Statistical Testing for Comparing Machine Learning Algorithms

Abstract—**There are many Machine Learning (ML) algorithms for classification tasks today. How to compare different algorithms and choose the best one becomes important because people don't want their choices to be fooled by randomness or variation in comparisons. Statistical testing serves as an effective and convincing method to determine if observed difference between two classification algorithms is statistically significant. In this work we examine two hypothesis testing methods: McNemar's test and 5x2cv paired t test, and carry out experiments with them on the wine dataset. Results show that both tests are effective and there are different scenarios that each one of them is suited for. McNemar's test can be used for comparing resource-consuming algorithms such as deep learning models because it only requires one run of training and testing. Whereas 5x2cv paired t test is slightly more stable and best suited for comparing algorithms that are not resource consuming, or for comparisons that need to be deliberate and precise.**

1. Introduction

1.1. Problem Statement

The development of machine learning algorithms, especially deep learning has greatly improved our capability to solve real-world classification tasks. Researchers often face the situation of having to select the best classifier from a pool of candidates. This problem is nontrivial because 1) we have to determine how to compare the performance of different algorithms and 2) make sure that the difference in performance is statistically significant.

For classification tasks, we have quite a few performance evaluation metrics such as accuracy, precision & recall[1], F1 score[2], and even auroc[3]. Different evaluation metrics may lead to different results in terms of which classifier is better and the choice of evaluation metrics is highly task specific. How to choose the right performance evaluation strategy is beyond the scope of this work. We will focus on the most common evaluation metric for classification: the accuracy or the error rate. Accuracy is the ratio of correct predictions of one classifier over all predictions and error rate is just another way of putting it, that is, the ratio of wrong predictions among all predictions.

Now that the difference in performance can be viewed as the extent to which classifiers disagree from each other in making errors. It looks like we just need to directly compare the error rate of two classifiers to get to know which one is better, which is what we commonly do in research projects. This practice, however, can be problematic. Take a simple binary classification task for example. Suppose we have two classifiers A and B, and train them on the same training dataset and test on a testing dataset. In case 1 classifier A gets an error rate of 25%, whereas classifier B gets an error rate of 15%. The difference in error rate is big, so that we can almost confidentially determine that B is better than A. However, in the other case 2, the error rate of A is improved and decreased to 17%, with the performance of B staying unchanged. The difference in error rate this time is relatively small that we may not yield

1. https://en.wikipedia.org/wiki/Precision_and_recall
2. https://en.wikipedia.org/wiki/F1_score
3. https://en.wikipedia.org/wiki/Receiver_operating_characteristic

a conclusion confidentially. This is because the difference between two algorithms can be very small that it is hard to determine if the difference is caused by the two algorithms themselves or just by any randomness or variation in the way the comparison is carried out.

In light of the above mentioned problems, we would like to introduce statistical testing on the comparison of two classification algorithms. We conduct McNemar's test and 5x2cv paired t test on four different machine learning classification algorithms on a small dataset and demonstrate the effectiveness of both test methods. The time cost of each test is also recorded to give a good sense of suited scenarios to use one of the two tests.

1.2. Sources of Variation

To design and perform statistical tests, we first need to figure out possible sources of variations that must be controlled by each test. There are a few sources we think are common and important.

The first source is the variation in data split of train and test sets. In a typical classification task, we split the dataset randomly into two sets, one for training and one for testing. Due to the property of random-draw, one algorithm may outperform another even though the two algorithms do not have any profound difference. This phenomenon is especially obvious when the available dataset is small. Small variations in the training data may lead to big changes in the classifier produced by a learning algorithm, as demonstrated as "instability" by Breiman [1].

The second source of variation is the randomness of weight initiation in some machine learning algorithms, especially for those models in deep learning. Depending on different weight initialization states, the same deep neural network may be trained into classifiers with varied performances, even on the same training dataset. Actually how to best initialize a deep feed-forward neural network is still a hot topic in the field of machine learning and artificial intelligence [2].

The last but not the least source of variation is the size of available training data. The amount of data to train a good classifier is different depending on the learning algorithm. Usually deep learning classification models require far more training data than classical machine learning algorithms, and the deeper the neural network, the bigger the training set size. If we start with a small dataset and get to a conclusion that one algorithm is better than the other, it doesn't necessarily hold true when the training data is increased and enhanced.

