COD: Measuring the similarity of classifiers

by

Adam H. Peterson

A thesis submitted to the faculty of

Brigham Young University

in partial fulfillment of the requirements for the degree of

Master of Science

Department of Department of Computer Science

Brigham Young University

November 2004

BRIGHAM YOUNG UNIVERSITY

GRADUATE COMMITTEE APPROVAL

of a thesis submitted by

Adam H. Peterson

This thesis has been read by each member of the following graduate committee and by majority vote has been found to be satisfactory.

Date	Tony Martinez, Chair
Date	Michael Goodrich
Date	Dennis Ng

BRIGHAM YOUNG UNIVERSITY

As chair of the candidate's graduate committee, I have read the thesis of Adam H. Peterson in its final form and have found that (1) its format, citations, and bibliographical style are consistent and acceptable and fulfill university and department style requirements; (2) its illustrative materials including figures, tables, and charts are in place; and (3) the final manuscript is satisfactory to the graduate committee and is ready for submission to the university library.

Date

Tony Martinez
Chair, Graduate Committee

Accepted for the Department

David W. Embley
Graduate Coordinator

Accepted for the College

G. Rex Bryce,
Associate Dean, College of College of Physical and Mathematical Sciences

ABSTRACT

COD: Measuring the similarity of classifiers

Adam H. Peterson

Department of Department of Computer Science

Master of Science

In the practice of machine learning, one must select a learning algorithm to employ for a problem. Common questions are: Are some algorithms basically the same, or are they fundamentally different? How different? Does an algorithm that solves a particular problem well exist? What algorithms should be tried for a particular problem? Could performance be improved by combining more than one solution? This research presents the COD (*Classifier Output Difference*) distance metric for finding similarity between classifiers and classifier families. This metric is a tool which begins to address such questions, in both theoretical and practical aspects. It goes beyond simple accuracy comparisons and provides insights about fundamental differences between learning algorithms and the effectiveness of algorithm variations, fills a niche in meta-learning, may be used to improve the effectiveness of the construction of ensemble classifiers, and can give guidance in research towards hybrid systems. This paper describes how COD works and provides examples showing its utility in providing research insights. Results from this research show that there are clearly

measurable differences in the behaviors of hypotheses produced by different learning algorithms, as well as clearly measurable differences between learning paradigms.

ACKNOWLEDGMENTS

I would like to acknowledge the support and work of my graduate advisor, Dr. Tony Martinez, in always making sure this research would be as polished as it could be. Additionally, I am indebted to the input and contributions of Dr. Michael Goodrich.

I am also grateful for the encouragement of my parents who always support me in my life choices and urge me to strive for success and service. My siblings (Ryan, Tyler, Ann, Garret, Faith, and Juliana) have also supported and encouraged me in this work.

Contents

List of Tables

List of Figures

Chapter 1

Measuring classifier similarity

1.1 Background

Machine learning is the field of study concerned with understanding and creating computational processes that learn. (A process can be said to learn a task if its measured performance on a task improves with experience or data.) Ideally, it would be desirable to fashion computer programs which could solve any task set to them. Unfortunately, no algorithm will ever be able to provide that guarantee.

The *No Free Lunch Theorem* (see, e.g. [DHS01], pp. 454-458 for a classical discussion, or [WM95] for research applying the *No Free Lunch Theorem* to search techniques) is well known in Machine Learning and indicates that if no assumptions can be made about any particular learning task (other than examples given from the task), every machine learning solution is as likely to generalize well on that task as any other. This theorem does not mean that machine learning is impossible. Indeed, it is not a difficult stretch to apply the logic of the No Free Lunch Theorem to any learning process, machine or otherwise. Yet, it is indisputable (except perhaps on a philosophical level) that humans learn. Because all learning tasks are not equally probable, the premise of the theorem where nothing can be assumed about a

learning task is rarely if ever met. In practice, the No Free Lunch Theorem means that no learning algorithm can be proven formally to outperform all other known or hypothetical learning algorithms or processes on all problems.

The problems that the field of machine learning centers on are, then, not all possible problems. Rather they are the set of problems likely to be of import to ML practitioners (or their clients). Learning tasks from real experience, learning tasks that occur as components in heuristic systems, and other problems that are actually encountered in nature or technology are what machine learning tries to solve. We will call this set of machine learning tasks P_I, the "interesting problems." In contrast to the set of all problems (which include a vast number of random problems with no regularity), these problems exhibit some regularity. Unfortunately, P_I is difficult to characterize formally.

Several machine learning algorithms and algorithm families have been specified and developed by the machine learning research community (including neural networks, decision trees, instance based learners, rule based systems, induction systems, etc.). Each machine learning algorithm makes some assumptions about the problems it is applied to, and performs most effectively on problems where those assumptions are more or less satisfied. These assumptions bias the learner towards one solution or another, and this bias is a necessary part of learning [Mit80]. A fuzzy concept of "coverage" for each learning algorithm can be conceived of as the set of problems on which the learning algorithm performs well. The *No Free Lunch Theorem* guarantees that a learning algorithm or small set of learning algorithms will not be able to "cover" the entire space of possible learning tasks. On the other hand, it is only coverage of P_I that is desired.

It is an open question whether current state of the art in machine learning has good coverage over the set of interesting problems. For many tasks, good accuracy may be

achieved with one or more of the learning algorithms or algorithm families commonly used. With others, even the best current learning algorithms do not perform as well. Although this research does not answer all these questions, it proposes *COD* (the *Classifier Output Difference* distance metric), a tool that (among other things) may be used to discover where P_I has dense coverage and where coverage is not as thorough. COD provides the machine learning researcher and practitioner a straightforward way to measure the distance between learning algorithms in relation to a collection of learning tasks. By finding the distance between machine learners and learning families, it can be determined where a number of different learning algorithms behave similarly to each other and thus overlap regions of P_I. Conversely, finding learning algorithms that have a high distance to all other learning algorithms indicate that the portion of P_I covered by that learning algorithm is thinner and there may be higher potential for improving coverage of P_I by finding new learning algorithms similar to the "orphaned algorithms."

Although there are a variety of types of machine learning tasks (agent decision problems, regression, etc.), this research will focus on classification problems. Ultimately, this research shows that the learning algorithms studied here do exhibit differences between each other in terms of measurable output behaviors.

1.2 Concepts

A *learning task* is a problem from P_I. In other words, it is a problem that someone is likely to want solved at some point. A *learning algorithm* (or simply *algorithm*) is a procedure that, when given a learning task, produces a *hypothesis*, or a model that attempts to solve the learning task. In this research, *hypothesis* and *classifier* will sometimes be used interchangeably, since it is constrained to classification problems. To *solve* a learning task, a hypothesis is constructed that generalizes on unseen examples with good accuracy. Often the generalization accuracy of the model on the

task is being maximized directly or indirectly and solutions to a learning task may vary in quality, but at a minimum, a hypothesis must generalize better than base line accuracy (the frequency of the most frequent class) in order to be considered a "solution" to the task.

A number of approaches could be taken to compare learning algorithms. One approach would be to compare the types of decision surfaces employed by one learning algorithm as opposed to another. An alternate approach would be to evaluate the capacity for one learning algorithm to simulate another. (Naturally this would be somewhat problematic when dealing with learning algorithms capable of shattering, i.e. arbitrarily dichotomizing, any set of consistent patterns and therefore capable of simulating any other hypothesis.) COD takes a different approach by first providing a mechanism to measure the distance between two hypotheses. COD postulates a distance between two hypotheses (for the same task) that is proportionate to the frequency with which they disagree on the classification of patterns. This metric goes beyond simply measuring the relative accuracies of the two learning algorithms by looking at the pairwise behavior on each instance from the task. A distance between two hypotheses over a particular data set can be estimated by observing the frequency that the hypotheses would disagree with each other on the classification of the patterns from the given set. A distance between two learning algorithms is then the expected distance between the hypotheses produced by the algorithm for a task. A distance between two learning algorithms over a collection of learning tasks may be estimated by taking the average of the distance between the hypotheses they produce over each individual task.

1.3 Applicability

A distance metric such as COD which gives an estimation of how often two learning algorithms will behave differently can be used in a number of ways:

- Knowing which learning algorithms behave similarly can guide a practitioner in evaluating algorithms for a desired learning task. Knowing, for instance, that algorithm A performs well and algorithm B performs poorly, a practitioner may use measures of similarity to try algorithm C (perhaps because it is similar to algorithm A) next and discount algorithm D (which may be similar to algorithm B).

- Quantifiable measurements of similarity between learning algorithms or hypotheses may be applied in *meta-learning*, where machine learners are used to guide the selection of other machine learners.

- Knowing which learning algorithms are likely to behave similarly to each other can guide a practitioner in constructing an ensemble classifier such that the greatest diversity can be obtained through the fewest number of models. Additionally, finding the distance between hypotheses can make it easier to insure that a disproportionate number of similar hypotheses is not included in an ensemble (which would reduce the effectiveness of the underrepresented hypotheses).

- Similarly, research into hybrid learning algorithms can be guided by knowledge of which algorithms consistently exhibit different behaviors. Hybrid approaches are more likely to give improved performance if based on learning algorithms with different strengths. Attempts to build a hybrid learning algorithm from powerful algorithms which achieve high accuracy while in actuality giving largely the same behaviors is likely to be less productive than attempts to build a hybrid learning algorithm from algorithms which genuinely exhibit diverse behaviors on learning tasks.

