Judging competitions and benchmarks: a candidate election approach

Adrien Pavao[1], Michael Vaccaro[1], and Isabelle Guyon[1,2]

1- LISN/CNRS/INRIA, Université Paris-Saclay, France
2- ChaLearn, California, USA

Abstract. Machine learning progress relies on benchmarking algorithms on various datasets or tasks. The problem we are addressing is that of declaring a winner or making a single total ranking of candidates (algorithms or competition participants), based on results obtained by several judges or voters (performance scores on multiple datasets or tasks). Inspired by social science and game theory literature, devoted to conducting elections in a fair way, we compare a number of ranking functions, ranging from simple score averaging across tasks, to Condorcet methods. We devise novel empirical criteria to assess the quality of ranking, including generalization to new tasks and stability under judge or candidate perturbation. We conduct an empirical comparison on the results of 5 competitions and benchmarks (one artificially generated). While theoretical analyses of ranking functions from previous authors indicate that no single ranking function satisfies all desired properties, our empirical analysis reveals that the classical average rank method fares well and is computationally efficient. However, pairwise/Condorcet methods offer better theoretical guarantees and also get better empirical results.

1 Introduction

`https://github.com/Didayolo/ranking-esann-2021`

The problem of aggregating individual preferences into one global ranking is encountered in many application domains: politics, economics, sports, web pages ranking, and more. We are interested in a specific instance of this problem, where candidates are machine learning algorithms and judges are tasks to solve. In machine learning, the problem is typically that of ranking algorithms by their performance on several datasets, particularly when conducting benchmarks or organizing scientific competitions.

Organizers of machine learning and benchmarks usually simply average scores or average the ranks of the competitors obtained on the various tasks, to obtain a ranking function. However, theory has been developed around the problem of ranking in the fields of social choice theory, economics, and game theory, characterizing the properties satisfied or not by the different ranking functions. Arrow [7] and later Gibbard [2] have shown that this problem is not trivial and that no aggregation method can satisfy all the desired properties. Desired properties include that a candidate ranked first by a majority of judges must be declared winner (*winner criterion*), and that the ranking should be stable under perturbations of the set of judges (*judge criterion*) and the set of candidates (*candidate*

criterion). The theoretical criteria are qualitative rather than quantitative: satisfied or not. The charter of this paper is to provide theoretically motivated criteria, allowing us to quantitatively evaluate ranking functions empirically on machine learning use cases. Our goal is to determine whether, in spite of pessimistic theoretical predictions, some ranking functions offer a good compromise between all criteria. To that end, we devise empirical quantitative equivalent of the theoretical criteria, and estimate them using the bootstrap [5].

In line with [3], we make connection with meta-learning. We include meta-generalization as part of our criteria. Thus, our stated goal is that benchmarks or competitions go beyond determining which algorithms performs best on the particular given set of tasks provided for the evaluation: their purpose is to uncover an algorithm that will fare well on *future* tasks of a similar nature. This setting arises in particular in AutoML competitions (e.g. [6, 9]). We introduce a form of cross-validation at the task level to evaluate such meta-generalization performance.

We perform an empirical evaluation of ranking functions using the criteria we propose on the results of five competitions and benchmarks (and one artificially generated, to put methods to a harder test in which judges have diverging opinions). We compare and contrast the results with theoretical predictions and provide guidelines to select methods according to circumstances.

