Bayesian Comparison of Machine Learning Algorithms on Single and Multiple Datasets

Alexandre Lacoste Alexandre.Lacoste.1@ulaval.ca
François Laviolette Francois.Laviolette@ift.ulaval.ca
Mario Marchand Mario.Marchand@ift.ulaval.ca
Département d'informatique et de génie logiciel, Université Laval, Québec, Canada

Abstract

We propose a new method for comparing learning algorithms on multiple tasks which is based on a novel non-parametric test that we call the Poisson binomial test. The key aspect of this work is that we provide a formal definition for what is meant to have an algorithm that is better than another. Also, we are able to take into account the dependencies induced when evaluating classifiers on the same test set. Finally we make optimal use (in the Bayesian sense) of all the testing data we have. We demonstrate empirically that our approach is more reliable than the sign test and the Wilcoxon signed rank test, the current state of the art for algorithm comparisons.

1 Introduction

In this paper, we address the problem of comparing machine learning algorithms using testing data. More precisely, we provide a method that verifies if the amount of testing data is sufficient to support claims such as : "Algorithm $\mathcal{A}$ is better than Algorithm $\mathcal{B}$". This is particularly useful for authors who want to demonstrate that their newly designed learning algorithm is significantly better than some state of the art learning algorithm.

Many published papers simply compare the empirical test risk of the classifiers produced by their learning algorithms. This is insufficient. Since the testing data is randomly sampled from the original task, repeating the experiment might lead to different conclusions. Moreover, when the goal is to compare the generalization performances of learning algorithms, more than one task must be taken into consideration.

Appearing in Proceedings of the 15th International Conference on Artificial Intelligence and Statistics (AISTATS) 2012, La Palma, Canary Islands. Volume 22 of JMLR: W&CP 22. Copyright 2012 by the authors.

In an effort to quantify the differences between learning algorithms, some authors estimate the variance of the risk over the different folds of cross validation. However, since the different training sets are correlated, this violates the independence assumption required by the variance estimator. Moreover, Bengio and Grandvalet [BG04, BG05] proved that there is no unbiased variance estimator for k-fold cross validation. To overcome this problem, Dietterich [Die98, Alp] developed the 5×2 cross validation which performs an average over a *quasi*-unbiased variance estimator.

Langford [Lan06] observed that the number of classification errors follows a binomial distribution and proposed a probabilistic testing set bound for the risk of classifiers. A lower and an upper bound on the true risk is then used to determine if the observed difference in empirical testing errors implies that the true risks differ with high confidence. While this non-parametric approach is rigorous and statistically valid, it has low statistical power. Indeed, in practice, the method often claims that there is not enough data to assert any statistical differences.

When the goal is to identify if an algorithm is more suited for general learning than another, both algorithms are analyzed over several datasets. In this situation, some authors propose to average the risk from the different tasks. This is also incorrect. Indeed, consider an algorithm that fails to obtain a good classifier on a task where the risk usually lies around 0.1. This would draw shadow on all the good work this algorithm could perform on tasks having low risk (around 0.01). Thus, we adopt the common viewpoint that the risk is *incommensurable* [Dem06] across the different tasks.

To address the incommensurability issue, methods such as the sign test [Men83] choose to ignore the magnitude of the difference on each task and simply count how many times the algorithm is better than its competitor on the set of tasks. When more than one learning algorithm is available, the Friedman test [Fri37] averages the rank of the classifiers across the different tasks, where each rank is obtained by sorting the empirical test risk of the produced classifiers for a particular task. This metric is able to partially take into account the amount of differences between the risk of classifiers. However, when only two algorithms are compared, it becomes equivalent to the sign test. Alternatively, it is possible to use the Wilcoxon signed rank (WSR) test [Wil45]. Instead of simply counting the number of times an algorithm is better than its competitor, like the sign test does, each count is weighted by the rank of the amount of differences in the risk. More precisely, the absolute differences between the empirical test risk of the produced classifiers are sorted across the different tasks. Then the rank of this magnitude is used to weight the counts for both algorithms.

To demystify which methods are appropriate, Demšar [Dem06] performed a benchmark and concludes that non-parametric tests such as the sign test, the WSR test [Wil45] and the Friedman test [Fri37] are safer to use than methods that assume a normal distribution of the data such as a the t-test or ANOVA. He also concludes that methods assuming some but limited commensurability such as the Friedman test and the WSR test, have more power than the sign test.

Our Contributions

Inspired by the work of Langford [Lan06], we use the fact that each error performed by a classifier on a test set follows a Bernoulli law. In other words, the *true risk* [1] of a classifier is defined as the probability of performing an error on the given task. Instead of obtaining bounds on the *true risk* like Langford did, we answer a simpler question : "Does classifier h have a smaller *true risk* than classifier g?". It is simpler in the sense that instead of having to estimate two real values, we only have a single binary variable to estimate. Since the test set has only a finite amount of samples, we use the Bayesian methodology to obtain the probability of either outcome. Using this approach, we are also able to take into account the dependencies induced when evaluating classifiers on the same test set.

To be able to compare two learning algorithms on several tasks, we introduce a new concept called a *context*. It represents a distribution over the different tasks a learning algorithm is meant to encounter. Then, each time a task is sampled from the context, whether or not algorithm $\mathcal{A}$ produces a better classifier than algorithm $\mathcal{B}$ follows a Bernoulli law. This means that we do not need to explicitly know the underlying distribution of the context to obtain a probabilistic answer to the following question : "Does algorithm $\mathcal{A}$ have a higher chance of producing a better classifier than algorithm $\mathcal{B}$ in the given context?".

To compare our new methodology to the sign test and the WSR test, we apply the different methods on a wide range of synthetic contexts[2]. Then, an analysis of the false positives and false negatives shows that the newly proposed method constantly outperforms these widely used tests.

The key aspect of this work is that we provide a formal definition for what is meant to have an algorithm that is better than another. Also, we are able to take into account the dependencies induced when evaluating classifiers on the same test set. Finally we make optimal use (in the Bayesian sense) of all the testing data we have. Also, note that all algorithms described in this paper are available as open source software on http://code.google.com/p/mleval/.

2 Theoretical Setup

We consider the classification problem. In the single task case, the input space $\mathcal{X}$ is an arbitrary set and the output space $\mathcal{Y} \subset \mathbb{N}$ denotes the set of possible *classes*. An *example* is an input-output pair (x, y) where $x \in \mathcal{X}$ and $y \in \mathcal{Y}$. Throughout the paper we make the usual assumption that each example is drawn according to an unknown distribution $\mathcal{D}$ on $\mathcal{X} \times \mathcal{Y}$. A *classifier* is a function $h : \mathcal{X} \longrightarrow \mathcal{Y}$. The *risk* $R_{\mathcal{D}}(h)$ of classifier h on distribution $\mathcal{D}$ is defined as $\underset{(x,y)\sim\mathcal{D}}{\mathbf{E}} I(h(x) \neq y)$ where $I(a) = 1$ if predicate a is true and $I(a) = 0$ otherwise. We say that classifier h is better than classifier g with respect to $\mathcal{D}$ (denoted as $h \overset{\mathcal{D}}{\succ} g$) when $R_{\mathcal{D}}(h) < R_{\mathcal{D}}(g)$.