- The ability to determine the similarity of hypotheses can lead to a better understanding of which variables result in genuine variance in the hypotheses and to what degree. A concept of algorithm *dispersion* may be measured by producing several hypotheses from the same algorithm on the same task while varying parameters to the task (or other factors such as sampling). Determining which factors affect the dispersion of a learning algorithm more can allow a researcher to properly discount variables which in practice do not affect behavior, allowing more attention to be paid to variables which in fact do. For example, if it could be determined that during neural network training, presentation order causes significantly more variance than initial weight settings, a researcher may then properly focus repeated runs of experiments more on alternate presentation orders and less on initial weight settings to obtain more effective statistics.

Much of the published research in the field of machine learning compares learning algorithms with each other. However, these comparisons are mostly concentrated on accuracy. Comparing the *performance* of learning algorithms is a natural step in publishing incremental improvement research. COD delves deeper into the *behavioral* differences between learning algorithms, even between algorithms which perform comparably (performance-wise) on a variety of problems. Two learning algorithms which consistently achieve comparable performance results on some learning tasks may in fact behave quite differently, if they tend to learn different aspects of the problem in different ways.

Chapter 2

Related Work

The quantification of distances between learning algorithms (as COD proposes) has tie-ins with a branch of machine learning called *meta-learning*, where machine learning strategies are applied at more than one level in the problem solving process. This idea is used in *landmarking*, put forward by [PBGC00]. Landmarking works to locate learning tasks in the space of all learning tasks. Problems are conceived of as existing in a space of learning tasks and their relative positions are estimated by evaluating the success of simple learning algorithms on the task. Landmarking is in some respects the inverse approach of COD. Whereas in landmarking, learning tasks are measured relative to some simple learning algorithms, COD measures distances between learning algorithms and hypotheses by using their performance on learning tasks.

A more traditional approach to learning task similarity is taken in [Zhe93] by positing several simple metrics that may be used to categorize learning tasks (e.g. number of inputs and whether all inputs are real-valued). Zheng's approach does not apply meta-learning ideas, and also tries to solve the inverse problem to COD.

Brodley approaches the problem of different learning algorithms covering some

tasks better than others by advocating a hybrid approach. "Concepts or subconcepts that are difficult to represent well in one formalism may be easy to represent in another. A system that can combine formalisms can represent succinctly and accurately a larger class of concepts." [Bro92] While this research does not explore the possible applications of COD to hybrid approaches, this is an area of future work. (See section 5.)

Generally speaking, the research literature has numerous instances where two learning algorithms are being compared. For example, [OM99] compares several ensemble methods by giving the error rates of neural network, decision tree, and ensemble methods and graphing the reduction in error for several ensemble types. In another example [SJW93] compares several methods for training neural networks, giving error values in the "best results" table with almost no explanation, assuming that the target audience is accustomed to reading error measurements in results sections. In the results table of [AE99] (a comparison between active learning neural network algorithms), the sum-squared error of several algorithms is given for comparison. Also, in [CBM96], the Rankprop algorithm is motivated in part by showing improvements in sum-squared error over zero/one targets. This is a common step in the demonstration of merit for a new learning algorithm, especially a variation on a previously known learning algorithm. (This is observed in [Aha92], which puts forward recommendations about how to use case studies to obtain useful empirical results.) However, these comparisons seldom if ever go beyond the measured error or accuracy of the algorithms in question. If, e.g. algorithm 1 performs on average around 75% accuracy while algorithm 2 performs around 80%, the two learning algorithms may exhibit different behaviors anywhere from 5% to 45% of the time.

Chapter 3

COD: The distance between classifiers

The objective of COD is to measure an estimate of the frequency that two hypotheses (or, by extension, two learning algorithms) will exhibit different behaviors on a task (or set of tasks). For the distance between two individual hypotheses, ideally it would be desirable to integrate the output difference between the two hypotheses across all possible inputs, weighted by the input probability. That is, the ideal COD distance metric over a learning task would be computed something like this:

$$D(H_1, H_2) = \int_{\boldsymbol{x} \in X} |H_1(\boldsymbol{x}) - H_2(\boldsymbol{x})| \cdot \mathrm{P}(\boldsymbol{x}) d\boldsymbol{x} \tag{3.1}$$

where $\boldsymbol{x}$ is each possible input from the input space X, the quantity $|H_1(\boldsymbol{x}) - H_2(\boldsymbol{x})|$ is 1 when hypothesis 1 outputs a different value from hypothesis 2 and 0 otherwise, and $\mathrm{P}(\boldsymbol{x})$ is the probability of input $\boldsymbol{x}$ occurring for the desired task.

For example, take the artificial classification problem given in figure 3.1. The decision regions for two hypotheses trained on this problem are given in figure 3.2. The region of the input space over which the hypotheses disagree is shown in figure 3.3. The proportion of the input space covered by this region then forms the basis for

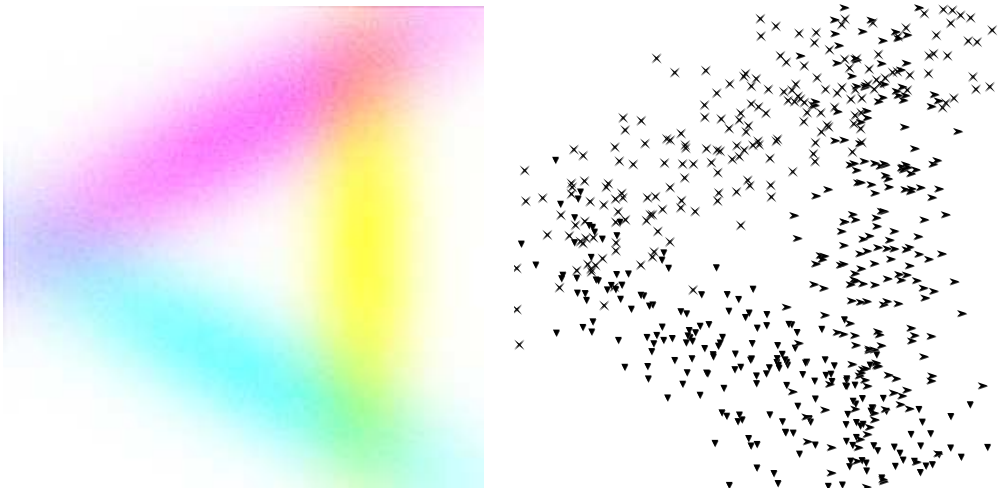

Figure 3.1: (Left:) An artificial example of a classification learning task with two real-valued input features and three output classes (each normally distributed in the feature space). (Right:) A sampling of the three classes plotted with three different monikers (for grayscale printouts).

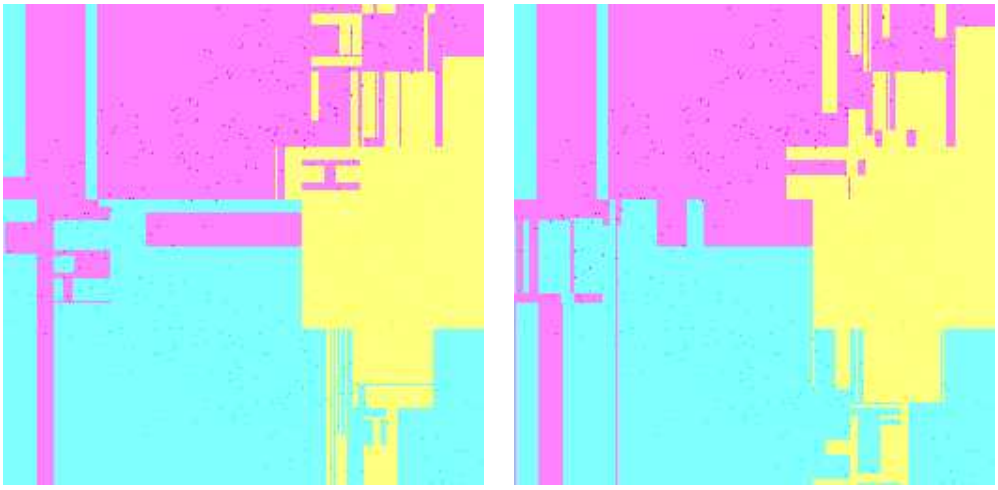

Figure 3.2: The decision surfaces for two different hypotheses (generated by two different decision tree learning algorithms) on the problem given in figure 3.1.

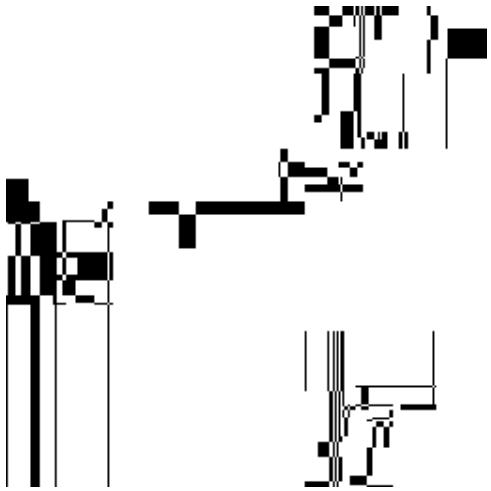

Figure 3.3: The region where the two hypotheses from figure 3.2 disagree.

the COD metric, after weighting the covered region by the probability of each input occurring, as shown in figure 3.4.