2 Problem setting

We consider a list of "candidates" $\mathcal{C} = (\mathbf{c_1}, ..., \mathbf{c_n})$ representing models (or algorithms) to be evaluated, and a list of "judges" $\mathcal{J} = (\mathbf{j_1}, ..., \mathbf{j_m})$ representing the tasks to be solved by the candidates. A score matrix M of size $n \times m$ is obtained by scoring the performance of each algorithm on each task. It can be thought of as a competition leaderboard in a multi-task competition. The scoring method used to evaluate a given task (including choices of metric, data split, use of cross-validation, etc.) may vary from task to task, but all the algorithms should be scored using the same scoring method for a *particular* task. The problem we are addressing is to obtain a single ranking of candidates $\mathbf{r} = \texttt{rank}(f(M))$ from the score matrix, using a ranking function $f : \mathbb{R}^{n \times m} \to \mathbb{R}^n$. The fractional ranking function $\texttt{rank} : \mathbb{R}^n \to \mathbb{R}^n_+$ is defined as follows: $\forall i \in \{1, ..., n\}$, $\texttt{rank}(\mathbf{v})_i = 1 + \sum_{j \neq i} \mathbb{1}_{\mathbf{v}_j > \mathbf{v}_i} + \frac{1}{2} \sum_{j \neq i} \mathbb{1}_{\mathbf{v}_j = \mathbf{v}_i}$. We are looking for a ranking function f that performs well according to given criteria.

2.1 Ranking functions

Ranking functions associate a score to each candidate based on an aggregation of the score matrix M. We can then derive a ranking from this set of scores. In this work, the ranking functions under study are: *mean*, *median*, *average rank* and two methods based on *pairwise comparisons* of candidates. *Mean* and *median* are average judges, obtained by either taking the mean or median values over all judges, for each candidate.

Average rank is defined as following: $f(M) = \frac{1}{m} \sum_{\mathbf{j} \in \mathcal{J}} \texttt{rank}(\mathbf{j})$. It has the interesting property of computing a ranking which minimizes the sum of the Spearman distance with all the input judges [8].

Pairwise comparisons methods give scores based on comparisons of all pairs of candidates: $f(M) = \left(\frac{1}{(n-1)} \sum_{j \neq i} w(\mathbf{c_i}, \mathbf{c_j}) \right)_{1 \leq i \leq n}$

where $w(\mathbf{c_i}, \mathbf{c_j})$ represents the performance of $\mathbf{c_i}$ against $\mathbf{c_j}$. We can define different pairwise methods by designing different w functions:

- *Success rate*: $w(\mathbf{u}, \mathbf{v}) = \frac{1}{m} \sum_{k=1}^{m} \mathbb{1}_{u_k > v_k}$.
- *Relative difference*: $w(\mathbf{u}, \mathbf{v}) = \frac{1}{m} \sum_{k=1}^{m} \frac{u_k - v_k}{u_k + v_k}$.
- *Copeland's method*: $w(\mathbf{u}, \mathbf{v}) = 1$ if the candidate $\mathbf{u}$ is more frequently better than the candidate $\mathbf{v}$ across all judges, 0.5 in case of a tie, and 0 otherwise.

2.2 Theoretical criteria

We summarize in Table **??** the main theoretical results on ranking functions studied in this paper [7, 2], defined below. The first two relate to properties of the winning solution, criteria 3 and 4 relate to resilience to judge perturbation and the last two to resilience to candidate perturbation.

Majority criterion [12]: If one candidate is ranked first by a majority (more than 50%) of judges, then that candidate must win.

Condorcet criterion: The Condorcet winner is always ranked first if one exists. The Condorcet winner is the candidate that would obtain majority against each of the others when every pair of candidates is compared. The Condorcet criterion is stronger than the Majority criterion.

Consistency: Whenever the set of judges is divided (arbitrarily) into several parts and elections in those parts garner the same result, then an election on the entire judge set also garners that result.

Participation criterion: The removal of a judge, where candidate $\mathbf{u}$ is strictly preferred to candidate $\mathbf{v}$, from an existing score matrix, should only improve the final position of $\mathbf{v}$ relatively to $\mathbf{u}$.

Independence of irrelevant alternatives (IIA): The final ranking between candidates $\mathbf{u}$ and $\mathbf{v}$ depends only on the individual preferences between $\mathbf{u}$ and $\mathbf{v}$ (as opposed to depending on the preferences of other candidates as well).