A learning algorithm is a function $\mathcal{A}$ that takes, as input, a set of examples called the *training set* and returns, as output, a classifier. Our goal is to find a metric that determines if an algorithm $\mathcal{A}$ is better than another algorithm $\mathcal{B}$. In order to do so, we assume that the input-output space $\mathcal{X} \times \mathcal{Y}$, the data-generating distribution $\mathcal{D}$ on $\mathcal{X} \times \mathcal{Y}$, and the number m of training examples are sampled i.i.d. from an (unknown) distribution $\mathcal{W}$. We refer to $\mathcal{W}$ as being the *context* from which the different datasets are drawn.

[1] The *true risk* is the limit value obtained with probability 1 when the size of the test set goes to infinity.

[2] Note that, if one wants to compare these different methodology, the resulting experiments will have to be performed on contexts for which the underlying distribution $\mathcal{W}$ is known, because, otherwise, it is impossible to state which algorithm is "really" better.

Given two learning algorithms, $\mathcal{A}$ and $\mathcal{B}$, and a context $\mathcal{W}$, we define $\tilde{q}_{\mathcal{AB}|\mathcal{W}}$ as the probability that $\mathcal{A}$ is better than $\mathcal{B}$ within the context $\mathcal{W}$, *i.e.*,

$$\tilde{q}_{\mathcal{AB}|\mathcal{W}} \overset{\text{def}}{=} \mathop{\mathbf{E}}_{(\mathcal{D},m)\sim\mathcal{W}} \mathop{\mathbf{E}}_{S\sim\mathcal{D}^m} I\left[\mathcal{A}(S) \overset{\mathcal{D}}{\succ} \mathcal{B}(S)\right] \quad (1)$$

Consequently, we say that $\mathcal{A}$ is better than $\mathcal{B}$ within the context $\mathcal{W}$ iff $\tilde{q}_{\mathcal{AB}|\mathcal{W}} > 1/2$, *i.e.*,

$$\mathcal{A} \overset{\mathcal{W}}{\succ} \mathcal{B} \quad \Leftrightarrow \quad \tilde{q}_{\mathcal{AB}|\mathcal{W}} > \tfrac{1}{2} \quad (2)$$

The definition of $\tilde{q}_{\mathcal{AB}|\mathcal{W}}$ is simple but, in practice, we do not observe $\mathcal{W}$ nor any of the sampled $\mathcal{D}$. It is thus not possible to directly determine if $\mathcal{A} \overset{\mathcal{W}}{\succ} \mathcal{B}$. However, we are able to provide a probabilistic answer by using Bayesian statistics.

3 Bounding the Risk

We cannot evaluate the true risk of a classifier h with respect to $\mathcal{D}$. However, using a test set $T \sim \mathcal{D}^n$, we can evaluate a probability distribution over the possible values of $R_{\mathcal{D}}(h)$.

This is done by first observing that $k_h \overset{\text{def}}{=} nR_T(h)$ follows a binomial law of parameter $p \overset{\text{def}}{=} R_{\mathcal{D}}(h)$. Next, we will use Bayes' theorem to obtain a posterior distribution over p. But first, we need to provide a prior distribution.

Since the beta distribution is the conjugate prior of the binomial distribution, it is wise to use it for the prior over p, with parameters $\alpha' > 0$ and $\beta' > 0$:

$$B(p; \alpha', \beta') \overset{\text{def}}{=} \tfrac{\Gamma(\alpha'+\beta')}{\Gamma(\alpha')\Gamma(\beta')} p^{\alpha'-1}(1-p)^{\beta'-1},$$

where Γ denotes the gamma function. It follows from Bayes' theorem, that the posterior will also be a beta distribution, but this time, with parameters $\alpha \overset{\text{def}}{=} \alpha' + k_h$ and $\beta \overset{\text{def}}{=} \beta' + n - k_h$.

To complete the choice of the prior, we still need to specify the values for α' and β'. This could be achieved by using some information coming from the training set or some knowledge we might have about the learning algorithm. However, to stay as neutral as possible as advocated by Jaynes [Jay57], we chose the least informative prior—which is the uniform prior given by $\alpha' = \beta' = 1$.

Now that we have a probability distribution over $R_{\mathcal{D}}(h)$, we can use the cumulative distribution, to obtain the probability that the risk is smaller than x :

$$B_c(x; \alpha, \beta) \overset{\text{def}}{=} \int_0^x B(t; \alpha, \beta)\, dt$$

From the nature of $B(\,\cdot\,; \alpha, \beta)$, $B_c(\,\cdot\,; \alpha, \beta)$ is a one to one relation. Therefore, it has an inverse that we denote by $B_c^{-1}(\,\cdot\,; \alpha, \beta)$.

Using this inverse function, we can now obtain a probabilistic upper bound of the risk

$$\Pr\left(\, R_{\mathcal{D}}(h) \;\le\; B_c^{-1}(1{-}\delta; \alpha, \beta) \,\right) \ge 1{-}\delta \quad (3)$$

where δ is the probability that our bound fails and is commonly set to small values such as 0.05. Since the bound also depends on δ, smaller values loosen the bound while higher values tighten the bound.

When $\alpha' = 1$ and in the limit where $\beta' \to 0$, we converge to the bound described in Theorem 3.3 of Langford's tutorial [Lan06]. This can be shown using the following identity [AS64]

$$\sum_{i=0}^{k} \binom{n}{i} p^i (1-p)^{n-i} = 1 - B_c(p, k{+}1, n{-}k)\,, \quad (4)$$

to rewrite the Langford's test-set bound as follows : $\Pr\left(\, R_{\mathcal{D}}(h) \;\le\; B_c^{-1}(1{-}\delta; k_h{+}1, n{-}k_h) \,\right) \ge 1{-}\delta$. Consequently, the probabilistic risk upper bound of Equation (3), given by the Bayesian posterior, becomes Langford's test-set bound when the prior has all its weight on $R_{\mathcal{D}}(h) = 1$ (*i.e.*, when we assume the worst). Finally, a numerical evaluation of B_c^{-1} can be computed using a Newton method and is available in most statistical software.

4 Probabilistic Discrimination of Learning Algorithms on a Single Task

Let $\mathcal{T}$ be any fixed classification task, or equivalently, let $\mathcal{D}$ be any fixed (but unknown) probability distribution on $\mathcal{X} \times \mathcal{Y}$. Let also $\mathcal{A}$ and $\mathcal{B}$ be two learning algorithms. In this section, we want to evaluate, for the task $\mathcal{T}$, the probability that a classifier obtained using $\mathcal{A}$ is better than a classifier obtained using $\mathcal{B}$.