Actually performing the integration given above is impractical (and in most cases impossible) for at least two reasons:

- In most cases, the input space X is of large dimension and/or contains a large number of possible inputs for each feature. Tasks where this is not the case tend to be academic in nature. Although they are properly part of P_I, they form a relatively small portion even when taken together.

- For a large number of learning tasks (if not most), the likelihood of an input vector for the task ($P(\boldsymbol{x})$) is not precisely known. Although this can be estimated in some cases, such estimations rely on assumptions which may apply imperfectly or not at all. Inaccuracies in the estimation can easily compound over a large integration to result in an inaccurate distance metric.

COD presents a method for estimating this distance measure between hypotheses in a practical manner, based on differences in their output behaviors. Two hypothe-

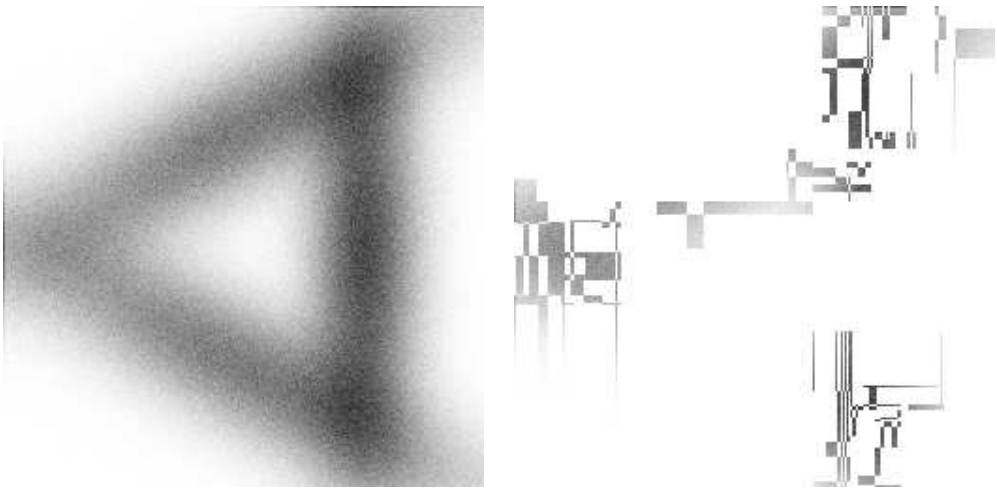

Figure 3.4: (Left:) The probability of each input occurring in the input space (darker values are more probable). (Right:) The region from figure 3.3 weighted by the probability of the input occurring. The proportion of the overall coverage of the right image against the coverage on the left is the COD distance metric.

ses may be compared using COD by evaluating each hypothesis on patterns that neither has seen during training and comparing their outputs. The number of patterns disagreed upon is counted and divided by the number of patterns used in the comparison, giving the COD metric value. Taking typical training set/testing set methodology (where a data set gathered from a task is divided into two disjoint sets, one used for training hypotheses and one used for evaluating their accuracy), the test set may be used to instead find the distance between two hypotheses trained on the same training set (or any training set disjoint from the testing set). Assuming the test set is representative of the learning task, the COD metric can be estimated as:

$$\hat{D}_T(H_1, H_2) = \frac{\sum_{\boldsymbol{x} \in T} |H_1(\boldsymbol{x}) - H_2(\boldsymbol{x})|}{|T|} \tag{3.2}$$

where T is the test set and $|H_1(\boldsymbol{x}) - H_2(\boldsymbol{x})|$ is 1 where hypothesis 1 disagrees with hypothesis 2 on the classification of pattern $\boldsymbol{x}$ (and 0 otherwise).

Measuring distance between two hypotheses on a problem in this way includes the following beneficial effects:

- The measurements are always taken of the behavior of the hypothesis on data that was not used in the training process, so the measurement reflects the behavior of the hypothesis during generalization rather than its idiosyncrasies in handling training data. By extension, the measurements will tend to reflect the generalizing behavior of hypotheses produced by the learning algorithm when several such measurements are aggregated.

- The data are a sampling from the distribution likely to occur during generalization, so the distance metric tends to reflect the difference in behavior likely to be observed during actual use. One might be tempted to instead train a hypothesis on all of the given data and then measure the distance by sampling manufactured patterns across the input domain of the problem and observing how often the two hypotheses disagree on the output. Such a measure would be less relevant since it would represent an integration across the input domain distributed substantially differently than is likely to occur during generalization.

- The distance is nonzero only when the hypotheses have different *observable* behaviors at the output. If two hypotheses give the same answer most of the time, even though they perform very different computations internally to arrive at their result, the internal differences are largely irrelevant for most purposes employing a hypothesis or algorithm distance (such as determining coverage over a set of learning tasks or measuring diversity for an ensemble).

Instead of using the basic training set/testing set method as a base, n-fold cross-validation may also be used [Die98]. In the standard application of cross-validation,

multiple hypotheses are trained and tested on rotating subsets of the data set such that each portion of the data set is tested against once (using a classifier not trained on that portion). The COD metric can be similarly measured between two learning algorithms by obtaining a pair of hypotheses, one from each algorithm, over each partitioning of the data set (as would be employed in cross-validation) and finding the pairwise distance between the corresponding hypotheses. The individual distances so measured can then be averaged as in cross-validation to obtain an estimation of the general distance between two learning algorithms.

In contrast to accuracy measurement methods (such as training set/testing set and cross-validation) which only operate against one hypothesis or learning algorithm, COD operates against a pair of such at a time (to find the distance between them). Where accuracy measurements gathered in such a way during research are often averaged to give an indication of the accuracy of the learning algorithms over a set of learning tasks, COD metrics are averaged to give a *matrix* describing the general distances between learning algorithms over a set of tasks (with an entry corresponding to each pair of algorithms and giving the average distance between them). Such a matrix can then be used in analysis to determine the type of coverage a set of learning algorithms has over the set of tasks.

The COD metric satisfies the triangle inequality between more than two hypotheses. It also satisfies the triangle inequality between two hypotheses and the "perfect classifier" (that is, the hypothetical classifier that gets every example correct). The distance between two hypotheses and their distance to the "perfect classifier" (that is, each hypothesis's error rate) can be used to form an abstract triangle, as visualized in figure 3.5. By applying the Law of Cosines, a concept of angle between two hypotheses (or learning algorithms, when used in aggregate) can be defined.

The angle value from this triangle may be used as an indication of useful diver-

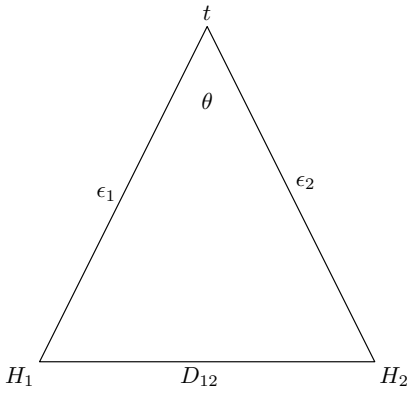

Figure 3.5: A visual representation of the conceptual triangle that can be formed by the COD metric between two hypotheses and the "perfect classifier." The value ϵ_1 is the error of H_1 on task t (and likewise for ϵ_2). D_{12} is the COD distance between H_1 and H_2. The sides of this triangle (ϵ_1, ϵ_2, and D_{12}) may be used to find angle θ by applying the formula $\theta = \arccos \frac{\epsilon_1^2+\epsilon_2^2-D_{12}^2}{2\epsilon_1\epsilon_2}$.

sity between two hypotheses. When learning algorithms produce hypotheses that perform comparably on a problem while behaving differently (as pictured in figure 3.6), potential for improvement exists by combining the strengths of the two learning algorithms. The different behaviors of two such hypotheses are the result of either the hypotheses misclassifying different examples or misclassifying the same examples while giving different wrong answers. On the other hand, if there is not much behavior difference between two hypotheses relative to their error rates (giving a smaller angle, as in figure 3.7), the potential for improvement by combining the hypotheses (or algorithms used to produce them) is less, as improvement tends to center around where the hypotheses do not already agree. Similarly, the potential for improvement is less between two hypotheses whose angle is small, even if their distance is large (as in figure 3.8). In such a case, the distance between them is largely the result of one hypothesis's superiority over the other and the second hypothesis's added value

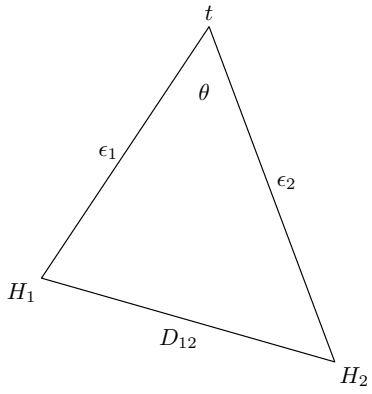

Figure 3.6: A COD triangle for two hypotheses that are relatively diverse, giving a fairly large value for θ.

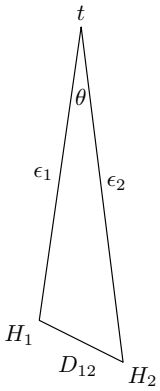

Figure 3.7: A COD triangle for two similar hypotheses, giving a relatively small value for θ.

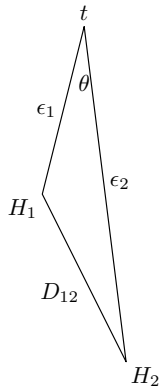

Figure 3.8: A COD triangle for two hypotheses where one hypothesis significantly dominates the other. When the difference between two hypotheses is mostly the result of patterns the first hypothesis classifies correctly that the second misclassifies, the first tends to dominate the second.

through diversity is lessened.