Local IIA (LIIA) (weaker): If candidates in a subset are in consecutive positions in the final ranking, then their relative order must not change if all other candidates get removed.

Independence of clones (clone-proof): Removing or adding clones of candidates must not change the final ranking between all other candidates.

2.3 Empirical criteria

In order to compare ranking functions, we must specify desirable properties, with respect to our end goals. Theoretical properties are strictly binary in nature, either satisfied or not. However, a method which does not satisfy a property could, in practice, satisfy it in most of the case. To remedy this problem and have a more thorough comparison of methods, we propose empirical criteria.

The **average rank of the winner** is the average rank across all input judges of the candidate ranked first in $f(M)$. To obtain a score between 0 and 1 and to maximize, we normalize it using the following formula: $1 - \frac{\texttt{average rank}-1}{m-1}$.

The **Condorcet rate** is the rate of ranking the Condorcet winner first when one exist. This rate can be evaluated on a set of score matrices and is the direct empirical equivalent of the Condorcet criterion.

The **generalization** is the ability for a ranking function to predict the ranking of the candidates on new unknown tasks, which are not part of the set used for evaluation (of the benchmark or competition). $\texttt{generalization}(f) = \sum_{\mathbf{j} \in \mathcal{J}^{valid}} \frac{1}{m} \sigma(f(\mathcal{J}^{train}), \texttt{rank}(\mathbf{j}))$, where σ is the chosen rank correlation coefficient, and $\mathcal{J}^{train}$ and $\mathcal{J}^{valid}$ are two disjoint sets of judges taken from M. There exist many ways of computing rank correlation, i.e. the degree of consensus between two rankings. In this work, we have tried Spearman's ρ [4] and Kendall's τ [11] (more precisely Kendall's τ_b, accounting for ties [1]), and decided to stick to Spearman's ρ as the results were similar in both cases.

The **stability** of a ranking method f is the concordance of the output rankings it produces under some variability of the input. The stability against perturbation on $\mathcal{C}$ and the stability against perturbation on $\mathcal{J}$ can be estimated separately, by performing the variations either across the candidate axis[1] or across the judge axis respectively. The concordance measures the overall consensus between a set of rankings and can be computed the average of the rank correlations (typically using Spearman's ρ or Kendall's τ) between all $\binom{m}{2}$ possible pairs of rankings: $\texttt{stability}(f) = \frac{1}{m(m-1)} \sum_{i \neq j} \sigma(X_i, X_j)$, where X is a matrix whose columns are the rankings $f(M')$ produced on several variation M' of the score matrix M, and X_i is the i^{th} column of X.

3 Experimental results

3.1 Settings

The data used for the experiments are matrices of performance of machine learning algorithms on a set of tasks. We have collected such matrices, summarized in Table 1, from past benchmarks as well as from a new benchmark we have put in place involving 66 datasets and 13 algorithms from the AutoDL Challenge [9]. In this new benchmark, the datasets represent a wide variety of classification tasks on images, videos, time series, text and tabular data. The algorithms are automated pipelines designed to perform as best as possible on any classification

[1] When perturbing candidates, σ needs to align the input vectors before computing the correlation.

task, and most of the approaches are based on deep learning. The benchmark was computed in a systematic way with the same experimental environment for each training and testing.

In order to estimate the values of the empirical criteria, we used bootstrapping. The generalization score, average rank of winner and Condorcet rate are estimated on 10000 trials. Each trial is based on a new version of the matrix M sampled with replacement both on the candidate axis and on the judge axis (in the case of generalization, the validation set is constituted of the out-of-bag judges). For stability, we've separated two cases: one where bootstrap is done on the candidate axis and one on judge axis. We've done 10 re-run with 100 bootstraps for each. Note that, to compare the ranking functions, we compute their respective scores on the same bootstraps of the score matrix M.

