Judging competitions and benchmarks: a candidate election approach

Adrien Pavao, Michael Vaccaro, and Isabelle Guyon
LISN/CNRS/INRIA, Université Paris-Saclay, France

Abstract. Machine learning progress relies on algorithm benchmarks. We study the problem of declaring a winner, or ranking "candidate" algorithms, based on results obtained by "judges" (scores on various tasks). Inspired by social science and game theory on fair elections, we compare various ranking functions, ranging from simple score averaging 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 prior theoretical analyses indicate that no single ranking function satisfies all desired properties, our empirical study reveals that the classical "average rank" method fares well. However, Condorcet methods get better empirical results.

1 Introduction

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 (ML) algorithms and "judges" are test performances on tasks to be solved (e.g. classification, regression or reinforcement learning problems). Organizers of competitions (or benchmarks) usually simply average scores or competitor ranks on the various tasks, to obtain a global ranking. 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 various ranking functions. Arrow [7] and later Gibbard [2] have shown that this problem is not trivial and that no known aggregation method can satisfy all desired properties. Such properties include that a candidate ranked first by a majority of judges must be declared winner, and that the ranking should be stable under perturbations of the set of judges and the set of candidates. Our goal is to determine whether, in spite of pessimistic theoretical predictions, some ranking functions offer a good compromise between all criteria on specific data. 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 and include meta-generalization as part of our criteria, to evaluate whether ranking functions identify algorithms that fare well on *future* tasks of a similar nature. This setting arises in particular in AutoML competitions (e.g. [6, 9]). We perform empirical evaluations on five competitions and benchmarks, 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 ($\mathcal{C}$ is the list of rows of M and $\mathcal{J}$ is the list of columns of M). 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 algorithms considered should be scored using the same scoring method for a *given* task. The problem we are addressing is to obtain a single ranking of candidates $\mathbf{r} = \mathtt{rank}(f(M))$ from the score matrix, using a ranking function $f : \mathbb{R}^{n\times m} \to \mathbb{R}^n$. The function $\mathtt{rank} : \mathbb{R}^n \to \mathbb{R}^n_+$ is defined as follows: $\forall i \in \{1, ..., n\},\ \mathtt{rank}(\mathbf{v})_i = 1 + \sum_{j\neq i} \mathbb{1}_{\mathbf{v}_j > \mathbf{v}_i} + \frac{1}{2}\sum_{j\neq i} \mathbb{1}_{\mathbf{v}_j = \mathbf{v}_i}$. We are looking for a ranking function f that performs well according to given criteria.

2.1 Ranking functions

Ranking functions associate a global score to each candidate based on an aggregation of the columns of the score matrix M. We can then derive a ranking from such global scores. In this work, the ranking functions under study are: *Mean*, *Median*, *Average Rank* and three 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 follows: $f(M) = \frac{1}{m}\sum_{\mathbf{j}\in\mathcal{J}} \mathtt{rank}(\mathbf{j})$. 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 2 (left) the main theoretical results on ranking functions studied in this paper, defined below. The first two relate to properties of the winning solution, consistency and participation criteria 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 rankings in those parts garner the same result, then a ranking on the entire judge set also garners that result.

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

Independence of irrelevant alternatives (IIA): The final ranking between candidates **u** and **v** depends only on the individual preferences between **u** and **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 cases. 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 exists. 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 $\mathcal{J}$ and the stability against perturbation on $\mathcal{C}$ can be estimated separately, by performing variations either across the judge axis or the candidate axis respectively. To measure the overall agreement between a set of q rankings resulting from perturbations, we compute an index of concordance by averaging correlations between $\binom{q}{2}$ pairs of ranking function outputs (typically using Spearman's ρ or Kendall's τ): $\texttt{stability}(f) = \frac{1}{m(m-1)} \sum_{i \neq j} \sigma(X_i, X_j)$, 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. When perturbing candidates, σ is restricted to the subset of matching candidates.

3 Experimental results

Data used for experiments are performance matrices of ML algorithms on a set of tasks in several *benchmarks* (Table 1). To estimate the values of empirical criteria, we use bootstrapping. The generalization score, average rank of winner, and Condorcet rate are estimated on $10,000$ repeat trials. Each trial is based on a new version of 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 criteria, we consider two separate cases: one where bootstrap is done on the candidate axis and one on the judge axis. We use $q = 100$ bootstraps yielding each a new version of matrix M (and a corresponding ranking), thus yielding $q(q-1)/2$ comparisons of pairs of rankings. The procedure is repeated 10 times. The code of the experiments is public[1] and based on the Python package Ranky[2].

	# 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 (DA) matrices used in the experiments. ALC refers to the Area under the Learning Curve, on which each point is a ROC AUC over time. The first DA matrices are only counted as one in standard error calculations (Table 2), because they come from the same benchmark. W is the concordance between datasets (as "judges") within a benchmark, evaluating their degree of agreement. "Norm" means that the matrix was globally standardized.

[1] `https://github.com/Didayolo/ranking-esann-2021`
[2] `https://github.com/Didayolo/ranky`

	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 of ranking functions (left), estimated values of empirical properties (center) and corresponding correlation matrix (right). Center results are averaged over bootstraps and all $b = 5$ benchmarks. Error bars computed as $\mathrm{std}(\text{bootstraps})/\sqrt{b}$ govern the number of significant digits.

Table 2 shows that empirical results are more nuanced than theoretical results. The following observations can be made:

Mean and *Median* are, by nature, insensitive to candidate-wise perturbations (ratings do not involve comparisons with other candidates). Accordingly, they satisfy 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, which make cross-candidate comparisons, and which are robust to non-normalized scores.

Average Rank performs better than *Mean* and *Median* in every respect, except candidate stability, but worse than *Copeland* and *Relative Difference*. It may be criticized for not taking actual differences between candidates into account and for generating ties. Thus, despite its simplicity, it is not our favorite.

Success rate should (in non pathological cases) provide the same ranking as *Average Rank*. However, unlike *Average Rank*, it is sensitive to introducing clones and gets a very bad candidate stability score. We see no advantage of it.

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 IIA and 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 such superiority of *Relative Difference* over other methods. 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* orders candidates to either favor little variance or large variance in judges' opinions. 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 with respect to judge variance, hence more faithful to its previous opinion, thus enjoying 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 "the higher the better" or "the lower the 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 prohibitive (polynomial in number of judges and candidates), 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 *ECML 2000*, Lecture Notes in Computer Science, pages 63–74, 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, 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 TPAMI*, 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 *Wshp on Interactive Adaptive Learning @ ECML-PKDD*, 2018.

5 Supplemental material

Supplemental material can be found here:

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

It includes:

- Detailed heatmaps of the Datasets-Algorithms matrices,
- Proofs of the theoretical criteria satisfied or not by *success rate* and *Relative Difference* methods,
- Formulas of Kendall's τ and Kendall's W (a computationally efficient concordance measure),
- Some scores detailed for each benchmark and interesting plots,
- Discussion about *Optimal Rank Aggregation* methods, especially *Kemeny-Young* method (not presented here).