This concept combines the measure of accuracy with the measure of classifier distance in a way that can be useful in evaluating hypotheses or learning algorithms for ensembles. The angle between two hypotheses will be large when the distance between them is large and when the errors of the hypotheses are small. In an ensemble, it may be more desirable to include marginally less accurate individual hypotheses if their diversity is greater, since higher diversity can give the ensemble a greater chance of overcoming the individual hypotheses' weaknesses. The angle measure so given can be thought of as a measurement on how skewed the triangle is. If the angle is small, then either one hypothesis generally dominates over another (a bulk of the patterns one hypothesis gets wrong, the other gets wrong in the same way), or the hypotheses get similar accuracies but do not differentiate themselves much over which patterns they get wrong (and where both miss the same pattern, they tend to give the same wrong answer). See section 4.4 for results related to this idea.

In the case of a classification task that is more than just a concept learning task (i.e. has more than two classifications), this measurement (and the associated angle measurement) can be particularly useful. COD will not only indicate how often two hypotheses or learning algorithms miss the same problems, it will also take into account how often they miss them in the same way, i.e. provide the same incorrect classification. If individual hypotheses in an ensemble tend to provide different incorrect answers, the chance is greater for the subset of hypotheses in the ensemble which classify a pattern correctly to "outvote" (depending on which flavor ensemble is being employed) the incorrect classifiers.

Chapter 4

Results and Analysis

4.1 Methods

In this research, the distances between several learning algorithms were measured over machine learning data sets from the Irvine Machine Learning Repository [MM96]. These particular tasks (listed in table 4.1 with a few basic statistics about each) were selected from the UCI Repository to obtain a sampling across the repository, but artificial data sets from the UCI Repository were explicitly excluded. Artificial data sets such as the two spirals problem are properly part of P_I, but they tend to be overrepresented in research and occur per se much less often in practice.

The learning algorithms among which the COD metric was measured are listed below. They were selected to represent several paradigms within the field of machine learning (decision trees, connectionist, Bayesian, and instance based), and also so as to include a few examples from each paradigm for comparison with each other and to other learning algorithms. Although these algorithms have not been tightly optimized and fine-tuned to fit the learning tasks tested in this paper, this is not critical to the thrust of this research. The utility of the metric to show *algorithm differences* is the focus, rather than the relative accuracies of the algorithms. In that spirit, several

Data set	Real Features	Nom. Features	Nom. Feature Values	Output classes
balanc	0	4	20	3
bupa	6	0	0	2
iono	34	0	0	2
iris	4	0	0	3
lymph	0	18	60	4
musk1	166	0	0	2
newthy	5	0	0	3
pima	8	0	0	2
segm	19	0	0	7
sonar	60	0	0	2
stvehi	18	0	0	4
vowel	10	0	0	11
wine	13	0	0	3
zoo	0	16	36	7

Table 4.1: The metrics for the Irvine Machine Learning Repository tasks used in this research.

reasonable learning algorithms from diverse learning paradigms were chosen.

DTg (Decision Tree with gain): A simple decision tree algorithm based on ID3 [Qui86] with *information gain* as a split metric and no pruning.

DTr (Decision Tree with gain ratio): A decision tree algorithm similar to *DTg* except *information gain ratio* is used instead of information gain (as recommended in [Qui93]).

DTe (Decision Tree with expectation ratio): Another decision tree algorithm similar to the first two with a different split metric. Quinlan notes that information gain tends to favor wide splits while gain ratio favors uneven splits. The expectation ratio modifies the gain ratio metric to favor more even splits. (To our knowledge this metric has not been used in the research literature. Nevertheless *DTe* performs

competitively with the other decision tree based solutions.)

MLP (Multilayer Perceptron): A feed-forward backpropagation neural network [RHW86] using the logistic activation function and sum-squared error as the objective function and a single hidden layer with $\lceil \log_2 i \cdot o \rceil$ hidden nodes, where i is the number of input features and o is the number of output classes.

SLP (Single Layer Perceptron): A single layer neural network [Ros58] using the logistic activation function.

NB (Naive Bayes): A simple naive Bayes algorithm as described in [Lan95] and [Mit97]. [Lan95] presents naive Bayes in the context of a variable number of (assumed to be) independent identically distributed enumerated features, and [Mit97] also discusses naive Bayes in an enumerated feature context. Neither describes how to handle real-valued inputs, so the three naive Bayes approaches used herein differ in the strategy for handling them. This particular variant gathers mean and variance statistics for each real-valued input feature and uses them to construct a Gaussian conditional prior probability for each class (which is then used with the other conditional priors to form a conditional probability of the pattern, as needed by Bayes).

NBi (Naive Bayes with binned inputs): This naive Bayes variation converts real valued features into enumerated features by binning such that each continuous input feature is broken up into bins, each containing $\sqrt{n_i}$ patterns (where n_i is the number of distinct input values for the feature). Each bin then becomes an enumeration value.

NBs (Naive Bayes with segmented distribution): Instead of binning con-

tinuous input values before presentation to the naive Bayes learner, they are binned by the learner on a per-class level (using the same binning technique from above). The bins are then used to estimate the distribution of each feature given the class, which is used to compute the conditional prior for the feature.

1NN (One Nearest Neighbor): A standard k-Nearest Neighbor approach [CH67] with $k = 1$. Real valued inputs are scaled to have a mean of zero and unit variance. (This scaling is used for all nearest neighbor approaches here.)

5NN (Five Nearest Neighbor): A standard k-NN algorithm with $k = 5$.

1NNp (One Nearest Neighbor, pruned): A standard k-NN approach with $k = 1$ where as each pattern is added during model construction, the pattern is first tested to see if it would be misclassified. If not, it is not added to the classifier.

5NNp (Five Nearest Neighbor, pruned): A k-NN approach with $k = 5$, with patterns pruned in a similar fashion to *1NNp*.

The distance between two learning algorithms was measured by employing the cross-validation inspired COD method. That is, each data set was divided into several disjoint subsets and each subset was withheld in turn while the remaining data were used for training a pair of hypotheses (one from each learning algorithm) that was then measured against the withheld data. These measurements were paired across all learning algorithms (i.e. the same partitioning was used for all learning algorithms), and the partitioning was *stratified* [ZM00] to reduce variance (i.e. the proportion of each class in each partition was made as close to equal as possible). The number of partitions used was 5.

4.2 Results: COD metrics between algorithms on several learning tasks

In the sections that follow, results from the following sets of experiments will be given and discussed:

- The COD distances between learning algorithms was measured by performing ten sets of paired experiments using the above discussed algorithms and averaging the results. The data set for each learning task was divided up five ways and the COD metric for each fifth of the data was measured using a hypothesis obtained from the remaining four fifths. The same five-way split of the data was used for all learning algorithms (giving paired results) and the split was stratified so that each class from the learning task was equally represented in each fifth (to the extent possible). This was repeated ten times, with ten different five-way splits, and averaged. The results of these experiments are given in tables showing the distances (which fall between 0 and 1) at three levels:
  - The individual learning task level (where results are reported for each combination of learning task and algorithm),
  - The learning algorithm level (where results are reported across all learning tasks for each algorithm), and
  - The learning algorithm paradigm level (with results from different algorithms in the same learning paradigm combined together).
- The COD dispersion measurement (indicating how much variation can occur from a single learning algorithm as a result of variations in its input) was approximated by comparing the distance between hypotheses trained on the same learning task with the same algorithm and different cuts of the data. The outputs of the hypotheses trained in the above ten sets of paired experiments were

used to find the distance between hypotheses given by the same algorithm on different samplings of each learning task. The different data cuts result in different input samplings and provide an indication as to how much variation is introduced into the learning algorithm through sampling effects.

- Results from a single learning task are examined in more detail as an example of how to use the COD metric to aid in the investigation of ensemble potential.

Tables A.1–A.14 (in the appendix) give the COD metrics measured between the learning algorithms listed above on each individual problem. To compose these tables, ten sets of paired experiments were run (with sampling varying between the different sets of paired experiments, as described above) and the resulting COD distance measures were averaged. Each entry in these tables indicate the proportion where the two algorithms disagreed on an output classification. For example, on the *balanc* learning task (table A.1 in the appendix, also reproduced as table 4.2 here), *DTg* disagreed with *DTr* 17% of the time over ten runs (each folded five ways). Similarly, on *bupa* (table A.2), *NB* disagreed with *MLP* 41% of the time. These tables are symmetrical because the COD distance relationship is symmetrical.

Table 4.3 combines the result of tables A.1–A.14 (in the appendix) to look at the average distance between algorithms over 14 learning tasks. For example, by looking at table 4.3 it can be found that on average over all fourteen learning tasks, the output of hypotheses trained with *MLP* disagreed with those given by *SLP* 11% of the time.

Table 4.3 is further condensed in table 4.4, with the learning algorithms from the same learning paradigm combined together. This table will be discussed further in the analysis in section 4.3.