	# Datasets	# Algorithms	Metric	W	Norm	Source
AutoDL-AUC	66	13	AUC	0.38	No	AutoDL [9]
AutoDL-ALC	66	13	ALC	0.60	No	
AutoML	30	17	BAC or R^2	0.27	Yes	AutoML [6]
Artificial	50	20	*None*	0.00	Yes	Authors of [13]
OpenML	76	292	Accuracy	0.32	Yes	Alors [10] website
Statlog	22	24	Error rate	0.27	Yes	Statlog in UCI repository

Table 1: Datasets-Algorithms matrices used in the experiments. ALC refers to the Area under the Learning Curve, on which each point is a ROC AUC over time. W is the concordance between the datasets, telling in some way how strong is the general consensus between them. "Norm" means that the matrix got globally standard normalized.

3.2 Results

	Theoretical properties							Empirical properties					Correlation matrix					
	Winner		Judge		Candidate			Winner		Judge		Candidate						
	Maj.	Condorcet	Consist.	Particip.	IIA	LIIA	Clone-proof	Winner rank	Condorcet rate	Generalization	Stability (judge)	Stability (candidate)	Mean	Median	Av. rank	Suc. rate	Rel. diff.	Copel.
Mean	0	0	1	1	1	1	1	0.68	0.4	0.36	0.753	1.000	1.00	0.81	0.75	0.74	0.63	0.74
Median	0	0	0	0	1	1	1	0.70	0.5	0.37	0.702	1.000	0.81	1.00	0.84	0.84	0.73	0.85
Average rank	0	0	1	1	0	0	0	0.74	0.8	0.41	0.780	0.954	0.75	0.84	1.00	1.00	0.88	0.97
Success rate	0	0	1	1	0	0	0	0.73	0.8	0.40	0.777	0.839	0.74	0.84	1.00	1.00	0.88	0.96
Relative diff.	0	0	1	1	0	0	0	0.73	0.8	0.41	0.884	0.941	0.63	0.73	0.88	0.88	1.00	0.83
Copeland	1	1	0	0	0	0	0	0.73	1.0	0.41	0.771	0.965	0.74	0.85	0.97	0.96	0.83	1.00

Table 2: Theoretical properties satisfied or not by the ranking functions (left) and estimated values of the empirical properties (center). The correlation matrix [...]. The results are averaged over all benchmarks. Error bars, etc. (TODO)

The empirical results are summarized in Table **??**. As can be seen, the empirical results are more nuanced than the theoretical results.

Mean and *Median* are, by nature, insensitive to candidate-wise perturbations (their individual candidate ratings do not involve comparisons with other candidates). Thus, not surprisingly, they satisfy the IIA, LIIA, and clone-proof criteria and they are empirically perfectly stable against candidate perturbations. However, their winner rates and generalization scores are poor in comparison

to other ranking functions. Thus performing cross-candidate comparisons may help good generalization, particularly if scores are not normalized well.

Average rank performs better than *Mean* and *Median* in every respect, except candidate stability (but it still does pretty well). It often criticized for enlarging small differences between nearby candidates and reducing differences between far away candidates. Also it often generates many ties. We do not consider it a bad ranking method because it is fast to compute and easy to understand, but it is not our favorite.

Copeland's method and relative difference outplay the others. By design, *Copeland's method* has a perfect Condorcet rate and it gets the best empirical candidate stability (even though it is not theoretically clone-proof). *Relative difference* closely approaches the performances of *Copeland's method* on these two criteria. But, *Relative difference* is much better than ALL other ranking functions in terms of judge-wise perturbation.

We performed side experiments on toy examples to understand the superiority of *Relative difference* over other methods in terms of judge-wise perturbation stability. One limit case is instructive. Consider two pairs of identical judges, the second two providing rankings in the exact inverse order as the first two. Furthermore, the scores equate the rankings. ALL methods (except *Relative difference*) return a n-way tie. *Relative difference* comes up with an ordering, which favors either (i) candidates with little variance in judges opinions (good generalists) or (ii) candidates with a lot of variance in judges opinions (good specialists). The former occurs when higher scores are better and the latter when lower scores are better. If one of the judges is suppressed, all methods (except *Relative difference*) drastically change their opinion in favor of the majority of judges (two vs. one having now the same opinion). *Relative difference* also changes its opinion, but it remains biased towards generalist or specialist candidates, and thus resembles more its previous opinion. Therefore, *Relative difference* enjoys more judge-wise perturbation stability.