The aforementioned sources of variation are not exclusive and in fact due to the limited time, we are not going to examine every aspect of them. We will instead focus on the design of statistic testing methods, given a fixed dataset and already well-initiated learning algorithms.

1.3. Hypothesis Testing

We are comparing two classifiers each time. The hypothesis test will be performed at a significance level (α) of 0.05 and the null hypothesis and alternative hypothesis can be formulated as follows:

$$H_0 : Two\ classifiers\ make\ errors\ in\ the\ same\ way$$

$$H_a : Two\ classifiers\ make\ errors\ differently$$

2. Method

2.1. McNemar's test

Test for the difference of two proportions: Before introducing McNemar's test, we will first introduce the test for the difference of two proportions, because it is more intuitive and correlates well with what we have learned in this course on the comparison of two population proportions. Basically, it is based on the test of difference between the error rates of two classifiers [3]. The assumption underlying this statistic test is that if

the error rate of one classifier A is p_A, then the possibility that A makes error is also p_A. We have thus got two proportions p_A and p_B on the test set size n, and we can carry out the usual test on the two population proportions. The test is quite straight-forward and easy to perform because it only needs one hold out test. It has a problem, however, that the two proportions p_A and p_B are actually not independent since they are evaluated on the same test dataset. A better approach which also needs only one hold out training and testing is the McNemar's test [4].

McNemar's test: In the McNemar's test, we train both classifiers A and B on the same training set and test on the same testing set, and record the results and construct a contingency table:

TABLE 1. CONTINGENCY TABLE

	B wrong	B right
A wrong	n_{00}	n_{01}
A right	n_{10}	n_{11}

where n_{00} means the number of samples both A and B predict correctly, and n_{01} means the number of samples only B gets right, and so on for n_{10} and n_{11}. Note $n = n_{00} + n_{01} + n_{10} + n_{11}$ is the total number of test samples.

McNemar's test is based on a χ^2 test that compares the distribution of counts expected under the null hypothesis. The following statistic is computed and is proved to be distributed (approximately) as χ^2 with 1 degree of freedom:

$$\frac{(|n_{01} - n_{10}| - 1)^2}{n_{01} + n_{10}}$$

If the null hypothesis is correct, then the probability that this statistic is greater than $\chi^2_{1,0.95} = 3.841$ is less than 0.05. We will reject the null hypothesis H_0 and conclude that the two classifiers A and B make errors differently, if the p value is less than 0.05.

2.2. 5x2cv paired t test

Before introducing 5x2cv paired t test, we would like to introduce resampled and k-fold cv paired t test .

Resampled paired t test: A natural way to circumvent the effect of randomness resulting from only one train/test run as applied in the McNemar's test, is to randomly sample multiple train/test data and evaluate the classifiers to get a series of paired error rates. Paired t test then can be performed to determine if the two sets of error rates differ from each other, just like the *inference for two population means* we have learned in this course.

K-fold cross-validated (cv) paired t test: Another way to do paired t test is through a k-fold cross validation. This time the train/test sets are not randomly drawn from the dataset pool, but rather the dataset pool is divided into k folds. Each time one fold is taken as a test set and the rest folds as a train set. It will result in k paired error rates and we can perform a usual paired t test on them.

Both the two paired t test methods have some potential drawbacks. As demonstrated by Dietterich [5], they tend to elevate the probability of Type I error, that is incorrectly reject null hypothesis while in reality it cannot be rejected. More specifically, the two methods tend to conclude classifiers to perform differently more often than they actually have to do. The source of the increased Type I error for resampled paired t test method comes from two factors: 1) The fact that train and test sets are randomly drawn will inevitably lead to overlapping among different training sets and also among testing sets, which will reduce the variance and thus increase the t statistic. That finally leads to a more frequent rejection of null hypothesis than expected. 2) Again because of randomly drawing data, each train and test sets are likely to contain an imbalance of points from different classes, that makes it not representative of the whole dataset. Dietterich has experimentally demonstrated that paired t test can

detect and magnify this difference until it is "statistically significant", and in fact it is the major reason of resulting in a high Type I error rate [5]. K-fold cv paired t test, on the other hand, greatly reduces the occurrence of imbalance by managing to use all the available data, in turn. The design of cross validation also avoids overlapping among different testing sets, with the only source of a slightly high Type I error to be the overlapping among different training sets. To deal with the problem of overlapping in training sets, a new approach is proposed by Dietterich: 5x2cv paired t test.

