Model Selection in Data Mining: A Statistical Approach

José Alberto Frydman, Francisco J. Cantú Ortiz[⋆]
Jorge H. Sierra, Rubén Morales-Menéndez

ITESM Monterrey campus
Monterrey NL, México
`afrydman@sbs.gob.pe, {fcantu,jsierra,rmm}@itesm.mx`

Abstract. Choosing the best algorithm for a machine learning task is an open problem. Statistical techniques such as *paired t-test* and k-fold cross validation have been proposed, but they require independence assumptions that are not met in many data mining tasks and for large data sets these techniques suffer from the multiplicative effect. We propose Analysis Of Variance as an alternative for comparing learning algorithms in large databases. We show that this approach avoids the drawbacks of the *paired t-test* and *k-fold* cross validation. We describe several experiments on a DELVE database that show the advantages of our proposal in comparing the decision tree learning algorithms Gain Ratio, Gini and Chi Squared Automatic Interaction Detection as well as the naive Bayes algorithm.

Keywords: Machine Learning, Decision Trees, *Paired t-test*, ANOVA.

1 Introduction

The development of learning algorithms in Machine Learning (*ML*) is quite vast and varied since there are many ways of searching the space of hypotheses for constructing ways of learning called models. Learning algorithms for model construction include Decision trees, and Bayesian nets. One of the fundamental questions in *ML* is this: given a specific learning problem and context, which algorithm is the one that generates the best model?. Alternatively the question may be in this way: given several learning algorithms and a large database which algorithm will produce the most accurate predictions?. This article intends to give an approximation to the second question. Our starting premise is that the most important factor to look at when selecting an algorithm is its error in predicting unseen cases; however, we take into account that other factors like speed of learning and understandability of the generated model can be more important in some domains.

A model, also called a classifier, is a function that given an example, the function assigns the example a class among several classes. A learning algorithm

[⋆] Corresponding author

is a function that given a database of instances and its classes constructs a classifier for assigning classes to unseen instances. The learning function is searched from a huge space of possible hypotheses.

Model validation means the evaluation of algorithms by comparing the accuracy of the models generated by the algorithms. Validation can be divided basically in three independent processes: **(1)** Measuring the error in the algorithm, **(2)** Improving the performance, **(3)** Comparing two or more algorithms. We propose a method for comparing more than two learning algorithms to select the best one depending upon the problem to be solved.

In this article we start from some premises related to points 1 and 2. Measuring the error deals with establishing a way of predicting the error in future cases; there are many different techniques to measure this, that can be chosen according to the number of examples. [16] suggests re-sampling techniques for data bases with less than $1,000$ examples, and the holdout method for data bases where more than $1,000$ examples can be used for testing. The Holdout method consists of separating the data in two sections: training and testing. We use the holdout method and justify why cross validation is not appropriate for model comparison.

For improving the performance of a model there are three strategies: a) manipulate the database reducing the training series, perform attribute selection or create new ones b) combine several models and weigh the results (Bagging or Boosting [1]) c) optimize the parameters of the algorithm. When comparing Data Mining (DM)algorithms it is important to recognize that any of these three strategies could produce a better result on one algorithm over another, and this fact would make algorithm selection a much more complicated process, [12]. However, [16] suggests the use of each algorithm in its best form, that is, if the algorithm is improved by some kind of manipulation, then it should be used. In this work we used the first approach.

We propose the use of *Analysis of Variance* ($ANOVA$)as an alternative test to the *paired t test*. Following the same line of reasoning, no costs for different errors were taken into account (false negative or false positive) because not all of the learning algorithms can deal with them. We worked only with the 0-1 loss function.

This work is organized as follows: In section 2 we present a review of works for comparing models of *Data Mining*, and we discuss some common problems. In section 3 we present the $ANOVA$ as a proposed solution to these problems. In section 4 we describe the experiments and the results. Finally, section 5 presents the conclusions.

