\varphi^d = \{x_i | s_i^d < s_k^d \text{ OR } s_i^d > s_t^d; s_i^d, s_t^d, s_k^d \in S^d; x_i \in X; \forall\, x_k \in P; \forall\, x_t \in N\} \quad (1)$$

where X = P ∪ N, P is a set of positive objects and N is a set of negative objects; s_i^d is the classifier d's prediction score of the object x_i. A set of standard simple objects φ can be defined as:

$$\varphi = \{x_i | x_i \in \varphi^d; \forall\, d \in D\} \quad (2)$$

where D represents either the set of all existing classifiers or a set of classifiers given.

Hence, the set of standard simple objects is simply the expression of all sets of local simple objects. Similarly, the more classifiers in D, the closer the local simple object φ^d approaches the standard simple objects.

B. Simple objects' effect on F1 score

The F1 score measures the accuracy of classifiers. Given a set of prediction scores from a classifier and a pre-defined threshold value, we can obtain the number of correctly predicted positive objects, falsely predicted positive objects, and falsely predicted negative objects, denoted as true positive (TP), false positive (FP), and false negative (FN), respectively. Then, the F1 score can be expressed as follows:

$$\text{F1 score} = 1/(1 + 0.5(\text{FP} + \text{FN})/\text{TP}). \quad (3)$$

When the number of simple positive objects increases, this increases the TP rate and thus the F1 score increases proportionally. In contrast, the number of simple negative objects has no effect on the F1 score. Therefore, when the F1 score increases, it could be attributed to a larger proportion of simple positive objects in the dataset rather than a more robust classifier.

C. Simple objects' effect on ROC

ROC is defined as the area under curve plotted with receiver operating characteristic [10]. ROC is equivalent to the Wilcoxon test of ranks [19], hence can be expressed by the following Wilcoxon-Mann-Whitney equation:

$$\text{ROC} = \frac{\sum_{t,k} I\left(s_t^d, s_k^d\right)}{pq}, \quad (4)$$

$$I\left(s_t^d, s_k^d\right) = \begin{cases} 1, \text{IF } s_t^d > s_k^d \\ 0 \end{cases}, \quad (5)$$

where p and q denote the number of the positive objects and the number of negative objects respectively.

When the dataset contains w simple positive objects and m simple negative objects, the ROC is calculated as follows after simplification:

$$\mathrm{ROC} = 1 - \frac{pq - \sum_{t,k} I\left(s_t^d, s_k^d\right)}{(p+w)(q+m)}. \tag{6}$$

From the above equation, we deduce that when w or m increases, the ROC increases proportionally. Hence both simple positive and negative objects have similar effect on ROC. Akin to the prior deduction from the F1 score, higher ROC values do not indicate better classifiers. The higher ROC value could be merely due to the presence of a larger number of simple objects. An example illustrating this phenomenon is shown in Fig. 1. With increasing quantities of simple objects, the ROC of both the robust Classifier 1 and subpar Classifier 2 can be improved to 1.00.

III. Evaluation without simple objects (NOSIMPLE) and discussion

This section introduces NoSimple and discusses its potential for industry applications.

A. NoSimple

Regardless of the evaluation metric used, simple objects invariably impair the evaluation metric's ability to identify an accurate classifier. With ROC as an accepted method of evaluating classifiers, the crux of the issue lies with the positive influence of simple objects on ROC value. Similarly, by utilizing the F1 score, simple objects can increase TP rate and thus the F1 score. However, this falsely elevated value does not truly reflect the performance of the classifier. This problem is confounded by other covariates such as the number of relevant objects or more advanced classifier models.

NoSimple aims to resolve this issue by reducing the effect of simple objects on evaluation of classifiers, by simply excluding these objects from the evaluation phase. However, the simple objects should not be permanently removed from its native dataset as its removal will bias other further evaluation such as model training. NoSimple is summarized in Algorithm 1. It essentially a three-step process: Firstly, the model attempts to identify the simple objects with Eq. (1) and (2). Next, the model removes the simple objects from the prediction scores of classifiers. Lastly, the model performs evaluation metric analysis on the remaining prediction scores of classifiers.

