Data Mining and Knowledge Discovery, 1, 317–327 (1997)
© 1997 Kluwer Academic Publishers, Boston. Manufactured in The Netherlands.

On Comparing Classifiers: Pitfalls to Avoid and a Recommended Approach

STEVEN L. SALZBERG salzberg@cs.jhu.edu
Department of Computer Science, Johns Hopkins University, Baltimore, MD 21218, USA

Editor: Usama Fayyad

Abstract. An important component of many data mining projects is finding a good classification algorithm, a process that requires very careful thought about experimental design. If not done very carefully, comparative studies of classification and other types of algorithms can easily result in statistically invalid conclusions. This is especially true when one is using data mining techniques to analyze very large databases, which inevitably contain some statistically unlikely data. This paper describes several phenomena that can, if ignored, invalidate an experimental comparison. These phenomena and the conclusions that follow apply not only to classification, but to computational experiments in almost any aspect of data mining. The paper also discusses why comparative analysis is more important in evaluating some types of algorithms than for others, and provides some suggestions about how to avoid the pitfalls suffered by many experimental studies.

Keywords: classification, comparative studies, statistical methods

1. Introduction

Data mining researchers often use classifiers to identify important classes of objects within a data repository. Classification is particularly useful when a database contains examples that can be used as the basis for future decision making; e.g., for assessing credit risks, for medical diagnosis, or for scientific data analysis. Researchers have a range of different types of classification algorithms at their disposal, including nearest neighbor methods, decision tree induction, error back propagation, reinforcement learning, and rule learning. Over the years, many variations of these algorithms have been developed and many studies have been produced comparing their effectiveness on different data sets, both real and artificial. The productiveness of classification research in the past means that researchers today confront a problem in using those algorithms, namely: how does one choose which algorithm to use for a new problem? This paper addresses the methodology that one can use to answer this question, and discusses how it has been addressed in the classification community. It also discusses some of the pitfalls that confront anyone trying to answer this question, and demonstrates how misleading results can easily follow from a lack of attention to methodology. Below, I will use examples from the machine learning community which illustrate how careful one must be when using fast computational methods to mine a large database. These examples show that when one repeatedly searches a large database with powerful algorithms, it is all too easy to "find" a phenomenon or pattern that looks impressive, even when there is nothing to discover.

It is natural for experimental researchers to want to use real data to validate their systems. A culture has evolved in the machine learning community that now insists on a convincing evaluation of new ideas, which very often takes the form of experimental testing. This is a healthy development and it represents an important step in the maturation of the field. One indication of this maturation is the creation and maintenance of the UC Irvine repository of machine learning databases [17], which now contains over 100 datasets that have appeared in published work. This repository makes it very easy for machine learning researchers to compare new algorithms to previous work. The data mining field, although a newer area of research, is already evolving a methodology of its own to compare the effectiveness of different algorithms on large databases. Large public databases are becoming increasingly popular in many areas of science and technology, bringing with them great opportunities, but also technical dangers. As we will see below, however, one must be very careful in the design of an experimental study using publicly available databases.

Although the development and maintenance of data repositories has in general been positive, some research on classification algorithms has relied too heavily on the UCI repository and other shared datasets, and has consequently produced comparative studies whose results are at best confusing. To be more precise: it has become commonplace to take two or more classifiers and compare them on a random selection of datasets from the UCI repository. Any differences in classification accuracy that reach statistical significance (more on that below) are provided as supporting evidence of important differences between the algorithms. As argued below, many such comparisons are statistically invalid. The message to the data mining community is that one must be exceedingly careful when using powerful algorithms to extract information from large databases, because traditional statistical methods were not designed for this process. Below I give some examples of how to modify traditional statistics before using them in computational evaluations.

2. Comparing algorithms

Empirical validation is clearly essential to the process of designing and implementing new algorithms, and the criticisms below are not intended to discourage empirical work. Classification research, which is a component of data mining as well as a subfield of machine learning, has always had a need for very specific, focused studies that compare algorithms carefully. The evidence to date is that good evaluations are not done nearly enough—for example, Prechelt [18] recently surveyed nearly 200 experimental papers on neural network learning algorithms and found most of them to have serious experimental deficiencies. His survey found that a strikingly high percentage of new algorithms (29%) were not evaluated on any real problem at all, and that very few (only 8%) were compared to more than one alternative on real data. In a survey by Flexer [9] of experimental neural network papers, only 3 out of 43 studies in leading journals used a separate data set for parameter tuning, which leaves open the possibility that many of the reported results were overly optimistic.