2 Literature Review

Usually, model error is measured by a technique known as ten-fold cross validation since it has been shown that it gives an almost unbiased estimator of the true error [10]. Ten-fold cross validation consists on randomly splitting the database in ten sections. Then, one section is left aside at a time and the algorithm is trained with the 9/10 remaining, after that, the error is computed in the series that was left aside. Hence the learning algorithm is performed ten times, each time with a different training series (even though they share many examples) and each time tested in a completely different test series. Finally, the average of the ten estimated errors is taken to produce the estimate of the error for the classifier. Traditionally, when we need to compare the error of two learning algorithms A and B in a database D_0, it is customary to apply the following procedure based on k fold cross validation, [13]:

1. Database D_0 is split into k sections $T_1, T_2, \ldots, T_k$ (folds) where each section contains at least 30 examples (heuristic rule based on the central limit.)
2. For $i = 1$ to k , use T_i as the testing series and the remaining of the data as the training series S_i.
   – Measure the error of the algorithms, $E_A(S_i)$, $E_B(S_i)$ in the testing series T_i
   – Measure the difference in the computed error obtained by each of the two algorithms $\delta_i = E_A(S_i) - E_B(S_i)$
3. Compute the average of the differences of the errors obtained $\overline{\delta}$, where $\overline{\delta} = \frac{1}{k}\sum_{i=1}^{k}\delta_i$

This procedure is identical to ten-fold cross validation, with the only difference that instead of obtaining the error in only one algorithm, the difference in errors of the two algorithms is obtained. It is worth stating that the number k can be 10 or a larger number with the only restriction that there are at least 30 examples in each section (fold).

Now, if the two algorithms have the same performance it would be expected that the value of $\overline{\delta}$ is approximately zero. But, how close to zero? The well-known *paired t test* is generally used to determine if the value of $\overline{\delta}$ is small enough to conclude that the two algorithms have the same error. It is called *paired t test* because the two values obtained, $E_A(S_i)$ and $E_B(S_i)$, are statistically dependent (in this case they are dependent because the algorithms are trained in the same training series and are also tested on the same testing series). In the *paired t test* the test statistic is: $t = \frac{\overline{\delta} - \Delta_0}{S_D/\sqrt{n}}$ where Δ_0 tends to 0.

The null hypothesis states that the difference in error is practically zero whereas the alternative hypothesis states that the difference is less than, greater than or not equal to zero. However, one of the assumptions of the *paired t test* is that the **pairs of values** $(E_A(S_1), E_B(S_1)), \ldots, (E_A(S_k), E_B(S_k))$ are independent, assumption that is violated in the previously explained procedure. But how and where is this assumption violated?

The *paired t test* is based on the premise that all individual measures are independent (measures in this *paired t test* are differences δ_i on the k estimated errors of the two algorithms).

Unfortunately, cross validation (as well as bootstrapping) introduces some degree of **dependence** amongst these measures, and this produces a bias on the *t test* towards rejecting the hypothesis of equality of means. But, why are the measures dependent? Each measure depends on two quantities: a) the training series and b) the models constructed from the corresponding training series. Testing series are independent, but the models are not. Each algorithm develops a model in *10* training series, (assuming $k = 10$) which are almost identical.

Each training series shares 89% (8/9) of the data with all the other training series on the 10 iterations, therefore the training series overlap, and this introduces a dependence amongst the models generated from them, hence, the errors produced by these models will also be dependent.

But this is not the only drawback on using this test. When we want to compare more than two population means, the *paired t test* is not recommended for the following reason. Suppose that from a certain population five samples are considered from which their corresponding means are computed: $\overline{x}_1, \ldots, \overline{x}_5$. It is required to perform the following 10 *t tests* to compare the 10 pairs of sample means $(\overline{x}_1, \overline{x}_2), \ldots, (\overline{x}_1, \overline{x}_5)$. If more sample means are to be compared, more t tests should be performed; for example, a total of six samples increase the number of pairs to fifteen. However, when each sample mean is paired with another and a *t test* is performed for each pair of means, the possibility of it showing significant differences for some pairs of means is increased, even if the sample means proceed from the same population! This phenomenon is known in statistics as the multiplicative effect. Then, whenever comparing three or more sample means, amongst themselves, the *t test* is inappropriate. Another technique should be used to test the hypothesis of equality of means [1].