Examples with and without using NoSimple are shown in Fig. 1. Given label = {0, 0, 0, 1, 1}, prediction results of Classifier 1 = {1, 2, 110, 6, 120}, and prediction results of Classifier 2 = {100, 150, 2, 130, 3}, equal zeroes were added to serve as the prediction results of simple objects into the prediction results of both classifiers. We plotted their ROC with and without NoSimple varying the numbers of simple objects. Without NoSimple, the evaluation results in an illusion that the performance of the two classifiers are similar. With NoSimple, the true disparity of the performance between the two classifiers is reflected regardless of the number of simple objects.

Algorithm 1: NoSimple

```
Input: A set of classifiers: D, A set of prediction scores from D: S;
       A set of data objects: X; An evolution metric E(*)
Output: A set of measurements for each classifier in D: M
1. $\varphi \leftarrow$ Find simple objects $\varphi$ using Eq. (1) and (2);
2. FOR EACH CLASSIFER $d \in D$:
3.     $t^d \leftarrow \{s_i^d | x_i \notin \varphi, s_i^d \notin S\}$, remove simple objects
4.     $m_i \in M \leftarrow E(t^d)$, calculate measurement
5 END FOR
```

B. Discussion: potential for cut-edge problems

Industry applications such as click through rate (CTR) prediction often suffer substantial discrepancy between the offline and online performance of the predictive models. Essentially, there is variation of real-time performance of models as compared to retrospective analyses or offline analyses. Studies on this disparity [21] reported that classifiers with higher ROC value in offline testing stage may perform worse in online performance. This phenomenon is related to the evaluation metric of ROC. As described previously that both negative and positive simple objects influence the evaluation of classifiers, simple objects should be considered in training, testing and evaluation. Apart from this, ROC values are highly dependent on the underlying distribution of data [20]. For this reason, higher ROC score for a model trained with higher rate of negative samples do not necessarily imply the model has better predictive performance. In a binary classification task, samples are labelled as *negative samples* and *positive samples*. In terms of outlier detection, the normalities are negative samples. Our studies about simple objects' effect on evaluation metrics support the claim by Yi et al. regarding the significance of the distribution of data [20].

IV. Experiments with case studies

Experimental results with case studies in the outlier detection domain are reported in Table I. Outlier classifiers (or detectors) normally calculate a score for each object to determine if an object is an outlier or a normality. We used 30 real-world datasets which can be found via UCI data repository or in the article by Campos et al. [23]. The information of the datasets is summarized in first four columns of Table I. We employed MOD [24], LOF [25], IF [26], and COP [27] as classifiers. ROC is utilized as the evaluation metric. The results with and without NoSimple are shown in Table I. The datasets are listed in order of increasing percentages of simple objects in data.

From the row labeled *AVG (Average)*, with 18.87% simple objects on average, the average ROC of all classifiers are significantly reduced with -0.04, -0.05, -0.06, and -0.05 to ROC values respectively. When the percentage of simple objects is less than 2%, there is no significant difference between the results with and without NoSimple. When the percentage of simple objects is between 2% and 30%, the difference between the results with and without NoSimple is approximately -0.02 to -0.10 ROC. When the percentage of simple objects is more than 34%, the difference between the results with and without NoSimple approaches -0.40 ROC. This illustrates that simple objects improve ROC proportionally.

In addition, the application of NoSimple preserves the relative performance of classifiers but essentially cleans the dataset involved. Hence, addition of NoSimple will improve accuracy of evaluation metrics of classifiers, without impairing comparison between different classifiers.