Classification research comes in a variety of forms: it lays out new algorithms and demonstrates their feasibility, or it describes creative new algorithms which may not (at first) require rigorous experimental validation. It is important that work designed to be primarily comparative does not undertake to criticize work that was intended to introduce creative new ideas or to demonstrate feasibility on an important domain. This only serves to suppress creative work (if the new algorithm does not perform well) and encourages people instead to focus on narrow studies that make incremental changes to previous work.

On the other hand, if a new method outperforms an established one on some important tasks, then this result would be worth reporting because it might yield important insights about both algorithms. Perhaps most important, comparative work should be done in a statistically acceptable framework. Work intended to demonstrate feasibility, in contrast to purely comparative work, might not always need statistical comparison measures to be convincing.

2.1. *Data repositories*

In conducting comparative studies, classification researchers and other data miners must be careful not to rely too heavily on stored repositories of data (such as the UCI repository) as its source of problems, because it is difficult to produce major new results using well-studied and widely shared data. For example, Fisher's iris data has been around for 60 years and has been used in hundreds (maybe thousands) of studies. The NetTalk dataset of English pronunciation data (introduced by Sejnowski and Rosenberg, [21] has been used in numerous experiments, as has the protein secondary structure data (introduced by Qian and Sejnowski [19]), to cite just two examples. Holte [12] collected results on 16 different datasets, and found as many as 75 different published accuracy figures for some of them. Any new experiments on these and other UCI datasets run the risk of finding "significant" results that are no more than statistical accidents, as explained in Section 3.2. Note that the repository still serves many useful functions; among other things, it allows someone with a new algorithmic idea to test its plausibility on known problems. However, it is a mistake to conclude, if some differences do show up, that a new method is "significantly" better on these datasets. It is very hard – sometimes impossible – to make such an argument in a convincing and statistically correct way.

2.2. *Comparative studies and proper methodology*

The comparative study, whether it involves classification algorithms or other data extraction techniques, does not usually propose an entirely new method; most often it proposes changes to one or more known algorithms, and uses comparisons to show where and how the changes will improve performance. Although these studies may appear superficially to be quite easy to do, in fact it requires considerable skill to be successful at both improving known algorithms and designing the experiments. Here I focus on design of experiments, which has been the subject of little concern in the machine learning community until recently (with some exceptions, such as [15] and [3]). Included in the comparative study category are papers that neither introduce a new algorithm nor improve an old one; instead, they consider one or more known algorithms and conduct experiments on known datasets. They may also include variations on known algorithms. The goal of these papers is ostensibly to highlight the strengths and weaknesses of the algorithms being compared. Although the goal is worthwhile, the approach taken by such papers is sometimes not valid, for reasons to be explained below.

3. Statistical validity: a tutorial

Statistics offers many tests that are designed to measure the significance of any difference between two or more "treatments." These tests can be adapted for use in comparisons of classifiers, but the adaptation must be done carefully, because the statistics were not designed with computational experiments in mind. For example, in one recent machine learning study, fourteen different variations on several classification algorithms were compared on eleven datasets. This is not unusual; many other recent studies report comparisons of similar numbers of algorithms and datasets.[1] All 154 of the variations in this study were compared to a default classifier, and differences were reported as significant if a two-tailed, paired t-test produced a p-value less than 0.05.[2] This particular significance level was not nearly stringent enough, however: if you do 154 experiments, then you have 154 chances to be significant, so the expected number of "significant" results at the 0.05 level is $154 * 0.05 = 7.7$. Obviously this is not what one wants. In order to get results that are truly significant at the 0.05 level, you need to set a much more stringent requirement. Statisticians have been aware of this problem for a very long time; it is known as the *multiplicity effect*. At least two recent papers have focused their attention nicely on how classification researchers might address this effect [10, 8].

In particular, let α be the probability that if no differences exist among our algorithms, we will make at least one mistake; i.e., we will find at least one significant difference. Thus α is the percent of the time in which we (the experimenters) make an error. For each of our tests (i.e., each experiment), let the nominal significance level be α^*. Then the chance of making the right conclusion for one experiment is $1 - \alpha^*$.