COD	***DTg***	***DTr***	***DTe***	***MLP***	***SLP***	***NB***	***NBi***	***NBs***	***1NN***	***5NN***	***1NNp***	***5NNp***
DTg	0	0.17	0.00	0.35	0.32	0.31	0.31	0.31	0.40	0.32	0.44	0.39
DTr	0.17	0	0.17	0.37	0.33	0.33	0.33	0.33	0.39	0.33	0.45	0.41
DTe	0.00	0.17	0	0.35	0.32	0.31	0.31	0.31	0.40	0.32	0.44	0.39
MLP	0.35	0.37	0.35	0	0.07	0.08	0.08	0.08	0.48	0.25	0.43	0.36
SLP	0.32	0.33	0.32	0.07	0	0.05	0.05	0.05	0.45	0.22	0.39	0.33
NB	0.31	0.33	0.31	0.08	0.05	0	0.00	0.00	0.44	0.22	0.40	0.32
NBi	0.31	0.33	0.31	0.08	0.05	0.00	0	0.00	0.44	0.22	0.40	0.32
NBs	0.31	0.33	0.31	0.08	0.05	0.00	0.00	0	0.44	0.22	0.40	0.32
1NN	0.40	0.39	0.40	0.48	0.45	0.44	0.44	0.44	0	0.38	0.32	0.44
5NN	0.32	0.33	0.32	0.25	0.22	0.22	0.22	0.22	0.38	0	0.39	0.33
1NNp	0.44	0.45	0.44	0.43	0.39	0.40	0.40	0.40	0.32	0.39	0	0.41
5NNp	0.39	0.41	0.39	0.36	0.33	0.32	0.32	0.32	0.44	0.33	0.41	0

Table 4.2: The table of COD metrics for the *balanc* task. Average: 0.30.

COD	***DTg***	***DTr***	***DTe***	***MLP***	***SLP***	***NB***	***NBi***	***NBs***	***1NN***	***5NN***	***1NNp***	***5NNp***
DTg	0	0.16	0.02	0.19	0.22	0.25	0.23	0.35	0.22	0.21	0.25	0.26
DTr	0.16	0	0.16	0.20	0.22	0.25	0.23	0.35	0.22	0.22	0.25	0.26
DTe	0.02	0.16	0	0.19	0.21	0.25	0.22	0.35	0.21	0.21	0.25	0.25
MLP	0.19	0.20	0.19	0	0.11	0.19	0.18	0.32	0.20	0.17	0.23	0.23
SLP	0.22	0.22	0.21	0.11	0	0.20	0.19	0.33	0.23	0.20	0.26	0.25
NB	0.25	0.25	0.25	0.19	0.20	0	0.17	0.28	0.26	0.24	0.29	0.28
NBi	0.23	0.23	0.22	0.18	0.19	0.17	0	0.28	0.24	0.21	0.27	0.26
NBs	0.35	0.35	0.35	0.32	0.33	0.28	0.28	0	0.36	0.35	0.38	0.38
1NN	0.22	0.22	0.21	0.20	0.23	0.26	0.24	0.36	0	0.14	0.13	0.21
5NN	0.21	0.22	0.21	0.17	0.20	0.24	0.21	0.35	0.14	0	0.19	0.18
1NNp	0.25	0.25	0.25	0.23	0.26	0.29	0.27	0.38	0.13	0.19	0	0.23
5NNp	0.26	0.26	0.25	0.23	0.25	0.28	0.26	0.38	0.21	0.18	0.23	0

Table 4.3: The Average COD table averaged over all tasks. Average: 0.23.

COD	**Dec. Tree**	**Connect.**	**N. Bayes**	**Inst. Based**
Dec. Tree	0.11	0.21	0.28	0.23
Connect.	0.21	0.11	0.24	0.22
N. Bayes	0.28	0.24	0.24	0.29
Inst. Based	0.23	0.22	0.29	0.18
Average to other	0.24	0.23	0.27	0.25

Table 4.4: Paired distance table between learning algorithm paradigms. This table was obtained by averaging the elements from regions of table 4.3. The zero diagonal elements of table 4.3 were ignored to compose this table, and the diagonal elements in this table represent an average distance between learning algorithms of the same learning paradigm. The bottom row gives the average of the elements from the row, except the diagonal element (distance between learning algorithms from the same family).

Table 4.5 will be referred to from time to time in the analysis section. It lists the average error rate obtained by each algorithm on each task. For example, the *SLP* algorithm exhibited an error rate of 0.17 on the *lymph* task, and thus had an accuracy rate of 83%.

Table 4.6 compares the ten sets of experiments run to compute the algorithm *dispersion* over each data set (rather than as paired tests). Algorithm dispersion is the average COD distance between two hypotheses produced by the same learning algorithm. In order to obtain dispersion, some parameter which affects the hypothesis produced must be varied so that the same hypothesis is not returned each time. In this case, the different samplings between the multiple foldings is being used to produce hypothesis variation, as was described near the top of this section in the second bullet point describing the dispersion measurement experiments.

Error	*DTg*	*DTr*	*DTe*	*MLP*	*SLP*	*NB*	*NBi*	*NBs*	*1NN*	*5NN*	*1NNp*	*5NNp*	**Avg**
balanc	0.36	0.37	0.36	0.03	0.08	0.09	0.09	0.09	0.48	0.26	0.43	0.37	**0.25**
bupa	0.35	0.36	0.35	0.32	0.31	0.43	0.37	0.43	0.39	0.34	0.43	0.42	**0.37**
iono	0.11	0.11	0.11	0.14	0.13	0.15	0.14	0.64	0.13	0.16	0.16	0.17	**0.18**
iris	0.06	0.07	0.06	0.07	0.10	0.05	0.09	0.12	0.04	0.03	0.07	0.06	**0.07**
lymph	0.29	0.30	0.26	0.18	0.17	0.21	0.21	0.21	0.20	0.19	0.27	0.26	**0.23**
musk1	0.20	0.22	0.20	0.13	0.18	0.26	0.18	0.23	0.15	0.14	0.17	0.19	**0.19**
newthy	0.06	0.07	0.06	0.06	0.07	0.04	0.05	0.09	0.06	0.08	0.09	0.10	**0.07**
pima	0.29	0.30	0.29	0.24	0.23	0.25	0.24	0.30	0.32	0.28	0.38	0.34	**0.29**
segm	0.03	0.04	0.03	0.03	0.07	0.20	0.08	0.86	0.04	0.06	0.06	0.09	**0.13**
sonar	0.26	0.26	0.26	0.22	0.23	0.32	0.27	0.36	0.19	0.22	0.22	0.31	**0.26**
stvehi	0.27	0.29	0.27	0.18	0.22	0.54	0.39	0.47	0.35	0.35	0.39	0.38	**0.34**
vowel	0.20	0.21	0.20	0.16	0.46	0.35	0.42	0.36	0.01	0.09	0.06	0.18	**0.22**
wine	0.07	0.09	0.07	0.03	0.03	0.02	0.06	0.16	0.25	0.29	0.29	0.32	**0.14**
zoo	0.04	0.04	0.06	0.09	0.06	0.08	0.08	0.08	0.03	0.05	0.05	0.07	**0.06**
Avg	**0.19**	**0.20**	**0.19**	**0.13**	**0.17**	**0.21**	**0.19**	**0.31**	**0.19**	**0.18**	**0.22**	**0.23**	**0.20**

Table 4.5: Average error on each learning task

Disp.	*DTg*	*DTr*	*DTe*	*MLP*	*SLP*	*NB*	*NBi*	*NBs*	*1NN*	*5NN*	*1NNp*	*5NNp*	**Avg**
balanc	0.21	0.20	0.21	0.05	0.05	0.06	0.06	0.06	0.38	0.28	0.46	0.37	**0.20**
bupa	0.32	0.35	0.32	0.19	0.13	0.11	0.18	0.23	0.14	0.14	0.31	0.35	**0.23**
iono	0.10	0.11	0.10	0.07	0.07	0.02	0.07	0.00	0.04	0.03	0.13	0.16	**0.07**
iris	0.03	0.04	0.03	0.05	0.07	0.01	0.06	0.05	0.00	0.01	0.07	0.08	**0.04**
lymph	0.21	0.24	0.22	0.11	0.11	0.06	0.06	0.06	0.13	0.09	0.27	0.24	**0.15**
musk1	0.23	0.26	0.23	0.11	0.13	0.07	0.10	0.06	0.08	0.08	0.20	0.23	**0.15**
newthy	0.04	0.06	0.04	0.04	0.03	0.01	0.02	0.04	0.04	0.02	0.10	0.10	**0.04**
pima	0.26	0.25	0.26	0.10	0.06	0.03	0.07	0.09	0.12	0.10	0.29	0.30	**0.16**
segm	0.04	0.05	0.04	0.03	0.02	0.02	0.04	0.00	0.02	0.03	0.07	0.11	**0.04**
sonar	0.28	0.28	0.28	0.14	0.15	0.07	0.13	0.16	0.07	0.09	0.19	0.28	**0.18**
stvehi	0.26	0.29	0.26	0.15	0.10	0.12	0.14	0.18	0.12	0.15	0.28	0.33	**0.20**
vowel	0.25	0.26	0.25	0.16	0.36	0.16	0.41	0.24	0.01	0.08	0.08	0.23	**0.21**
wine	0.08	0.11	0.08	0.03	0.02	0.01	0.06	0.07	0.08	0.11	0.21	0.27	**0.09**
zoo	0.02	0.03	0.03	0.06	0.06	0.02	0.02	0.02	0.00	0.02	0.05	0.09	**0.04**
Avg	**0.17**	**0.18**	**0.17**	**0.09**	**0.10**	**0.05**	**0.10**	**0.09**	**0.09**	**0.09**	**0.19**	**0.22**	**0.13**

Table 4.6: Average distance between hypotheses trained using the same training algorithm and a different sampling.