For the **wbc** dataset, originally the LOF classifier performs relatively well with 0.96 ROC and the difference of ROC between LOF and other classifiers is no more than 0.02 ROC. However, after removing simple objects, the performance is reduced to 0.56 ROC, which is only marginally better than the random guessing. Surprisingly, the gap between LOF and other classifier increased to up to 0.34 ROC, because most of the local simple objects for LOF are standard simple objects. Similar observations were made for **stamps** (sta.) and **shuttle** (shu.) datasets. This indicates that the presence of simple objects may mislead users into selecting unsuitable classifiers.

TABLE I DATASET INFORMATION AND EXPERIMENT RESULTS

Set	**N**	**D**	**O**	**Simple objects** (%)	**Original**				**NoSimple**			
					MOD [24]	LOF [25]	IF [26]	COP [27]	MOD [24]	LOF [25]	IF [26]	COP [27]
let.	1600	32	100	0.01<	0.62	0.83	0.62	0.56	0.62	0.83	0.62	0.56
bre.	683	9	239	0.01<	0.99	0.49	0.99	0.99	0.99	0.49	0.99	0.99
spa.	4207	57	1679	0.01<	0.58	0.43	0.61	0.69	0.58	0.43	0.61	0.69
sate.	6435	36	2036	0.11	0.73	0.57	0.70	0.63	0.73	0.57	0.70	0.63
pim.	768	8	268	0.13	0.75	0.66	0.68	0.65	0.75	0.66	0.68	0.65
car.	2114	21	466	0.14	0.49	0.55	0.70	0.66	0.49	0.55	0.70	0.66
wil.	4819	5	257	0.64	0.45	0.55	0.44	0.34	0.44	0.55	0.43	0.34
ann.	7129	21	534	0.83	0.62	0.69	0.63	0.69	0.61	0.69	0.63	0.68
wpb.	198	33	47	1.01	0.58	0.51	0.51	0.52	0.57	0.50	0.50	0.52
wav.	3443	21	100	1.05	0.64	0.73	0.73	0.73	0.63	0.73	0.73	0.73
pag.	5393	10	510	1.08	0.90	0.87	0.90	0.88	0.90	0.86	0.90	0.87
hea.	270	13	120	1.11	0.72	0.61	0.61	0.69	0.71	0.60	0.60	0.69
ver.	240	6	30	2.08	0.31	0.36	0.35	0.33	0.30	0.34	0.34	0.32
arr.	450	259	206	2.89	0.70	0.73	0.74	0.76	0.68	0.71	0.73	0.74
vow.	1456	12	50	2.95	0.91	0.94	0.79	0.50	0.91	0.94	0.78	0.48
opt.	5216	64	150	4.39	0.31	0.38	0.74	0.68	0.27	0.36	0.72	0.67
mni.	7603	100	700	4.95	0.85	0.79	0.78	0.77	0.84	0.78	0.77	0.76
par.	195	22	147	5.13	0.51	0.46	0.49	0.54	0.47	0.43	0.46	0.51
mus.	3062	166	97	7.94	0.99	0.39	1.00	0.95	0.99	0.33	1.00	0.94
ion.	351	32	126	11.97	0.62	0.89	0.85	0.79	0.53	0.86	0.81	0.74
sati.	5803	36	71	30.64	1.00	0.87	0.99	0.97	1.00	0.81	0.99	0.96
win.	129	13	10	32.56	0.93	0.93	0.81	0.87	0.89	0.89	0.70	0.79
lym.	148	3	6	34.46	0.81	0.89	0.85	0.83	0.71	0.83	0.78	0.74
pen.	9868	16	20	38.71	0.97	0.93	0.87	0.58	0.94	0.89	0.79	0.31
glas.	214	7	9	46.73	**0.76**	0.78	0.77	0.76	**0.54**	0.57	0.55	0.52
sta.	340	9	31	58.24	0.93	0.82	0.90	0.93	0.82	0.50	0.73	0.81
wdb.	367	30	10	58.86	0.90	0.93	0.93	0.97	0.74	0.81	0.83	0.93
shu.	1013	9	13	63.97	0.99	0.99	**0.87**	**0.82**	0.96	0.96	**0.64**	**0.48**
thy.	3772	6	93	65.35	0.97	0.95	0.98	0.94	0.92	0.84	0.93	0.82
wbc	454	9	10	88.11	0.98	**0.96**	0.99	0.99	0.81	**0.56**	0.90	0.89
AVG	-	-	-	18.87	0.75	0.71	0.76	0.73	0.71	0.66	0.72	0.68

Where N refers to objects, D refers to Dim, O refers to Outliers.