If we conduct n independent experiments, the chance of getting them all right is then $(1-\alpha^*)^n$. (Note that this is true only if all the experiments are independent; when they are not, tighter bounds can be computed. If a set of different algorithms are compared on the same test data, then the tests are clearly not independent. In fact, a comparison that draws the training and test sets from the same sample will not be independent either.) Suppose that in fact no real differences exist among the algorithms being tested; then the chance that we will *not* get all the conclusions right — in other words, the chance that we will make at least one mistake is

$$\alpha = 1 - (1 - \alpha^*)^n$$

Suppose for example we set our nominal significance level α^* for each experiment to 0.05. Then the odds of making at least one mistake in our 154 experiments are $\alpha = 1 - (1 - .05)^{154} = 0.9996$. Clearly, a 99.96% chance of drawing an incorrect conclusion is not what we want! Again, we are assuming that no true differences exist; i.e., this is a conditional probability. (More precisely, there is a 99.96% chance that at least one of the results will incorrectly reach "significance" at the 0.05 level.)

In order to obtain results significant at the 0.05 level with 154 tests, we need to set $1 - (1 - \alpha^*)^{154} \leq 0.05$, which gives $\alpha^* \leq 0.0003$. This criterion is over 150 times more stringent than the original $\alpha \leq 0.05$ criterion.

The above argument is still very rough, because it assumes that all the experiments are independent of one another. When this assumption is correct, one can make the adjustment of significance described above, which is well known in the statistics community as the

Bonferroni adjustment. When experiments use identical training and/or test sets, the tests are clearly not independent. The use of the wrong p-value makes it even more likely that some experiments will find significance where none exists. Nonetheless, many researchers proceed with using a simple t-test to compare multiple algorithms on multiple datasets from the UCI repository; see, e.g., Wettschereck and Dietterich [23]. Although easy to conduct, the t-test is simply the wrong test for such an experimental design. The t-test assumes that the test sets for each "treatment" (each algorithm) are *independent*. When two algorithms are compared on the same data set, then obviously the test sets are not independent, since they will share some of the same examples—assuming the training and test sets are created by random partitioning, which is the standard practice. This problem is widespread in comparative machine learning studies. (One of the authors of the study cited above has written recently that the paired t-test has "a high probability of Type I error ... and should never be used" [5].) It is worth noting here that even statisticians have difficulty agreeing on the correct framework for hypothesis testing in complex experimental designs. For example, the whole framework of using alpha levels and p-values has been questioned when more than two hypotheses are under consideration [20].

3.1. Alternative statistical tests

One obvious problem with the experimental design cited above is that it only considers overall accuracy on a test set. But when using a common test set to compare two algorithms, a comparison must consider four numbers: the number of examples that Algorithm A got right and B got wrong ($A > B$), the number that B got right and A got wrong ($B > A$), the number that both algorithms got right, and the number that both got wrong. If one has just two algorithms to compare, then a simple but much improved (over the t-test) way to compare them is to compare the percentage of times $A > B$ versus $B > A$, and throw out the ties. One can then use a simple binomial test for the comparison, with the Bonferroni adjustment for multiple tests. An alternative is to use random, distinct samples of the data to test each algorithm, and to use an analysis of variance (ANOVA) to compare the results.

A simple example shows how a binomial test can be used to compare two algorithms. As above, measure each algorithm's answer on a series of test examples. Let n be the number of examples for which the algorithms produce different output. We must assume this series of tests is independent; i.e., we are observing a set of Bernoulli trials. (This assumption is valid if the test data is a randomly drawn sample from the population.) Let s (successes) be the number of times $A > B$, and f (failures) be the number of times $B > A$. If the two algorithms perform equally well, then the expected value $E(s) = 0.5n = E(f)$. Suppose that $s > f$, so it looks like A is better than B. We would like to calculate

$$P(s \geq \text{observed value}|p(\text{success}) = 0.5)$$

which is the probability that A "wins" over B at least as many times as observed in the experiment. Typically, the reported p-value is double this value because a 2-sided test is used. This can be easily computed using the binomial distribution, which gives the probability of s successes in n trials as

