Choosing learning algorithms using sign tests with high replicability

Remco R. Bouckaert

Xtal Mountain Information Technology and Computer Science Department,
University of Waikato, New Zealand
rrb@xm.co.nz, remco@cs.waikato.ac.nz

Abstract. An important task in machine learning is determining which learning algorithm works best for a given data set. When the amount of data is small the same data needs to be used repeatedly in order to get a reasonable estimate of the accuracy of the learning algorithms. This results in violations of assumptions on which standard tests are based and makes it hard to design a good test.
In this article, we investigate sign tests to address the problem of choosing the best of two learning algorithms when only a small data set is available. Sign tests are conceptually simple and no assumption about underlying distributions is required. We show that simplistic sample generation can lead to flawed test outcomes. Furthermore, we identify a test that performs well based on Type I error (showing a difference between algorithms when there is none), power (showing a difference when it indeed exists) and replicability.
Replicability is a novel measure of a quality of a test that gives an indication how likely it is that the test outcome will be the same when the same test on the same data with the same sampling scheme and same pair of algorithms is executed, but with a different randomization of the data. A new definition of replicability is provided and its benefits highlighted. Empirical evidence is provided to show the test is robust under a varied range of circumstances.

1 Introduction

Choosing learning algorithms for classification tasks when a single data set is given is an area marked with unexpected pitfalls [3, 10]. As a result, many methods can indicate one algorithm outperforms another while the experimental data actually does not support such a claim. If the algorithms are closely related, standard techniques such as minimum description length, information criteria or Bayesian techniques [4] can be used for comparing the algorithms. However, if the algorithms are based on a range of varied techniques such as naive Bayes [5], C4.5 [8], or nearest neighbor [11], those techniques cannot be applied straightforwardly. A commonly used technique is to repeatedly split the data set in different training/test sets, train the algorithms on the traing set and use the classification accuracies of the test sets as input for a hypothesis test. Inevitably, the various samples of accuracy thus obtained are somewhat dependent because they are partially based on the same data. Consequently, the thus far widely used t-test [6] based on resampling was found to have a high probability of resulting in a wrong recommendation [3]. Also the popular repeated cross validation test [11] suffers from this problem [2].

This article tries to draw out a small part of the map by studying sign tests for choosing between two learning algorithms when a small data set is given (question 8 in Dietterich's taxonomy of problems [3]). Sign tests are attractive because they are conceptually simple: when multiple samples agree on the outcome more than a threshold number of times, we decide to choose the preferred algorithm, otherwise we accept the null hypothesis that the algorithms perform equally well. Furthermore, sign tests require no assumptions about the underlying distribution, unlike t-tests. The only assumption underlying sign tests is that samples are independent. In the remainder of this paper we will see how easily this assumption can be violated.

Replicability of experiments is routinely addressed in areas like the social sciences and physics. However, in experimental computer science and machine learning in particular not a lot of attention is given to this issue. In a machine learning setting, it became recently clear that many popular tests give different outcomes when repeating the test on the same data, but with different randomizations [2]. Clearly, this is an undesirable property of a test, because a researcher or data analyst can be misled unintentionallys in thinking one method outperforms another, while the data may not support such claim except for the specific randomization that was used in the experiment. In this article, we consider replicability when judging the quality of a test.

This article is organized as follows. We start with an introduction to sign tests and explain how to apply such tests to learning algorithm selection. In Section 3, we discuss how to determine the quality of these tests, with special emphasis on replicability. Section 4 describes experimental evaluation under a range of circumstances. We conclude with some final remarks, recommendations and directions for further research.

2 Sign tests for selecting learning algorithms

The null hypothesis of tests for determining which of two learning algorithms performs best is that both algorithms performs the same. A sample to perform the test is typically obtained by performing an experiment where the two algorithms are repeatedly trained on one part of the data and tested on another part of the data. Based on this sample (details follow later in this section), the probability of the sample assuming the hypothesis is true can be calculated, the so called p-value. If the p-value is below the significance level, which is a predetermined threshold, we reject the null hypothesis and decide that one of the algorithms outperforms the other, otherwise we conclude that there is no support for such a claim.