Up to here we have mentioned only the *paired t test*, however, there exist other approaches to comparing learning algorithms. For instance Dietterich [5] suggests the test $5 \times 2cv$ as the best test to compare errors (means) for algorithms that can be executed efficiently 10 times, he warns however that this test violates some of the independence assumptions. Other authors even question if the type I error is the one to compare (Type I error consists on falsely rejecting the hypothesis of equality of means). Also, due to how unstable learning algorithms are, any comparison between two or more of them must be done using multiple runs on different training series to remove the variability of the algorithm [4], [7]; this is the approach we take here. There is not a consensus amongst the research community about the method to use for comparison studies [11].

3 Comparing Algorithms using Analysis of Variance

We have mentioned two drawbacks about using the *paired t test*; on one hand the overlapping of the training series due to cross validation and on the other the multiplicative effect when comparing more than two algorithms. To correct the former, the method proposed here is applicable only for large databases[2], so that the holdout method can be used to measure error and hence eliminate the dependence produced by cross validation. Typically, the databases are huge (thousands or millions) in the data mining community. To avoid the latter, a statistical test should be found that does the same function as the *paired t test* but that can be used for more than two populations; such a test is the Analysis of

[1] Sample mean corresponds in this work to the algorithm error.

[2] In some domains $2,000$ examples can be a large number whereas in another domain would be totally insufficient. We propose the use of learning curves to determine the number of examples needed to learn an underlying pattern.

Variance (**ANOVA**). ANOVA has been proposed for learning in support vector machines [15].

ANOVA is a statistical procedure for comparing the means of a population under the effect of different treatments. It is called a one-way ANOVA when only one variable effects the population, in a several-way ANOVA there are more variables effecting the population. For this work the treatments studied with ANOVA are the learning algorithms that have been trained with the training series, and the effects these treatments produce on the testing series will be studied, that is, the errors produced by the classifiers. Before applying ANOVA we must verify that its statistical assumptions are satisfied [3]. These assumptions are: **(1)** The observations in each sample are independent. **(2)** The underlying distribution in each population is normal. **(3)** All the populations have the same variance.

4 Experiments and Results

We present two experiments and for each four learning algorithms are compared; three decision trees generated with different split rules, (Gain ratio, Gini, and CHAID or Chi squared Automatic Interaction Detection) and the Bayesian classifier (Naive Bayes). Several approaches have been proposed for learning decision trees from databases [8]. To compute the errors obtained by each of the algorithms, we used the *DM* software Mineset 3.1 (Silicon Graphics). In the decision trees the level of pruning was established at 0.7(default value) and in the Bayesian classifier the Laplace corrector was set to automatic.

The database used in the experiments consists of $45,000$ entries and was obtained from the Delve archives, it can be found under the name Adult Dataset. Each entry presents 13 attributes; 6 continuous, 7 nominal, and it does not show any missing values. The dependent variable is Income, and can be classified in greater or less than $50,000$ US dollars annually. The purpose of this database is to classify if a person has an income greater or smaller than $50,000$ dollars (dichotomous variable). From now on we call a *sample* from the population the series of errors obtained by the algorithms in the database, being the population the universe of all errors that would be obtained from an infinite database from the same domain of application.

The first experiment consists of a two way ANOVA (one factor being the algorithms and the other as the training and testing series - the two of them together -), using blocking in the training and testing factor. We split at random the database in 20 sections, that is, each part consisted of $2,250$ examples. In each of them, we chose at random $1,250$ examples for training and $1,000$ examples for testing; in this way we got 4 samples of 20 errors each. See left side in Fig 1.

