A Bayesian Bradley-Terry model to compare multiple ML algorithms on multiple data sets

Jacques Wainer WAINER@IC.UNICAMP.BR
Artificial Intelligence Lab, RECOD.ai
Institute of Computing
University of Campinas
Campinas 13083-970, Brazil.

Abstract

This paper proposes a Bayesian model to compare multiple algorithms on multiple data sets, on any metric. The model is based on the Bradley-Terry model, that counts the number of times one algorithm performs better than another on different data sets. Because of its Bayesian foundations, the Bayesian Bradley Terry model (BBT) has different characteristics than frequentist approaches to comparing multiple algorithms on multiple data sets, such as Demsar (2006) tests on mean rank, and Benavoli et al. (2016) multiple pairwise Wilcoxon tests with p-adjustment procedures. In particular, a Bayesian approach allows for more nuanced statements regarding the algorithms beyond claiming that the difference is or it is not statistically significant. Bayesian approaches also allow to define when two algorithms are equivalent for practical purposes, or the region of practical equivalence (ROPE). Different than a Bayesian signed rank comparison procedure proposed by Benavoli et al. (2017), our approach can define a ROPE for any metric, since it is based on probability statements, and not on differences of that metric.

This paper also proposes a local ROPE concept, that evaluates whether a positive difference between a mean measure across some cross validation to the mean of some other algorithms is should be really seen as the first algorithm being better than the second, based on effect sizes. This local ROPE proposal is independent of a Bayesian use, and can be used in frequentist approaches based on ranks.

A R package and a Python program that implements the BBT is available.

Keywords: Bayesian, Bradley-Terry model, Comparison of classifiers, Comparison of regressors, Multiple data sets, Multiple algorithms

1. Introduction

Almost any research in Machine Learning that proposes a new model or a new algorithm it will compare with some previous research, likely on multiple data sets. Machine learning algorithms are measured using different metrics; to list a few in classification tasks: accuracy, Cohen kappa, balanced accuracy, AUC (area under the ROC curve), F1 , MCC (Matthew's correlation coefficient), AP (average precision or area under the PR-curve), g-means, Bier score, and so on. For regression tasks: RMSE (root mean squared error), MAE (mean absolute error), R^2 (coefficient of determination) and adjusted R^2, MAPE (mean absolute percentage error), maximum residual error, etc. There are other metrics for hierarchical classes (Kosmopoulos et al., 2015), for multi-label problems (Zhang and Zhou, 2014) and metrics for NLP generation (Celikyilmaz et al., 2020), multi-objective optimization (Riquelme et al., 2015), among a myriad of evaluation metrics for other different tasks.

It is commonly understood that one should not aggregate measures on different data sets. That is, it is commonly understood that a 0.76 accuracy in one data set and a 0.96 accuracy in another data set should not be aggregated together, to compute, for example, the mean accuracy of 0.86 on the two data sets. The intuition seems to be that the same gain in accuracy *does not mean the same* on different values of the base accuracy; an improvement of the accuracy from 76% to 78% is much less "impressive" or "significant" than a gain from 96% to 98%, although in both cases the gain is of 2 percentage points. Therefore two algorithms, one with with 78% and 96% accuracy in both data sets, and the other with 76% and 98%, have the same mean accuracy, but the second seems "better" than the first, since the second algorithm's gain of 2% on the second data set is more "significant", or more "impressive" than its 2% loss on the first data set!

However, metrics like accuracy, and AUC, and others, are at least **comparable**, in the sense that one can say when one measure in one data set is better than the measure on another data set. If algorithm A achieved an AUC of 0.78 in one data set and an AUC of 0.86 on another data set, one can claim that the second measure (0.85) was better than the first – a statement so obvious that it is odd to see it written down. And if the comparison procedure needs to order the measures for each algorithm in order to compute the median value or the mean value (which, as discussed, is not considered a "correct" step) one can do it. There are measures, specially in regression tasks, that are **not comparable** across different data sets. For example, is an RMSE of $30'000,00 on predicting housing values of Boston suburbs better or worse than a RMSE of 3.3 on predicting the quality of red wine (a number between 0 and 10)?

The comparable property is important because some comparison strategies, those based on pairwise Wilcoxon tests (Benavoli et al., 2017, 2016; Stapor et al., 2021), compute for each algorithm the median of the measures on the different data sets and then compare these medians, but to compute the median one must have a comparable metric across different data sets. If the $30'000,00 and the 3.3 are RMSE errors from the same algorithm, how would one rank them in order to compute the median RMSE of this algorithm across the different data sets?

The goals of a comparison procedure are in order of importance:

1. The procedure should tell which algorithm is better, which is second place, which one is third place and so on when measured on a particular set of data sets. We will call this the **aggregated ranking** of the algorithms for that set of data sets.

2. The procedure should compute how confident or how hopeful one should be that the ordering will remain true when one tests the same algorithms on a different data set.

3. The procedure should state how much one algorithm is better than another and specially when one algorithm is not much better than another one, that is, when both algorithm have similar performances or when they are equivalent.

4. The procedure should not require that all algorithms must be evaluated in all data sets.

There are many alternatives to compute the aggregated ranking. The usual one is to compute each algorithms' rank within each data set: 1 for the best, 2 for the second, and

the average rank in case of ties, and compute the **mean rank** for each algorithms. The ranking of the mean ranks is the final aggregated rank. Another alternative, which has been justified in the literature (Benavoli et al., 2016; Stapor et al., 2021), is to compute the median measure for each algorithm (provided the metric is comparable) and the aggregated ranking is the rank of the medians. There are also numerous alternatives for ranking aggregation discussed in other disciplines: ranking are called preferences in social choice theory and there is a large set of voting procedures to aggregate preferences (List, 2022). But the aggregation procedures proposed in other disciplines usually are not associated with some measure of how confident one can be that a particular item is really "better" than another one in the aggregated rank.

The second goal (evaluation of how confident one should be about the aggregated ranking) is needed if only a sample of the relevant the population was measured and ranked. In case of deciding the winners of an election, or the winners of a sport championship, one is dealing with the whole population. There is no concern on whether the senator who won the seat, or the team that won the championship had a "statistically significant" victory. The point of the comparison procedure is just to compute the aggregated ranking. But when comparing algorithms for machine learning, one is never interested on the raw aggregated ranking on the set of data sets used in the comparison; instead, one is interested on claims that can be made about future data sets regarding the ranking of the algorithms.

The step of computing the confidence on the aggregated ranking is usually performed using statistical tests that produce for each pair of algorithms in the aggregated ranking a binary decision or a continuous measure on whether one can trust that algorithm A is better than algorithm B or not, for example. Historically, the more common approach is the use of a null hypotheses significant tests (NHST), also called a **frequentist** approach. Frequentist approaches aim at the binary decision on whether one should accept or not each pair comparison. The usual statement "the difference is statistically significant" indicates that one should be confident on that pair comparison. More recently, there has been a move towards a **Bayesian** approach to statistical tests which will not return a binary decision on accepting or not a pair comparison in the aggregated ranking, but define a probability that "A is better than B".

Regarding the third goal, there is a clear interest in stating that one algorithm is similar to another for practical purposes. The usual frequentist methods cannot make this claim[1]; some researchers wrongly assume that if there is no statistical difference between two algorithms, they are similar or equivalent. A non significant difference in frequentist test only indicates that the sample size was not large enough to find the statistical significance. With a large enough sample size, all p-values go to 0 (Shalizi, 2015; Kruschke and Liddell, 2015) and thus all differences become significant. On the other hand, Bayesian methods do allow one to make claims of practical equivalence as we will see below (section 2.3).

Finally, the fourth goal: it is desirable not to require that all algorithms run all data sets. Algorithms may not converge for some cases, may demand more memory than it is available, or may exceed the allotted computational time to perform the tests. In these cases there will be no corresponding metric for that algorithm for that data set. The comparison

1. There are frequentist tests that determine when two alternatives do not have practical difference. These are known as equivalence tests [@eqvtest] but are not of common use in Machine Learning or other areas of Computing.

procedure should handle this gracefully. For frequentist approaches, there is no clear or agreed upon way to deal with missing measures, as we will discuss below.

This paper proposes a new comparison procedure based on the number of times an algorithm is better than another on the different data sets. In more details:

- The statistical framework for our proposal is the Bradley-Terry model for ranks, which assumes that each item being compared, algorithms in this case, has a latent "merit number" or "ability", and that the probability that one item is better than another is a function only of the two items' merit numbers.

- The aggregated ranking of the algorithms is the ordering of these merit numbers.

- We propose a Bayesian implementation of the Bradley-Terry model. The Bayesian implementation will allow to compute probabilities that one algorithm is better than another, and that is the measure of one's confidence in the ordering.

- Because we are using a Bayesian model we can define when two algorithms are equivalent for practical purposes. In Bayesian testing, the range of values for which one claims there is no practical differences is called a region of practical equivalence or ROPE.

- Our proposal of ROPE is in the probability space (a claim on probabilities) and not on the metric space (a claim on differences of accuracy or differences of RMSE). This allows our notion of ROPE to be generic across different metrics, and perfectly understandable (and modifiable) by different researchers even in the case they do not have experience on comparing algorithms on that particular metric.

- We defined a concept of local ROPE, a decision criteria when comparing to algorithms on a particular data set, based on the fold data for the two algorithms, that decides whether one algorithm is better than another. This decision is not just based on the difference between the two mean measures, but takes into consideration the "noise level" or formally, the effect size of the differences.

This paper is laid out as follows: Section 2 is a short tutorial on frequentist tests for multiple comparisons in general, and Bayesian tests. A reader that knows the basics of frequentist test can skip section 2.1; a reader that understands p-value adjustments in multiple comparisons can skip Section 2.2. Section 3 discusses the previous frequentist and Bayesian approaches to comparing multiple algorithms on multiple data sets. Section 4 discusses this proposal, the use of a Bayesian Bradley Terry model. Section 5 shows some first results of using the BBT model. Section 6 discusses our proposal of considering more than the mean across different cross-validations when determining if one algorithm is better than the other. Section 7 compare the results of the BBT model with the previous approaches. Section 8 discusses the quality of the BBT model as a predictive estimation of the future behavior of the algorithms on new data sets. Section 9 discusses some of the advantages and shortcomings of the BBT model, and section 10 summarizes the main conclusions.

2. A short tutorial on frequentist and Bayesian multiple comparisons

Let us assume that different algorithms tested on multiple data sets are measured using some metric for which calculating the average is an acceptable procedure (which is not the usual case in Machine Learning results, as discussed above), and let us also assume that many data sets were used in the comparison. These two assumptions are necessary for this tutorial so: a) we can use a **parametric** frequentist test to discover which algorithms are significantly different from each other and b) use a **Gaussian** based Bayesian modeling of the problem.

2.1 General aspects of frequentist approach

The frequentist approaches, also known as **NHST** or null hypothesis statistical testing, will select a few statements of the form "the mean of results for algorithm A is different than the mean of results for algorithm B", or more succinctly that "algorithm A is different than algorithm B" and claim that *they are true* (but not in these terms), and for the other possible statements, it will claim that it cannot decide whether the statement is true or not true. More formally, NHST will indicate that a difference between two algorithms or groups is true by the statement "the difference between algorithm A and B is *statistically significant*". The test will indicate that a difference cannot be shown to be true or false by the statement "the difference between algorithms A and B is *not* statistically significant".

Internally, most frequentist test models the problem by assuming that there is no difference between the populations that *generated* the samples, and compute an approximation of the probability that the unique source of data for all groups would generate (by chance) samples that have mean values as different, or more different, than the ones encountered in the real data. This approximation of the probability is called **p-value** and the standard in Machine Learning and most other Science areas is is that if p-value ≤ 0.05 then one claims that the null hypothesis (that all data came from a single population) is false. Therefore, that the "differences between the algorithms are real".

For a statistical test, a **type 1** error is the test claiming that the difference between two sets of measures is "real" (or more precisely, the test claims that there is enough evidence to state that there is a difference) when the difference is not real - a false positive error. A **type 2** error is the test claiming that the difference is not real when "it really is" - a false negative error.

Usually, the probability of a type 1 error, that is, the probability of a false positive error, or the probability of making a false positive claim is denoted by α. In the Sciences, one is very concern that false claims will not be made, and thus one sets α to 0.05 or lower, which is the threshold of 0.05 for the p-value. The number $1 - \alpha$ is called the **significance**, and a frequentist test which uses p-value ≤ 0.05 will state that the difference between groups A and B is statistically significant with 95% significance. The probability of a type 2 error is denoted by β - the probability of a false negative error, or the probability of missing a true claim. The number $1-\beta$ is called **power** of a test. There is no generally scientific standards for power although some textbooks when teaching experiment planning or experimental design mention that one should aim at a power of 80%.

2.2 Frequentist test - ANOVA plus post-hoc tests

When comparing the mean of multiple groups, the standards frequentist approach is to perform an **omnibus** test first, followed, if needed, by some **post hoc** tests. The omnibus test assumes as null hypothesis that all groups came from the same population or distribution. If the p-value of this test is low enough, then the conclusion is that not all groups came from the same population, but this does not tell us which groups are different from each other. This will be the role of the post-hoc tests. In parametric multiple comparison procedures, the omnibus test is called ANOVA (Analysis of Variance) or repeated measures ANOVA, depending on whether the data is not-paired or paired respectively.

Paired data refer to situations where there is a correspondence among each data items in each of the groups. For example, each algorithm was tested on the same set of data sets, and thus one can make the correspondence of that result in each algorithm to one particular data set. Usually, in machine learning, one deals with paired data, but that is not necessary. The alternative to paired data is called *independent* samples, or *non-paired* data[2]. Statistical tests differ for paired and non-paired data, but one can always use a non-paired test on paired data, but usually with some decrease in power - some differences that would be significant with the paired test may not be so using the non-paired test.