The COD measure between two hypotheses is bounded by the error rate of the two hypotheses. Two hypotheses can not disagree on patterns on which they both give the (same) correct classification, and thus if one hypothesis has an error of ϵ_1 and another an error of ϵ_2, the COD measure can be no greater than $\epsilon_1 + \epsilon_2$. (By similar logic, the COD metric of two such hypotheses can not be less than $|\epsilon_1 - \epsilon_2|$.)

Following from this, the dispersion of a set of hypotheses generated by an algorithm will be bounded by twice the average error for a task. However, the results show that the dispersion tends to be tighter: the average dispersion is less than the average error for each learning algorithm. (Compare with table 4.5.) For most of the learning algorithms, the dispersion values tend to mirror the error values: when the error is high, the dispersion tends to be high. The exceptions to this trend are the naive Bayes variants, which consistently show a low dispersion across several tasks, even where high error occurs.

4.3 Analysis

By looking at the results presented above, we can immediately make some observations. First, comparing the average dispersion values for each learning algorithm (found on the bottom row of table 4.6) with the average error of each algorithm (the bottom row of table 4.5), hypotheses produced by the same learning algorithm have a stronger tendency to agree with each other than they do with the target values. That is, their distance between each other (the COD metric) is lower than their distance to the target (the error rates). The average dispersion over all problems is 0.13 and the average error rate is 0.20, noticeably greater. The trend continues across the bottom of the tables. The average dispersion (from table 4.6) for each algorithm is consistently lower or at least no higher than the corresponding average error (table 4.6).

Comparing hypotheses generated from different algorithms shows the opposite

trend. The average distance between two algorithms (found on table 4.3) is 0.23, marginally higher than 0.20 (the average error rate from table 4.5). The effect is even stronger when comparisons are made only between algorithms of different paradigms. Every off-diagonal distance between algorithms of two paradigms (table 4.4) is greater than 0.20.

One observation can be made here by examining table 4.3. Of the learning algorithms surveyed, the most similar algorithms are *DTg* (Decision Tree using information gain as a split criterion) and *DTe* (Decision Tree using expectation gain ratio as a split criterion), with a distance of only 0.02 (i.e. on average *DTg* and *DTe* will exhibit different behaviors on 2% of unseen patterns across the given range of problems). *DTe* is actually a variant of *DTr* (Decision Tree using information gain ratio as a split criterion), intended to compensate for the information gain ratio metric's tendency to select uneven splits. As *DTe* is a variant of *DTr*, one would expect that *DTe* would be more similar to *DTr* than other learning algorithms. Yet COD shows that its behavior is in fact more like that of *DTg*.

However, it can be observed that on most of the individual tasks, *DTe* and *DTg* have a distance of 0 (every data set except those in table A.5 (*lymph*) and table A.14 (*zoo*)). Looking at the characteristics of the tasks with a zero distance, we discover that every task except *balanc* has only real-valued input features (see table 4.1). *Balanc*, on the other hand, only has enumerated input features. However, it has the same number of enumerations (five) for each input. Similarly, real-valued features are always split in a binary fashion at each level, so all the tasks for which the classifier distance was 0 have the property that the arity of all the splits available to the decision tree learning algorithm is the same at every decision point. The expectation gain ratio split criterion used in *DTe* turns out to be equivalent to the information gain criterion used in the standard *DTg* learning algorithm, augmented

with a multiplier to compensate for the tendency for high arity to lower entropy faster than lower arity. When the arity of all splits is equal, this multiplier is the same for all candidate splits and can be factored out and discarded, resulting in a criterion equivalent to information gain. On the two tasks where DTe and DTg are different (*lymph* and *zoo*), the three decision tree learning algorithms are relatively equidistant.

To go beyond these observations and perform analysis in greater depth, it is useful to examine another aspect of classifier similarity. A distance of 1.0 indicates that the two models disagree on the classification of every pattern. A pair of classifiers with this property may not necessarily be maximally different in the abstract. A concept learning task hypothesis which always disagrees with another hypothesis is functionally equivalent to the original hypothesis with an inverted output. Such a pair of hypotheses, though exhibiting a COD distance of 1.0 (the maximum), are not functionally distinct. Classifiers with a unit distance are probably rare in practice. However, it may then be asked, if a distance of 1.0 does not necessarily indicate a high degree of functional difference, what should the distance between two genuinely different hypotheses be?

If two functions are genuinely different, the output of one function will generally give no useful information for determining the output of the other. If the behaviors of two different hypotheses are statistically independent, the likelihood that the hypotheses are similar from a functionality point of view is low. However, learning algorithms tend to maximize some aspect of output accuracy and therefore hypotheses from even very different algorithms are unlikely to produce statistically independent outputs. Two powerful (even if different) algorithms solving a relatively easy learning task are likely to produce hypotheses whose outputs are highly correlated simply by virtue of achieving good accuracy.

A hypothesis performs a functional mapping from input feature values to output

classifications. It is desirable to construct a hypothesis whose output classification is always the correct classification. In practice, this generally does not occur, and some patterns are classified incorrectly. If two hypotheses always classify the same patterns incorrectly, they exhibit functional similarity. Knowing the output of one hypothesis provides information that can help predict the output of the other. By a similar token, if two imperfect hypotheses (i.e. hypotheses that have nonzero error) *never* classify the same patterns incorrectly, there is likewise some functional similarity. At some level, the two hypotheses are extracting similar structure from the learning task, and simply doing different things with it.

In order to account for this, we use statistical independence conditional on the output accuracy of the individual classifiers and on the prior distribution of the classes. If two hypotheses are genuinely different, their distance will tend toward the distance that would be expected between two hypotheses whose particular misclassified patterns are independent of each other. This expected COD distance (or *ECOD* distance $\tilde{D}$) can be computed as follows:

$$\begin{aligned}\tilde{D}_T(H_1, H_2) =& P(H_1(\boldsymbol{x}) = T(\boldsymbol{x})) \cdot P(H_2(\boldsymbol{x}) \neq T(\boldsymbol{x})) \\ &+ P(H_1(\boldsymbol{x}) \neq T(\boldsymbol{x})) \cdot P(H_2(\boldsymbol{x}) = T(\boldsymbol{x})) \\ &+ P(H_1(\boldsymbol{x}) \neq T(\boldsymbol{x})) \cdot P(H_2(\boldsymbol{x}) \neq T(\boldsymbol{x})) \cdot \\ &\quad P(H_1(\boldsymbol{x}) \neq H_2(\boldsymbol{x}) | H_1(\boldsymbol{x}) \neq T(\boldsymbol{x}) \wedge H_2(\boldsymbol{x}) \neq T(\boldsymbol{x}))\end{aligned} \tag{4.1}$$

where $\boldsymbol{x}$ is a pattern sampled from the learning task, $H_i(\boldsymbol{x})$ is the classification H_i gives for pattern $\boldsymbol{x}$, and $T(\boldsymbol{x})$ is the correct label for $\boldsymbol{x}$. There are five cases for each pattern in T to consider in computing the value of $\tilde{D}$:

1. The case where both hypotheses classify the pattern correctly.

2. The case where hypothesis 1 classifies the pattern correctly and hypothesis 2 misclassifies it.

3. The case where hypothesis 2 classifies the pattern correctly and hypothesis 1 misclassifies it.

4. The case where both hypotheses misclassify the pattern and give the same incorrect classification.

5. The case where both hypotheses misclassify the pattern and give different incorrect classifications.

Patterns falling under cases 1 and 4 are instances where the two hypotheses agree on the output, and thus contribute zero to the COD metric. The COD metric could be computed, then, by finding the proportion of patterns falling under cases 2, 3, and 5. To compute the ECOD metric, the probablity of cases 2, 3, and 5 are estimated (corresponding to the three terms of equation 4.1), assuming that the particular patterns misclassified by H_1 and H_2 are independent of each other. The first term in equation 4.1 (an estimate of case 2) is:

$$P(H_1(\boldsymbol{x}) = T(\boldsymbol{x}) \wedge H_2(\boldsymbol{x}) \neq T(\boldsymbol{x}))$$

which (assuming independence) can be factored into:

$$P(H_1(\boldsymbol{x}) = T(\boldsymbol{x})) \cdot P(H_2(\boldsymbol{x}) \neq T(\boldsymbol{x}))$$

(as given in the equation). The second term is the symmetrical case for H_2. The third term is found by estimating case 5:

$$P(H_1(\boldsymbol{x}) \neq T(\boldsymbol{x}) \wedge H_2(\boldsymbol{x}) \neq T(\boldsymbol{x}) \wedge H_1(\boldsymbol{x}) \neq H_2(\boldsymbol{x}))$$

which, by the probability transformation $P(A \wedge B) = P(A) \cdot P(B|A)$, is transformed into:

$$P(H_1(\boldsymbol{x}) \neq T(\boldsymbol{x}) \wedge H_2(\boldsymbol{x}) \neq T(\boldsymbol{x})) \cdot P(H_1(\boldsymbol{x}) \neq H_2(\boldsymbol{x}) | H_1(\boldsymbol{x}) \neq T(\boldsymbol{x}) \wedge H_2(\boldsymbol{x}) \neq T(\boldsymbol{x}))$$

The assumption of conditional independence allows one "$\wedge$" to be transformed into a product, giving:

$$P(H_1(\boldsymbol{x}) \neq T(\boldsymbol{x}))\cdot P(H_2(\boldsymbol{x}) \neq T(\boldsymbol{x}))\cdot P(H_1(\boldsymbol{x}) \neq H_2(\boldsymbol{x})|H_1(\boldsymbol{x}) \neq T(\boldsymbol{x})\wedge H_2(\boldsymbol{x}) \neq T(\boldsymbol{x}))$$

as found in equation 4.1. This term must be computed using the priors of the individual classes from the learning tasks. (It is assumed that the hypotheses tend to output classifications in proportion to the priors of the problem.) For a concept learning task, the probability that the two hypotheses disagree, given that they are both incorrect, is zero. For a multiclass problem, the probability is not quite trivial, but can be computed fairly straightforwardly.