The numbers may seem arbitrary but there is a reason for this; first, the database was broken in 20 parts because the analysis of variance gets a better consistency when the number of observations is relatively large (20 is an acceptable value). The decision to assign $1,250$ examples for training to each subsections due to the fact that the error made by the algorithms when training with $1,250$ examples was almost the same as the error made with all the

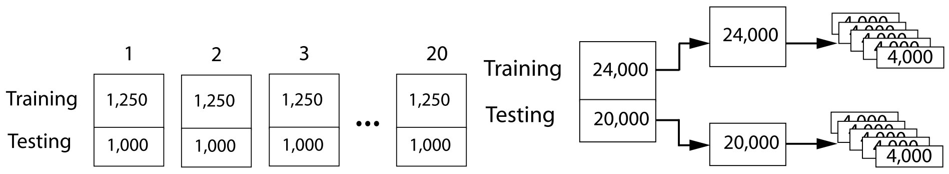

Fig. 1. Training and Testing examples : 1st and 2nd experiments

45, 000 examples (this was verified using learning curves), therefore 1, 250 training examples are a reasonable number to train the algorithms. Also, the 1, 000 test examples represent the minimum acceptable number of examples that are needed to compute with a good approximation the error in an algorithm (with only 1% deviation of the true error). To have an exact judgment on the error in a model, 5, 000 testing examples are needed (with a deviation in the computation of the error of 0.5%) [16]. Although the way the database was split may seem arbitrary, the reader may verify with has been said that there is a theoretical and also an empirical reason. The errors obtained with the four algorithms are shown in Table 1. See errors in Table 1 for these algorithms.

Table 1. Performance of the different methods

	Error					
Algorithm	1	2	...	19	20	Average
Gain Ratio	17.8	14.5	...	16.7	14.6	16.65 %
Chi-Squared	18.9	18.8	...	19	16.6	20.05 %
Gini	17.1	16.1	...	20.2	17.1	19.05 %
naive Bayes	18.5	17.3	...	17.5	18.4	17.92 %

The model generated in this first experiment is a model of mixed effects since we are considering a non-random factor (algorithms) and a random one (the training and testing series). This random factor represents the blocking. Blocking is done when an external variable is present and can affect the results of the test [2]. In the case of comparing algorithms, the external variable is the random selection of the training and testing examples. At first sight we can see that Gain Ratio is the treatment with less error and this is proven using analysis of variance. Once we have the observed errors (that conform four samples)we verify that the statistical assumptions of ANOVA are satisfied.

Assumption (1) is satisfied due to the form in which the samples were constructed. Both the training and testing examples are randomly chosen. All the samples have a size of twenty observations ($2, 250 \times 20$). To check assumption (2) a statistical test known as Shapiro-Wilks was applied. In this test the null hypothesis is that the population is normally distributed. When applying the test to the 20 observations obtained from the treatment Gain Ratio we got a p-value of 0.4088 ($W = 0.9519$). As can be seen, the p-value is large and therefore the null hypothesis cannot be rejected. Similar results were obtained with the other three treatments (algorithms). So, assumption (2) is also satisfied in the four treatments. To test for assumption (3) 4 tests were conducted (in all four of them the null hypothesis is that the variances are equal). The tests: O'Brien, Brown-Forsythe, Levene and Barlett did not reject the null hypothesis, that is, we cannot reject the equality of variances in the average errors of the algorithms. It must be clear that assumption (3) states that the population variances must be equal regardless of the probability distribution of the data (the distribution may be binomial). The normality was tested by the classical tests.

Once we have verified that all the assumptions are satisfied, we can proceed to perform the analysis of variance. In Table 2 we can see the summary of the analysis of variance, as if there was only one factor. Since the p-value is less than 0.0001 we can safely reject the null hypothesis, which means the algorithms behave significantly different. In another results, not shown here, the treatments as well as the blocks are significant (we reject the null hypothesis), that is, they have an influence on the degree of error.

Table 2. Analysis of Variance like a whole

Source	DF	Sum of Squares	Mean Square	F Ratio
Model	22	235.629	10.7104	7.5926
Error	57	80.4065	1.4106	Prob>F
C Total	79	316.0355		<.0001

When the null hypothesis in the analysis if variance is not rejected, the analysis is concluded and no significant differences between the means (in this case average errors) were found [3]. But if the null hypothesis is rejected, it is of interest to know which means differ from the others. The method to continue with this analysis is called multiple comparisons. There are several tests in statistical literature to do this. One such tests is Tukey's test. For the case of comparing algorithms we would like to compare all pairs formed from the four treatments. When applying this procedure to the factor *algorithms* we get[3]: $Q_{0.05,4,57} = 3.75\sqrt{MSE/J} = 3.75\sqrt{1.41/20} = 0.995$.