Once the comparison passes the omnibus test, one performs the post-hoc test, which verifies which of the claims "group A is different than group B" has sufficient evidence. There are different ways of classifying post-hoc tests, but for the purpose of this paper, the post-hoc test may compute a single critical difference, for example Tukey's range test. If the difference between two group means is larger than this critical difference, then one can claim that the difference between the two groups is statistically significant, otherwise, the difference is not statistically significant. The alternative for critical difference post-hoc tests is to compute pairwise p-values, that is, to compute a separate p-value for each pair of groups. In the case of multiple p-values, the post hoc test must also adjust each of the p-values, using some (statistical) **multiple comparison** or **p-value adjustment** procedure.

Statistical multiple comparison procedures is a complex topic in frequentist statistics (Feise, 2002; Rothman, 1990; Bender and Lange, 2001). We refer the reader to (Farcomeni, 2008; Garcia and Herrera, 2008) for a deeper understanding of the issue and the techniques. But in general terms, let us assume that each pairwise test uses the 0.05 threshold for p-value (this is called the comparison-wise or individual error rate). If one makes k *independent* tests, and summarizes them into a single statement "T_1 and T_2 and T_3 and $\dots T_k$" the probability that any one of the sub-statement (T_i) is correct is $1 - 0.05$. If we assume that the tests are independent, the probability that all sub-statements are correct is $(1-0.05)^k$, and therefore, the probability that the statement as a whole is false is $1 - (1 - 0.05)^k$. This is called the family-wise error rate (FWER). If there are 100 tests, the FWER is 0.994, instead of the 0.05 one would expect from each of the individual tests! If one wants the family-wise error to be 0.05 one must use a much smaller individual error rate. On first approximation, if the individual error rate is α, and if one wants the FWER to be 0.05, then:

2. In older statistical texts, or texts used for other scientific areas, the paired situation is also called *within-subjects*, while the non-paired is called *between-subjects*.

$$\begin{aligned} 1-(1-\alpha)^k &= 0.05 \\ 1-(1^k - k1^{k-1}\alpha + O(\alpha^2)) &= 0.05 \\ k\alpha &= 0.05 - O(\alpha^2)) \\ \alpha &= \frac{0.05}{k} \end{aligned}$$

The calculations above reflects what is known as the Bonferroni test or Bonferroni correction. The term $O(\alpha^2)$ indicates that there is a term with α^2, another with α^3 and so on. Since α is small, this calculation assumes that the $O(\alpha^2)$ term is very small and does not change the results in any important way. Therefore, to maintain a FWER of 0.05, when $k = 100$ each individual test should be accepted if its p-value is less than 0.0005, that is 0.05/100. Statistical multiple test procedures are usually expressed in terms of adjusting or modifying the original p-values from each individual test so that one will still use the usual 0.05 threshold. In the example above, the Bonferroni procedure would multiply each of the original p-values by 100 (limiting the product to 1.00 since p-value is probability), and if any of the adjusted p-value is lower than 0.05 then one accepts that comparison as statistically significant.

There are problems with the assumptions for the Bonferroni correction:

- Not all tests are independent especially if one is comparing n different algorithms (or any items). In this case the k are all pairwise comparisons, that is $k = \frac{n(n-1)}{2}$. If one test determines that algorithm A is better than B and B is better then C, then the third test, which compares A and C is fully determined, and it is not independent of the others. If, on the other hand, a test determines that A is better than B and C, the third test which compares B and C is indeed independent of the other two.

- Not all k tests will return that a particular comparison/difference is statistically significant. If, in the last example, the third test determines that there is not enough evidence to claim that B is better than C, the final claim will be "A is better than B and A is better than C". In this case, only the result of two tests were incorporated into the final claim.

The effective k one should use in the Bonferroni correction is lower than the total number of tests and therefore the threshold p-value is lower "than it should be." This increases the type 2 error of the procedure - some statements will not achieve the low p-value required by the Bonferroni correction, but would have low enough p-value if the correct k was used.

Beyond the Bonferroni procedure, there are dozen of different p-value adjustment procedures, among them, Holm, Hochberg, Hommel, Benjamini and Hochberg, and Benjamini and Yekutieli (R core, 2022). There is no clear guidelines to select one or other p-adjustment procedures, but all are more powerful than Bonferroni.

Because of the dependence on the number of comparisons, frequentist approaches make a distinction between two goals of a multiple comparisons. The first, and probably more common case, which is called **comparisons against a control**, is the situation that a research proposed a new algorithm and the researcher wants to compare that algorithm,

known as *the control*, against others which are considered as competitors. The goal of the comparisons is to show that the control is better or worse than each of its competitors; and the internal order among the competitors is not needed, nor it is reported. The second goal, called **all pairwise comparisons**, reflects a case where the researcher wants to rank a set of algorithms on multiple data sets and all pair comparisons are of interest and are reported. When one is comparing a set of algorithms against a control, much less comparisons must be made, which will require a less severe adjustment of the p-values.

2.3 Bayesian tests

All Bayesian tests (also known as **Bayesian estimation procedures**) compute the posterior join distribution of some parameters of the model that are important for the analysis. The simpler case to discuss is a Bayesian version of the two samples (non paired) t-test for the means. A simple Bayesian test will model each set of data (X and Y) as being samples from two Gaussian distributions with mean μ_X and μ_Y and with standard deviation σ_X and σ_Y respectively. The model also defines that μ_X and μ_Y are themselves sampled from another Gaussian with mean μ and standard deviation σ, and that σ_X and σ_Y are sampled from a uniform distribution between L and H. A more complex model of the same problem was proposed by Kruschke (2013). This is expressed by the following notation:

$$\begin{aligned} x_i &\sim \text{Normal}(\mu_X, \sigma_X) \\ y_i &\sim \text{Normal}(\mu_Y, \sigma_Y) \\ \mu_X &\sim \text{Normal}(\mu, \sigma) \\ \mu_Y &\sim \text{Normal}(\mu, \sigma) \\ \sigma_X &\sim \text{Unif}(L, H) \\ \sigma_Y &\sim \text{Unif}(L, H) \end{aligned} \tag{1}$$

The variables of interests are called **parameters** and in this case are μ_X and μ_Y (and maybe σ_X and σ_Y).

The distributions Normal(μ, σ) and Unif(L, H) are sometimes called **hyper-priors**. The **hyper-parameters** μ, σ, L, and H are set externally so that the distributions for the true data X and Y are likely. For example, if the measured average of the data in X is 5 and the measured average of the data in Y is 5.2, then the random variables μ_X and μ_Y should be around 5, and thus the mean of the Gaussian from which both μ_X and μ_Y are sampled should have mean (the μ hyper-parameter) of 5.

The width of the hyper-priors, the hyper-parameters σ, L, and H, and even the hyper-prior distributions themselves, are constraints on the possible values of the parameters. If these distributions are too narrow, the parameters will likely remain around a limited range of possible values, and the result of the Bayesian estimation may be due more to these constraints than the data itself, which is undesirable. If these hyper-parameters are too large, they may impose little or too little constraints on the parameters and may have impacts on the convergence of the MCMC algorithm. A broader version of this discussion is named the non-informative vs weakly informative vs strongly informative priors (Lemoine, 2019; Gelman et al., 2017).

In general terms, a Bayesian test will compute (or sample, as we will see below) the posterior distribution of the parameters of interests, in this case μ_X and μ_Y given the data. That is, we want to compute:

$$P(\mu_X, \mu_Y|M, X, Y) = K \;\; P(X, Y|M, \mu_X, \mu_y)P(\mu_X, \mu_Y|M)$$

where X and Y are the data, and M is the model itself (equations 1), and K is a constant that normalizes the distribution $P(\mu_X, \mu_Y|...)$ so it adds to 1 (or whose integral is 1). Given the distribution $P(\mu_X, \mu_Y|M, X, Y)$ one can compute, for example, $P(\mu_X > \mu_Y|M, X, Y)$, that is, the probability that the mean of the Gaussian distribution that *generated* X (Gaussian distribution from which X was sampled) is larger than the mean of the distribution that generated Y. If probability is 0.8, then the researchers are 80% sure that X comes from a population whose mean is greater than the population from which Y comes from, and they can report that. This claim is different than the claim that researchers using a frequentist approach can make.

2.3.1 ROPE

Another useful statement to derive from the posterior distribution is the probability that the difference between μ_X and μ_Y is of no practical consequence. If δ is the value below which the difference is of no practical consequence, then the value of $P(|mu_X - mu_Y| < \delta|M, X, Y)$ is the probability that there is no important difference between μ_X and μ_Y. The value δ in Bayesian analysis is called **region of practical equivalence** or **ROPE**. Differences smaller than the ROPE are considered as differences that do not matter for practical purposes.

2.3.2 MCMC and convergence

Usually, Bayesian tests do not compute the probability distribution $P(\mu_X, \mu_Y|M, X, Y)$ analytically, but use some instance of the MCMC family of algorithms to sample tuples $\langle \mu_{X_j}, \mu_{Y_j} \rangle$ from that distribution, indicated as $\langle \mu_{X_j}, \mu_{Y_j} \rangle \sim P(\mu_X, \mu_Y|M, X, Y)$. In the case of a sample set of $\langle \mu_{X_j}, \mu_{Y_j} \rangle$, computing the probability that $\mu_X > \mu_Y$ is just counting the proportion of the samples for which $\mu_{X_j} > \mu_{Y_j}$.

MCMC algorithms will converge on the limit to the target distribution, but in practice the MCMC algorithm will run for a pre-determined number of steps, and the convergence diagnostics allow the user to assess whether the samples returned are "good samples" from the target distribution. We refer the reader to Roy (2019) for a comprehensive discussion on convergence diagnostics.

Convergence diagnostics are necessary every time one runs a Bayesian model. The diagnostic informs users that the set of tuples they receive from the algorithm are indeed samples of the posterior distribution of the parameters or that the user may need to run more steps of the MCMC algorithm, or even redefine their Bayesian model.

2.3.3 Posterior Predictive Check

Posterior predictive diagnostics attempts to measures how well the Bayesian model (in our case the model proposed in Equation 2) models the data itself.

If the MCMC converges, it will correctly return tuples sampled from the posterior distributions of the parameters, but if the model itself does not correctly describe the data, the posterior distributions of the parameters will be less useful than expected.

The goal of the posterior predictive check (PPC) is to verify that the data generated by the model (given the posterior values of the parameters) is similar to the observed data. In other words, when the parameters assume the "correct values" (the posterior distribution of the parameters), one can "run the model forward", in the case above compute a new value of x_{ik}.[3] The distribution of, for example x_{1k}, that is, all the generated data for the first x data should be compared with "the real data" x_1. If the model is appropriate to the problem, the real data should be very similar to the set of generated data.

There are other forms of evaluating how well a model fits the data based on approximations of the leave-one-out cross validation, among them, WAIC (Watanabe-Akaike information criteria) (Watanabe and Opper, 2010) and loo (Vehtari et al., 2017). We will not go into details regarding these approaches but we will notice they are useful when comparing different models among themselves. We will use them further below when discussing alternative modeling.

2.3.4 Bayesian tests for multiple comparisons

We have discusses above a particular Bayesian model to compare **two** sets of data. If one performs all pairwise comparisons using some frequentist test, that would require using some p-adjustment procedure. But it is yet unclear whether there is or not a problem in performing multiple Bayesian comparisons. It has been claimed that if the Bayesian model is hierarchical/multilevel there is no problem in performing multiple comparisons (Gelman et al., 2012). Hierarchical models are similar to the line 1 in the Bayesian model described above where the two parameters are sampled from a single distribution. In a hierarchical model with multiple comparisons, one would need an equation like 1 for each of the parameters of interest $\mu_X, \mu_Y, \mu_W, \ldots \mu_Z$, all sampled from the *same* distribution. This will pool the different estimates of μ_i towards each other, what is called *partial pooling* or *shrinkage*. One such hierarchical model is the Bayesian ANOVA (Kruschke, 2014, ch. 19).

2.4 Frequentist versus Bayesian approaches

There are many differences between frequentist and Bayesian approaches. We will not discuss them in this paper, but we point the reader to Benavoli et al. (2017) discussion on the limitations of the frequentist approach.

3. Previous Frequentist and Bayesian approaches

In this section we will discuss the previous approaches to comparing multiple algorithms on multiple data sets.

3. This statement is a simplification in order to provide the reader with an intuitive understanding of the process. Formally, the same MCMC algorithm also samples from $P(X_{rep}, Y_{rep}|M, X, Y)$ where X_{rep} is the data generated from the model (M) given the real data (X and Y).

3.1 Demsar's procedure (mean rank plus Nemenyi test) and extensions

The most standard frequentist procedure for comparison of multiple algorithms on multiple data sets is the one defined by Demsar (2006). The main steps of this comparison procedure are:

- each data set results are converted to ranks, 1 for the best result, 2 for the second best results;
- ties are treated as average rank. If two algorithms have the same measure on a data set and they are the fourth best ranked algorithms for a data set, then the two algorithms receive the rank of 4.5, the average of 4 and 5, and the next algorithm receives rank 6;
- the final order is the decreasing order of the mean rank across all data sets. That is, an algorithm with a lower mean rank is *better* than an algorithm with a higher mean rank;
- the degree of confidence in this order is given by a pairwise binary statement on whether one should be confident or not that the algorithm i is better than algorithm j. The statement that one should be confident on each pair comparison is indicated by the usual phrase in NHST that "the difference *is* statistically significant";
- the computation of the pair comparisons significance is accomplished by first applying the Friedman test (the omnibus test) and then applying the Nemenyi test which returns a critical difference on the mean rank; if the difference on the mean ranks of two algorithms is smaller than the critical difference, then the difference between the two algorithms is *not* statistically significant.

Demsar (2006) also discusses a different comparison procedure when comparing against a control.