In the previous section, we saw that, using a test set $T \sim \mathcal{D}^n$ we can obtain a probability distribution over $R_{\mathcal{D}}(\mathcal{A}(S))$. Similarly, with a second test set $T' \sim \mathcal{D}^{n'}$, we can obtain a probability distribution over $R_{\mathcal{D}}(\mathcal{B}(S'))$. When both training sets and both test sets are independent, the joint distribution over $R_{\mathcal{D}}(\mathcal{A}(S))$ and $R_{\mathcal{D}}(\mathcal{B}(S'))$ is simply the product of the individual distributions. From this joint distribution, we can therefore evaluate $\Pr\left(\mathcal{A}(S) \overset{\mathcal{D}}{\succ} \mathcal{B}(S')\right)$ by integrating the probability mass in the region where $R_{\mathcal{D}}(\mathcal{A}(S)) < R_{\mathcal{D}}(\mathcal{B}(S'))$.

However, in most experiments, learning algorithms are sharing the same training set and the same testing set. To take these dependencies into account, let us exploit the fact that the joint error of the pair of classifiers comes from a multinomial distribution.

More precisely, let $h \overset{\text{def}}{=} \mathcal{A}(S)$ and $g \overset{\text{def}}{=} \mathcal{B}(S)$ where $S \sim \mathcal{D}^m$. For each sample $(x,y) \sim \mathcal{D}$, we have the following three outcomes: (1) $h(x) \neq y \wedge g(x) = y$; (2) $h(x) = y \wedge g(x) \neq y$; (3) $h(x) = g(x)$ with probability p_h, p_g and $\overline{p}$ respectively. Moreover, $p_h + p_g + \overline{p} = 1$. Let k_h, k_g, and $\overline{k}$, be the respective counts of these events on a testing set T. Then $(k_h, k_g, \overline{k})$ follows a multinomial law with parameters $(|T|, (p_h, p_g, \overline{p}))$.

Since $R_{\mathcal{D}}(h) < R_{\mathcal{D}}(g)$ whenever $p_h < p_g$, our aim is to obtain a probability distribution over $(p_h, p_g, \overline{p})$, given $(k_h, k_g, \overline{k})$, to be able to integrate the probability mass in the region where $p_h < p_g$. This can be done using Bayes' theorem whenever we have a prior distribution.

Since the Dirichlet distribution is the conjugate prior of the multinomial, it is wise to use it for the prior information we have about p_h, p_g and $\overline{p}$. The Dirichlet distribution of parameters $(\alpha_h, \alpha_g, \overline{\alpha})$ is defined as

$$\begin{aligned} &D\left(p_h, p_g, \overline{p}; \alpha_h, \alpha_g, \overline{\alpha}\right) \\ &\overset{\text{def}}{=} \frac{\Gamma(\alpha_h+\alpha_g+\overline{\alpha})}{\Gamma(\alpha_h)\Gamma(\alpha_g)\Gamma(\overline{\alpha})} p_h^{\alpha_h-1} p_g^{\alpha_g-1} \left(1-p_g-p_h\right)^{\overline{\alpha}-1} \end{aligned}$$

where $\alpha_h > 0$, $\alpha_g > 0$ and $\overline{\alpha} > 0$.

If the prior is a Dirichlet with parameters $\left(\alpha'_h, \alpha'_g, \overline{\alpha}'\right)$, it follows from Bayes' law that, after observing k_h, k_g and $\overline{k}$ on the testing set, the posterior is also a Dirichlet with parameters $(\alpha_h, \alpha_g, \overline{\alpha})$ where $\alpha_h \overset{\text{def}}{=} \alpha'_h + k_h$, $\alpha_g \overset{\text{def}}{=} \alpha'_g + k_g$ and $\overline{\alpha} \overset{\text{def}}{=} \overline{\alpha}' + \overline{k}$. Consequently, the following theorem gives us the desired result for $\Pr\left(h \overset{\mathcal{D}}{\succ} g\right)$.

Theorem 4.1. *Let* $\alpha_h \overset{\text{def}}{=} \alpha'_h + k_h$, $\alpha_g \overset{\text{def}}{=} \alpha'_g + k_g$ *and* $\overline{\alpha} \overset{\text{def}}{=} \overline{\alpha}' + \overline{k}$, *where* $\alpha'_g > 0$, $\alpha'_h > 0$, $\overline{\alpha}' > 0$ *, then*

$$\begin{aligned} &\Pr\left(h \overset{\mathcal{D}}{\succ} g\right) \\ &= \int_0^1 \int_0^{\frac{1-\overline{p}}{2}} D\left(p_g, p_h, \overline{p}\,;\, \alpha_g, \alpha_h, \overline{\alpha}\right)\, dp_h\, d\overline{p} \\ &= B_c\left(\tfrac{1}{2}; \alpha'_h + k_h, \alpha'_g + k_g\right) \end{aligned}$$

Proof. The first equality follows from the explanations above. Now, using $C \overset{\text{def}}{=} \frac{\Gamma(\alpha_h+\alpha_g+\overline{\alpha})}{\Gamma(\alpha_h)\Gamma(\alpha_g)\Gamma(\overline{\alpha})}$, $\gamma \overset{\text{def}}{=} 1 - \overline{p}$ and $z \overset{\text{def}}{=} \frac{p_h}{\gamma}$, we have :

$$\begin{aligned} &\int_0^1 \int_0^{\frac{1-\overline{p}}{2}} D\left(p_g, p_h, \overline{p}\,;\, \alpha_g, \alpha_h, \overline{\alpha}\right)\, dp_h\, d\overline{p} \\ &= C \int_0^1 \overline{p}^{\overline{\alpha}-1} \int_0^{\frac{1-\overline{p}}{2}} p_h^{\alpha_h-1} \left(1-\overline{p}-p_h\right)^{\alpha_g-1}\, dp_h\, d\overline{p} \\ &= C \int_0^1 \overline{p}^{\overline{\alpha}-1} \int_0^{\frac{1}{2}} (\gamma z)^{\alpha_h-1} (\gamma-\gamma z)^{\alpha_g-1} \gamma\, dz\, d\overline{p} \\ &= C \int_0^1 \overline{p}^{\overline{\alpha}-1} \gamma^{\alpha_h+\alpha_g-1}\, d\overline{p} \int_0^{\frac{1}{2}} z^{\alpha_h-1} (1-z)^{\alpha_g-1}\, dz \\ &= \frac{\Gamma(\alpha_h+\alpha_g)}{\Gamma(\alpha_h)\Gamma(\alpha_g)} \int_0^{\frac{1}{2}} z^{\alpha_h-1} (1-z)^{\alpha_g-1}\, dz \\ &\overset{\text{def}}{=} B_c\left(\tfrac{1}{2}; \alpha'_h + k_h, \alpha'_g + k_g\right) \end{aligned}$$

□

4.1 About the Prior

From Theorem 4.1, we see that $\Pr\left(h \overset{\mathcal{D}}{\succ} g\right)$ does not depend on $\overline{k}$ nor $\overline{\alpha}'$. However, to complete the choice of the prior distribution, we still need to provide values for α_h and α_g. It might be possible to extract information from the training set using cross-validation. However, this approach is not straightforward and will require further investigation in future work. Also, we should not introduce favoritism by using an imbalanced prior. Hence, one should use $\alpha_h = \alpha_g \overset{\text{def}}{=} \widetilde{\alpha}$. This leaves us with only one parameter, $\widetilde{\alpha}$, for the prior. Using $\widetilde{\alpha} > 1$ is equivalent to supposing, *a priori*, that both classifiers are similar. On the opposite, using $0 < \widetilde{\alpha} < 1$ is equivalent to supposing, *a priori*, that both classifiers are different. Since they are no evidences supporting the choice of one over the other, we follow Jaynes' maximum entropy principle [Jay57] and we use $\widetilde{\alpha} = 1$, i.e., $\alpha_h = \alpha_g = 1$, which represents the uniform distribution.