$$\frac{n!}{s!(n-s)!}p^s q^{n-s}$$

If we expect no differences between the algorithms, then $p = q = 0.5$. Suppose that we had $n = 50$ examples for which the algorithms differed, and in $s = 35$ cases algorithm A was correct while B was wrong. Then we can compute the probability of this result as

$$\sum_{s=35}^{50} \frac{n!}{s!(n-s)!}(0.5)^n = 0.0032$$

Thus it is highly unlikely that the algorithms have the same accuracy; we can reject the null hypothesis with high confidence. (Note: the computation here uses the binomial distribution, which is exact. Another, nearly identical form of this test is known as McNemar's test [6], which uses the χ^2 distribution. The statistic used for the McNemar test is $(|s-f|-1)^2/(s+f)$, which is simpler to compute.) If we make this into a 2-sided test, we must double the probability to 0.0064, but we can still reject the null hypothesis. If we had observed $s = 30$, then the probability would rise to 0.1012 (for the one-sided test), or just over 10%. In this case we might say that we cannot reject the null hypothesis; in other words, the algorithms may in fact be equally accurate for this population.

The above is just an example, and is not meant to cover all comparative studies. The method applies as well to classifiers as to other data mining methods that attempt to extract patterns from a database. However, the binomial test is a relatively weak test that does not handle quantitative differences between algorithms, nor does it handle more than two algorithms, nor does it consider the frequency of agreement between two algorithms. If N is the number of agreements and $N >> n$, then it can be argued that our belief that the algorithms are doing the same thing should increase regardless of the pattern of disagreement. As pointed out by Feelders and Verkooijen [8], finding the proper statistical procedure to compare two or more classification algorithms can be quite difficult, and requires more than an introductory level knowledge of statistics. A good general reference for experimental design is Cochran and Cox [2], and descriptions of ANOVA and experimental design can be found in introductory texts such as Hildebrand [11].

Jensen [13, 14] discusses a framework for experimental comparison of classifiers and addresses significance testing, and Cohen and Jensen [3] discuss some specific ways to remove optimistic statistical bias from such experiments. One important innovation they discuss is *randomization testing*, in which one derives a reference distribution as follows. For each trial, the data set is copied and class labels are replaced with random class labels. Then an algorithm is used to find the most accurate classifier it can, using the same methodology that is used with the original data. Any estimate of accuracy greater than random for the copied data reflects the bias in the methodology, and this reference distribution can then be used to adjust the estimates on the real data.

3.2. Community experiments: Cautionary Notes

In fact, the problem in the machine learning community is worse than stated above, because many people are sharing a small repository of datasets and repeatedly using those same datasets for experiments. Thus there is a substantial danger that published results, even when using strict significance criteria and the appropriate significance tests, will be mere accidents of chance. The problem is as follows. Suppose that 100 different people are studying the effects of algorithms A and B, trying to determine which one is better. Suppose that in fact both have the same mean accuracy (on some very large population of datasets), although the algorithms vary randomly in their performance on specific datasets. Now, if 100 people are studying the effect of algorithms A and B, we would *expect* that five of them will get results that are statistically significant at the $p \leq 0.05$ level, and one will get significance at the 0.01 level! (Actually, the picture is a bit more complicated, since this assumes the 100 experiments are independent.) Clearly in this case these results are due to chance, but if the 100 people are working separately, the ones who get significant results will publish, while the others will simply move on to other experiments. Denton [4] made similar observations about how the reviewing and publication process can skew results. Although the data mining community is much broader than the classification community, it is likely that benchmark databases will emerge, and that different researchers will test their mining techniques on them. The experience of the machine learning community should serve as a cautionary tale.

In other communities (e.g., testing new drugs), experimenters try to deal with this phenomenon of "community experiments" by duplicating results. Proper duplication requires drawing a new random sample from the population and repeating the study. However, this is not what happens with benchmark databases, which normally are static. If someone wants to duplicate results, they can only re-run the algorithms with the same parameters on the same data, and of course the results will be the same. Using a different partitioning of the data into training and test sets does not help; duplication can only work if new data becomes available.