Garcia and Herrera (2008) proposed and tested different extensions to Demsar's procedure. The extensions use different post-hoc procedures: Shaffer's static and Bergmann-Hommel's procedures. The authors claim that these post-hoc procedure are stronger than the Nemenyi test. Garcia et al. (2010) also proposed new omnibus tests, the Friedman aligned ranks and Quade tests, and tested other post-hoc procedures for p-value adjustments, and concluded that the procedures Holm, Hochberg, Hommel, Holland and Rom produce equivalent results.

Demsar (2006) and extensions can be summarised as a family of *non-parametric* and *paired* multiple procedures computed using the rank of the algorithms within each data set. The omnibus procedure is Friedman (Demsar, 2006) or others (Garcia et al., 2010), and the post-hoc procedures can be based on critical difference on the ranks (Demsar, 2006; Garcia and Herrera, 2008) or can be even Wilcoxon pairwise procedures on the rank data followed by the many alternative p-adjustment procedures (Garcia et al., 2010).

3.2 Pairwise Wilcoxon plus p-value adjustment procedures

Benavoli et al. (2016) discuss that comparison procedures based on mean ranks have a problem: if the comparison of the algorithms A, B, C, D, and E yield that the differences

between A and B are significant, changing the other algorithms to say F, G, and H may result in a non-significant difference between A and B. That is, the result of the comparison of two algorithms depend on the other ones being compared. They suggest using a pairwise Wilcoxon signed rank test between the metrics obtained by each algorithms for all data sets, followed by an appropriate multiple comparisons adjustment procedure. This procedure is also suggested by Stapor et al. (2021).

Of relevance for this paper is that this approach will not create an aggregated ranking based on the mean rank of each algorithm across the data sets, but an aggregated ranking based on the median measure of each algorithm across the data sets. Of course, such procedure require comparable metrics. The distinction between comparing against a control and all pairwise comparison is trivially implemented in this approach.

Before we advance to Bayesian approaches, let us discuss the issues regarding missing data in the frequentist approaches discussed so far - the cases where an algorithm did not run for some data set. The problem is that there is no standard way of dealing with the missing data. For mean rank approaches such as Demsar's procedure one could not compute the rank for the algorithm that did not run for a particular data set, and not use this missing rank when computing the mean rank, but that would imply that not running on a data set would attribute to that algorithm its mean rank. This is possible but it is not standard. For example Fernandez-Delgado et al. (2014) did something similar but that procedure was criticised by Wainberg et al. (2016) as adding a positive bias to the algorithm that did not run on the data set. One could do the same for the pairwise Wilcoxon tests, but again that would mean that an algorithm that did not run on a data set receives its median measure for that data set. We do not have an opinion on whether this is the correct thing to do, or if one should assign to the algorithm the worst rank; the problem is that there is no agreed upon way of dealing with missing data for frequentist tests.

3.3 Bayesian pairwise signed-rank test

Benavoli et al. (2017) developed two forms of a Bayesian version of the Wilcoxon signed rank test (BSR), and argued for their use within the Machine Learning practice. The first form uses a single measure per algorithm and per data set, usually the mean of different measures using some form of cross-validation, a standard approach in Machine Learning. The second, more complex, form (called Bayesian hierarchical correlated t-test further expanded in Corani et al. (2017)) uses each measure from each of the cross-validations (which we will call a *fold* below regardless if k-fold is being used as the cross validation procedure) into the computation, but they suggest that practitioners should use the simpler form.

The model's parameter is the mean of the pairwise difference between of the accuracy of the algorithms on the different data sets, and the model computes a distribution of probability for this parameter. The authors state that if this parameter is between -0.01 and 0.01 (or a ROPE of 1%) there is no practical difference between the two algorithms being compared. The justification for the 1% ROPE *for accuracy* is not presented in the paper, but that number is not too different than the ROPE threshold proposed by Wainer (2016), but in this case based on some empirical evidence, and twice as large than the one proposed by Wainer and Cawley (2021), based on different set of empirical evidence.

On the data level, as we discussed above, a 1% of difference may or may not be important, depending on the accuracy itself. A 1% change for a 79% accuracy is likely insignificant but a 1% change for an accuracy of 98% is impressive. Given the range of accuracy values that appear in practical cases of comparing two classifiers (some high, some low) the authors are claiming that changes on the *mean value* of less than 1% are irrelevant from a practical point of view.

Even if one accepts the 0.01 ROPE for accuracy, there is no agreed upon, or even proposed (as far as this author is aware) ROPEs for AUC, F1, MCC, and other classification metrics and for comparable regression metrics. And, of course, there will be no ROPE for incomparable metrics, nor will the Bayesian signed rank method be applicable to incomparable metrics, given that the mean of the pairwise differences is not conceptually well defined (what is the mean of a difference of $7400,00 and a difference of 1.3 points in the RMSE of two algorithms?).

The Bayesian signed rank test was defined for comparing *two* algorithms on multiple data sets, and thus there may be problems in applying it to multiple comparisons. Benavoli et al. (2017) do not perform any multiple comparison in the paper. They do perform many signed-rank procedures with different algorithms but not with the goal of ranking those algorithms. At the end of the paper, they acknowledge that the Bayesian signed rank model does not have an hierarchical component that would allow its unproblematic use in multiple comparisons. But they use an argument put forth by Kruschke (2013) that the use of ROPE would mitigate the false alarm rate (making false positive claims). This is not a fully accepted argument, and even if it is correct, that would *require* the use of ROPE in any comparison. Since Benavoli et al. (2017) only proposes a ROPE for accuracy, unless other proposals for ROPEs for other metrics are advanced, using BSR for other metrics would be imprudent.

4. Bradley-Terry model

The Bradley-Terry (BT) model (Bradley and Terry, 1952) is mainly used to rank "players" in "tournaments" where the payers compete pairwise in matches, for example, soccer teams or chess payers. The BT model attributes to each player X_i an *intrinsic value* or *ability* w_i. The intrinsic value relates to the probability that player i will win player j in a match by:

$$P(X_i \text{ wins } X_j) = P(X_i \succ X_j) = \frac{w_i}{w_i + w_j}$$

The final ranking of the players is defined by the rank of their intrinsic values w_i. We will use the notation $i \succ j$ to indicate that X_i wins X_j. In Machine Learning "wining" depends on the metric. If the metric is accuracy or AUC, the highest value wins; if the metric is RMSE or other error metrics, or execution time, or energy consumption, the lower value wins.

The intrinsic values are invariant to a multiplicative constant, that is, if the set $\{w_i\}$ correctly models all the probabilities $P(a \succ b)$, so will $\{\alpha w_i\}$. Therefore, to specify a single set of intrinsic values, one also requires that $\sum w_i = 1$.

An alternative to the w_i is to use their natural logarithms $\beta_i = \log w_i$. The useful formula regarding β is:

$$\text{logit}(i \succ j) = \log \frac{P(i \succ j)}{1 - P(i \succ j)} = \log \frac{P(i \succ j)}{P(j \succ i)} = \beta_i - \beta_j$$

The β values are invariant to additive constant, since the w_i were invariant to multiplicative constants. The usual way to specify a single set of solutions is to require that $\sum \beta_i = 0$.

The standard BT model does not deal with ties and thus $1 - P(i \succ j) = P(j \succ i)$, but there are some extensions of the BT model that incorporate ties into the model (Rao and Kupper, 1967; Davidson, 1970; Baker and Scarf, 2021). We will discuss the model proposed by Davidson (1970) in Section 6.2. One common practice is to keep the standard BT model without ties but to change the data provided to the estimation procedure so that a tie between i and j is modeled a counting as both a victory for i and for j or sometimes as half a victory for each. We will elaborate this point further in Section 6.2.

4.0.1 MLE and Bayesian estimation of the w or β

Let us assume that players i and j play $N_{ij} = N_{ji}$ matches against each other, and W_{ij} is the number of matches that i wins and W_{ji} is the number of matches j wins (and thus $N_{ij} = W_{ij} + W_{ji}$ given that there are no ties). If we assume that $W_{ii} = 0$, and if there are t players, then the likelihood function is:

$$L(w) = \prod_{i,j=1}^{t} (\frac{w_i}{w_i + w_j})^{W_{ij}}$$

Given all the W_{ij}, the maximum likelihood (MLE) estimation is the set $w^* = \{w_1, w_2, \dots w_n\}$ such that

$$w^* = \text{argmax}_w L(w)$$

There has been proposals for different algorithms to compute the MLE (and MAP) solutions for the BT model (Hunter, 2004; Caron and Doucet, 2012). Some research refer to the case where the comparison graph is sparse (Li et al., 2021; Butler and Whelan, 2004), which is not the case in our problem – we assume that most of the algorithms will be compared to almost all other algorithms on most data sets.

The Bayesian model for BT is based on the beta coefficients:

$$\begin{aligned} W_{ij} &\sim \text{Binomial}(N_{ij}, \frac{e^{\beta_i}}{e^{\beta_i} + e^{\beta_j}}) \\ \beta_i &\sim \text{Normal}(0, \sigma) \\ \sigma &\sim \text{LogNormal}(0, 0.5) \end{aligned} \tag{2}$$

The binomial expression captures the fact that the number of times i wins from j is a binomial distribution given the total number of matches between i and j (N_{ij}) and the probability that i will win each match ($P(i \succ j) = \frac{w_i}{w_i + w_j}$).

The β parameters can have positive and negative values and thus it is reasonable to sample them from a normal distribution with mean 0, and variance σ (a hyper-parameter).

This is the hierarchical component of the Bayesian BT model (BBT): all β_i are sampled from the same distribution and thus the model can be used to compare multiple algorithms, since there will be partial pooling. The hyper-prior for σ is a log-normal distribution, as proposed by Carpenter (2018), but in section A we explore other hyper-priors and show that there is no difference on whether one uses them.

5. Exploration of the BBT

This section will explore some results of using the BBT model on a particular set of algorithms and data sets (Section 5.1). We will present some of the outputs of the model (Section 5.2) We also discuss two forms of diagnostics check (Sections 5.3 and 5.4), and a concept of ROPE appropriate for the model (Section 5.5). We finalize with a discussion on the two interpretation of the basic parameters of the model (Section 5.5.1) and a discussion on comparison against a control algorithm (Section 5.6).

5.1 Data

We will explore the use of the BBT model on four use-cases regarding the comparison of machine learning algorithms on multiple data sets. The details of how the data was obtained are not relevant for this paper, but for the sake of completeness we will briefly describe it. The four use cases are called small-small, small-large, medium-medium, and large-large.

The large-large (*ll*) use-case is the results of running 16 out of the box classifiers (without tuning their hyper-parameters) on 132 data sets. The classifiers are listed and briefly described below. The data sets are the first 132 smallest data sets from the set curated by Olson et al. (2017). We use the accuracy metric so we can compare the results with the BSR procedure.

The first set of 13 algorithms are implementations of classification algorithms available in the scikit-learn package (Pedregosa et al., 2011) (version 1.1.1), which also defines their particular set of default values for the hyper-parameters (not listed here).

- *dt*: Decision tree
- *gbm*: Gradient boosting classifier,
- *knn*: K-nearest neighbors
- *lda*: Linear discriminant analysis,
- *lr*: Logistic regression
- *mlp*: Multi-layer perceptron
- *nb*: Naive Bayes Gaussian classifier
- *passive*: Passive-aggressive classifier,
- *qda*: Quadratic discriminant analysis,
- *rf*: Random forest

- *ridge*: Ridge regression
- *svm*: SVM with a RBF kernel
- *svml*: SVM with linear kernel

The next set of two algorithms are implemented in the XGBoost package (Chen and Guestrin, 2016) (version 1.6.1):

- *xgb*: Gradient boosting classifier as implemented by the XGBoost package
- *xrf*: Random forest as implemented by the XGBoost package

And the final algorithm is from the LightGBM package (Ke et al., 2017) :

- *lgbm*: Gradient boosting classifier as implemented by the LightGBM

The large-large results reflect the case were a large number of algorithm is being compared on a large number of data sets. Most of the curated sets of data sets, for example PMLB (Olson et al., 2017), the KEEL imbalance data sets (Alcala-Fdez et al., 2011), the OpenML-CC18 Curated Classification benchmark (Bischl et al., 2019) include on the order of 100 data sets. We believe that, in a large case, a researcher will be comparing around 20 different algorithms, but we are aware of some published research that test many more algorithms, for example around 50 (Wainer and Fonseca, 2021), or around 100 (Fernandez-Delgado et al., 2014; Wang et al., 2021). But it is important to notice that these three examples did not perform statistical tests to confirm that the differences are statistically significant (within the frequentist framework) or made any other probabilistic claim (within a Bayesian framework). The ll-results are available as part of the R package developed for this research (see Section 9.1).

The other use cases are:

- the small-small (*ss*) use-case represents the situation where the researcher is comparing a small number of algorithms (5) on a small number of data sets (20);
- the small-large (*sl*) use-case represents the situation where 5 algorithms are being compared on 100 data sets; and finally
- the medium-medium (*mm*) use case is the case of 10 algorithms being compared on 50 data sets.

In this paper, the *ss*, the *sl*, and the *mm* cases will be used in *repeated experiments*: when we need to make some general claim regarding the BBT procedure, we will test the claim on 10 random *ss* results, 10 random *mm* results, and 5 random *sl* results, all sampled from the *ll-results*.

One fixed *ss* result will be used to illustrate the BBT procedure throughout this paper. We call this the *base* results, where we selected *lgbm*, *xgb*, *svm*, *lda*, and *dt* as the classification algorithms, and also selected 20 arbitrary data sets from the 132 in the ll-results.