The benefit of a sign test is that it does not make any assumption about the underlying distribution and is very well suited to data from matched pairs. Matched pairs naturally occur when comparing two algorithms A and B on n data sets giving a pair of accuracies $P_{A,i}$ and $P_{B,i}$ ($1 \leq i \leq n$) on the test set. Using the sign of the difference $x_i = P_{A,i} - P_{B,i}$, we can count the number of plusses and minuses. When accuracies $P_{A,i}$ and $P_{B,i}$ are the same, which occurs quite often when two algorithms are performing very similarly, we count this as half a plus and half a minus. The number of plusses and minusses can be used to test the null hypothesis H_0 that both algorithms perform the same. If this is true, the probability of generating a plus or a minus is 0.5, in other words $H_0 : p = 0.5$.

Assuming the samples are independent and the null hypothesis is true, the number of plusses and minuses follows a binomial distribution [9]. The probability of observing k plusses in n comparisons is $P(k) = \binom{n}{k} p^k (1-p)^{n-k}$, which with $p = 0.5$ is $P(k) = \binom{n}{k} \frac{1}{2}^n$. We accept the null hypothesis as long as the probability that the number of plusses is less than k or larger than $n - k$ is more than α.

A potential drawback of sign tests is that they do not exploit the magnitute of the difference, just the sign of the difference, thus ignoring potentially valuable information. The Wilcoxon sign-ranked test may aleviate this problem, but is out of scope of this article.

Note that the one and only assumption made for performing a sign test is that the samples are drawn independently. Unfortunately, we do not have the luxury of getting independent samples because we are forced to reuse the data in generating samples due to the small size of the data set. There are various ways to obtain these samples. We describe the popular methods of resampling, k-fold cross validation, and various ways to benefit from repeated cross validation. These make obvious candidates to provide input for sign tests because they have been used extensively in t-tests before.

Resampling: Resampling is repeatedly splitting the data randomly r times in a training and a test set. The algorithms A and B learn on the training set and accuracies $P_{A,i}$ and $P_{B,i}$, $1 \leq i \leq r$ are obtained by classifying instances on the accompanying test set. The sign of the difference $x_i = P_{A,i} - P_{B,i}$ is used for the sign test with $n = r$ comparisons.

k-fold cross validation: Cross validation splits the data D in k approximate equal parts $D_1, \ldots, D_k$, and learn on the data $D \backslash D_i$, $1 \leq i \leq k$ with one part left out. The part D_i left out is used as test set. This gives k accuracy differences $x_i = P_{A,i} - P_{B,i}$ from which the sign is used for a sign test with $n = k$ comparisons.

Use all data: Repeating k-fold cross validation r times and using the results is a way to increase replicability [2]. Such an experiment gives $r \times k$ differences x_{ij}, $1 \leq i \leq r$, $1 \leq j \leq k$. The signs can be used for a sign test with $n = r \times k$ comparisons.

Average over folds: Averaging over folds in a repeated cross validation experiment, the recommended method for Weka [11], may be expected to give

Fig. 1. Example illustrating the data used for the various tests. It shows the data involved in doing the various tests if the data of a 3x3 fold cross validation experiment were available. The 'use all data' test uses all points in the top rectangle.

Run	Fold 1	2	3	used when averaging over folds
1	3.33	10	-6.66	2.22
2	6.66	3.33	0	3.33
3	6.66	-10	-3.33	-2.22
used when averaging over runs	5.55	1.11	-3.33	

Run	Sorted Fold 1	2	3
1	-6.66	3.33	10
2	0	3.33	6.66
3	-10	-3.33	6.66
used when averaging over sorted runs	-5.55	1.11	7.77

a better estimate of the difference of accuracies and thus a way to improve the Type I error. To perform a sign test, take the sign of the average difference $x_{i.} = \sum_{j=1}^{k} x_{ij}$ (where x_{ij} as for the use all data test) and apply a sign test with $n = r$ comparisons.

Average over runs: Averaging over folds can be interpreted as an improved way of doing resampling. The natural extension is performing an improved way of k-fold cross validation, and instead of averaging over folds, average over runs. To perform a sign test, take the sign of the average difference $x_{.j} = \sum_{i=1}^{r} x_{ij}$ and apply the test with $n = k$ comparisons.