As we may see, 0.995 is smaller than any difference between the averages of the four algorithms $(17.92 - 16.65)$, $(19.05 - 17.92)$, $(20.05 - 19.05)$. Therefore, we can conclude that all four algorithms are significantly different amongst themselves: they are all different. It would be interesting to establish the reason

[3] MSE : Mean Square Error, Q: Critical Value of Studentized Range.

why this difference exists between the algorithms; a bias and variance analysis in each of them may help in finding the reason[4].

For the second experiment, we consider three factors instead of two: algorithms, training series and testing series. The three factor ANOVA has exactly the same assumptions as one way and two way ANOVA; and so, the results on verifying the assumptions in this second experiment is not shown here.

The database was split, one with $24,000$ examples for training and the other with $20,000$ examples for testing (both randomly chosen). Both, training an testing examples were also split into groups of 4,000 examples, that is, we got 6 groups of $4,000$ training examples and 5 groups of $4,000$ testing examples, look right side of figure 1. Each of the four algorithms was trained on the 6 training series and was tested in the 5 testing series, giving a total of 120 observations (4 algorithms x 6 training series x 5 testing series). This methodology produce 4 samples of 30 (6×5) observations each for the ANOVA experiment.

This experiment, as the first one, is a mixed effects one since the treatments are fixed, but the training and testing series are random. The interesting part of this experiment is that the interaction between the 3 factors is also considered (algorithms, training and testing), which was not present in the first experiment. The null hypothesis is the same as before; all errors are the same, Table 3[5].

Table 3. Three factors test of effects (A = Algorithm, T = Training, Te = Testing)

Source	SS	MS Num	DF Num	F Ratio	Prob >F
A	98.88	32.96	3	30.22	<0.0001
T	13.72	2.74	5	5.98	0.0025
A-T	6.03	0.40	15	3.10	0.0009
Te	9.70	2.42	4	2.77	0.07
A-Te	9.82	0.82	12	6.32	<0.0001
T-Te	3.72	0.19	20	1.44	0.1408

We can see that in the majority of the factors, the null hypothesis is rejected except in the factor testing, and in the factor interaction between testing and training ($T-Te$). This means that testing examples by themselves do not change the results. This may be due to the fact that the number of training examples is quite large $(4,000)$ as to assure the goodness of the approximation to the true error of the algorithm. In other words, using $4,000$ testing examples would not change significantly the mean error of the algorithm. As was mentioned earlier, with $5,000$ testing examples we can be completely certain that the computed error is the true error of the algorithm, so with 4,000 the approximation is almost perfect [16]. Treatment effects are highly significant, and we can conclude

[4] The bias of an algorithm is the systematic error that cannot be eliminated even if equally size infinite training series are obtained [6] .

[5] SS : Sum of Squares, MS: Mean Square, DF : Degree of Freedom.

that each algorithm has a different effect. Training examples present a significant difference also, from here we can see that when varying the training series the generated classifiers are significantly different (even the classifiers generated with the same algorithm). This behavior is what Breiman called an "unstable" algorithm [4]. The summary of the analysis is shown in Table 4. We can see that the hypothesis of equality of means is also rejected (p-value < 0.0001). The average errors obtained in this experiment are: Bayes (17.52%), Chi S. (18.36%), Gain R. (15.92%), Gini (17.82%). Note that the Gain Ratio is the best.

Table 4. Analysis of variance as a whole for three factors

Source	DF	SS	Mean Square	F Ratio
Model	59	141.87	2.40	18.56
Error	60	7.77	0.13	Prob>F
C Total	119	149.63		<.0001

We perform two independent experiments. None of them is better than the other, they are simply two ways of approaching the ANOVA for the same problem; the reader can choose the one that seems more appropriate. There is no agreement amongst the AI community about how to compare learning algorithms [9]. Even statisticians have some problems in agreeing on the correct approach in complex experimental design problems [14].