The table of values for the base results is displayed in Table 1. The table represents the *mean* accuracy on the *same* 4-fold evaluation of the algorithms on each data-set.

db	dt	lda	lgbm	xgb	svm
biomed	0.837	0.842	0.876	0.890	0.886
breast	0.931	0.951	0.964	0.961	0.957
breast_w	0.940	0.950	0.961	0.961	0.961
buggyCrx	0.790	0.861	0.867	0.867	0.861
clean1	1.000	1.000	1.000	1.000	0.968
cmc	0.455	0.513	0.525	0.524	0.544
colic	0.761	0.837	0.815	0.815	0.641
corral	1.000	0.900	1.000	1.000	1.000
credit_g	0.668	0.718	0.766	0.769	0.724
diabetes	0.714	0.772	0.747	0.742	0.758
ionosphere	0.869	0.866	0.940	0.932	0.934
irish	1.000	0.740	1.000	1.000	0.988
molecular_b...y_promoters	0.727	0.689	0.896	0.887	0.802
monk3	0.975	0.792	0.980	0.986	0.964
prnn_crabs	0.880	1.000	0.950	0.935	0.960
prnn_synth	0.800	0.852	0.824	0.828	0.856
saheart	0.626	0.723	0.660	0.671	0.712
threeOf9	0.996	0.809	1.000	0.998	0.992
tokyo1	0.902	0.920	0.928	0.926	0.931
vote	0.929	0.956	0.945	0.959	0.956

Table 1: The base results.

5.2 Basic outputs of the model

A **wintable** is the representation of the number of wins and losses for each pair of algorithms. This data is the input for the Bayesian model 2.

alg1	alg2	win1	win2	ties
dt	lda	6	13	1
dt	lgbm	0	17	3
dt	xgb	0	17	3
dt	svm	5	14	1
lda	lgbm	6	13	1
lda	xgb	5	14	1
lda	svm	5	15	0
lgbm	xgb	9	8	3
lgbm	svm	10	9	1
xgb	svm	11	8	1

(a) The wintable for the base results.

alg1	alg2	win1	win2	ties
dt	lda	7	14	0
dt	lgbm	2	19	0
dt	xgb	2	19	0
dt	svm	6	15	0
lda	lgbm	7	14	0
lda	xgb	6	15	0
lda	svm	5	15	0
lgbm	xgb	11	10	0
lgbm	svm	11	10	0
xgb	svm	12	9	0

(b) The final wintable for the base results once the ties have been added as half-victories to both algorithms.

Table 2: The wintable.

Table 2a shows some ties, for example, *dt* and *lda* both achieve an 1.0 accuracy in the *clean1* data set. As discussed above, the BT model does not deal with ties. We will model ties by adding half (rounded up) of the ties as victories for both algorithms, which we call the *spread* policy. Section 6.2 will discuss this further. Thus, the final wintable for the base results is shown in Table 2b using the spread policy to transform the ties into partial wins for the different algorithms.

The MCMC solution to the BBT model in 2 for the data in Table 2b is a set of tuples k for the parameters β_i, and for σ. In our case, we are interested in using, for example, each β_{1k} and β_{4k} to compute $P_k(1 \succ 4)$. That is, for each of the k samples, we compute an instance of $P_k(a \succ b)$, for all pairs of algorithms alg_a and alg_b, and we can plot or display a summary of these distributions.

The Bayesian solution to the BT model poses the issue of defining the aggregated ranking. One solution is to compute the ranking for each sample generated by the MCMC algorithm: for a sample k we order the algorithms by decreasing values of β_{ik}. This is the approach used in Carpenter (2018), and by the `bpcs` R package (Issa Mattos and Martins Silva Ramos, 2021). The problem with this approach is that, at the end, there is a distribution of k rankings, and it is unclear how to make any statement regarding a single, final ordering. One can select the most frequent ranking among the samples, or compute the rank of each algorithm in each ranking, and order them by the mean rank.

We believe that determining the order of the algorithms is an important part of the comparison procedure, and thus we prefer a different solution. We order the algorithms by their mean β across all samples. This results in a single ordering, which is the aggregated ranking of the BBT comparison procedure.

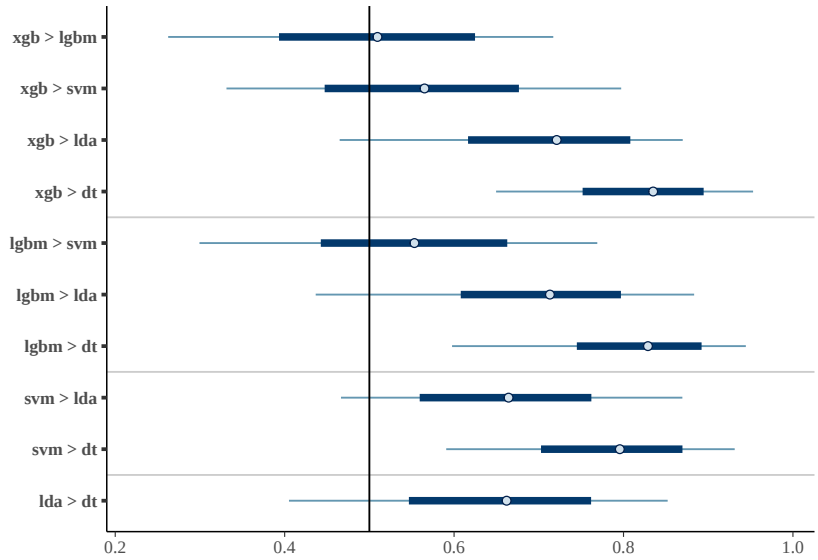

Figure 1: The graphical representation of the distribution of P (a better than b)

Figure 1 displays the distributions of the $P_k(a \succ b)$. The figure orders the algorithms from best to worse, where the best is compared with all the others, the second best with the remaining worse, and so on. The central dot is the median of the distribution of $P_k(a \succ b)$; the wider line represent the 89% HDI (highest density interval) of the distribution, while the thin line represents the full range of the distribution. Some researchers in Bayesian estimation use 89% (instead of 95%) as a jest to distinguish a *credible interval* which is an interval that concentrates some mass (in this case 89% of the mass) of a distribution from the "95% confidence interval" concept in frequentist statistics which has a subtly different meaning (Makowski et al., 2019). There are infinitely many intervals that contain 89% of the mass of a distribution. The HDI is the smallest of them (for unimodal distributions) (Kruschke, 2014).

Some of the data displayed in Figure 1 can be summarized in Table 3, containing the mean, the low and high limits of the 89% HDI.

5.3 Convergence Diagnostics and execution times

As discussed, every run of an MCMC algorithm would possibly require verifying that the algorithm did converge to the target distribution. In this research, we use Stan (Stan Development Team, 2022) as the language to describe the BBT model (2) and to run the MCMC. Stan collects many convergence diagnostics data and interprets them into claims on whether there was an acceptable convergence or not. The result of running the simplified Stan check is displayed below.

```
## Checking sampler transitions treedepth.
## Treedepth satisfactory for all transitions.
##
## Checking sampler transitions for divergences.
```

pair	mean	low	high
xgb > lgbm	0.51	0.40	0.63
xgb > svm	0.56	0.45	0.68
xgb > lda	0.72	0.62	0.81
xgb > dt	0.83	0.75	0.90
lgbm > svm	0.55	0.44	0.66
lgbm > lda	0.71	0.62	0.81
lgbm > dt	0.82	0.75	0.90
svm > lda	0.66	0.57	0.77
svm > dt	0.79	0.72	0.88
lda > dt	0.66	0.56	0.77

Table 3: The table representation of the distributions of probabilities

```
## No divergent transitions found.
##
## Checking E-BFMI - sampler transitions HMC potential energy.
## E-BFMI satisfactory.
##
## Effective sample size satisfactory.
##
## Split R-hat values satisfactory all parameters.
##
## Processing complete, no problems detected.
```

The MCMC sampling of the model is unproblematic – in all examples in this paper, including the ones in the sections below, we used 1000 steps of warm-up and 1000 steps of sampling, on 4 chains. In a modern laptop, for example, an Intel i5 at 1.4 GHz, each chain takes in the worse case 0.5 seconds and the four them can be run truly in parallel, one in each core.

5.4 Posterior Predictive Check and WAIC

Figure 2 is a usual way of displaying the posterior predictive check. The plot shows the `win1` variables from the wintable – one for each pair of algorithms. The histogram represent the data generated by the Bayesian model and the vertical bar, the actual value for that variable. If model is a good generator of the actual data, then the real data will be well centered in the histogram of possible values for that variable.

We also propose a non-graphical representation of the PPC: for each variable we compute the 50%, 90%, 95%, and 100% HDI of the generated values, and compute the proportion of the true data what falls within each of its own respective HDI. The proportion of data values that fall within its respective 90% HDI should be 0.9 or more. Table 4 displays this alternative representation of the PPC.

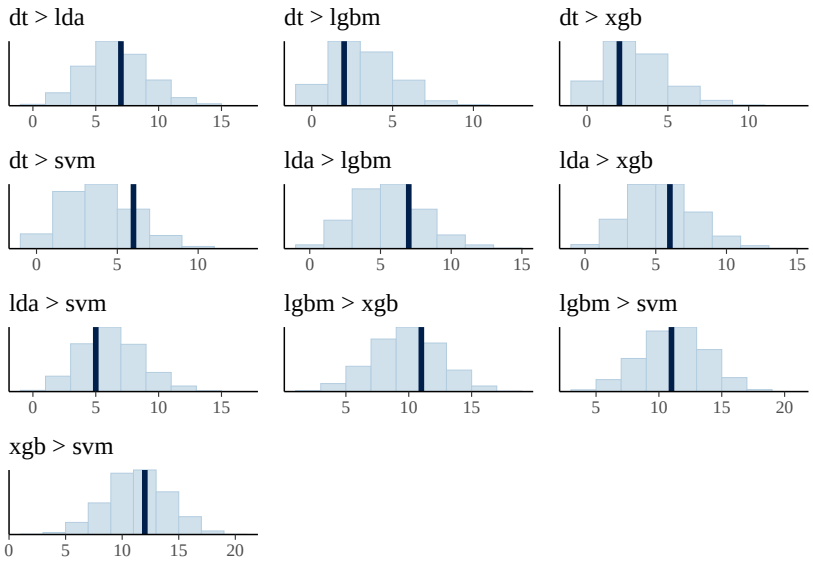

Figure 2: Graphical representation of the PPC

hdi	proportion
0.50	0.8
0.90	1.0
0.95	1.0
1.00	1.0

Table 4: The table representation of the PPC

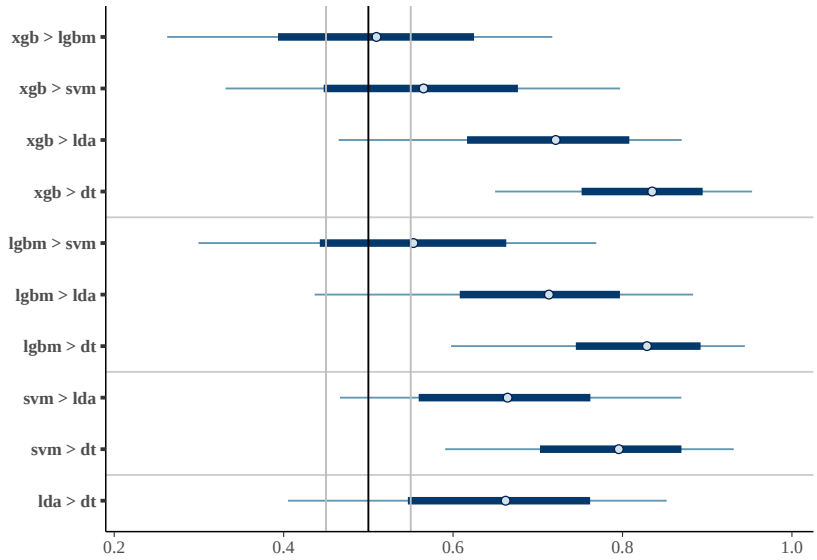

Figure 3: Graphical representation of the probability distributions with the rope information

5.5 ROPE

As discussed above, the Bayesian perspective allows one to define a difference between parameters that does not matter for practical purposes, and to make statements regarding the probability that the parameters do no differ in any practical sense.

The BBT model simplifies the adoption of the idea of practical equivalence. The final measure of the BBT model is the probability that a particular algorithm is better than some other. One can define an universal ROPE for probability statements, regardless of underlying metric being used to measure when one algorithm is "better" than another. In this paper we propose that if the probability that one algorithm is better than another is within the range of 0.45 to 0.55, we can make the claim that the two algorithms are equivalent for practical purposes.

Formally, if $P(A \succ B) \in [0.45, 0.55]$ we will make the claim that algorithms A and B are equivalent for practical purposes. This is not a claim based on the community's understanding of this problem, or the author's experience in dealing with the comparison of multiple algorithms. The ROPE limits are very easy to comprehend and anyone can propose a different ROPE for different applications. The author, somewhat arbitrarily, believes that an algorithm whose probability of being better than another is below 55% (and above 45%) is really not that better than the other one. Other researchers can have different intuitions regarding this and are well justified to change the ROPE for their application.

The ROPE can be included in both the distribution graph and the summary table. In Figure 3, the two gray vertical lines represent the low and upper bound of the rope.

Table 5 is the summary of the probability distributions, with two columns added. The *in.rope* column is the mass of probability that A and B are equivalent for all practical purposes, or more formally, the proportion of samples $P_k(A \succ B)$ that fall within the ROPE interval $[0.45, 0.55]$. The *above.50* column is the mass of the probability distribution

pair	mean	delta	above.50	in.rope
xgb > lgbm	0.51	0.23	0.56	0.51
xgb > svm	0.56	0.23	0.81	0.36
xgb > lda	0.72	0.19	1.00	0.00
xgb > dt	0.83	0.14	1.00	0.00
lgbm > svm	0.55	0.22	0.78	0.41
lgbm > lda	0.71	0.19	1.00	0.01
lgbm > dt	0.82	0.14	1.00	0.00
svm > lda	0.66	0.20	0.99	0.04
svm > dt	0.79	0.16	1.00	0.00
lda > dt	0.66	0.21	0.99	0.06

Table 5: Table with the summary of the probabilities distributions

that A is better than B, or more formally, the proportion of samples $P_k(A \succ B)$ that fall in the interval [0.50, 1.00].