Average over sorted runs: Averaging over runs combines results from different runs rather arbitrary. One gets better estimates of a k-fold cross validation experiment by first sorting the results for the individual k-fold cross validation experiments and then taking the average. This way, the estimate for the minimum value is calculated from the minimum values in all folds, the one but lowest from the one but lowest results in all folds, etc. Let $x_{\theta(ij)}$ be the jth highest value of accuracy difference $x_{i'j'}$ of run i. Then, take the sign of the average difference $x_{.j} = \sum_{i=1}^{r} x_{\theta(ij)}$ and apply a sign test with $n = k$ comparisons.

Figure 1 illustrates the difference between the data used for the tests. The figure shows an example of 3x3 fold cross validation outcomes in the box at the left half (though in practice a 10x10 fold cross validation is more appropriate). All the data in the box in Figure 1 is used for the "use all data" test. For resampling, essentially only the first column is required when performing a 2/3-1/3 split of training and test data. Cross validation uses only the first run, that is, the first row of a 3x3 fold cross validation outcome. Averaging over folds and runs is essentially summing over columns and rows respectively. For getting sorted means, first the results have to be sorted over folds, giving the table at the right of Figure 1. Then the means are obtained by summing over rows.

3 Quality of a test

There are various ways to judge the quality of a test. Firstly, the *Type I error* is the probability that a test rejects the null hypothesis while the hypothesis is true. If the Type I error is lower than the significance level, the test is called conservative. The *power* is the probability a test rejects the null hypothesis when the null hypothesis is indeed false. The more power, the better the test.[1] Typically, Type I error and power are related: the more power, the higher the Type I error. Ideally, the Type I error should coincide with the significance level and the power as high as possible.

Often, a test is only judged on its Type I error and power, but when a test is based on repeatedly using the same small data set the issue of replicability arises [2]. Replicability is the ability of a test to produce the same outcome on the same data with the same pair of algorithms but with a different randomization of the data. Obviously, this is an important property since it guarantees that an experiment performed by a researcher or data analyst can be repeated by others and that the outcome is unambiguous.

In [2], a rather ad hoc measure of replicability is proposed that demonstrates that replicability is indeed an issue. The measure proposed is obtained by counting how often a test gives the same outcome ten times when performing the test with ten different randomizations of a data set. By generating 1000 data sets from a single data source and counting the proportion of data sets for which the test produces the same outcome in 10 different runs, an indication of replicability for a particular data source is obtained. This procedure can be repeated for different data sources. Minimal replicability tend to occur for those data sources the hypothesis test has difficulty in deciding which outcome to prefer. This is exactly the area where machine learning researchers typically work and where data analysts make their most difficult decisions. So, the minimum replicability over various data sources is determined to be the overall replicability of a test in [2].

The problem with the proposed definition is that it depends on the number of times a test is performed on a data set. If the number of tests is increased from 10 to say 100, the probability that at least one test outcome differs from the others increases and consequently the replicability decreases. So, replicability cannot be mutually compared when different numbers of tests are performed on a data set. We propose a new definition of replicability that does not depend on the number of times a test is performed.

Definition: *Replicability* of a hypothesis test is the probability two runs of the test on the same data set, with the same pair of algorithms and the same method of sampling the data produce the same outcome minimized over data sets.

To distinguish between this definition and the one in [2], we will refer to the latter as *10x consistency.* The outcome of a hypothesis test is two valued (reject or not reject). Therefore, the probability two tests produce the same outcome is at worst 50%. So, replicability is valued between 50% and 100%, unlike 10x consistency which lies between 0% and 100%. In fact, our definition can be interpreted as an upper bound of 10x consistency.

[1] The Type II error (which explains Type I error's name) is the probability a test incorrectly does not reject the null hypothesis and it is related to the power through Type II error = 1 - power.