Thus, it is prudent to evaluate the classifiers by checking the percentage of simple objects. When the percentage is more than 30%, the application of NoSimple is key to distinguish whether the difference in perceived performance is simply due to dataset factors. NoSimple preserves the relative performance between classifiers and instead alters the absolute performance of classifiers based on the selected evaluation metric. If the application of NoSimple does not bring about significant differences in ROC, it is inferred that either classifier can be used and the performance evaluation is unbiased.

V. CONCLUSION

By introducing the concept of simple objects and their effect on existing evaluation metrics for binary classifiers, the NoSimple framework was proposed to reduce simple objects' bias for any evaluation metrics for binary classifiers. We discovered that with increasing numbers of simple objects, ROC and F1 score tend to approach 1 and falsely reflect good performance. Additionally, simple objects bring different impact to different classifiers and applications, but the relative performances of different classifiers are preserved. NoSimple provides a systemic method of evaluating binary classifiers and provides important clues for solving existing problems in the industry. Needless to say, NoSimple brings about a multitude of potential applications.

REFERENCES

[1] G. Santafe, I. Inza, and J. A. Lozano, "Dealing with the evaluation of supervised classification algorithms," *Artificial Intelligence Review*, vol. 44, no. 4, pp. 467–508, 2015.

[2] M. Wu and J. Ye, "A small sphere and large margin approach for novelty detection using training data with outliers," *IEEE transactions on pattern analysis and machine intelligence*, vol. 31, no. 11, pp. 2088–2092, 2009.

[3] P. Flach, "Performance Evaluation in Machine Learning: The Good, the Bad, the Ugly, and the Way Forward," *Proceedings of the AAAI Conference on Artificial Intelligence,* vol. 33, pp. 9808–9814, 2019.

[4] Q. Zhu, "On the performance of Matthews correlation coefficient (MCC) for imbalanced dataset," *Pattern Recognition Letters*, vol. 136, pp. 71–80, 2020.

[5] D. K. McClish, "Analyzing a Portion of the ROC Curve," *Medical decision making*, pp. 190–195, 1989.

[6] M. Thomson and W. Zucchini, "On the statistical analysis of ROC curves," *Statistics in Medicine*, vol. 8, pp. 1277–1290, 1989.

[7] D. K. McClish, "Evaluation of the Accuracy of Medical Tests in a Region around the Optimal Point," *Academic Radiology*, vol. 19, no. 12, pp. 1484–1490, 2012.

[8] A. M. Carrington, P. W. Fieguth, H. Qazi, A. Holzinger, H. H. Chen, F. Mayr, and D. G. Manuel, "A new concordant partial auc and partial c statistic for imbalanced data in the evaluation of machine learning algorithms," *Springer/Nature BMC Medical Informatics and Decision Making*, vol. 20, no. 1, pp. 1–12, 2020.

[9] T. Saito and M. Rehmsmeier, "The precision-recall plot is more informative than the ROC plot when evaluating binary classifiers on imbalanced datasets," *PLoS ONE*, vol. 10, no. 3, pp. 1–21, 2015.

[10] Fawcett, Tom. "An Introduction to ROC Analysis". *Pattern Recognition Letters*. 27 (8): 861–874, 2006.

[11] S.-Y. Shiu and C. Gatsonis, "The predictive receiver operating characteristic curve for the joint assessment of the positive and negative predictive values," *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, vol. 366, no. 1874, pp. 2313–2333, 2008.