5.5.1 Strong and weak interpretations of the probability estimates

We believe that the four important columns to report are *mean*, *delta* (the difference between the *high* and *low* values of the HDI) *in.rope* and *above.50*. In particular, the *mean* and the *above.50* measures will be central to what we will call the **strong** and the **weak interpretations** of the probability estimates. As discussed, the BBT model generates a set of numbers $P_k(A \succ B)$ which we have been calling probabilities that A is better than B. And in fact, these numbers are used in the BBT model as the parameters of the binomial distribution that are interpreted as probabilities of the event happening. Under the strong interpretation, we do understand these numbers as probabilities that A is better than B in a frequentist approach - in the long run, for a large number of data sets, the proportion of times A wins from B should approach that number. In the strong interpretation, the *mean* column is the best estimation of how much better algorithm A is from B. The *delta* column (or both *low* and *high*) are the estimates on how uncertain one is about that probability.

The weak interpretation sees each $P_k(A \succ B)$ as a number that measures how much A is better than B. It happens to be a number that ranges from 0.0 to 1.0, and if the number is lower than 0.5, it indicates that B is better than A. Under the weak interpretation, each $P_k(A \succ B)$ is an evidence that A is better than B, not a promise of future proportions of wins in the long run. In this case, the *above.50* is the measure on how confident one should be that A is better than B. If 90% of the evidence (each $P_k(A \succ B)$) are above 0.5, one can be 90% confident that A is better than B.

The *in.rope* measure falls oddly between the two interpretations. Following the weak interpretation, we are counting the proportion of evidence that falls within an interval (from 0.45 to 0.55). But to justify the range of the ROPE, we used a strong interpretation's point of view: the fact that, in the long run, one algorithm will win from the other at most 55% of the time is not relevant for practical purposes.

pair	mean	delta	above.50	in.rope
rf > xrf	0.60	0.05	1.0	0.00
lgbm > xrf	0.59	0.05	1.0	0.01
xgb > xrf	0.58	0.05	1.0	0.03
gbm > xrf	0.56	0.05	1.0	0.24
mlp > xrf	0.54	0.05	1.0	0.65
xrf > svm	0.51	0.05	0.8	0.99
xrf > lr	0.58	0.05	1.0	0.04
xrf > lda	0.62	0.05	1.0	0.00
xrf > svml	0.62	0.05	1.0	0.00
xrf > ridge	0.68	0.05	1.0	0.00
xrf > knn	0.69	0.05	1.0	0.00
xrf > dt	0.72	0.04	1.0	0.00
xrf > qda	0.77	0.04	1.0	0.00
xrf > nb	0.82	0.03	1.0	0.00
xrf > passive	0.85	0.03	1.0	0.00

Table 6: Table with the summary for the xrf control algorithm on the ll-results.

There are two reasons to put forth these two interpretations, and not just to follow the more direct strong interpretation. In section 7 we will see that in order to compare the BBT model with previous ones, we should use the weak interpretation; and in section 8 we will show that the BBT model may not be very well calibrated for predictions on future results for the algorithms under the strong interpretation, but it is better calibrated under the weak interpretation.

5.6 Comparisons against a control

In Bayesian tests, there is no need to distinguish between all pairwise comparisons and comparisons against a control. There is no penalty regarding the quality of assessments when performing only n instead of $n(n-1)/2$ pairwise comparisons. If one is only interested in displaying the comparisons of the control algorithm against its competitors, one need only to limit the rows of the summary table that are shown.

Using the ll-results, let us assume that the *xrf* algorithm is the control. One can run the full BBT model for all algorithms and display only the rows that include *xrf*. Table 6 show these results

6. What counts as a win? Folds and local rope

Usually, the final performance measure for a particular algorithm for a particular data set is the average of the measures on some form of repeated cross validation, where the algorithm trains on different subsets of the data set and its performance is measured on the corresponding test subsets. Standard forms of repeated cross-validations are: k-fold, repeated k-folds, repeated train/test split, and bootstrapped samples of the data set. In each

db	dt	lda	lgbm	xgb	svm
cmc	0.455	0.513	0.525	0.524	0.544

Table 7: Detail of the base results for the cmc data set

case, there are k pairs of subsets TR_i (train) and TE_i (test) such that $TR_i \cup TE_i = DS$ and $TR_i \cap TE_i = \emptyset$ (DS is the whole data set). We will call each of the TE_i as a **fold** although we do not assume that a k-fold cross validation is being used – almost all cross-validation procedures can be used.

Regarding the data in this research, we computed the mean of a 4-fold cross-validation of each algorithm for each data set. Furthermore, for each data set the folds were fixed and the same for all algorithms, that is, for the first fold, all algorithms were trained in the subset TR_1 and tested in the subset TE_1, and so on.

Let us consider a particular pair of measures from table 1 for the data set "cmc" repeated as Table 7. The entries for the *lgbm* and the *xgb* algorithms are 0.525 and 0.524 respectively. This small difference counts as a win for *lgbm*, as much as the much more impressive difference to the 0.455 accuracy for the *dt* algorithm also counts as a win.

If we consider the folds themselves for the "cmc" data set, displayed in table 8, we see that this small difference is even less convincing as a win for the *lgbm* algorithm. For these runs, we used the same fold 1, fold 2, and so on for all the algorithms, and thus it is reasonable to compare the performance of the algorithms on each fold separately. In this case, *lgbm* wins on two of the folds, but looses on the other two, of the 4 fold computation of the algorithm's accuracy. Therefore, if we use the folds as sources of evidence instead of working on the mean of the folds' results, *lgbm* would receive two wins and *xgb* also have to wins, as opposed to a single win for *lgbm* when using the means.

From another point of view, if we consider the standard deviations of the measures on the folds for each algorithm, also displayed on Table 8, the differences in the mean of both algorithms (0.001) is also much smaller than the standard deviations (0.02 and 0.03). In some intuitive sense, the difference in averages that yielded the win for *lgbm* is much smaller than some "noise level" of the evaluation procedure itself, considering the variability of the measures in the folds within each classifier itself.

This opens up two lines of reasoning: the one that considers each fold (provided they are the same for all algorithms) as sources of evidence to count the wins and losses of the algorithms, and one that deals with the difference between the means for all folds, but take into consideration both the difference and some "noise level" derived from the variability within each classifier. But in both cases we are aiming reducing the strength of evidence that *lgbm* has a victory in relation to *xgb*, and that should be considered a tie.

This research will follow the "noise level" line. We argue that a difference of 0.001 in the means given that for each classifier the variability of the measures in the folds is at least 10 times as high, should not count as a win but as a ties between the two algorithms. We will call this approach the **local ROPE**, some threshold below which one should consider the difference between two classifiers as unimportant. But as we will see, the local ROPE is not a fixed number but depend on the results of the two algorithms on the different folds.

db	fold	lgbm	xgb	diff
cmc	1	0.547	0.531	0.016
cmc	2	0.522	0.538	-0.016
cmc	3	0.503	0.481	0.022
cmc	4	0.527	0.546	-0.019
sd		0.018	0.029	0.001

Table 8: Detail of the base results for the cmc data set

Regarding the first line, using the folds themselves as source of evidence for wins and losses, we believe that issues such as the fact that the fold results are not independent of each other would make the analysis too complex (the same conclusion reached by Benavoli et al. (2017) for their analysis), and will leave that approach for future research.

6.1 Local ROPE

In almost all statistical tests one has two sets of measures, and one wants to make a claim regarding whether there is enough evidence that the difference of the means (or other summary measure) of the two sets is "real" or not. This is exactly the problem in hand: should one consider the difference of the means of the folds as "real" – and thus that one algorithm is better than the other – or not?

Cohen's D is a measure of effect size between two sets of data. For the case where the two sets have the same number of data, as it is our case, the Cohen d is computed as the difference between the means, divided by an "average" standard deviation of the two sets, where the "average" standard deviation is actually the square root of the average variance of the two sets. This is displayed in Equation 3. Cohen d is the measure of the separation between the two means, as a proportion of the standard deviation and can be seen as a signal to noise ratio measure: the difference in means is the signal, and the "average" standard deviation is the noise.

$$d = \frac{\mu_1 - \mu_2}{\sqrt{\frac{\sigma_1^2 + \sigma_2^2}{2}}} \tag{3}$$

We can compute the Cohen D of two set of fold measures, and consider that there is no important difference, and thus, a ties between the two algorithms, if the D is below a threshold $d_{\min}$ which we will call the **local ROPE threshold**. Therefore, if:

$$\mu_1 - \mu_2 \leq d_{\min}\sqrt{\frac{\sigma_1^2 + \sigma_2^2}{2}} \tag{4}$$

we should consider that there was a tie between algorithms 1 and 2 for that data set.

We will argue that the threshold can be safely set to the value of 0.4 using the theory of power analysis for t-tests. We refer the reader to the concepts of type 1 and type 2 errors, their probabilities α and β (section 2.1). The power analysis of t-tests relates α, β, the effect size of the measure, and the number of samples in each set. Unfortunately, the relation between these variables is almost never displayed as an equation, but as tables

(Cohen, 1988, ch. 2) or embedded into programs, such as G*power (Faul et al., 2007) or the `pwr` R package (Champely, 2020). We will show the results of running the `pwr` package.

For our present goals, there is no conceptual difference between false positive and false negative errors. We want to find out whether the two sets of fold measures indicate that the difference between the means is "real" or "not real", and erring to one side is not worse that to the other. Thus, let us assume a 70% probability of detecting both the presence of a real difference and a absence of a real non-difference, that is, $\alpha = 0.3$ and $\beta = 0.3$. For this case, and assuming a Cohen D of 0.4, the necessary number of data in each set is given by running the `pwr` function:

```
pwr.t.test(n = NULL, d = 0.4, sig.level = 0.3, power = 0.7)

Two-sample t test power calculation
          n = 30.18637
          d = 0.4
  sig.level = 0.3
      power = 0.7
alternative = two.sided
NOTE: n is number in *each* group
```

In this case one would need at least 30 measure in each set to be able to find a true difference or a true non-difference with 70% probability. But the traditional cross validations in machine learning are from 3 to 10 folds. That is, using the usual cross validation in machine learning, a minimum effect size of 0.4 is very safe - one would not be able to detect differences whose effect sizes are 0.4 or below, if one requires a 70% of sensitivity and specificity. If one is using 10 repetitions of 10-folds as cross validation, one can use $d_{\min} = 0.2$.

The discussion above assumes that the two samples of fold measures are not paired, that is, that possible different folds were used in the evaluation of the different algorithms. But if the researchers have control over it, they can use the same folds for all algorithms. For the paired case, the definition of the Cohen's D is somewhat different than the one presented in 3. Instead of dealing with the mean and standard deviations of the two sets, one should compute the mean and standard deviation of the differences between the corresponding paired data in the two sets.

$$d_z = \frac{\mu_{X_1 - X_2}}{\sigma_{X_1 - X_2}} = \frac{\mu_1 - \mu_2}{\sigma_{X_1 - X_2}} \tag{5}$$

The power analysis for paired samples is also somewhat different, and with the same numbers as before ($\alpha = 0.3$ and $\beta = 0.3$), and using Equation 5 for the effect size calculation, the resulting lower bound for the number of samples is 15, lower than the case for unrelated samples, but still well above the usual number of folds used in machine learning evaluations.

```
pwr.t.test(n = NULL, d = 0.4, sig.level = 0.3, power = 0.7, type="paired")

Paired t test power calculation
          n = 15.53464
```

```
          d = 0.4
  sig.level = 0.3
      power = 0.7
alternative = two.sided
NOTE: n is number of *pairs*
```

The same decision process as described in 4 can be followed, using the same d_{min} threshold of 0.4, but using the paired definition for the effect size.

$$\mu_1 - \mu_2 \leq d_{\min}\sigma_{X_1 - X_2} \tag{6}$$

6.2 How to deal with ties?

The local ROPE concept introduces many ties to the wintable, as it is designed to do. The standard ways of dealing with ties in the Bradley-Terry model are:

- add: add the ties as victories to both players involved.
- spread: add the ties as half a victory to each player involved
- forget: do not add ties to as victories to any of the players.

Another alternative is to change the Bradley-Terry model into a model that includes ties, for example the one proposed by Davidson (1970). The Davidson model is displayed in Equation 7. It includes a new parameter ν similar to the β_i. ν controls how likely are ties "in that sport", despite the differences between the players. If $\nu \to -\infty$, $P(i \text{ ties } j)$ will be 0, that is, there are no ties; if $\nu \to \infty$, $P(i \text{ties} j)$ will be 1, regardless of the players' different β. Finally, for $\nu = 0$, if $\beta_i = \beta_j$ then the probability of a tie is 1/3.

$$\begin{aligned} P(i \succ j | \text{ no tie }) &= \frac{\exp \beta_i}{\exp \beta_i + \exp \beta_j + \exp(\nu + (\beta_i + \beta_j)/2)} \\ P(i \text{ ties } j) &= \frac{\exp(\nu + (\beta_i + \beta_j)/2)}{\exp \beta_i + \exp \beta_j + \exp(\nu + (\beta_i + \beta_j)/2)} \end{aligned} \tag{7}$$

We will compare the different policies, using a repeated experiment as described above, and comparing how well the different solutions model the data. We will use the posterior predictive check and the WAIC to measure how well each policy fits the real data. Regarding the WAIC, the number itself is hard to interpret, but when comparing two models, the lower the value of the WAIC, the better (as it is with other information measures such as AIC and BIC).