5 The Poisson Binomial Test

In this section we generalize the results of the previous section to contexts. In context $\mathcal{W}$, whether or not algorithm $\mathcal{A}$ outputs a better classifier than algorithm $\mathcal{B}$ is a Bernoulli random variable of parameter $\tilde{q}_{\mathcal{AB}|\mathcal{W}}$ (Equation (1)). Therefore, after observing N datasets, the number of wins of $\mathcal{A}$ over $\mathcal{B}$ is a binomial distribution of parameter $\tilde{q}_{\mathcal{AB}|\mathcal{W}}$ and N.

Knowing the number of wins of $\mathcal{A}$ on N trials would allow us to directly integrate the Beta distribution to evaluate the probability that $\mathcal{A} \overset{\mathcal{W}}{\succ} \mathcal{B}$. However, since we only have a probabilistic answer when discriminating learning algorithms on a single task, we need to take the expectation over the different number of wins.

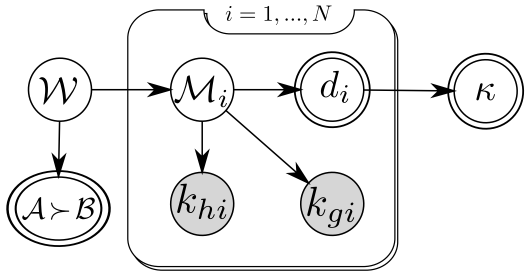

Figure 1: A graphical model representing the dependencies between the variables during the process of evaluating algorithms on multiple datasets.

Putting this more formally, comparing $\mathcal{A}$ and $\mathcal{B}$ on dataset i yields the multinomial $\mathcal{M}_i$ of parameters p_{hi}, p_{gi} and $|T_i|$. As seen in Section 4, this multinomial is hiding the process of comparing h_i and g_i, trained on S_i and tested on T_i. Let k_i be the observations obtained on the i^{th} test set (e.g., $k_i = (k_{hi}, k_{gi})$). To simplify equations, we also define $d_i \overset{\text{def}}{=} I(p_{hi} < p_{gi})$, $\kappa \overset{\text{def}}{=} \sum_i d_i$, $\mathbf{k} \overset{\text{def}}{=} (k_1, k_2, ..., k_N)$, $\mathbf{d} \overset{\text{def}}{=} (d_1, d_2, ..., d_N)$, and $r \overset{\text{def}}{=} \tilde{q}_{\mathcal{AB}|\mathcal{W}}$. As represented in Figure 1 the only observed variables are the ones in $\mathbf{k}$.

Consequently, we have :

$$\begin{aligned}
\Pr\left(\mathcal{A} \overset{\mathcal{W}}{\succ} \mathcal{B} \,\middle|\, \mathbf{k}\right) &= \int_{\frac{1}{2}}^{1} \Pr(r \mid \mathbf{k})\, dr \\
&= \int_{\frac{1}{2}}^{1} \sum_{\kappa=0}^{N} \Pr(r, \kappa \mid \mathbf{k})\, dr \\
&= \sum_{\kappa=0}^{N} \int_{\frac{1}{2}}^{1} \Pr(r \mid \kappa)\, dr \Pr(\kappa \mid \mathbf{k}) \\
&= \sum_{\kappa=0}^{N} \Pr(\kappa \mid \mathbf{k})\, B_c\left(\tfrac{1}{2}, N-\kappa+\widehat{\beta}_{\mathcal{B}}, \kappa+\widehat{\beta}_{\mathcal{A}}\right)
\end{aligned}$$

where we have used the beta distribution with parameter $\widehat{\beta}_{\mathcal{A}}$ and $\widehat{\beta}_{\mathcal{B}}$ for the prior distribution over r and, using Bayes' theorem, have obtained a posterior also given by a beta distribution. To avoid favoritism over $\mathcal{A}$ or $\mathcal{B}$, it is crucial to use $\widehat{\beta}_{\mathcal{A}} = \widehat{\beta}_{\mathcal{B}} \overset{\text{def}}{=} \widehat{\beta}$. Also, to maximize entropy, we use $\widehat{\beta} = 1$.

We are now left with one term to evaluate:

$$\begin{aligned}
&\Pr(\kappa \mid \mathbf{k}) \\
&= \sum_{\mathbf{d} \in \{0,1\}^N} \Pr(\kappa \mid \mathbf{d}) \Pr(\mathbf{d} \mid \mathbf{k}) \\
&= \sum_{\mathbf{d} \in \{0,1\}^N} \Pr(\kappa \mid \mathbf{d}) \prod_{i=1}^{N} \Pr(d_i \mid k_i) \\
&= \sum_{\mathbf{d} \in \{0,1\}^N} I\left(\sum_i d_i = \kappa\right) \prod_{i=1}^{N} p_i^{d_i} (1-p_i)^{1-di} \qquad (5)
\end{aligned}$$

where $p_i \overset{\text{def}}{=} \Pr\left(h_i \overset{\mathcal{D}}{\succ} g_i\right)$ is given by Theorem 4.1. In this form, Equation (5) is computationally hard to track. However, this represents a Poisson-binomial distribution [Wan93], which is the generalization of the binomial distribution when the Bernoulli trials have different probabilities. There exist several ways to compute such probability distribution [FW10, CDL94, CL97]. In our case, we use the following dynamic programming algorithm. We have $\Pr(\kappa) = q_N(\kappa)$ where $q_i(\kappa)$ is defined recursively as:

$$q_i(\kappa) \overset{\text{def}}{=} \begin{cases} 1 & \text{if } i = 0 \wedge \kappa = 0 \\ 0 & \text{if } \kappa < 0 \vee \kappa > i \\ p_i q_{i-1}(\kappa - 1) \\ \quad +(1-p_i) q_{i-1}(\kappa) & \text{otherwise} \end{cases}$$

This algorithm has $O(N^2)$ complexity. It is possible to build a $O\left(N \log^2(N)\right)$ algorithm, using spectral domain convolution to combine the solution of a divide and conquer strategy. But the $O\left(N^2\right)$ algorithm is simpler and fast enough for our needs.