5x2cv paired t test: In 5x2cv paired t test, instead of a usual k-fold cv, the split is reduced to 2-fold so that training sets would never overlap in each run of cv. The procedure of 5x2cv paired t test can be described in the following table:

TABLE 2. PROCEDURE OF EACH CV

Train	Classifier	Error Rate	Difference
1^{st} fold	A	P_A^1	$P^1 = P_A^1 - P_B^1$
	B	P_B^1	
2^{nd} fold	A	P_A^2	$P^2 = P_A^2 - P_B^2$
	B	P_B^2	

In each run of cv, both classifiers A and B are firstly trained on the 1st fold and tested on the 2nd fold to yield the error rates P_A^1 and P_B^1. They are then trained on the 2nd fold and tested on the 1st fold to yield P_A^2 and P_B^2. Difference between A and B are computed to get P^1 and P^2. The variance of this run of cv is given as:

$$s^2 = (p^1 - \bar{p})^2 + (p^2 - \bar{p})^2$$

$$where\ \bar{p} = \frac{p^1 + p^2}{2}$$

Let s_i^2 be the variance computed from the i-th run of cv, and let p_1^1 be the p^1 from the first run of cv, we compute the following statistic:

$$\tilde{t} = \frac{p_1^1}{\sqrt{\frac{1}{5}\sum_{i=1}^{5} s_i^2}}$$

Under the null hypothesis, this $\tilde{t}$ statistic has approximately a t distribution, with 5 degrees of freedom. We will reject the null hypothesis if $\tilde{t} > t_{0.05,df}$ or p value < 0.05, with $df = 5$.

3. Experiment

We implement the two test methods in Python and conduct experiments using them on four classifiers on the wine dataset. The source code is designed to be easily extendable to various datasets and pairs of classifiers, and is available at *https://github.com/liyu10000/algorithms_testing*.

3.1. Dataset

The dataset we use is about wine recognition data which is provided by the UCI Machine Learning Repository[4]. These data are the results of a chemical analysis of wines grown in the same region in Italy but derived from three different cultivars. Three types of wine are represented in the 178 samples, with the results of 13 chemical analyses recorded for each sample. The data contains no missing values and consists of only numeric data, with a three class target variable for classification. Table 3 shows the number of instances in each class.

TABLE 3. NUMBER OF INSTANCES PER CLASS

Class	Number of Instances
class 1	59
class 2	71
class 3	48

3.2. Classifiers

The four classifiers are used as given in the Python packages such as *scikit-learn*. Since the purpose of this work is to examine the statistic

4. https://archive.ics.uci.edu/ml/datasets/Wine

tests themselves, we do not tune the parameters of classifiers intentionally. The process of finding the best algorithm for this specific classification task is for demonstrating the effectiveness of the two tests. The four classifiers we choose are as follows:

- XGBoost (XGB)[5]: It stands for eXtreme Gradient Boosting, which is an implementation of Gradient Boosted decision trees.
- Random Forest (RF)[6]: It is a classification algorithm consisting of many decisions trees.
- Support Vector Machine (SVM)[7]: It is a discriminative classifier formally defined by a separating hyperplane.
- Logistic Regression (LR)[8]: It is a statistical model that in its basic form uses a logistic function to model a binary dependent variable, although many more complex extensions exist.

We can just view them as classifiers with symbols XGB, RF, SVM, and LR, without losing the explainability of this work.

3.3. Test Result & Discussion

We run the two tests on each pair of classifiers and record the error rate, running time, testing result (whether or not to reject the null hypothesis). The results are listed in the following tables.

Table 4 records the error rates of four classifiers under two tests. Since the wine dataset size is small and there is only one train/test split in the McNemar's test, the error rates under it is not very stable, whereas the error rates under 5x2cv paired t test is much more stable because of the cross validation. The error rates shown in the table is from one run out of our multiple runs, but it should be representative. If not with the results of statistical testing, the conclusions we can draw from the comparison of error rates are that: 1) Under one-time train/test split, both RF and SVM are the best classifiers. 2) Under 5 runs of 2-fold cv, RF is the best classifier, with XGB the second best, LR the third, and SVM the worst. We will see how our conclusions can be changed with testing.