3.3. Repeated tuning

Another very substantial problem with reporting significance results based on computational experiments is something that is often left unstated: many researchers "tune" their algorithms repeatedly in order to make them perform optimally on at least some datasets. At the very least, when a system is being developed, the developer spends a great deal of time determining what parameters it should have and what the optimal values should be. For example, the back propagation algorithm has a learning rate and a momentum term that greatly affect learning, and the architecture of a neural net (which has many degrees of freedom) has an enormous effect on learning. Equally important for many problems is the representation of the data, which may vary from one study to the next even when the same basic dataset is used. For example, numeric values are sometimes converted to a discrete set of intervals, especially when using decision tree algorithms [7].

Whenever tuning takes place, every adjustment should really be considered a separate experiment. For example, if one attempts 10 different combinations of parameters, then significance levels (p-values) would have to be, e.g., 0.005 in order to obtain levels comparable to a single experiment using a level of 0.05. (This assumes, unrealistically, that the experiments are independent.) But few if any experimenters keep careful count of how many adjustments they consider. (Kibler and Langley [15] and Aha [1] suggest, as an alternative, that the parameter settings themselves be studied as independent variables, and that their effects be measured on artificial data. A greater problem occurs when one uses an algorithm that has been used before: that algorithm may already have been tuned on public databases. Therefore one cannot even know how much tuning took place, and any attempt to claim significant results on known data is simply not valid.

Fortunately, one can perform virtually unlimited tuning without corrupting the validity of the results. The solution is to use cross validation[3] entirely within the training set itself. The recommended procedure is to reserve a portion of the training set as a “tuning” set, and to repeatedly test the algorithm and adjust its parameters on the tuning set. When the parameters appear to be at their optimal settings, accuracy can finally be measured on the test data.

Thus, when doing comparative evaluations, *everything* that is done to modify or prepare the algorithms must be done in advance of seeing the test data. This point will seem obvious to many experimental researchers, but the fact is that papers are still appearing in which this methodology is not followed. In the survey by Flexer [9], only 3 out of 43 experimental papers in leading neural network journals used a separate data set for parameter tuning; the remaining 40 papers either did not explain how they adjusted parameters or else did their adjustments after using the test set. Thus this point is worth stating explicitly: any tweaking or modifying of code, any decisions about experimental design, and any other parameters that may affect performance must be fixed before the experimenter has the test data. Otherwise the conclusions might not even be valid for the data being used, much less any larger population of problems.

3.4. Generalizing results

A common methodological approach in recent comparative studies is to pick several datasets from the UCI repository (or some other public source of data) and perform a series of experiments, measuring classification accuracy, learning rates, and perhaps other criteria. Whether or not the choice of datasets is random, the use of such experiments to make more general statements about classification algorithms is not necessarily valid.

In fact, if one draws samples from a population to conduct an experiment, any inferences one makes can only be applied to the original population, which in this case means the UCI repository itself. It is not valid to make general statements about other datasets. The only way this might be valid would be if the UCI repository were known to represent a larger population of classification problems. In fact, though, as argued persuasively by Holte [12] and others, the UCI repository is a very limited sample of problems, many of which are quite easy for a classifier. (Many of them may represent the same concept class, for example many might be almost linearly separable, as suggested by the strong performance of the perceptron algorithm in one well-known comparative study [22].) Thus the evidence is strong that results on the UCI datasets do not apply to all classification problems, and the repository is not an unbiased “sample” of classification problems.

This is not by any means to say that the UCI repository should not exist. The repository serves several important functions. Having a public repository keeps the community honest, in the sense that any published results can be checked. A second function is that any new idea can be “vetted” against the repository as a way of checking it for plausibility. If it works well, it might be worth further investigation. If not, of course it may still be a good idea, but the creator will have to think up other means to demonstrate its feasibility. The dangers of repeated re-use of the same data should spur the research community to continually enlarge the repository, so that over time it will become more representative of classification problems in general. In addition, the use of artificial data should always be considered as a way to test more precisely the strengths and weaknesses of a new algorithm. The data mining community must likewise be vigilant not to come to rely too heavily on any standard set of databases.

There is a second, related problem with making broad statements based on results from a common repository of data. The problem is that someone can write an algorithm that works well on some of these datasets specifically; i.e., the algorithm is designed with the datasets in mind. If researchers become familiar with the datasets, they are likely to be biased when developing new algorithms, even if they do not consciously attempt to tailor their algorithms to the data. In other words, people will be inclined to consider problems such as missing data or outliers because they know these are problems represented in the public datasets. However, these problems may not be important for different datasets. It is an unavoidable fact that anyone familiar with the data may be biased.