5.1 Transitivity and Ordering

The operator $\overset{\mathcal{D}}{\succ}$ is transitive for any $\mathcal{D}$ since it is based on the risk of the classifiers. We then have $h \overset{\mathcal{D}}{\succ} g \wedge g \overset{\mathcal{D}}{\succ} f \implies h \overset{\mathcal{D}}{\succ} f$. However, this transitive property doesn't hold for $\overset{\mathcal{W}}{\succ}$. This can be shown with a small counterexample presented on Figure 2. In this example, we are comparing algorithms $\mathcal{A}$, $\mathcal{B}$ and $\mathcal{C}$ on datasets a, b and c. We work in the theoretical setup. Therefore we know $\mathcal{W}$ and assume that it is the uniform distribution on the 3 datasets. Also, the exact risk for each classifier is expressed in Figure 2. Hence, from Definition (2), we have $\mathcal{A} \overset{\mathcal{W}}{\succ} \mathcal{B}$, $\mathcal{B} \overset{\mathcal{W}}{\succ} \mathcal{C}$ and $\mathcal{C} \overset{\mathcal{W}}{\succ} \mathcal{A}$. This implies that $\overset{\mathcal{W}}{\succ}$ cannot be used for ordering algorithms.

	$\mathcal{A}$	$\mathcal{B}$	$\mathcal{C}$
a	0.1	0.2	0.3
b	0.3	0.1	0.2
c	0.2	0.3	0.1

Figure 2: A minimalist counterexample on the transitivity of comparison on multiple datasets.

The non-transitivity of $\overset{\mathcal{W}}{\succ}$ is a consequence of ignoring the amount of differences between the two risks. This also affects other methods such as the sign test (defined in Section 6.1). In most practical situations, the finite amount of samples have the side effect of weighting the differences between classifiers. Thus, alleviating the chances of observing a high probability cycle.

6 Simulations With Synthetic Contexts

In this section we compare the performances of the Poisson binomial test with the sign test and the WSR test. Ideally, we would run learning algorithms with the different evaluation methods on contexts for which we know the real underlying distributions and we would repeat this process several times. However, we do not have such data and this process would take forever. Fortunately, we saw that the outcome of testing two learning algorithms on a task is equivalent to sampling from a multinomial of parameters p_g, p_h, $\overline{p}$, and $n \overset{\text{def}}{=} |T|$. Therefore, by defining a probability distribution over the four parameters of this multinomial, we build what we call a *synthetic context.*

With those contexts, it is now easy to sample the data we need to compare the answer of the different methods with the real answer. However, the behavior of the three methods depends on a confidence threshold which offer a trade-off between the success rate and the error rate. It is thus hard to compare them with a fixed threshold. To overcome this difficulty, we use the area under the ROC curve (AUC) as a metric of performances.

The following subsections provide the details on how to obtain the p-values for the sign test and the WSR test and on how to compute the AUC. Finally, in Section 6.4, we present the results obtained on different synthetic contexts.

6.1 Sign Test

The sign test [Men83] simply counts the number of times that $\mathcal{A}$ has a better empirical metric than $\mathcal{B}$ and assumes that this comes from a binomial distribution of parameters $\tilde{q}_{\mathcal{AB}|\mathcal{W}}$ and N. In our case, we use $\operatorname{sgn}(k_{hi} - k_{gi})$ and any sample i where $k_{hi} = k_{gi}$ is ignored, for $i \in \{1, 2, ..., N\}$.

To assert that the observed difference is significant enough, the sign test is based on hypothesis testing. In our case, this corresponds to whether $\mathcal{A} \overset{\mathcal{W}}{=} \mathcal{B}$ or not. More formally, $H_0 : \tilde{q}_{\mathcal{AB}|\mathcal{W}} = 0.5$ and $H_1 : \tilde{q}_{\mathcal{AB}|\mathcal{W}} \neq 0.5$. Let κ_+ be the number of times we observe a positive value of $\operatorname{sgn}(k_{gi} - k_{hi})$ and κ_- be the number of times we observe a negative value. Then, the two tailed p-value of H_0 is given by

$$\rho_s(\kappa_+, \kappa_-) = 2 \sum_{i=0}^{\min(\kappa_+,\kappa_-)} \binom{\kappa_+ + \kappa_-}{i} \frac{1}{2^{\kappa_+ + \kappa_-}}$$

It is common to reject H_0 when $\rho_s(\kappa_+, \kappa_-)$ goes below 0.05 (or any reasonable threshold). In that case, we proceed as follows : if $\kappa_+ > \kappa_-$, we conclude that $\mathcal{A} \overset{\mathcal{W}}{\succ} \mathcal{B}$, and we conclude the converse when $\kappa_+ < \kappa_-$.

6.2 Wilcoxon Signed Rank (WSR) Test

Instead of simply counting the number of wins like the sign test does, the WSR test [Wil45] weight each count by the rank of the absolute difference of the empirical risk. More precisely, the empirical risk difference is

$$d_i \overset{\text{def}}{=} \frac{k_{gi} - k_{hi}}{m_i}.$$

The samples where $d_i = 0$ are rejected and we use J as the set of indices where $d_i \neq 0$. To take into account the fact that there might be samples of equal rank, we use the following formula to compute the rank :

$$r_i \overset{\text{def}}{=} \frac{1}{2}\left(1 + \sum_{j\in J} I\left(|d_j| < |d_i|\right) + \sum_{j \in J} I\left(|d_j| \leq |d_i|\right)\right)$$

The sum of positive rank is $c_+ \overset{\text{def}}{=} \sum_{j\in J} r_j I\left(d_j > 0\right)$ and the sum of negative rank is $c_- \overset{\text{def}}{=} \sum_{j\in J} r_j I\left(d_j < 0\right)$.

The WSR test assumes that the d_i are sampled i.i.d. from a symmetric distribution around a common median[3]. Then, under H_0, the values of $I\left(d_i < 0\right)$ are i.i.d. and have equal probabilities for either outcome. This allows us to recursively compute the probability distribution for the values of c_+ under H_0 as follows :

$$w_n(c) \overset{\text{def}}{=} \begin{cases} 0 & \text{if } c \notin [0, \frac{n(n+1)}{2}] \\ 1 & \text{if } n = 1 \text{ and } c \in [0, \frac{n(n+1)}{2}] \\ \frac{w_{n-1}(c) + w_{n-1}(c-n)}{2} & \text{otherwise} \end{cases}$$

Finally, the two tailed p-value is given by :

$$\rho_w(c_+, c_-) \overset{\text{def}}{=} 2 \sum_{c=0}^{\min(c_+,c_-)} w_{|J|}(c)$$

and whenever $\rho_w(c_+, c_-) \leq \delta$, we reject H_0. In that case, we proceed as follows : if $c_+ > c_-$, we conclude that $\mathcal{A} \overset{\mathcal{W}}{\succ} \mathcal{B}$, and conclude the converse when $c_+ < c_-$.