A naive approach to estimating the replicability would be to generate pairs of randomization of a data set and run the test to obtain samples with which to estimate replicability. However, assuming randomizations of a data set are uniformly distributed we can perform a test n times and take every pair of outcomes as a sample. There are $\binom{n}{2}$ such pairs. Suppose k $(0 \leq k \leq n)$ tests reject the null hypothesis and the remaining $n-k$ do not. Then $\binom{k}{2}$ pairs of rejecting pairs and $\binom{n-k}{2}$ pairs of non rejecting pairs can be formed. This gives an estimate of replicability as $R(k,n) = (\binom{k}{2} + \binom{n-k}{2})/\binom{n}{2} = \frac{k(k-1)+(n-k)(n-k-1)}{n(n-1)}$. By drawing m data sets from a data source and performing the test n times on each data source, we can estimate replicability for the data source as follows. Let i_k $(0 \leq k \leq n)$ be the number of times the test agrees k times (so $\sum_{k=0}^{n} i_k = m$). Then, replicability is $R = \sum_{k=0}^{n} \frac{i_k}{m} R(k,n)$. By comparing replicability for various data sources a better indication of replicability of a test can be obtained.

4 Simulation study

We performed some experiments to study the behavior of the sign tests by measuring Type I error, power and replicability under different circumstances. A random data source was created over 10 binary variables with a class probability of 50% and all attributes independent. From this data source, 1000 data sets of 300 instances were drawn and one test set of 10.000 instances. Training naive Bayes [5] and C4.5 [8] as implemented in Weka version 3.3[2] [11] on the 1000 data sets and averaging the accuracy measured on the test set showed that both algorithms have a 50% accuracy, as expected. Another three data sources were created by randomly generating Bayes networks over 10 binary variables and generating 1000 data sets and one test set similarly. Table 1 shows the average accuracies of those data sets. For sets 2, 3 and 4, C4.5 outperforms naive Bayes with an increasing margin. The results for Set 1 are used to measure Type I error because it is a data source for which both algorithm perform exactly the same on average, guaranteeing that the null hypothesis is true. For the other three sets we know that the null hypothesis is false and should be rejected. The proportion of datasets for which this happens is a measure of the power of a test.

Table 1. Average accuracies (in percentage) on test data by training on the 1000 data sets and measuring on the test set.

Algorithm	Set 1	Set 2	Set 3	Set 4
Naive Bayes:	50.0	87.84	71.92	81.96
C4.5:	50.0	90.61	77.74	93.23
Difference	0.0	2.77	5.83	11.27

[2] Available from http://www.cs.waikato.ac.nz/ml.

The input for the sign tests was obtained by training naive Bayes and C4.5 on each of the 4000 data sets using 10 fold cross validation repeated 1000 times. For each of the tests, a subset of these 1000x10 accuracies was used. For example, for a 10x10 use all data test, only the first 10 runs of 10 fold cv was used. Default options for naive Bayes and C4.5 in Weka were used. Table 2 shows the results for ten trials of each test on a data set for the 1000 data sets at 5% significance level. In each test, naive Bayes or C4.5 can be preferred or alternatively the null hypothesis that both perform equal can be accepted. So, a total of 10.000 wins can be reached maximally for an algorithm.

The last two columns of Table 2 shows replicability for the tests. The first is 10x consistency as defined in [2], which is calculated by determining the proportion of data sets for which the outcome of a test is the same with 10 different randomizations. The minimum of the four data sources is shown. The last column is replicability as defined in Section 3. Note that the same ranking of tests is obtained by both forms of replicability.

Table 2. Type I error (for Set 1), power (for Set 2, 3 and 4) and replicability of various sign tests at 5% significance level (all numbers in percentages).

test	Set 1 Type I	Set 2 Power	Set 3 Power	Set 4 Power	10x consistency	Replicability
Resampling	15.23	25.36	45.45	93.77	16.0	65.2
k-fold cv	11.30	29.79	43.31	95.64	28.8	71.9
Use all data	48.35	68.49	86.64	100.00	43.1	78.3
Average over folds	61.80	78.86	90.22	100.00	34.7	75.3
Average over runs	53.74	69.07	86.28	100.00	32.6	74.2
Average over sorted runs	5.00	21.24	48.61	99.05	66.7	87.6

Some observations on Table 2:

- The Type I error for resampling is just over 15% at the 5% significance level. This elevated Type I error is considerably lower than the over 50% reported on resampled t-test [3], but still higher than the expected 5%. The power is reasonable, but the replicability is lowest of all test considered, 65.2% (or even 16% more dramatically highlighted through 10x consistency), which is realized on Set 3. This means that the probability of getting the same outcome twice is only increased from 50% to 65%. Alternatively, this means that for 84% of the data sets in Set 3 this test can provide different outcomes depending on the particular randomization of the data set when running the test twice.
- The k-fold cv test performs slightly better than the resampling test. The Type I error is elevated at just over 11% due to the dependence between samples because they are based on the same data. This increase is comparable to the elevated Type I error for t-test where typical values are 10% [3,

7]. The power is quite reasonable, however the replicability (see last column of Table 2) is rather low at almost 72%, which is considerably worse than for tests based on repeated cross validation.
- The use all data test has an unacceptable high Type I error due to the independence assumption underlying the sign test being violated. The dependence between the various outcomes of a k-fold cv experiment is already high, but increases when performing a repeated cross validation experiment.
- Also the Type I error for the average over folds and average over runs is unacceptable high at over 50%. The reason is that the signs are based on means over a number of cases and due to the dependence between the various cases, these means will have the same sign very often.
- The Type I error for the average over sorted runs is exactly on the mark at 5.00% and the power is quite acceptable as well. Sorting the folds before averaging effectively breaks the dependence between the various cases. Also notable is that the replicability is highest of all the tests considered. The ranking of tests is in fact the same according to the two definitions of replicability, however our definition makes it easier to compare replicability with other experiments.

Table 2 shows that the Type I error for some tests is so bad at the 5% significance level that we looked into trying to repair the test by changing the threshold value artificially by rejecting the null hypothesis only if all signs are the same. However, even then the Type I error for average over folds and average over runs was unacceptable high (over 15%) while the use all data test lost its power.

The overall conclusion is that judged on Type I error, power and replicability, only the average over sorted runs test shows any sign of promise and the others will not be considered further.[3]

4.1 Varying the number of runs

Another experiment was set up to measure the impact of the number of runs and Table 3 shows the result for 5, 10, 15, 20, 25, 30, 40, 60 and 100 runs. Note, only the outcomes of the sorted runs test is shown. Type I error for set 1 and power for sets 2, 3 and 4 are listed and only one test per data set counts, so the numbers are based on a sample of 1000. The last two rows list 10x consistency and replicability.

Table 3 suggests that there is a slight improvement of Type I error and power with increasing numbers of runs. More interestingly, the replicability increases to a point where it exceeds the highest known replicability of the acceptable t-tests [2] according to 10x consistency. The 10x consistency of almost 90% for 100 runs comes at the cost of extra computational effort. It is also comforting to realize that with 100 runs, the probability of getting the same outcome using this test is over 96%.

[3] Actually, the numbers were calculated for the tests for following tables, but it only served to confirm the flaws just mentioned.

Table 3. Type I error (for Set 1), power (for Set 2, 3 and 4) and replicability for average over sorted runs test with 10 folds at 5% significance level for various numbers of runs (in percentages).

test	5	10	15	20	25	30	40	60	100
Set 1	5.4	5.1	4.7	4.5	4.9	4.4	4.6	4.9	4.6
Set 2	20.4	21.7	22.2	22.1	22.0	22.7	22.6	23.1	23.3
Set 3	48.0	49.7	49.8	50.0	49.9	49.7	49.4	49.6	50.6
Set 4	98.4	99.0	99.4	99.2	99.4	99.6	99.4	99.3	99.4
10x consistency	54.1	66.7	72.5	74.9	79.3	80.1	82.2	85.3	89.4
Replicability	83.2	87.6	90.1	91.2	92.3	93.1	93.7	94.6	96.3

4.2 Varying class probability

Data sources similar to the one used for Set 1 were created with class probabilities ranging from 10% to 50% and 1000 data sets generated with each of them (Set 1 was used for the 50% data source). The Type I error measured based on a sample of 1000 was 0.0, 0.0, 1.3, 3.8 and 5.1% for class probabilities of 10, 20, 30, 40 and 50% respectively. Note that the Type I error decreases with decreasing class probability because the classifiers tend to become more similar and predict the majority class only. The Type I error is very acceptable for the desired significance level.