Tables 9 and 10 display the WAIC, and the PPC summary results for the repeated experiments for comparing different ways of dealing with ties. The results are the average of the repeated experiments on ss, mm, sl use cases, and the ll-results, also averaging whether the local ROPE was used or not, and whether the paired local ROPE was used. The results clearly show that the Davidson model is much worse than the others regarding both WAIC and PPC. Add, forget and spread policies are all equivalent, and somewhat arbitrarily, we decided on using the spread policy in this paper.

	ss					mm				
policy	**waic**	**h50**	**h90**	**h95**	**h100**	**waic**	**h50**	**h90**	**h95**	**h100**
add	42.41	0.87	1.00	1.00	1	225.51	0.73	1.00	1.00	1.00
davidson	118.59	0.40	0.78	0.86	1	873.01	0.29	0.58	0.66	0.88
forget	40.27	0.78	0.99	1.00	1	227.61	0.61	0.96	0.99	1.00
spread	41.41	0.82	1.00	1.00	1	223.76	0.68	0.99	1.00	1.00

Table 9: The PPC of ss and mm use cases for the different policies.

	sl					ll				
policy	**waic**	**h50**	**h90**	**h95**	**h100**	**waic**	**h50**	**h90**	**h95**	**h100**
add	61.62	0.81	0.97	0.99	1.00	760.49	0.57	0.95	0.97	1.00
davidson	325.65	0.21	0.51	0.56	0.78	4730.36	0.17	0.38	0.43	0.70
forget	65.46	0.67	0.95	0.97	0.99	823.52	0.39	0.85	0.91	0.99
spread	62.25	0.75	0.97	0.99	1.00	773.98	0.47	0.92	0.95	1.00

Table 10: The PPC of sl and ll use cases the different policies.

The low quality of the Davidson model is somewhat surprising, given that the model explicitly deals with ties, and the other policies seems *ad hoc* solutions. Table 11 show the results of the PPC summary for the ll-results. One can see that neither the wins nor the ties are well calibrated with their corresponding HDI.

7. Comparison with standard approaches

Let us compare the results of using the BBT procedure with that of using some of the standard comparisons procedures. Section 7.1 compare the BBT with Demsar's procedure on a different set of use-cases; section 7.2 compares with the pairwise Wilcoxon procedure, and section 7.3 compares with the Bayesian signed rank procedure.

7.1 Demsar's

The Demsar's procedure starts with the evaluation of the Friedman test, which results in a p-value ≤ 0.05 for the base results. The next step is the Nemenyi test, which results in the

hdi	**proportion**	**ties**
0.50	0.23	0.26
0.90	0.46	0.46
0.95	0.52	0.49
1.00	0.91	0.82

Table 11: The Davidson model on the ll-results.

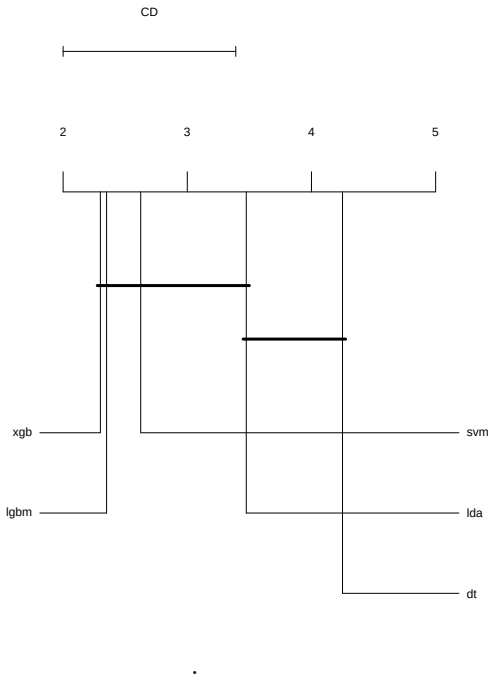

Figure 4: The CD plot of the base results

algorithm	mean rank
xgb	2.30
lgbm	2.35
svm	2.62
lda	3.48
dt	4.25

Table 12: The table representation of the CD plot

Critical Difference plot displayed in Figure 4. The results of significant and non significant differences can be displayed in a tabular for as in Table 12.

We refer the reader to Table 5 for the BBT results regarding the base results. Here, we are faced with the strong vs weak interpretations of the probability estimates. Under the *strong* interpretation, it would be reasonable to make an equivalence between a frequentist test making a claim of statistical significant difference between algorithm A and B (with 95% confidence) and that the mean estimate that $P(A \succ B) \geq 0.95$. These two concepts are not the same, but it is reasonable to make the equivalence to compare them. In this case, none of the claims of superiority made by the BBT model reaches the level of 95%. The highest mean is 0.84 (*xgb* against *dt*).

But under the weak interpretation, using $above.50 \geq 0.95$ as the equivalence of statistical significance, the BBT procedure finds 5 differences that one would call significant, as opposed to three found by the Demsar's procedure. In the base results, the differences between *lgbm* and *lda*, and *xgb* and *lda* are detected as "significant" by BBT, but they were not detected as such by the Demsar's procedure.

We ran the repeated experiments to verify how frequent are three results when comparing the BBT model with the traditional frequentist approach proposed by Demsar: how many times does BBT find a significant difference (in the sense that the *above.50* result is larger than 0.95) where Demsar does not find it significant, how many times the reverse happens, and how many times the aggregated ranking computed by one method did agree with the one computed by the other method.

The results are:

- *ss:* For the 100 pairs of comparisons (10 from each ss), the BBT method classified 40 of them as significantly different, when Demsar did not determined them to be. No pairs were found significantly different by Demsar's procedure and not by BBT. Finally, only 3 of the 10 aggregated rankings were found to be the same.
- *mm:* For the 450 pairs, BBT found 187 that were not significant according to the Demsar procedure, none the other way, and 5 of the 10 aggregated rankings were the same.
- *sl:* For the 50 pairs, 11 were found as significant only by BBT and none the other way, and 5 aggregated rankings were in agreement.
- *ll results:* For 120 pairs of comparisons, BBT found 41 that were not significant by Demsar's, none the other way, and the aggregated ranking did not agree.

Thus there is strong evidence that the BBT procedure is stronger, in the sense that it finds more significant differences than Demsar's procedure, and that it supersedes Demsar's procedure, in the sense that it does not miss any significant difference determined by Demsar's. Furthermore, usually 50% of the time, the two procedures do not fully agree on the final aggregated ranking.

For all but the ll-results, there was no case in which $in.rope \geq 0.95$. For the ll results, there were 8 cases in which the Demsar's procedure sees not significant difference but the BBT can make a stronger claim that the two algorithms are equivalent. There is an interesting case, which did not occur with our data, but Benavoli et al. (2017) report, of two algorithms whose difference is significant but are practically equivalent. The frequentist approach has enough evidence to claim that the algorithms are different, but the Bayesian approach claims that the difference does not matter! But this is expected: with enough data all algorithms will be found statistically different from each other, even those whose difference is irrelevant.

7.2 Pairwise Wilcoxon

Table 13 reports the result of the pairwise Wilcoxon tests, using the Horchberg p-value adjustment procedure. The column *p-value* indicates the adjusted p-value of the comparison, and as usual, the comparisons where p.value ≤ 0.05 are considered significant.

Following the same procedure described above of considering $above.50 \geq 0.95$ as "equivalent" to p.value ≤ 0.05, we see that the pairwise Wilcoxon tests only detect two significant differences, for the five detected by the BBT model. Also the aggregated rank is not the same for the pairwise Wilcoxon and BBT. The second and third best algorithms, and the fourth and fifth (according to BBT) are in reverse order for the pairwise Wilcoxon procedure, although it does not find these to pairs of algorithms as significantly different from each other.

We ran the replication experiments as above and the results are:

- *ss:* For the 100 pairs the BBT found 21 beyond the pairwise Wilcoxon, only one pair was detected by the pairwise Wilcoxon and missed by the BBT, and 2 of the 10 aggregated ranking were the same.

algorithm	median
lgbm	0.93
svm	0.93
xgb	0.93
dt	0.87
lda	0.85

(a) Pairwise Wilcoxon: Order of the algorithms

name	p.value
lgbm > svm	0.98
lgbm > xgb	0.98
lgbm > dt	0.00
lgbm > lda	0.47
svm > xgb	0.98
svm > dt	0.11
svm > lda	0.46
xgb > dt	0.00
xgb > lda	0.46
dt > lda	0.98

(b) Pairwise Wilcoxon: Significance of differences displayed as a table.

Table 13: Pairwise Wilcoxon tests - order of the algorithms and significance of differences displayed as a table."

- *mm:* For the 450 pairs, BBT found 126 beyond the pairwise Wilcoxon, 6 missed, and no aggregated ranking was the same.
- *sl:* For the 50 pairs, BBT found 6 beyond the pairwise Wilcoxon, 1 missed, and only one aggregated ranking was the same.
- *ll results:* For 120 pairs of comparisons, BBT found 21 beyond the pairwise Wilcoxon, 1 missed, and the methods did not agree on the aggregated ranking.

BBT seems to be stronger than the pairwise Wilcoxon but not all pairs deemed significant by the pairwise Wilcoxon are found by BBT. The two procedures fully agree on very few of the aggregated rankings. Also, for the ll-results, there were 7 cases in which the pairwise Wilcoxon procedure sees not significant difference but the BBT can make a stronger claim that the two algorithms are equivalent.

7.3 Pairwise Bayesian signed rank

Following the caveat that it is unclear whether one should use the Bayesian signed rank (BSR) on multiple comparisons, Table 14 displays the results of the comparisons. Benavoli et al. (2017) suggest that one should report the probabilities that the difference falls within the 0.1 ROPE (*in.rope*) and the probability that it falls above the ROPE (*above.rope*). We also report the probability that the probability is above 0 (*above.0*), that is, that the difference between the two mean parameter values is positive, that is, that one algorithm is better than the other.

We believe that the only fair comparison is between the *above.50* from BBT and the *above0* from BSR. Both measures have the same semantics: *above0* counts the proportion of evidence within BSR that one algorithm is better than another and *above.50* under the

name	above0	in.rope	above.rope
lgbm > svm	0.72	0.62	0.33
lgbm > xgb	0.50	1.00	0.00
lgbm > dt	1.00	0.00	1.00
lgbm > lda	0.95	0.01	0.96
svm > xgb	0.21	0.60	0.06
svm > dt	1.00	0.00	1.00
svm > lda	0.97	0.03	0.93
xgb > dt	1.00	0.00	1.00
xgb > lda	0.98	0.01	0.96
dt > lda	0.20	0.00	0.24

Table 14: Results of the pairwise Bayesian Wilcoxon test

weak interpretation also counts the proportion of evidence that one algorithm is better than another. Figure 5 plot the BSR *above0* measure against the corresponding BBT *above.50* measure, where the dark line is the $y = x$ line. The figure also displays the Pearson correlation coefficient of the two measures. As one can verify, there is a low correlation between the results.

We do not know how to interpret the differences between the two procedures, nor if the differences are important or not.

8. BBT as a prediction

Frequentist methods are *explanatory* in the sense that they explain the data at hand, but they make no prediction about future data. The fact that the difference between two sets of data is found not to be significant is not a prediction (with 95% probability, for example) that future data for these sets will not be different from each other. Instances of new data are almost always different from each other. Furthermore, with more data, maybe the difference that was found not to be significant becomes significant!

Frequentist tests do not make predictions, but Bayesian tests can. In particular the BBT model's output as displayed in Table 5) can be seen as probabilities that one algorithm will better than the others *on future data*. There are two distinct predictions, as discussed in section 5.5.1. In the strong interpretation, the *mean* measure is a prediction of the proportion of wins for the better algorithm against the worse for future data sets. In the weak interpretation, the *above.50* is the prediction of the proportion of wins for the better algorithm against the worse. In this section we will test both predictions.

We ran the usual set of repeated experiments, but with the goal of measuring how correct are these predictions on future data sets. For each sample of an ss results (the "training" case), we also sampled 10 data sets not used in the training ss, and measure in this "test" case, for each pair of algorithms, the number of times one algorithm was better than the other. For algorithms A1 and A2, let us call *win1* the number of data sets for which algorithm A1 performed better, and *win2* the number of data sets in which algorithm A2 performed better. We must point out that we did not use local ROPE on the test data. The

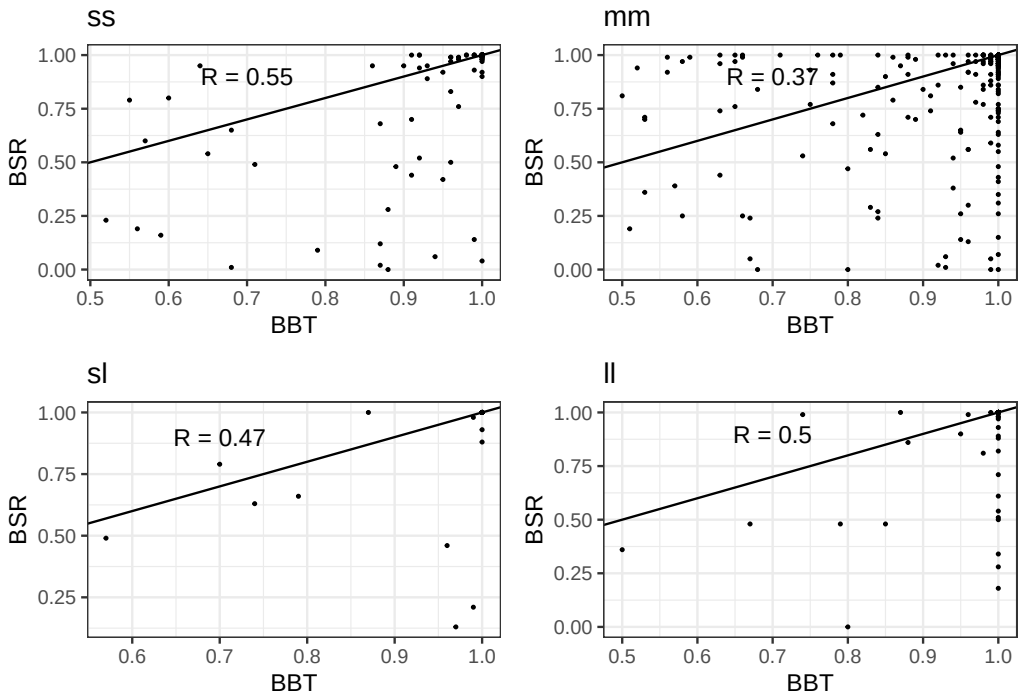

Figure 5: Comparison of the above0 from BSR and *above.50* (BBT) for the ss, mm , sl and ll use cases

ratio $win1/(win1 + win2)$ is the *empirical estimation* of the probability that algorithm A1 is better than algorithm A2 on the test data sets.