TABLE 4. ERROR RATE

Classifier	McNemar	5x2cv
XGB	0.028	0.051
LR	0.028	0.059
RF	0.0	0.035
SVM	0.0	0.073

TABLE 5. TEST RESULT COMPARISON OF TWO TESTS

	McNemar	5x2cv
XGB vs. LR	False	False
XGB vs. RF	**True**	False
XGB vs. SVM	**True**	False
LR vs. RF	False	False
LR vs. SVM	False	False
RF vs. SVM	**True**	**True**

Table 5 shows the testing results. The mark 'True' means we can reject the null hypothesis, that is, the two classifiers make errors differently. The mark 'False' means we fail to reject the null hypothesis and two classifiers make errors in the same way. Loosely speaking, the True mark indicates that two classifiers are different, and we can infer which one is better with the help of error rates. From table 5, it is interesting to find that although error rates suggest RF and SVM to be the same (make zero errors actually), McNemar's test still suggests they are different. It is weird because two perfect classifiers can hardly be different! After we check the implementation, we find out that it makes no sense to compare n_{01} and n_{10}

5. https://xgboost.readthedocs.io/en/latest/python/python_api.html#module-xgboost.sklearn

6. https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestRegressor.html

7. https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html

8. https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html

when they are too small, which partly explains why we get weird result. Apart from that, we get the test result of XGB vs. LR to be 'False', suggesting they are the same at making errors, which agrees well with their error rates.

Different from McNemar's test, the 5x2cv paired t test gives more convincing results. It can be seen from the table that only the pair RF vs. SVM gets a 'True' mark which means that they are different. It is contradictory to our observation from the table of error rates, where all four classifiers are different. Now that if we define the **best** classifier to be the one such that all others are no better than it, we can safely conclude that RF is the best classifier. On the other hand, we can never reach any conclusions about the other three classifiers. They are the same according to the test, even though they have different error rates.

TABLE 6. TIME COST COMPARISON OF TWO TESTS

Time(/s)	McNemar	5x2cv
XGB vs. LR	0.067	0.339
XGB vs. RF	0.500	3.608
XGB vs. SVM	0.570	1.621
LR vs. RF	0.454	3.290
LR vs. SVM	0.569	1.298
RF vs. SVM	0.884	4.894

We also record the time cost for the two test methods under each pair of classifiers, in the hope to prove that the time complexity is different. The results are shown in Table 6, with all values in seconds. It is clear that regardless of which pair we run, McNemar's test is always substantially faster than the 5x2cv paired t test. The result is expected because of the design of two methods. Even though the longest time is no more than 5 seconds in our experiment, it is worth noting that the time cost difference could be enlarged in some situations when we have a huge amount of data to work on and a big model like the ones in deep learning. In this case, we have to carefully evaluate which method to use and usually we can only rely on the McNemar's test, since the other one is too resource consuming.

4. Conclusion

In conclusion, we have studied two statistical testing methods for comparing the performance of machine learning classifiers. We implement the methods and test them on four different classifiers, on the wine dataset. The results show that both the McNemar's test and 5x2cv paired t test are effective in determining if the observed difference in performance is statistically significant, with the later test method to be more stable and convincing. We also record and compare the time cost of two methods, and find that McNemar's test is suitable for situations that running multiple train/test rounds are expensive, whereas the 5x2cv paired t test is best for the situations that multiple train/test runs are allowed and more stable comparisons are expected.

References

[1] Leo Breiman et al. Heuristics of instability and stabilization in model selection. *The annals of statistics*, 24(6):2350–2383, 1996.

[2] Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 249–256, 2010.

[3] Douglas H Jones. Book review: Statistical methods, george w. snedecor and william g. cochran ames: Iowa state university press, 1989. xix+ 491 pp. *Journal of Educational Statistics*, 19(3):304–307, 1994.

[4] Brian S Everitt. *The analysis of contingency tables*. Chapman and Hall/CRC, 1977.

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