[12] C. O'Reilly and T. Nielsen, "Revisiting the ROC curve for diag- nostic applications with an unbalanced class distribution," *2013 8th International Workshop on Systems, Signal Processing and Their Applications*, WoSSPA 2013, pp. 413–420, 2013.

[13] D. J. Hand, “Measuring classifier performance: A coherent alterna- tive to the area under the ROC curve,” *Machine Learning*, vol. 77, no. 1, pp. 103–123, 2009.

[14] P. A. Flach, J. Hernández-Orallo, and C. F. Ramirez, “A coherent interpretation of auc as a measure of aggregated classification performance,” in *ICML*, 2011.

[15] A. M. Carrington *et al*., "Deep ROC Analysis and AUC as Balanced Average Accuracy, for Improved Classifier Selection, Audit and Explanation," in *IEEE Transactions on Pattern Analysis and Machine Intelligence*,

[16] A. P. Bradley, “The use of the area under the {ROC} curve in the evaluation of machine learning algorithms,” *Pattern Recognition*, vol. 30, pp. 1145–1159, 1997.

[17] J. M. Lobo, A. Jiménez-valverde, and R. Real, “AUC: a misleading measure of the performance of predictive distribution models,” *Global Ecology and Biogeography*, no. 17, pp. 145–151, 2008.

[18] S. Mallett, S. Halligan, M. Thompson, G. S. Collins, and D. G. Altman, “Interpreting diagnostic accuracy studies for patient care,” *Bmj*, vol. 345, 2012.

[19] Mason, Simon J.; Graham, Nicholas E. "Areas beneath the relative operating characteristics (ROC) and relative operating levels (ROL) curves: Statistical significance and interpretation". *Quarterly Journal of the Royal Meteorological Society*. 128 (584): 2145–2166. 2022.

[20] Jeonghee Yi, Ye Chen, Jie Li, Swaraj Sett, and Tak W. Yan. 2013. Predictive model performance: offline and online evaluations*. In Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*. Association for Computing Machinery, New York, NY, USA, 1294–1302, 2013.

[21] Heng-Tze Cheng, Levent Koc, Jeremiah Harmsen, Tal Shaked, Tushar Chandra, Hrishi Aradhye, Glen Anderson, Greg Corrado, Wei Chai, Mustafa Ispir, et al. Wide & deep learning for recommender systems. *In Proceedings of the 1st Workshop on Deep Learning for Recommender Systems. ACM*, 7–10, 2016.

[22] H. Zhu, J. Jin, C. Tan, F. Pan, Y. Zeng, H. Li, et al., "Optimized cost per click in taobao display advertising", *Proc. 23rd ACM SIGKDD Int. Conf. Knowl. Discovery and Data Mining*, pp. 2191-2200, 2017.

[23] G.O. Campos, A. Zimek, J. Sander, R.J.G.B. Campello, B. Micenkova, E. Schubert, I. Assent, and M.E. Houle, "On the evaluation of unsupervised outlier detection: measures, datasets, and an empirical study, " *Data Mining and Knowledge Discovery*, 30 (4), 891–927, 2016.

[24] J. Yang, S. Rahardja, P. Fränti, "Mean-shift outlier detection and filtering," *Pattern Recognition*, 115, 107874, 2021

[25] M.M. Breunig, H. Kriegel, R.T. Ng, and J. Sander, "LOF: Identifying density-based local outliers," *ACM SIGMOD Int. Conf. on Management of Data,* 29 (2), 93-104, 2000.

[26] F. Liu, T. Ting, K. Ming, and ZH. Zhou, "Isolation-based anomaly detection," *ACM Transactions on Knowledge Discovery from Data* (TKDD), 6 (1), 3:1-3:39, 2012.

[27] Z. Li, et al. "COPOD: Copula-Based Outlier Detection," *IEEE International Conference on Data Mining* (ICDM), 2020