5 Conclusions

The results obtained for both experiments are the same: in first place, algorithms are significantly different and, in second place, Gain Ratio is the best algorithm to predict if a person has an income greater to $50,000$ dollars. Therefore, we conclude that ANOVA is able to compare several learning algorithms and select the best algorithm for this particular database. In this case the best algorithm is Gain Ratio. It may happen that algorithms don't show differences, and in such a case, we could chose the fastest algorithm or the easiest to interpret (for example, a decision tree would be preferred over a neural net) .

ANOVA is an alternative technique that does not violate the statistical assumptions of commonly used tests. Three statistical assumptions for ANOVA must be satisfied and also a sufficiently large database should be used (to be able to break it into smaller databases). Although these requirements may seem burdensome and restricted only to domains that satisfy them, it is worth trying this technique when the degree of error is the most important factor to deal with. We propose a method to determine the best algorithm in a particular problem.

Experimental design was not developed with computational algorithms in mind, therefore, any application of the statistical techniques for this purpose must be seen as a mere approximation to the solution of the problem, but not as the only form to compare algorithms.

Choosing the best algorithm for a particular problem is an exploratory process; it depends directly on the knowledge the analyst has of the algorithms and the domain of application, hence involving some science and also some art. Fundamental research in *ML* is inherently empirical, because the performance of *ML* algorithms is determined by how well their underlying assumptions match the structure of the world [4].

There is not a general theory about how to choose an algorithm a priori. What we would like is to find a series of measures obtained from the data from which we could predict with a certain degree of confidence which algorithm will behave the best performance[11] but currently, this is not possible. If we want to prove that an algorithm is in general better than another (this is perhaps one of the fundamental questions in *ML* [4], [5]), a database used in the experiment should be chosen first, however, the question about which databases are representative is still unanswered.

References

1. E Bauer and R Kohavi. An empirical comparison of voting classification algorithms: Bagging, boosting, and variants. *Machine Learning*, 36:105–142, 1999.
2. G E P Box, W G Hunter, and Stuart. *Estadística para investigadores.* John Wiley and Sons Inc, 1988.
3. J L Devore. *Probabilidad y Estadística para Ingeniería y Ciencias.* Thomson and Learning, 2001.
4. T G Dietterich. Editorial. *Machine Learning*, 24(2):1–3, 1996.
5. T G Dietterich. Approximate statistical tests for comparing supervised classification learning algorithms. *Neural Computation*, 10(7):1895–1924, 1998.
6. T G Dietterich. Machine learning bias, statistical bias, and statistical variance of decision tree algorithms. *Machine Learning*, 1998.
7. T G Dietterich. Machine learning for sequential data: A review. In T Caelli, editor, *Structural, Syntactic, and Statistical Pattern Recognition*, volume 2396, pages 15–30. Springer-Verlag, 2002.
8. W Fan, H. Wan, P S Yu, and H. S. Lo. Inductive learning in less than one sequential data scan. In *Proc IJCAI*, 2003.
9. A Feelders and W Verkooijen. *Learning from Data: AI and Statistics V, chapter On the Statistical Comparison of Inductive Learning.* Springer-Verlag, 1996.
10. R Kohavi. A study of cross-validation and bootstrap for accuracy estimation and model selection. In *Proc IJCAI*, 1995.
11. R Kohavi. Data mining using MLC++. *Machine Learning*, 6(4):234–245, 1996.
12. D Michie, D J Spiegelhalter, and C C Taylor. *Machine Learning, Neural and Statistical Classification.* Ellis Horwood, 1994.
13. T Mitchell. *Machine Learning.* McGraw-Hill, 1997.
14. S L Salzberg. On comparing classifiers: A critique of current research and methods. *Data Mining and Knowledge Discovey*, 1:1–12, 1999.
15. G. Valentini and T. G. Dietterich. Bias-variance analysis of support vector machines for the development of svm-based ensemble methods. *Machine Learning Research*, To appear, 2004.
16. S M Weiss and C A Kulikowski. *Computer Systems that Learn.* Morgan Kaufmann, 1991.