3.5. A recommended approach

Although it is probably impossible to describe a methodology that will serve as a "recipe" for all comparisons of computational methods, here is an approach that will allow the designer of a new algorithm to establish the new algorithm's comparative merits.

1. Choose other algorithms to include in the comparison. Make sure to include the algorithm that is most similar to the new algorithm.
2. Choose a benchmark data set that illustrates the strengths of the new algorithm. For example, if the algorithm is supposed to handle large attribute spaces, choose a data set with a large number of attributes.
3. Divide the data set into k subsets for cross validation. A typical experiment uses $k = 10$, though other values may be chosen depending on the data set size. For a small data set, it may be better to choose larger k, since this leaves more examples in the training set.
4. Run a cross-validation as follows.

   (A) For each of the k subsets of the data set D, create a training set $T = D - k$.
   (B) Divide each training set into two smaller subsets, $T1$ and $T2$. $T1$ will be used for training, and $T2$ for *tuning*. The purpose of $T2$ is to allow the experimenter to adjust all the parameters of his algorithm. This methodology also forces the experimenter to be more explicit about what those parameters are.
   (C) Once the parameters are optimized, re-run training on the larger set T.
   (D) Finally, measure accuracy on k.
   (E) Overall accuracy is averaged across all k partitions. These k values also give an estimate of the variance of the algorithms.

5. To compare algorithms, use the binomial test described in section 3.1, or the McNemar variant on that test.

(It may be tempting, because it is easy to do with powerful computers, to run many cross validations on the same data set, and report each cross validation as a single trial. However, this would not produce valid statistics, because the trials in such a design are highly interdependent.) The above procedure applies to a single data set. If an experimenter wishes to compare multiple data sets, then the significance levels used should be increased using the Bonferroni adjustment. This is a conservative adjustment that will tend to miss significance in some cases, but it enforces a high standard for reporting that one algorithm is better than another.

4. Conclusion

As some researchers have recently observed, no single classification algorithm is the best for all problems. The same observation can be made for data mining: no single technique is likely to work best on all databases. Recent theoretical work has shown that, with certain assumptions, *no* classifier is always better than another one [24]. However, experimental science is concerned with data that occurs in the real world, and it is not clear that these theoretical limitations are relevant. Comparative studies typically include at least one new algorithm and several known methods; these studies must be very careful about their methods and their claims. The point of this paper is not to discourage empirical comparisons; clearly, comparisons and benchmarks are an important, central component of experimental computer science, and can be very powerful if done correctly. Rather, this paper's goal is to help experimental researchers steer clear of problems in designing a comparative study.

Acknowledgments

Thanks to Simon Kasif and Eric Brill for helpful comments and discussions. Thanks to Alan Salzberg for help with the statistics and for other comments. This work was supported in part by the National Science Foundation under Grants IRI-9223591 and IRI-9530462, and by the National Institutes of Health under Grant K01-HG00022-01. The views expressed here are the author's own, and do not necessarily represent the views of those acknowledged above, the National Science Foundation, or the National Institutes of Health.

Notes

1. This study was never published; a similar study is [16], which used 22 data sets and 4 algorithms.
2. A p-value is simply the probability that a result occurred by chance. Thus a p-value of 0.05 indicates that there was a 5% probability that the observed results were merely random variation of some kind, and do not indicate a true difference between the treatments. For a description of how to perform a paired t-test, see reference [11] or another introductory statistics text.
3. Cross validation refers to a widely used experimental testing procedure. The idea is to break a data set up into k disjoint subsets of approximately equal size. Then one performs k experiments, where for each experiment the k^{th} subset is removed, the system is trained on the remaining data, and then the trained system is tested on the held-out subset. At the end of this k-fold cross validation, every example has been used in a test set exactly once. This procedure has the advantage that all the test sets are independent. However, the training sets are clearly not independent, since they overlap each other substantially. The limiting case is to set $k = n - 1$, where n is the size of the entire data set. This form of cross validation is sometimes called "leave one out."