6.3 Area Under the ROC Curve

With a synthetic context, we can directly sample a set of k_{hi} and k_{gi} for $i \in \{1, 2, \ldots, N\}$. Providing this information to one of the methods, we obtain an answer $\hat{a}$ and a confidence level γ. In the case of the sign test, the confidence level is $1 - \rho_s(\kappa_+, \kappa_-)$. For the WSR test, we use $1 - \rho_w(c_+, c_-)$. Finally, for the Poisson binomial test, we use $\max\left(\Pr\left(\mathcal{A} \overset{\mathcal{W}}{\succ} \mathcal{B}\right), \Pr\left(\mathcal{B} \overset{\mathcal{W}}{\succ} \mathcal{A}\right)\right)$.

Next, repeating this experiment M times for a given threshold τ and comparing the answer to the real answer a, we obtain a success count s_τ and an error count e_τ. More formally,

$$\begin{aligned} s_\tau &\overset{\text{def}}{=} \textstyle\sum_{j=1}^{M} I(\gamma_j > \tau) I(\hat{a}_j = a) \\ e_\tau &\overset{\text{def}}{=} \textstyle\sum_{j=1}^{M} I(\gamma_j > \tau) I(\hat{a}_j \neq a). \end{aligned}$$

To obtain the ROC curve, we have computed all pairs (s_τ, e_τ) by selecting τ from the set of the M obtained confidence levels γ_j. Next, to obtain the AUC, we use a trapezoidal approximation of the integral over the values $\left(\frac{s_\tau}{s_0}, \frac{e_\tau}{e_0}\right)$ where s_0 and e_0 correspond to the

[3]We will see that the symmetric assumption is inappropriate for machine learning algorithm evaluation.

success count and error count when the threshold is at its minimum value.

To obtain a good resolution for the ROC curve, we fix M to 10^5. Also, to make sure that there is no bias in the methods, we randomly swap the k_{hi} and k_{gi} and, we adjust the value of a accordingly.

6.4 Experimental results

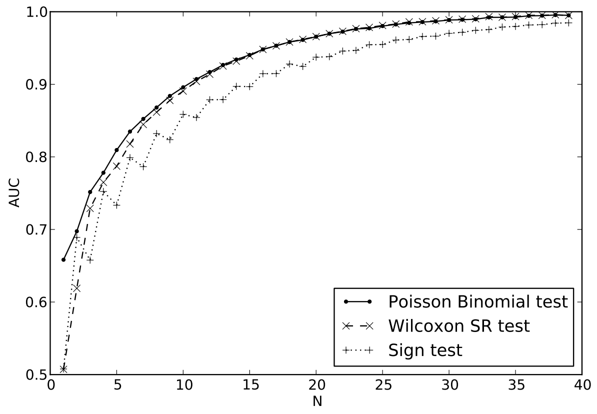

Figure 3: Comparison of the 3 methods, using AUC on a single Dirichlet synthetic context, for various values of N where n is fixed to 1001.

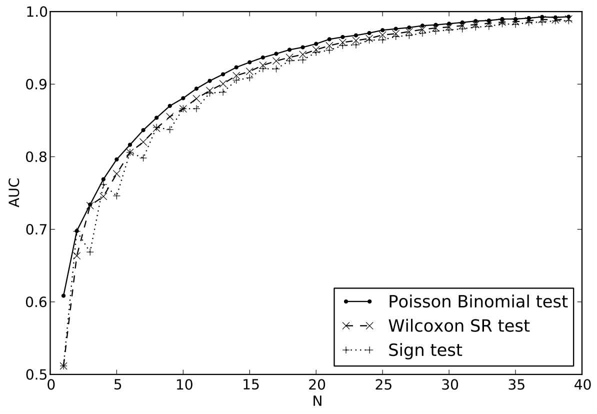

Figure 4: Comparison of the 3 methods, using AUC on a multiple Dirichlet synthetic context, for various values of N where n is fixed to 1001.

As explained before, a synthetic context is completely determined by a probability distribution over the parameters of the three outcome multinomial. For simplicity reasons, we fix n, and use Dirichlet distributions to represent synthetic contexts.

For our first experiment, we fix n to 1001 and use the Dirichlet distribution of parameters $(100, 110, 790)$. The expected values for p_h, p_g and $\overline{p}$ are then respectively 0.1, 0.11 and 0.79. This means that in this context $\mathcal{A} \overset{w}{\succ} \mathcal{B}$. Figure 3 expresses the AUC of the three different methods, for various values of N. From those results, we first observe that the Poisson binomial test constantly outperform the sign test. Also, we observe that the WSR test have performances similar to our approach. However, it is important to understand that this synthetic context is favorable to the WSR test. Indeed, the d_i (see Section 6.2) are sampled from a symmetric distribution, which is one of the assumptions required by the WSR test.

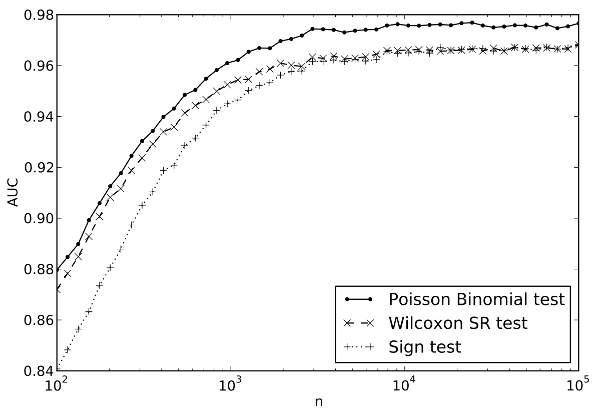

Figure 5: Comparison of the 3 methods, using AUC on a multiple Dirichlet synthetic context, for various values of n where N is fixed to 21.

To explore how the WSR test behaves with a non-symmetric distribution of the d_i, we use the following bimodal context : with probability $\frac{2}{3}$, we sample from the Dirichlet distribution of parameters $(100, 140, 9760)$, otherwise we sample from the Dirichlet distribution of parameters $(1400, 1000, 7600)$. Now, fixing n to 100001 and N to 14, we have an AUC over 0.8 for the sign test and the Poisson binomial test, while the AUC of the WSR test is 0.334, i.e., worse than random. This means that the WSR test may drastically fail in some situations. Such events can occur in practice when a learning algorithm is better than its competitor on tasks having many classes, but the two algorithms are exposed to a context where tasks having fewer classes are more frequent. Since the average risk typically raises with the amount of classes, this would create an asymmetrical distribution of the empirical risk differences.

Finally, to build a more plausible context, we use the error counts obtained when comparing svm with parzen window on 22 datasets coming from UCI and MNIST (Section 9.1). The synthetic context is thus a uniform distribution over 22 Dirichlet distributions where the parameters are provided in supplementary materials. The AUC for various values of N and various values of n are expressed in Figure 4 and Figure 5.