As an example, assume that H_1 has an error rate of 0.1 and H_2 has an error rate of 0.2 on a concept learning task. If the outputs of these hypotheses are conditionally independent of each other, the expected distance between them is $0.9\cdot0.2+0.8\cdot0.1+0 = 0.26$. (In a concept learning problem, case 5 never occurs. If two hypotheses disagree, they always give the same wrong answer.) The actual distance may vary anywhere between 0.1 and 0.3.

We do not necessarily expect the conditional independence condition to occur whenever two algorithms are truly different. Properties of the training set, test set, or other factors may impose dependence despite the distinctiveness of hypotheses or algorithms being compared. However, it does provide a landmark to look for which, significantly, scales well when the accuracy improves: when the hypotheses being compared are more accurate, their COD distance is constrained to be smaller, but the ECOD distance between different hypotheses also becomes smaller.

The ECOD distances for conditionally independent hypotheses have been computed and can be found in Appendix B. The values found in those tables have been averaged to form table 4.7. This table is again condensed into paradigm values in ta-

ble 4.8, for easy comparison with table 4.4. To the extent that these tables correlate, the learning paradigms will tend to learn in observably different ways (given the fact that they all try to maximize accuracy).

ECOD	***DTg***	***DTr***	***DTe***	***MLP***	***SLP***	***NB***	***NBi***	***NBs***	***1NN***	***5NN***	***1NNp***	***5NNp***
DTg	0	0.29	0.29	0.26	0.29	0.31	0.30	0.40	0.29	0.29	0.31	0.32
DTr	0.29	0	0.29	0.27	0.30	0.32	0.31	0.41	0.29	0.30	0.31	0.33
DTe	0.29	0.29	0	0.26	0.29	0.31	0.30	0.40	0.29	0.29	0.31	0.32
MLP	0.26	0.27	0.26	0	0.25	0.28	0.27	0.37	0.27	0.26	0.29	0.30
SLP	0.29	0.30	0.29	0.25	0	0.30	0.29	0.39	0.30	0.29	0.32	0.32
NB	0.31	0.32	0.31	0.28	0.30	0	0.31	0.39	0.32	0.31	0.34	0.34
NBi	0.30	0.31	0.30	0.27	0.29	0.31	0	0.39	0.31	0.30	0.33	0.33
NBs	0.40	0.41	0.40	0.37	0.39	0.39	0.39	0	0.40	0.39	0.42	0.42
1NN	0.29	0.29	0.29	0.27	0.30	0.32	0.31	0.40	0	0.29	0.30	0.32
5NN	0.29	0.30	0.29	0.26	0.29	0.31	0.30	0.39	0.29	0	0.31	0.32
1NNp	0.31	0.31	0.31	0.29	0.32	0.34	0.33	0.42	0.30	0.31	0	0.33
5NNp	0.32	0.33	0.32	0.30	0.32	0.34	0.33	0.42	0.32	0.32	0.33	0

Table 4.7: The Average Expected COD table averaged over all tasks. Average: 0.32.

COD	**Dec. Tree**	**Connect.**	**N. Bayes**	**Inst. Based**
Dec. Tree	0.29	0.28	0.34	0.30
Connect.	0.28	0.25	0.32	0.29
N. Bayes	0.34	0.32	0.36	0.35
Inst. Based	0.30	0.29	0.35	0.31
Expected to other	0.31	0.30	0.34	0.37

Table 4.8: Expected COD distance table between algorithm paradigms.

Based on table 4.4, the paradigms that exhibit the most behavioral difference are instance based learners and naive Bayes learners (with an average COD distance of 29%). The Bayesian approach of constructing conditionally independent prior probabilities and combining them to find a posterior probability seems to capture a very different behavior from the instance based approach of modeling and evaluating the

decision surface dynamically during execution. Naive Bayes style hypotheses also behaved quite differently from decision tree hypotheses (with a 28% COD distance). The assumption of conditional feature independence employed by naive Bayes approaches is certainly not employed by the decision tree paradigm, where every decision level but the topmost takes advantage of feature space culling performed by the parent decisions.

Of the four paradigms examined herein, the naive Bayes paradigm appears to exhibit the most difference from the other three. Its average distance to any other paradigm is 0.27, higher than the instance based approach's average distance of 0.25 and the decision tree's average distance of 0.24. The high distance between naive Bayes and the other paradigms is partly a result of higher error rate on average. When more patterns are being missed, there is more room for disagreement between models. However, further evidence that the naive Bayes paradigm is behaving genuinely differently from the others can be seen when the naive Bayes entry at the bottom of table 4.4 is compared with the corresponding ECOD value in 4.8. The naive Bayes distance is closer to its ECOD value of 0.34 (indicating a likelihood of true differentiation), proportionally, than any of the other paradigms. So far as the ECOD estimate is an accurate measure of expected distance given conditional independence, this indicates that the naive Bayes approaches behave genuinely differently from the others examined in this research. Also, the bottom of table 4.6 indicates that the individual naive Bayes algorithms exhibit nearly the least amount of variation within themselves (around 9%–10%) compared to the other algorithms. Thus the high COD distance between naive Bayes and other algorithms is not a consequence of high variance in the hypotheses produced by naive Bayes approaches (since the behavioral variance is in fact relatively low).

The ECOD estimate provides a landmark to which the COD metrics may be

compared to get an indication of when COD distance values are significant. Below are shown some observations that can be drawn with this information.

Observing the individual distances between learning algorithms in table 4.3, the highest distances between two learning algorithms occur between *NBs* and the two pruning instance based learning algorithms (0.38 to each). Not only are these high distances, but they are also near the corresponding ECOD distance value from table 4.7. These are good indications of behavioral differences. Also, the next highest distances are in the 0.34–0.37 range and again occur between naive Bayes algorithms and other learning algorithms, reinforcing the inference that naive Bayes algorithms consistently behave differently from the other learning algorithms surveyed here. However, one of the naive Bayes learning algorithm family, *NBi*, exhibits more similarity to other learning algorithms than the other two. Although the *NBi* distances are not unusually low, they are not unilaterally high either and are rather close to the average value of 0.25. It is not immediately clear why this is the case, although *NBi* differs from the other two naive Bayes approaches in that it does not handle real values directly. Instead, *NBi* preprocesses them into nominal values that are handled by naive Bayes in the standard fashion. *NBs* and *NB*, by contrast, use real-valued inputs directly to obtain feature probability estimates.

Also, observe that the naive Bayes algorithms on average have the highest distance to each other (compared to the other paradigms' distance to each other), despite the fact that on three tasks (*balanc*, *lymph*, and *zoo*) their distances to each other are zero. (These particular tasks have only nominal features and the naive Bayes variants only differ on the treatment of real values.) This is partly caused by unusually high distances on two tasks (*iono* and *segm*). Naive Bayes can fail catastrophically when the assumption of independent input features is substantially not met, sometimes causing the resulting hypothesis to always or almost always output a single classification.

In addition to showing which learning algorithms behave most differently, the COD metric and ECOD estimate provide insights into which algorithms behave similarly. As was observed in a previous section, the distance between *DTg* and *DTr* is very low. The next smallest distance from table 4.3 is that between *MLP* and *SLP* at 0.11, (with a corresponding ECOD value of 0.25), a somewhat low distance and comparatively high ECOD estimate. This one may be somewhat surprising since the *SLP* algorithm is constrained to linearly separable solutions which model many real world processes rather poorly, whereas one of the motivations in the development of the *MLP* algorithm was to compensate for this deficiency. Looking at the distances on the individual tasks, *MLP* and *SLP* have a consistently low distance to each other, with one exception: the *vowel* task (table A.12). This is the only task where *SLP* does markedly worse than the best learning algorithms listed, and is hence probably the only task that can not be approximately solved using a linear model effectively. (It is observed in [AM99] that despite the well-known limitations of the *SLP* learning algorithm, it often performs competitively.)

The third and fourth smallest distances from table 4.3 are between members of the Nearest Neighbor family (0.13 between *1NN* and *1NNp*, and 0.14 between *1NN* and *5NN*). All nearest neighbor learning algorithms operate on similar principles and it is not surprising when they behave in a similar fashion. However, within the Nearest Neighbor family the largest distance is between the two pruned Nearest Neighbor algorithms. The tendencies shown in the totals for the Nearest Neighbor family of algorithms hold pretty consistently across the individual learning tasks surveyed (with the exception of *balanc*). It appears that the minor differences in the behavior of k-Nearest Neighbor for higher k compound as the pruning process occurs, resulting in more different decision boundaries relative to the non-pruned variants.