References

1. D. Aha. Generalizing from case studies: A case study. In *Proc. Ninth Intl. Workshop on Machine Learning*, pages 1–10, San Mateo, CA, 1992. Morgan Kaufmann.
2. W. Cochran and G. Cox. *Experimental Designs*. Wiley, 2nd edition, 1957.
3. P. R. Cohen and D. Jensen. Overfitting explained. In *Prelim. Papers Sixth Intl. Workshop on Artificial Intelligence and Statistics*, pages 115–122, January 1997.
4. F. Denton. Data mining as an industry. *Review of Economics and Statistics*, 67:124–127, 1985.
5. T. Dietterich. Statistical tests for comparing supervised learning algorithms. Technical report, Oregon State University, Corvallis, OR, 1996.
6. B. Everitt. *The Analysis of Contingency Tables*. Chapman and Hall, London., 1977.
7. U. M. Fayyad and K. B. Irani. Multi-interval discretization of continuous valued attributes for classification learning. In *Proc. 13th Intl. Joint Conf. on Artificial Intelligence*, pages 1022–1027, Chambery, France, 1993. Morgan Kaufmann.
8. A. Feelders and W. Verkooijen. Which method learns most from the data? In *Prelim. Papers Fifth Intl. Workshop on Artificial Intelligence and Statistics*, pages 219–225, Fort Lauderdale, Florida, 1995.
9. A. Flexer. Statistical evaluation of neural network experiments: Minimum requirements and current practice. In R. Trappl, editor, *Cybernetics and Systems '96: Proc. 13th European Meeting on Cybernetics and Systems Res.*, pages 1005–1008. Austrian Society for Cybernetic Studies, 1996.
10. O. Gascuel and G. Caraux. Statistical significance in inductive learning. In *Proc. of the European Conf. on Artificial Intelligence (ECAI)*, pages 435–439, New York, 1992. Wiley.
11. D. Hildebrand. *Statistical Thinking for Behavioral Scientists*. Duxbury Press, Boston, MA, 1986.
12. R. Holte. Very simple classification rules perform well on most commonly used datasets. *Machine Learning*, 11(1):63–90, 1993.
13. D. Jensen. Knowledge discovery through induction with randomization testing. In G. Piatetsky-Shapiro, editor, *Proc. 1991 Knowledge Discovery in Databases Workshop*, pages 148–159, Menlo Park, CA, 1991. AAAI Press.
14. D. Jensen. Labeling space: A tool for thinking about significance testing in knowledge discovery. Office of Technology Assessment, U.S. Congress, 1995.
15. D. Kibler and P. Langley. Machine learning as an experimental science. In *Proc. of 1988 Euro. Working Session on Learning*, pages 81–92, 1988.
16. R. Kohavi and D. Sommerfield. Oblivious decision trees, graphs, and top-down pruning. In *Proc. 14th Intl. Joint Conf. on Artificial Intelligence*, pages 1071–1077, Montreal, 1995. Morgan Kaufmann.
17. P. M. Murphy. UCI repository of machine learning databases – a machine-readable data repository. Maintained at the Department of Information and Computer Science, University of California, Irvine. Anonymous FTP from ics.uci.edu in the directory pub/machine-learning-databases, 1995.
18. L. Prechelt. A quantitative study of experimental evaluations of neural network algorithms: Current research practice. *Neural Networks*, 9, 1996.
19. N. Qian and T. Sejnowski. Predicting the secondary structure of globular proteins using neural network models. *Journal of Molecular Biology*, 202:65–884, 1988.
20. A. Raftery. Bayesian model selection in social research (with discussion by Andrew Gelman, Donald B. Rubin, and Robert M. Hauser). In Peter Marsden, editor, *Sociological Methodology 1995*, pages 111–196. Blackwells, Oxford, UK, 1995.
21. T. Sejnowski and C. Rosenberg. Parallel networks that learn to pronounce English text. *Complex Systems*, 1:145–168, 1987.
22. J. Shavlik, R. Mooney, and G. Towell. Symbolic and neural learning algorithms: An experimental comparison. *Machine Learning*, 6:111–143, 1991.
23. D. Wettschereck and T. Dietterich. An experimental comparison of the nearest-neighbor and nearest-hyperrectangle algorithms. *Machine Learning*, 19(1):5–28, 1995.
24. D. Wolpert. On the connection between in-sample testing and generalization error. *Complex Systems*, 6:47–94, 1992.