For the strong interpretation we are interested on how well the *distribution* of the probability estimates $P_k(1 \succ 2)$ fits the empirical probability $win1/(win1 + win2)$. In this case, we follow the same idea as the PPC summary table: we compute the 50%, 70%, and 90% HDI of each of the distributions (for each pair of algorithms) and compute the proportion of the empirical probabilities that fall within their respective HDIs. As usual, for a calibrated probability distribution, at least 50% of the empirical probability should fall within their respective 50% HDI, and so on. We also computed the proportion of empirical probabilities that falls above the 90% HDI highest value, and below the 90% HDI lowest value. Finally, we computed the mean error and the median absolute difference (MAD) between the mean prediction and the empirical probability.

For the weak interpretation, we do not have a distribution and thus we cannot follow the same evaluation procedure as above. The *above.50* measure is a probability statement, and we want to measure how correct is that statement. We will follow the procedure for constructing a calibration plot for classifiers, for example. We divide the range of *above.50* values into bins, and for each bin we compare the real and the expected number of cases where $win1 > win2$. The real number in a bin is counted from the "test" data for the cases where the *above.50* falls within the limits of the bin. The expected number of cases is the sum of the *above.50* values in that bin. In calibration plots, one usually divide the probability estimation into 10 bins, but we feel that that would be too much information.

lrope	paired	within.90	within.70	within.50	above90	below90	err	mad
ss								
F	F	0.39	0.25	0.15	0.33	0.28	0.00	0.13
T	F	0.44	0.25	0.14	0.33	0.23	-0.01	0.12
T	T	0.42	0.23	0.13	0.33	0.25	-0.01	0.13
mm								
F	F	0.33	0.22	0.16	0.37	0.31	0.00	0.08
T	F	0.34	0.22	0.14	0.39	0.27	-0.01	0.08
T	T	0.32	0.22	0.16	0.41	0.27	-0.01	0.08
sl								
F	F	0.40	0.22	0.14	0.48	0.12	-0.06	0.07
T	F	0.38	0.26	0.14	0.50	0.12	-0.07	0.07
T	T	0.34	0.20	0.12	0.54	0.12	-0.06	0.06

Table 15: Prediction results - strong interpretation

We divided the *above.50* into 3 bins, from 0.5 to 0.7, from 0.7 to 0.9, and from 0.9 to 1.0. We believe that researchers will be interested on a few, wide ranges of values for the *above.50*: either low confidence (0.5 to 0.7), middle confidence (0.7 to 0.9), and high confidence (0.9 to 1.0) (confidence that A is better than B).

Table 15 reports the result for the strong interpretation. The predictions made by the BBT are not well calibrated. Much less than the expected proportion of empirical probabilities fall within the different HDIs. In all cases, less than 50% of the empirical probabilities fall within the 90% HDI, which should contain 90% of them. At a first approximation, the *above90* and *below90* values as somewhat similar, which seems to indicates that the BBT model is not systematically over-stating, or under-stating the probability that one algorithm is better than another. It is our understanding that the miss-calibration of the strong interpretation of the parameter is a problem of *variance* (predicting incorrectly the range of possible values) and not a problem of *bias* (predicting incorrectly the most probable value). The errors for the mean prediction are low from 0.02 to 0.01 (as expected for low bias). Thus we believe that the BBT is calibrated for its mean prediction but too overconfident on the range of possible values, its *credal interval*.

The local ROPE concept was included into the model both as an aesthetically point (why should small differences that are expected when computing the average of the cross-validations count as a win for one algorithm?) but also to reduce the model's overconfidence on its certainty regarding the estimates. The local ROPE decreases the number of wins of one algorithm against another which should reduce the BBT confidence on the probability that one is better than another, widening the credal interval. But, as Table 15 shows, the introduction of the local ROPE had little effect on widening the credal interval, although it did have a small effect in lowering the error.

Table 16 displays the results of the calibration of the prediction under the weak interpretation. The calibration seems better than that for the strong interpretation. Predictions of the *above.50* in the 50-to-70 range are few and they are close the empirical results for all

lrope	paired	pred50-70	real50-70	pred70-90	real70-90	pred90-100	real90-100
ss							
F	F	4.3	3	13.8	11	75.3	64
T	F	8.1	7	10.7	7	73.3	63
T	T	6.1	4	11.1	8	75.2	65
mm							
F	F	11.5	7	27.4	25	394.8	361
T	F	9.9	8	25.2	20	398.4	370
T	T	9.0	12	33.1	25	392.0	363
sl							
F	F	1.9	2	0.0	0	46.6	45
T	F	2.0	3	0.0	0	46.5	45
T	T	2.0	3	0.0	0	46.5	45

Table 16: Prediction results - weak interpretation

three use cases. The same is true for the 70-to-90 range. For the high confidence range, from 90-to-100, the predictions seems a little too confident, a little higher than the empirical value. The BBT model seems a little too confident on its predictions and the introduction of the local ROPE and the paired local ROPE also made no difference on that over-confidence.

9. Discussion

BBT as a comparison procedure has advantages over some of the standard frequentist approaches. Even if one remains within a dichotomous decision of accepting or not the difference between two algorithms in a aggregated ranking, BBT seems to supersede Demsar's procedure and seems to be stronger (but not necessarily supersedes) pairwise Wilcoxon based approaches. Thus, BBT makes more decisions of "significant" differences than these two frequentist tests. Benavoli et al. (2017) also show that the BSR procedure makes more decisions than Demsar's where comparing only pairs of classifiers.

Comparing BBT with BSR is harder, there is not much agreement between the numbers generated by the two procedures. But BSR has some limitations that are addressed in BBT: it is yet unclear whether BSR can be used for multiple comparisons, the ROPE proposed by BSR is valid only for accuracy and not other comparable metrics, and BSR cannot be used when comparing incomparable metrics.

We mentioned above that most of the comparison procedures are "explanatory" in the sense that they explain the data at hand: given this data how much confidence one has in stating claims of differences between the algorithms. The explanatory aspect of BBT seems very reasonable. The predictive posterior check show that the Bayesian model is indeed a good model of the data that was given, and it showed that a more complex model such as Davidson's is not needed and it worsens the fitness between the model and the data. The hyper prior tests (Section A) show that the model is not sensitive to different reasonable hyper priors, and the test on different policies to deal with ties (Section 6.2), besides using

the Davidson model, are basically equivalent. All this should point to the conclusion that the model is stable to different decisions, and it is generally a good fit to the data.

As we already mentioned, the mode is simple, the MCMC converges well with few samples, and our implementation (thanks to Stan's MCMC) runs in less than a second in a modern laptop.

Regarding the "predictive" part of the model it is yet unclear whether the apparent overconfidence of the model, specially under the strong interpretation is a problem. Since no frequentist test can make predictions, and the BSR did not test its predictive fitness, we do not have an alternative to compare against.

Regarding missing values, the cases where an algorithm cannot run on one or more data sets, the BBT model simply does not count a win or a loss for that algorithm in comparison to the others for that data set. For example, let us assume that the algorithm "xgb" does not run on the first two data sets in Table 1 (data sets "biomed" and "breast"). The resulting wintable is displayed in Table 17a, which should be contrasted with the wintable in Table 2b, and the summary results are displayed in Table 17b, which should be contrasted with the results in Table 5.

9.1 Code and data availability

An R package (bbtcomp) that implements the BBT model is available at `https://github.com/jwainer/bbtcomp`. To install it use `remotes::install_github("jwainer/bbtcomp")`. The R package uses the `cmdstanr` package to interface with Stan which implements the MCMC sampler. At the time of the writing, `cmdstanr` is not available in CRAN, and should be installed following the instructions in `https://mc-stan.org/cmdstanr/`.

A Python program that implements all functionalities of the R package implementation the BBT model with the exception of the graphic generating functions, is available in the github directory above, in the folder `python`. The program also uses the `cmdstanpy` interface to Stan.

The code that generated the examples and experiments in this paper, as well as the Rmarkdown version of this paper is available in the github directory above, in the folder `paper`.

9.2 How to use the BBT model - weak interpretation and 0.95 probabilities

We believe that there are two basic approaches on using the BBT model on one's own research. If the researcher or the researcher's audience is more accustomed to the frequentist approach, we suggest the researcher uses the summary values related to the weak interpretation of the parameters (the *above.50* and *in.rope* values) and 0.95 probability threshold. In summary:

- if the *in.rope* value is 0.95 or above the researcher can claim that the two algorithms are equivalent (according to the definition of ROPE)
- if the *above.50* is 0.95 or above the researcher can claim that one algorithm is better than the other. Notice that both *in.rope* and *above.50* can be above 0.95, in this case the first rule above applies, and one should claim that the algorithms are equivalent.

alg1	alg2	win1	win2	ties
dt	lda	7	14	0
dt	lgbm	2	19	0
dt	xgb	2	17	0
dt	svm	6	15	0
lda	lgbm	7	14	0
lda	xgb	6	13	0
lda	svm	5	15	0
lgbm	xgb	10	9	0
lgbm	svm	11	10	0
xgb	svm	10	9	0

(a) Wintable

pair	mean	delta	above.50	in.rope
lgbm > xgb	0.51	0.23	0.54	0.51
lgbm > svm	0.54	0.23	0.73	0.44
lgbm > lda	0.70	0.19	1.00	0.01
lgbm > dt	0.82	0.15	1.00	0.00
xgb > svm	0.53	0.23	0.68	0.45
xgb > lda	0.69	0.20	1.00	0.02
xgb > dt	0.82	0.16	1.00	0.00
svm > lda	0.66	0.21	0.99	0.05
svm > dt	0.79	0.17	1.00	0.00
lda > dt	0.66	0.22	0.99	0.06

(b) Results for the corresponding BBT model

Table 17: "Results when *xrg* does not run on the first two data sets."

- for all other cases, the researcher should make no claim.

This approach include threshold numbers that are familiar to the frequentist test user, namely the 0.95 or 95%, and as we have seen in section 7.1, this approach should be stronger than Demsar's procedure, in the sense that it will select more pairwise comparisons as "significant" and will not miss any of the comparisons deemed significant by Demsar's procedure. BBT will also select more comparisons as "significant" as the pairwise Wilcoxon procedure (section 7.2).

The BBT model also allows one to make claims of equivalence (when *in.rope* ≥ 0.95) that are not possible in the frequentist case.

Regarding the explanatory aspect of the procedure, for the pairs of algorithms with "significant" differences, the researcher can claim that for the *data given*, 95% or more of the estimates of the relative strengths of the algorithms indicates that algorithm A is better than algorithm B.

Finally, the BBT model is reasonable well calibrated although slightly overconfident regarding high values of *above.50*, so for the pairs with "significant" differences the researcher can make the claim that, *likely* with 90% probability or better, the best algorithm should perform better than the worse for *future data sets*.

9.3 How to use the BBT model - strong interpretation and 0.70 probabilities

If the researchers and their audiences are more comfortable with Bayesian results, we recommend following the strong interpretation (and the *mean* summary of the probabilities) and choosing a threshold of 0.70. The procedure is:

- if the *mean* is below 0.55 one claims that both algorithms are equivalent
- if the *mean* is above 0.70 one claims that one algorithm is "significantly" better than the other.

Regarding the explanatory aspect, the researcher can state the claim that for *the data given* the mean estimate to the probability that algorithm A is better than B is at least 70%, and the 89% (or 95% if the researcher prefers) of the estimates fall within the *low* to *high* credal interval.

Regarding *future data*, the researcher can make the claim that the *most likely* value for the probability that A is better than B is 70% or better. The low predictive bias allows one to make that claim, but the high variance does not allow one to define a credal interval for those estimates.

The strong interpretation with 0,70 threshold is somewhat comparable with the Demsar's results for small-small use cases, but become a more stringent criteria for the other use cases, that is, some pairs are determined to be significantly different in Demsar's procedure but *not* using the 0.70 threshold. For the repeated experiments:

- for **ss** use cases: the 0.70 threshold detects 18 new pairs of comparisons (out of 100) that Demsar's procedure does not find significantly different, and detects 9 pairs that the pairwise Wilcoxon does not find significantly different, but does not detect 5 that the pairwise Wilcoxon does.
- for **mm** use cases: the 0.70 threshold find 10 new pairs (out of 100), but misses 7 in comparison to Demsar, and finds new 27 but misses 73 in comparision to pairwise Wilcoxon.
- for **sl** use cases: the 0.70 threshold finds no new pairs, but misses 11 (out of 50) in comparison to Demsar's, and also misses 17 in comparison to the pairwise Wilcoxon.
- for the **ll** results: the 0.70 threshold misses 16 (out of 120) pairs in comparison to Demsar, and misses 38 in comparison to Wilcoxon.

Therefore, using the 0.70 strong interpretation threshold will result in less pairs flagged as "different" as the size of the comparisons increases. But, different than any frequentist-based "significantly different" pair of algorithms, one can still claim that for those flagged pairs in the BBT procedure, the best algorithm in each pair will likely perform better that the other one in 70% of the future data sets.