From those results, we conclude that the Poisson binomial test is a significantly more appropriate test for machine learning algorithm comparison.

7 Comparing Popular Algorithms

Table 1: Comparison of the test risk for 3 of the 4 algorithms on 5 of the 22 datasets. Individual significance is shown, using Theorem 4.1, for selected pair of classifiers.

	svm	$\overset{\mathcal{D}_i}{\succ}$	ann	$\overset{\mathcal{D}_i}{\succ}$	parzen
Adult	0.157	0.52	0.157	1.00	0.172
Glass	0.140	0.61	0.150	0.40	0.140
MNIST:08	0.003	1.00	0.012	0.04	0.006
Mushrooms	0.000	0.50	0.000	0.99	0.001
Sonar	0.154	0.84	0.202	0.42	0.192

Table 2: The pairwise Poisson binomial test showing Pr (row $\succ$ col). Gray values represent redundant information.

	svm	ann	parzen	adaBoost
svm	0.50	0.72	0.99	0.95
ann	0.28	0.50	0.88	0.87
parzen	0.01	0.12	0.50	0.52
adaBoost	0.05	0.13	0.48	0.50

Table 3: The pairwise sign test for 1 - $\rho_s(\kappa_+, \kappa_-)$. Cases where $\kappa_+ \not> \kappa_-$ are omitted.

	svm	ann	parzen	adaBoost
svm	-	0.88	1.00	0.92
ann	-	-	0.71	0.83
parzen	-	-	-	-
adaBoost	-	-	0.18	-

Table 1 presents the empirical test risk $R_{T_i}(\mathcal{A}_j(S_i))$ for three popular learning algorithms on five UCI datasets. A more complete table is provided in the supplementary material. Table 2 expresses the pairwise comparison of algorithms when using the Poisson binomial test. For comparison, we have also reported the p-value of the sign test in Table 3 and for the WSR test in Table 4. Here, the three methods yield comparable results but, in contrast with the experiment in Section 6, it is not possible to conclude if one method yields better conclusions than the others since the distribution $\mathcal{W}$ is unknown here.

When performing experiments on synthetic contexts, we observed that using a threshold of 0.85 for the Poisson binomial test yield a lower type I error rate than the sign test and the WSR test with a threshold of 0.90 on the confidence level. Using this remark, the sign test perform 2 assertions, the WSR test perform 3 assertions and, the Poisson binomial test perform 4 assertions. While it seems favorable to our approach, another test might have yield different results. However, the results provided in Section 6 are reliable and support the same conclusion.

Table 4: The pairwise WSR test for 1 - $\rho_w(c_+, c_-)$. Cases where $c_+ \not> c_-$ are omitted.

	svm	ann	parzen	adaBoost
svm	-	0.30	0.97	0.92
ann	-	-	0.95	0.41
parzen	-	-	-	-
adaBoost	-	-	-	-

8 Future Work

In this work we did not addressed the problem of multiple comparisons. This case occurs when more than one probabilistic comparison has to be made. Then, the probability of having a false conclusion inrcreases with the number of comparisons. To address this problem, it is common to use methods to control the familywise error rate [HT87] or the false discovery rate [BH95]. However, the current approaches are made to work with frequentist comparisons and are not directly applicable to our Bayesian tool. To this end, we are currently working on a Bayesian method to control the false discovery rate under dependencies.

We are also investigating on how to extend the Poisson binomial test to work outside of the classification paradigm. This would allow us to compare algorithms meant to work on regression or structured output tasks.

Together, these new methods should provide the machine learning community with a wide range of tools to shed light on the discovery of new learning algorithms.

Acknowledgement

Thanks to Pascal Germain, Alexandre Lacasse, Jean-Francis Roy, and Francis Turgeon-Boutin for their help in the software implementation. Thanks to CLUMEQ for providing support and access to Colosse's high performance computer grid. Work supported by NSERC Discovery Grants 122405 (M. M.) and 262067 (F. L.).

References

[Alp] E. Alpaydın. Combined 5 × 2 cv F Test for Comparing Supervised Classification Learning Algorithms.

[AS64] M. Abramowitz and I.A. Stegun. *Handbook of mathematical functions with formulas, graphs, and mathematical tables*. Dover publications, 1964.

[BG04] Y. Bengio and Y. Grandvalet. No unbiased estimator of the variance of k-fold cross-validation. *The Journal of Machine Learning Research*, 5:1089–1105, 2004.

[BG05] Y. Bengio and Y. Grandvalet. Bias in estimating the variance of k-fold cross-validation. *Statistical modeling and analysis for complex data problems*, pages 75–95, 2005.

[BH95] Y. Benjamini and Y. Hochberg. Controlling the false discovery rate: a practical and powerful approach to multiple testing. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 289–300, 1995.

[CDL94] X.H. Chen, A.P. Dempster, and J.S. Liu. Weighted finite population sampling to maximize entropy. *Biometrika*, 81(3):457, 1994.

[CL97] S.X. Chen and J.S. Liu. Statistical Applications of the Poisson-Binomial and conditional Bernoulli distributions. *Statistica Sinica*, 7:875–892, 1997.

[CV95] C. Cortes and V. Vapnik. Support-vector networks. *Machine learning*, 20(3):273–297, 1995.

[Dem06] J. Demšar. Statistical comparisons of classifiers over multiple data sets. *The Journal of Machine Learning Research*, 7:1–30, 2006.

[Die98] T.G. Dietterich. Approximate statistical tests for comparing supervised classification learning algorithms. *Neural computation*, 10(7):1895–1923, 1998.

[Fri37] M. Friedman. The use of ranks to avoid the assumption of normality implicit in the analysis of variance. *Journal of the American Statistical Association*, 32(200):675–701, 1937.

[FS95] Y. Freund and R. Schapire. A desicion-theoretic generalization of on-line learning and an application to boosting. In *Computational learning theory*, pages 23–37. Springer, 1995.

[FW10] M. Fernandez and S. Williams. Closed-Form Expression for the Poisson-Binomial Probability Density Function. *Aerospace and Electronic Systems, IEEE Transactions on*, 46(2):803–817, 2010.

[HT87] Y. Hochberg and A.C. Tamhane. *Multiple comparison procedures*, volume 82. Wiley Online Library, 1987.

[Jay57] E.T. Jaynes. Information theory and statistical mechanics. II. *Physical review*, 108(2):171, 1957.

[Lan06] J. Langford. Tutorial on practical prediction theory for classification. *Journal of Machine Learning Research*, 6(1):273, 2006.

[Men83] W. Mendenhall. Nonparametric statistics. *Introduction to Probability and Statistics*, 604, 1983.

[MP69] M. Minsky and S. Papert. Perceptrons. *MIT Press,Cambridge, MA*, 1969.

[Par62] E. Parzen. On estimation of a probability density function and mode. *The annals of mathematical statistics*, 33(3):1065–1076, 1962.

[Ros56] M. Rosenblatt. Remarks on some nonparametric estimates of a density function. *The Annals of Mathematical Statistics*, 27(3):832–837, 1956.