4 Conclusion

Theoretical considerations hint that finding a ranking function that fulfills all desirable criteria is impossible. Our empirical evaluation on machine learning use cases reveals however that several ranking functions obtain a good compromise for the empirical criteria we defined, which mimic quantitatively theoretical properties. Of all ranking functions, the pairwise method that we baptized "relative difference" is our favorite among pairwise methods and "average rank" is our favorite among averaging methods. A distinct advantage of "relative difference" is its stability with respect to judge perturbation. It is explained by a bias towards choosing candidates for which judges have a lower (or higher) variance in opinion (depending on whether the score is "higher is better" or "lower is better"). Its candidate stability (particularly resistance to clones) is a little lower than that of Copeland's method, but in most practical settings, this may not be an issue (as clones can be avoided). Its computational complexity is not pro-

hibitive, therefore we recommend it. Further work includes studying variants of hybrids of the proposed ranking functions to improve on the various criteria and studying consistency and rate of convergence of the ranking functions, thought of as meta-learning algorithms.

References

[1] A. Agresti. *Analysis of Ordinal Categorical Data*. New York: John Wiley & Sons, 2010.

[2] G. Allan. Manipulation of voting schemes: A general result. *Econometrica*, 1973.

[3] P. Brazdil and C. Soares. A comparison of ranking methods for classification algorithm selection. In R. L. de Mántaras and E. Plaza, editors, *ECML 2000, Proceedings*, volume 1810 of *Lecture Notes in Computer Science*, pages 63–74. Springer, 2000.

[4] W. W. Daniel. Spearman rank correlation coefficient. In *Applied Nonparametric Statistics (2nd ed.)*, pages 358 – 365. Boston: PWS-Kent, 1990.

[5] B. Efron and R. J. Tibshirani. *An Introduction to the Bootstrap*. Number 57 in Monographs on Statistics and Applied Probability. Chapman & Hall/CRC, Boca Raton, Florida, USA, 1993.

[6] I. Guyon et al. *Analysis of the AutoML Challenge Series 2015–2018*, pages 177–219. Springer International Publishing, Cham, 2019.

[7] A. K. J. A difficulty in the concept of social welfare. *Journal of Political Economy*, 1950.

[8] M. Kendall and J. Gibbons. *Rank Correlation Methods.* 5th Edition, Edward Arnold, London., 1990.

[9] Z. Liu, A. Pavao, Z. Xu, et al. Winning solutions and post-challenge analyses of the ChaLearn AutoDL challenge 2019. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, page 17, 2020.

[10] M. Misir and M. Sebag. Alors: An algorithm recommender system. *Artif. Intell.*, 244:291–314, 2017.

[11] R. Nelsen. Kendall tau metric. In *Encyclopedia of Mathematics*. EMS Press, 2001.

[12] J. Rothe. *Economics and Computation: An Introduction to Algorithmic Game Theory, Computational Social Choice, and Fair Division.* Springer, 2015.

[13] L. Sun-Hosoya, I. Guyon, and M. Sebag. Activmetal: Algorithm recommendation with active meta learning. In *Proceedings of the Workshop on Interactive Adaptive Learning co-located with ECML 2018 and PKDD 2018, Dublin, Ireland, September 10th, 2018*, volume 2192, pages 48–59. CEUR-WS.org, 2018.

5 Supplemental material

Reference the supplemental material in the text. – Isabelle

Supplemental material can be found here:

`https://github.com/Didayolo/ranking-esann-2021`