4.3 Varying significance level

In the experiments, the significance level of the test was varied ranging from 1 to 11 percent for the sorted runs test only and results are shown in Table 4. For Set 1, the Type I error is listed and for Set 2, 3 and 4 the power is listed. Only the first of the ten tests on each data set counts in this experiment, so the Type I error/power is based on a sample of 1000 (which explains the slight difference in Type I error at 5% in this table and Table 2).

Table 4. Type I error (for Set 1) and power (for Set 2, 3 and 4) for average over sorted runs test with 10 runs and 10 folds at various significance levels (in percentages).

test	1%	5%	10%	11%
Set 1	0.5	5.1	5.6	24.3
Set 2	7.2	21.7	26.7	48.2
Set 3	19.9	49.7	52.9	75.6
Set 4	92.6	99.0	99.3	100.0

Table 4 shows that the Type I error is acceptably close to the desired level for 1%, 5% and 10% levels, but jumps to almost 25% where 11% is desired (increasing the number of runs shows the same effect shown in a separate experiment).

Note that there is a threshold effect: for ten folds, an algorithm is recommended if all 10 signs are the same at $\alpha = 0.1\%$, if 9 or more signs are the same at $\alpha = 1.1\%$, and 8 or more at $\alpha = 5.5\%$. So when putting the significance level at say 2.5% the number of signs at which an algorithm is preferred over the other is the same as at a 5% significance level for 10 runs. This threshold effect is inevitable when using the small sample of 10 folds.

4.4 Varying number of folds

Table 5 lists the results when varying the number of folds based on a sample of 1000. This table highlights the aforementioned threshold effect and shows that one has to be careful in selecting an appropriate number of folds.

Table 5. Type I error (for Set 1) and power (for Set 2, 3 and 4) for average over sorted runs test with 10 runs at 5% significance level for various numbers of folds (in percentages).

test	5	10	20	30	40	50
Set 1	0.4	5.1	9.4	11.2	9.8	13.8
Set 2	9.4	21.7	22.6	20.1	14.5	14.7
Set 3	25.3	49.7	48.7	46.5	43.8	44.5
Set 4	97.0	99.0	98.7	97.6	96.6	96.6

4.5 Varying algorithms and variable cardinality

To see the behavior of the sign test with other algorithms, we trained nearest neighbor, tree augmented naive Bayes and support vector as implemented in Weka [11] with default parameters. To get an impression of how well non-binary data is handled, two new data sources were used: one with ten ternary variables and one with ten four-valued variables with all variables independent and uniformly distributed. Using the outcomes of 10 times repeated 10 fold cross validation, pair-wise sign tests were performed at 5% significance level. Table 6 shows the Type I error for pair-wise comparisons of algorithms (based on a sample of 1000). Weka's support vector implementation works on binary variables only, so the outcome for ternary and four valued variables is omitted. Remarkably, all Type I errors do not exceed the 5% significance level substantially, suggesting that this is a reasonable conservative test for comparing pairs of algorithms. Also, tests with closely linked underlying mechanisms, like naive Bayes and tree augmented naive Bayes, show the dependency of algorithms in the very low Type I error.

Table 6. Type I error (in percentage) for average over sorted runs test with 10 runs and 10 folds on a binary (Set 1), ternary and four valued data for various algorithms (nb = naive Bayes, nn=nearest neighbor, tan = tree augmented naive Bayes, sv = support vector).

	Binary				Ternary			Four valued		
	C4.5	nb	nn	tan	C4.5	nb	nn	C4.5	nb	nn
nb	5.1				3.9			4.1		
nn	2.7	4.9			2.3	5.3		2.4	4.7	
tan	5.4	2.1	3.7		3.8	0.2	5.6	4.0	0.0	4.8
sv	1.6	0.6	0.8	0.7						

4.6 Replicability on real datasets

In order to see how the test behaves on real world data sets, we performed an experiment running naive Bayes (NB), C4.5 and nearest neighbor (NN) using 27 UCI [1] datasets[4]. Though this does not allow us to measure Type I error or power, since it is not known whether the null hypothesis is true or not, it makes it possible to get an indication of replicability of a hypothesis test. Ten times running a 10x10 sorted runs test as before, 20 instances returned consistent outcomes in all ten runs for NB against C4.5 and against NN. Comparing C4.5 against NN even gives consistent outcomes for 24 datasets. Calculating replicability by pooling the outcomes of the runs gives 89% for NB vs C4.5, 92% for NB vs NN and 95% for C4.5 vs NN. So, the sorted runs test has a rather acceptable level of replicability on real live dataset as well.