[Wan93] YH Wang. On the number of successes in independent trials. *Statistica Sinica*, 3(2):295–312, 1993.

[Wil45] F. Wilcoxon. Individual comparisons by ranking methods. *Biometrics Bulletin*, 1(6):80–83, 1945.

9 APPENDIX—SUPPLEMENTARY MATERIAL

9.1 Synthetic Contexts

Table 5: List of Dirichlet parameters used in the multimodal synthetic context.

kh+1	kg+1	n-kh-kg+1
11	14	1223
12	20	1232
10	16	4078
13	15	1228
31	17	320
22	22	148
27	35	154
14	18	112
17	25	286
24	17	151
149	240	5555
16	30	762
19	14	791
11	16	237
79	127	3834
15	20	1213
10	11	803
15	15	122
11	29	176
14	18	352
46	64	410
31	1019	2694

9.2 Experimentation details

In this section, we compare the following commonly-used learning algorithms:

- **svm:** Support Vector Machine [CV95] with the RBF kernel.
- **parzen:** Parzen Window [Par62, Ros56] with the RBF kernel.
- **adaBoost:** AdaBoost [FS95] using stumps as weak classifiers.
- **ann:** Artificial Neural Networks [MP69] with two hidden layers, sigmoid activation function and L^2 regularization of the weights.

These learning algorithms are compared on 18 binary classification datasets coming from UCI and 4 other datasets coming from MNIST. Our goal is to explore the behavior of the Poisson binomial test on commonly-used learning algorithms applied on real datasets rather than discriminating the quality of these learning algorithms. The details on the tuning of these algorithms are provided in the appendix.

Experimental Setup

Each dataset is split into two equal-size dataset called S_i and T_i. Then, each learning algorithm is trained on S_i and tested on T_i using the zero-one loss. Finally, all pairs of learning algorithms are compared using both the Poisson binomial test and the sign test.

Since each learning algorithm comes with adjustable hyperparameters, we use the 10-fold cross validation estimate of the error rate on S_i to select the most appropriate set of values.

Details on Learning Algorithms

We present here the details about the tuning of the learning algorithms used in the experimental setup.

To concisely define the list of explored hyperparameters, we use $\text{Lin}_k(a,b) \overset{\text{def}}{=} \{a, a+s, a+2s, ..., b\}$ where $s \overset{\text{def}}{=} \frac{b-a}{k-1}$. This represents k values uniformly selected in the interval $[a,b]$. Similarly, we define $\text{Log}_k(a,b) \overset{\text{def}}{=} \{10^x \mid x \in \text{Lin}_k(a,b)\}$.

When searching the appropriate parameter for the RBF kernel, we use the following approach. First recall that the RBF kernel values are given by

$$k_{\text{RBF}}(x,x') \overset{\text{def}}{=} \exp\left(\frac{-r}{2\hat{\sigma}^2}\|x-x'\|^2\right), \tag{6}$$

where r is found using cross-validation and $\hat{\sigma}$ is obtained using the train set. To obtain an appropriate value for $\hat{\sigma}$, we used the 10th percentile over the distribution of distances between all pairs of examples in S_i.

To build weak learners for boosting algorithms, we use c stumps per attribute where c is constant across all attributes and is found using cross validation. For each attribute, each of the c threshold values are uniformly distributed across the interval of values realized on the training data.

svm: Support Vector Machine [CV95] with RBF kernel. The width r of the kernel (Equation (6)) is selected from $\text{Log}_{20}(-5,5)$ and the soft-margin parameter C is selected from $\text{Log}_{20}(-2,5)$.

parzen: Parzen Window [Par62, Ros56] with RBF kernel. The width r of the kernel is selected from $\text{Log}_{30}(-4,4)$.

adaBoost: AdaBoost [FS95], using stumps as weak classifiers. The number of iterations is selected using

Table 6: Comparison of $R_{T_i}(\mathcal{A}_j(S_i))$ for $i \in \{1, 2, ..., 22\}$ and $j \in \{1, 2, 3, 4\}$. Individual significance is shown, using Theorem 4.1, for selected pair of classifiers.

	svm	$\succ$	**ann**	$\succ$	**parzen**	$\succ$	**adaBoost**
Adult	0.157	0.52	0.157	1.00	0.172	0.00	0.151
Breast	0.041	0.50	0.041	0.87	0.053	0.50	0.053
Credit	0.187	0.00	0.129	0.84	0.144	0.43	0.141
Glass	0.140	0.61	0.150	0.40	0.140	0.95	0.206
Haberman	0.279	0.57	0.286	0.05	0.231	0.50	0.231
Hearts	0.196	0.03	0.135	0.98	0.196	0.97	0.257
Ionosphere	0.057	0.77	0.074	1.00	0.160	0.05	0.109
Letter:AB	0.000	0.94	0.004	0.12	0.001	0.99	0.009
Letter:DO	0.014	0.73	0.017	0.04	0.008	1.00	0.040
Letter:OQ	0.009	0.76	0.013	0.98	0.027	0.89	0.038
Liver	0.349	0.71	0.366	0.75	0.395	0.07	0.331
MNIST:08	0.003	1.00	0.012	0.04	0.006	1.00	0.016
MNIST:17	0.007	0.69	0.007	0.64	0.008	0.36	0.007
MNIST:18	0.011	1.00	0.037	0.00	0.017	0.42	0.016
MNIST:23	0.017	1.00	0.035	0.00	0.022	1.00	0.041
Mushrooms	0.000	0.50	0.000	0.99	0.001	0.01	0.000
Ringnorm	0.015	1.00	0.054	1.00	0.282	0.00	0.027
Sonar	0.154	0.84	0.202	0.42	0.192	0.59	0.202
Tic-Tac-Toe	0.161	0.00	0.052	1.00	0.198	1.00	0.357
USVotes	0.069	0.25	0.065	0.98	0.092	0.01	0.051
WDBC	0.049	0.02	0.021	1.00	0.077	0.03	0.042
Waveform	0.068	0.26	0.067	1.00	0.080	0.41	0.079

cross validation for values in $\{2^n \mid n \in \{1, 2, ..., 10\}\}$ and the number of stumps is selected from { 1, 2, 3, 4, 6, 10, 14, 21, 31 }.

ann: Artificial Neural Networks [MP69] with two hidden layers, sigmoid activation function, and L^2 regularization of the weights. The input space is normalized on the training set, such that each attribute has zero mean and unit variance. The number of neurons on the second layer is $\lceil \sqrt{N_{l1}} \rceil$ where N_{l1} is the number of neurons on the first layer and is selected from { 3, 4, 5, 6, 7, 9, 11, 13, 16, 19, 23, 28, 33, 40, 48, 57, 69, 83, 100 }. Finally, the weight of the L^2 regularizer is selected from $\text{Log}_{20}(-2, 2)$. The network is trained using conjugate gradient descent with no early stopping.

In all cases, when there are two hyperparameters, all pair of proposed values are explored.