10. Conclusion

The BBT model is a comparison procedure based on the Bradley-Terry model that defines a merit number for each of K players that compete on pairwise matches, and relates these merit numbers to the probability that a player will win a match. The BBT model is a Bayesian implementation of the Bradley-Terry model, with the advantages of a Bayesian estimation procedure over a frequentist one:

- one can use a more subtle language regarding each pair of algorithms in the aggregated ranking, beyond just claiming that the difference is significant or not
- one can define a threshold below which two algorithms are equivalent for practical purposes, and make claims regarding whether two algorithms are equivalent or not.
- one has some understanding on the uncertainties regarding the claims

Beyond a Bayesian advantage, the BBT model also:

- works on any metric of interest, whether comparable or not
- the main parameters are probabilities, so one can have a more intuitive understanding on how to define ROPEs, how to understand uncertainty, and so on
- the procedure can accommodate missing data regarding algorithms that did not run on particular data sets

Finally, we also defined the concept of a local ROPE, which is a procedure to decide when to state that one algorithm is "really" better than another for a particular data set, based on their average measure over different folds. We believe that local ROPE is a useful procedure for frequentist tests, in particular rank based tests.

This paper did not spend too much space defending the use of Bayesian tests – we refer the reader to Benavoli et al. (2017) which not only presents the BSR model but makes convincing and well justified arguments for the Machine Learning community to move away from frequentist tests and into Bayesian tests. This author agrees with those arguments and proposals.

A. Alternative hyper-priors

The BBT Model uses a lognormal distribution with a 0.5 scale parameter as hyper-prior for the σ parameter, as proposed in Carpenter (2018). But that author also suggested using different scales for the lognormal distribution as well as a Cauchy distribution (with scale = 1.0). On the other hand, Issa Mattos and Martins Silva Ramos (2021) proposes a Normal distribution with standard deviation 3.0 as that hyper-prior. We tested these alternatives and they make no important difference. We performed the usual repeated experiments, measuring WAIC and the calibration of the PPC as in Tables 18 and 19

	ss					mm					
hyper	**scale**	**waic**	**h50**	**h90**	**h95**	**h100**	**scale**	**waic**	**h50**	**h90**	**h95**	**h100**
cauchy	0.50	40.85225	0.83	1	1	1	0.50	219.7626	0.728	0.998	1	1
cauchy	1.00	40.76751	0.83	1	1	1	1.00	219.7257	0.715	0.996	1	1
lognormal	0.25	40.58881	0.87	1	1	1	0.25	219.6236	0.708	0.996	1	1
lognormal	0.50	40.56506	0.90	1	1	1	0.50	219.6175	0.699	0.996	1	1
lognormal	1.00	40.73847	0.85	1	1	1	1.00	219.6493	0.728	0.996	1	1
normal	2.00	40.71471	0.84	1	1	1	2.00	219.5983	0.722	0.994	1	1
normal	5.00	40.69713	0.86	1	1	1	5.00	219.6517	0.705	0.996	1	1

Table 18: The PPC of ss and mm use cases for the different hyper-priors

	sl					ll					
hyper	**scale**	**waic**	**h50**	**h90**	**h95**	**h100**	**scale**	**waic**	**h50**	**h90**	**h95**	**h100**
cauchy	0.50	62.51531	0.64	1.00	1	1	0.50	767.2083	0.48	0.92	0.96	1
cauchy	1.00	62.33314	0.66	1.00	1	1	0.25	766.7715	0.47	0.92	0.96	1
lognormal	0.25	62.36310	0.68	1.00	1	1	1.00	767.1563	0.46	0.92	0.97	1
lognormal	0.50	62.50422	0.62	1.00	1	1	1.00	767.2467	0.46	0.92	0.96	1
lognormal	1.00	62.59410	0.62	0.98	1	1	0.50	767.6128	0.48	0.93	0.96	1
normal	2.00	62.45112	0.64	1.00	1	1	2.00	767.7501	0.47	0.92	0.97	1
normal	5.00	62.54199	0.70	1.00	1	1	5.00	768.0781	0.48	0.92	0.96	1

Table 19: The PPC of sl and ll use cases the different hyper-priors

References

Jesus Alcala-Fdez, Alberto Fernandez, Julian Luengo, Joaquin Derrac, Salvador Garcia, Luciano Sanchez, and Francisco Herrera. Keel data-mining software tool: data set repository, integration of algorithms and experimental analysis framework. *Journal of Multiple-Valued Logic & Soft Computing*, 17, 2011.

Rose Baker and Philip Scarf. Modifying bradley–terry and other ranking models to allow ties. *IMA Journal of Management Mathematics*, 32(4):451–463, 2021.

Alessio Benavoli, Giorgio Corani, and Francesca Mangili. Should we really use post-hoc tests based on mean-ranks? *The Journal of Machine Learning Research*, 17(1):152–161, 2016.

Alessio Benavoli, Giorgio Corani, Janez Demsar, and Marco Zaffalon. Time for a change: a tutorial for comparing multiple classifiers through bayesian analysis. *The Journal of Machine Learning Research*, 18(1):2653–2688, 2017.

Ralf Bender and Stefan Lange. Adjusting for multiple testing—when and how? *Journal of clinical epidemiology*, 54(4):343–349, 2001.

Bernd Bischl, Giuseppe Casalicchio, Matthias Feurer, Frank Hutter, Michel Lang, Rafael G. Mantovani, Jan N. van Rijn, and Joaquin Vanschoren. Openml benchmarking suites. *arXiv:1708.03731v2 [stat.ML]*, 2019.

Ralph Allan Bradley and Milton E Terry. Rank analysis of incomplete block designs: I. the method of paired comparisons. *Biometrika*, 39(3/4):324–345, 1952.

Kenneth Butler and John T. Whelan. The existence of maximum likelihood estimates in the bradley-terry model and its extensions, 2004. URL `https://arxiv.org/abs/math/0412232`.

Francois Caron and Arnaud Doucet. Efficient bayesian inference for generalized bradley–terry models. *Journal of Computational and Graphical Statistics*, 21(1):174–196, 2012.

Bob Carpenter. The bradley-terry model of ranking via paired comparisons, 2018. URL `https://github.com/stan-dev/example-models/tree/master/knitr/bradley-terry`.

Asli Celikyilmaz, Elizabeth Clark, and Jianfeng Gao. Evaluation of text generation: A survey, 2020. URL `https://arxiv.org/abs/2006.14799`.

Stephane Champely. *pwr: Basic Functions for Power Analysis*, 2020. URL `https://CRAN.R-project.org/package=pwr`. R package version 1.3-0.

Tianqi Chen and Carlos Guestrin. XGBoost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, pages 785–794, New York, NY, USA, 2016. ACM. ISBN 978-1-4503-4232-2. doi: 10.1145/2939672.2939785. URL `http://doi.acm.org/10.1145/2939672.2939785`.

Jacob Cohen. *Statistical power analysis for the behavioral sciences*. Lawrence Erlbaum Associates, 1988.

Giorgio Corani, Alessio Benavoli, Janez Demsar, Francesca Mangili, and Marco Zaffalon. Statistical comparison of classifiers through bayesian hierarchical modelling. *Machine Learning*, 106(11):1817–1837, 2017.

Roger R Davidson. On extending the bradley-terry model to accommodate ties in paired comparison experiments. *Journal of the American Statistical Association*, 65(329):317–328, 1970.

Janez Demsar. Statistical comparisons of classifiers over multiple data sets. *The Journal of Machine Learning Research*, 7:1–30, 2006.

Alessio Farcomeni. A review of modern multiple hypothesis testing, with particular attention to the false discovery proportion. *Statistical methods in medical research*, 17(4):347–388, 2008.

Franz Faul, Edgar Erdfelder, Albert-Georg Lang, and Axel Buchner. G* power 3: A flexible statistical power analysis program for the social, behavioral, and biomedical sciences. *Behavior research methods*, 39(2):175–191, 2007.

Ronald J Feise. Do multiple outcome measures require p-value adjustment? *BMC medical research methodology*, 2(1):1–4, 2002.

Manuel Fernandez-Delgado, Eva Cernadas, Senen Barro, and Dinani Amorim. Do we need hundreds of classifiers to solve real world classification problems? *The journal of machine learning research*, 15(1):3133–3181, 2014.

Salvador Garcia and Francisco Herrera. An extension on" statistical comparisons of classifiers over multiple data sets" for all pairwise comparisons. *Journal of machine learning research*, 9(12), 2008.

Salvador Garcia, Alberto Fernandez, Julian Luengo, and Francisco Herrera. Advanced non-parametric tests for multiple comparisons in the design of experiments in computational intelligence and data mining: Experimental analysis of power. *Information sciences*, 180 (10):2044–2064, 2010.

Andrew Gelman, Jennifer Hill, and Masanao Yajima. Why we (usually) don't have to worry about multiple comparisons. *Journal of research on educational effectiveness*, 5(2): 189–211, 2012.

Andrew Gelman, Daniel Simpson, and Michael Betancourt. The prior can often only be understood in the context of the likelihood. *Entropy*, 19(10):555, 2017.

David R Hunter. Mm algorithms for generalized bradley-terry models. *The annals of statistics*, 32(1):384–406, 2004.

David Issa Mattos and Erika Martins Silva Ramos. Bayesian paired comparison with the bpcs package. *Behavior Research Methods*, pages 1–21, 2021.

Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. Lightgbm: A highly efficient gradient boosting decision tree. *Advances in neural information processing systems*, 30, 2017.

Aris Kosmopoulos, Ioannis Partalas, Eric Gaussier, Georgios Paliouras, and Ion Androutsopoulos. Evaluation measures for hierarchical classification: a unified view and novel approaches. *Data Mining and Knowledge Discovery*, 29(3):820–865, 2015.

John Kruschke. *Doing Bayesian data analysis: A tutorial with R, JAGS, and Stan*. Academic Press, 2014.

John K Kruschke. Bayesian estimation supersedes the t test. *Journal of Experimental Psychology: General*, 142(2):573, 2013.

John K Kruschke and Torrin M Liddell. The bayesian new statistics: Two historical trends converge. *SSRN Electronic Journal*, 2606016, 2015.

Nathan P Lemoine. Moving beyond noninformative priors: why and how to choose weakly informative priors in bayesian analyses. *Oikos*, 128(7):912–928, 2019.

Wanshan Li, Shamindra Shrotriya, and Alessandro Rinaldo. The performance of the mle in the bradley-terry-luce model in ℓ_∞-loss and under general graph topologies, 2021. URL `https://arxiv.org/abs/2110.10825`.

Christian List. Social Choice Theory. In Edward N. Zalta, editor, *The Stanford Encyclopedia of Philosophy*. Metaphysics Research Lab, Stanford University, Spring 2022 edition, 2022.

Dominique Makowski, Mattan S Ben-Shachar, and Daniel Ludecke. bayestestr: Describing effects and their uncertainty, existence and significance within the bayesian framework. *Journal of Open Source Software*, 4(40):1541, 2019.

Randal S. Olson, William La Cava, Patryk Orzechowski, Ryan J. Urbanowicz, and Jason H. Moore. Pmlb: a large benchmark suite for machine learning evaluation and comparison. *BioData Mining*, 10(36):1–13, Dec 2017. ISSN 1756-0381. doi: 10.1186/s13040-017-0154-4. URL `https://doi.org/10.1186/s13040-017-0154-4`.

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

R core. p.adjust: Adjust p-values for multiple comparisons (version 3.6.2). https://www.rdocumentation.org/packages/stats/versions/3.6.2/topics/p.adjust, 2022.

PV Rao and Lawrence L Kupper. Ties in paired-comparison experiments: A generalization of the bradley-terry model. *Journal of the American Statistical Association*, 62(317): 194–204, 1967.

Nery Riquelme, Christian Von Lucken, and Benjamin Baran. Performance metrics in multi-objective optimization. In *2015 Latin American computing conference (CLEI)*, pages 1–11. IEEE, 2015.

Kenneth J Rothman. No adjustments are needed for multiple comparisons. *Epidemiology*, pages 43–46, 1990.

Vivekananda Roy. Convergence diagnostics for markov chain monte carlo, 2019. URL `https://arxiv.org/abs/1909.11827`.

Cosma Shalizi. Any p-value distinguishable from zero is insufficiently informative. http://bactra.org/weblog/1111.html, May 2015.

Stan Development Team. Stan modeling language users guide and reference manual, version 2.29, 2022. URL `https://mc-stan.org`.

Katarzyna Stapor, Pawel Ksieniewicz, Salvador Garcia, and Michal Wozniak. How to design the fair experimental classifier evaluation. *Applied Soft Computing*, 104:107219, 2021. ISSN 1568-4946. doi: https://doi.org/10.1016/j.asoc.2021.107219. URL `https://www.sciencedirect.com/science/article/pii/S1568494621001423`.

Aki Vehtari, Andrew Gelman, and Jonah Gabry. Practical bayesian model evaluation using leave-one-out cross-validation and waic. *Statistics and computing*, 27(5):1413–1432, 2017.

Michael Wainberg, Babak Alipanahi, and Brendan J Frey. Are random forests truly the best classifiers? *The Journal of Machine Learning Research*, 17(1):3837–3841, 2016.

Jacques Wainer. Comparison of 14 different families of classification algorithms on 115 binary datasets, 2016. URL `https://arxiv.org/abs/1606.00930`.

Jacques Wainer and Gavin Cawley. Nested cross-validation when selecting classifiers is overzealous for most practical applications. *Expert Systems with Applications*, 182:115222, 2021. ISSN 0957-4174.

Jacques Wainer and Pablo Fonseca. How to tune the rbf svm hyperparameters? an empirical evaluation of 18 search algorithms. *Artificial Intelligence Review*, 54(6):4771–4797, 2021.

Shenhao Wang, Baichuan Mo, Stephane Hess, and Jinhua Zhao. Comparing hundreds of machine learning classifiers and discrete choice models in predicting travel behavior: an empirical benchmark. *arXiv preprint arXiv:2102.01130*, 2021.

Sumio Watanabe and Manfred Opper. Asymptotic equivalence of bayes cross validation and widely applicable information criterion in singular learning theory. *Journal of machine learning research*, 11(12), 2010.

Min-Ling Zhang and Zhi-Hua Zhou. A review on multi-label learning algorithms. *IEEE Transactions on Knowledge and Data Engineering*, 26(8):1819–1837, 2014. doi: 10.1109/TKDE.2013.39.