5 Conclusions

We demonstrated how to successfully apply a sign test to selecting between two learning algorithms when a small data set is available. The benefits of sign tests are that they are conceptually simple and based on very few assumptions. We investigated various methods for generating samples from a single small data set. Most of these methods are well known for generating samples used in t-tests, but surprisingly all these methods turned out to have serious flaws. Only one test among the investigated tests exhibited an acceptable Type I error, reasonable power and a good level of replicability .

This test is based on a repeated 10 fold cross-validation experiment where the folds are sorted and the accuracy of each fold is averaged. Then the signs of the 10 averaged accuracies can be used for a standard sign test. Increasing the number of runs improves replicability, which can reach over 96% at 100

[4] Namely anneal, arrhythmia, audiology, autos, balance-scale, breast-cancer, credit-rating, ecoli, German credit, glass, heart-statlog, hepatitis, horse-colic, Hungarian heart disease, ionosphere, iris, labor, lymphography, pima-diabetes, primary-tumor, sonar, soybean, vehicle, vote, vowel, Wisconsin breast cancer and zoo.

runs. This compares favorably with results for t-tests [2]. Though replicability is already reasonably good at 10 runs, the number of runs should only be limited by computational considerations in order to guarantee replicability. Changing the number of folds is not recommened due to threshold effects inherent to the sign test. Empirical evidence suggests this test performs robustly under a varied range of circumstances.

Another contribution of this paper is a refined definition of replicability. Its benefits are highlighted and it is demonstrated that various tests are undesirable due to their low replicability.

There are two obvious areas in which to extend the tests presented here. One is in selecting the best out of multiple algorithms instead of selecting the best of just two algorithms. Another one is in selecting the best of two algorithms when data sets from multiple domains are available since often machine learning algorithms are compared on benchmark data sets like from the UCI repository. Combined, those two problems form the final goal of this exploration. All these questions pose various multiple comparison problems; the probability that a pair-wise comparison indicates a preference for an algorithm, while in reality there is none, increases when comparing multiple algorithms and multiple data sets. Given the issues highlighted in this paper for the simple case of just two algorithms, it is worth studying those problems in greater detail.

Acknowledgements

I thank Eibe Frank for his stimulating discussions on this topic.

References

1. C.L. Blake and C.J. Merz. UCI Repository of machine learning databases. Irvine, CA: University of California, 1998.
2. R.R. Bouckaert. Choosing between two learning algorithms based on calibrated tests. International Conference on Machine Learning, 51–58, 2003.
3. T.G. Dietterich. Approximate Statistical Tests for Comparing Supervised Classification Learning Algorithms. Neural Computation, 10(7) 1895–1924, 1998.
4. T. Hastie, R. Tibshirani and J. Friedman. The elements of statistical learning. Springer Series in Statistics. Springer-Verlag, New York, 2001.
5. G.H. John and Pat Langley. Estimating Continuous Distributions in Bayesian Classifiers. Uncertainty in Artificial Intelligence, 338–345, 1995.
6. T. Mitchell. Machine Learning. McGraw Hil, 1997.
7. C. Nadeau and Y. Bengio. Inference for the generalization error. Advances in Neural Information Processing Systems 12, MIT Press, 2000.
8. R. Quinlan. C4.5: Programs for Machine Learning. Morgan Kaufmann Publishers, San Mateo, CA, 1993.
9. C.J. Wild and G.A.F. Weber. Introduction to probability and statistics. Department of Statistics, University of Auckland, New Zealand, 1995.
10. S. Salzberg. On Comparing Classifiers: Pitfalls to Avoid and a Recommended Approach. Data Mining and Knowledge Discovery 1:3, 317-327, 1997.
11. I.H. Witten and E. Frank. Data mining: Practical machine learning tools and techniques with Java implementations. Morgan Kaufmann, San Francisco, 2000.