4.4 Example: COD and Ensemble Work

The discussion provided above shows the potential value of the COD metric in theoretical work. By contrast, the results in this section give a concrete motivation for use of the COD metric. It is shown how COD can be used to aid in evaluating the improvement potential for using an ensemble classifier to solve a particular learning task. For this example, the *lymph* task was chosen. (The *lymph* task was selected because it is one of the two tasks where DTg and DTe have a nonzero distance.) This section also shows how the angle measurement mentioned in section 3 can be used to combine notions of classifier distance and error rate into a single metric.

The experiments discussed above on the *lymph* learning task were used to compute COD angle measurements between several hypotheses generated from the same learning algorithms. These angle measurements are given in tables 4.9–4.18. Although in this case only the sampling was varied to construct the various hypotheses, other ways of inducing variance in the hypotheses can be used as well when constructing an ensemble classifier (such as presentation order or initial conditions for some algorithms). The COD metric would be used in a similar fashion to estimate how much behavioral variation is produced by using these methods of varying the hypothesis behavior.

Tables 4.9, 4.10, and 4.11 give the results for the three decision tree variations. The average angles between individual instances of the same learning algorithm with different sampling (48.4, 46.7, and 32.5) are moderately high, indicating a reasonable diversity in the answers given by the hypotheses. This is promising for ensemble potential.

These results can be compared against the results for naive Bayes, table 4.14. Since the naive Bayes algorithms employed here differ only in their treatment of real valued inputs and *lymph* has only enumerated inputs, only one naive Bayes variation has

Angle	Seed 1	Seed 2	Seed 3	Seed 4	Seed 5	Seed 6	Seed 7	Seed 8	Seed 9	Seed 10
Seed 1	0	37.0	44.4	46.2	41.9	48.1	54.5	44.8	37.4	42.6
Seed 2	37.0	0	49.0	40.3	29.2	44.9	34.0	42.0	41.9	32.6
Seed 3	44.4	49.0	0	50.7	44.8	48.1	51.5	49.1	34.4	54.5
Seed 4	46.2	40.3	50.7	0	41.2	41.7	33.4	42.7	37.7	38.9
Seed 5	41.9	29.2	44.8	41.2	0	52.5	38.0	46.0	35.0	35.3
Seed 6	48.1	44.9	48.1	41.7	52.5	0	47.1	42.7	40.8	50.3
Seed 7	54.5	34.0	51.5	33.4	38.0	47.1	0	42.4	35.7	45.5
Seed 8	44.8	42.0	49.1	42.7	46.0	42.7	42.4	0	43.7	35.6
Seed 9	37.4	41.9	34.4	37.7	35.0	40.8	35.7	43.7	0	46.0
Seed 10	42.6	32.6	54.5	38.9	35.3	50.3	45.5	35.6	46.0	0

Table 4.9: The COD angle table for ***DTg*** on the *lymph* task (in degrees). Average angle is 42.6.

Angle	Seed 1	Seed 2	Seed 3	Seed 4	Seed 5	Seed 6	Seed 7	Seed 8	Seed 9	Seed 10
Seed 1	0	46.7	51.9	50.3	50.6	59.0	66.8	67.0	44.0	58.9
Seed 2	46.7	0	44.8	37.3	50.0	57.6	60.0	43.5	31.4	49.2
Seed 3	51.9	44.8	0	29.0	40.6	40.0	53.1	47.0	42.8	40.4
Seed 4	50.3	37.3	29.0	0	47.4	41.8	46.7	45.3	38.4	43.0
Seed 5	50.6	50.0	40.6	47.4	0	45.8	58.9	53.0	44.5	45.3
Seed 6	59.0	57.6	40.0	41.8	45.8	0	67.9	56.0	42.4	52.3
Seed 7	66.8	60.0	53.1	46.7	58.9	67.9	0	59.9	43.8	58.3
Seed 8	67.0	43.5	47.0	45.3	53.0	56.0	59.9	0	42.4	43.5
Seed 9	44.0	31.4	42.8	38.4	44.5	42.4	43.8	42.4	0	41.3
Seed 10	58.9	49.2	40.4	43.0	45.3	52.3	58.3	43.5	41.3	0

Table 4.10: The COD angle table for ***DTe*** on the *lymph* task (in degrees). Average angle is 48.4.

Angle	Seed 1	Seed 2	Seed 3	Seed 4	Seed 5	Seed 6	Seed 7	Seed 8	Seed 9	Seed 10
Seed 1	0	43.2	48.4	52.5	39.3	56.7	44.5	43.2	49.7	50.7
Seed 2	43.2	0	47.2	53.2	41.2	39.3	43.0	43.0	43.0	50.7
Seed 3	48.4	47.2	0	50.1	42.5	49.7	44.4	34.9	49.9	55.0
Seed 4	52.5	53.2	50.1	0	58.5	62.5	47.1	51.7	53.2	58.5
Seed 5	39.3	41.2	42.5	58.5	0	48.9	33.2	41.2	31.9	57.1
Seed 6	56.7	39.3	49.7	62.5	48.9	0	40.8	39.3	48.2	54.9
Seed 7	44.5	43.0	44.4	47.1	33.2	40.8	0	36.3	32.3	50.7
Seed 8	43.2	43.0	34.9	51.7	41.2	39.3	36.3	0	41.7	45.2
Seed 9	49.7	43.0	49.9	53.2	31.9	48.2	32.3	41.7	0	53.5
Seed 10	50.7	50.7	55.0	58.5	57.1	54.9	50.7	45.2	53.5	0

Table 4.11: The COD angle table for ***DTr*** on the *lymph* task (in degrees). Average angle is 46.7.

Angle	Seed 1	Seed 2	Seed 3	Seed 4	Seed 5	Seed 6	Seed 7	Seed 8	Seed 9	Seed 10
Seed 1	0	21.6	37.0	37.3	25.0	32.8	27.9	34.1	12.9	23.1
Seed 2	21.6	0	32.2	35.1	8.4	34.4	31.8	32.5	32.8	24.8
Seed 3	37.0	32.2	0	45.7	31.1	44.0	35.6	39.7	40.8	30.7
Seed 4	37.3	35.1	45.7	0	30.8	42.8	40.3	36.2	38.9	37.9
Seed 5	25.0	8.4	31.1	30.8	0	35.5	26.6	30.9	37.3	14.8
Seed 6	32.8	34.4	44.0	42.8	35.5	0	34.5	39.3	32.1	36.8
Seed 7	27.9	31.8	35.6	40.3	26.6	34.5	0	38.7	37.9	30.1
Seed 8	34.1	32.5	39.7	36.2	30.9	39.3	38.7	0	28.7	26.1
Seed 9	12.9	32.8	40.8	38.9	37.3	32.1	37.9	28.7	0	35.6
Seed 10	23.1	24.8	30.7	37.9	14.8	36.8	30.1	26.1	35.6	0

Table 4.12: The COD angle table for ***MLP*** on the *lymph* task (in degrees). Average angle is 32.5.

Angle	Seed 1	Seed 2	Seed 3	Seed 4	Seed 5	Seed 6	Seed 7	Seed 8	Seed 9	Seed 10
Seed 1	0	23.0	43.8	38.1	28.7	28.4	45.1	33.5	27.5	26.7
Seed 2	23.0	0	45.0	25.8	22.3	29.8	35.2	29.5	32.5	27.7
Seed 3	43.8	45.0	0	38.7	45.0	45.0	38.9	41.8	56.1	48.4
Seed 4	38.1	25.8	38.7	0	36.2	25.8	45.1	28.2	37.5	31.8
Seed 5	28.7	22.3	45.0	36.2	0	30.8	37.4	38.9	42.0	29.9
Seed 6	28.4	29.8	45.0	25.8	30.8	0	40.0	43.5	40.0	25.4
Seed 7	45.1	35.2	38.9	45.1	37.4	40.0	0	48.4	45.2	47.8
Seed 8	33.5	29.5	41.8	28.2	38.9	43.5	48.4	0	35.2	36.6
Seed 9	27.5	32.5	56.1	37.5	42.0	40.0	45.2	35.2	0	34.4
Seed 10	26.7	27.7	48.4	31.8	29.9	25.4	47.8	36.6	34.4	0

Table 4.13: The COD angle table for ***SLP*** on the *lymph* task (in degrees). Average angle is 36.4.

Angle	Seed 1	Seed 2	Seed 3	Seed 4	Seed 5	Seed 6	Seed 7	Seed 8	Seed 9	Seed 10
Seed 1	0	21.5	22.6	12.5	15.0	19.4	21.1	18.2	29.0	19.2
Seed 2	21.5	0	12.3	12.5	16.6	11.8	15.4	11.0	23.6	9.1
Seed 3	22.6	12.3	0	16.9	14.8	15.8	13.0	14.5	16.9	11.1
Seed 4	12.5	12.5	16.9	0	9.2	15.3	14.5	17.1	18.5	16.5
Seed 5	15.0	16.6	14.8	9.2	0	19.5	13.0	10.8	20.7	15.0
Seed 6	19.4	11.8	15.8	15.3	19.5	0	9.5	19.4	13.5	11.5
Seed 7	21.1	15.4	13.0	14.5	13.0	9.5	0	12.4	7.6	13.4
Seed 8	18.2	11.0	14.5	17.1	10.8	19.4	12.4	0	20.1	18.2
Seed 9	29.0	23.6	16.9	18.5	20.7	13.5	7.6	20.1	0	17.3
Seed 10	19.2	9.1	11.1	16.5	15.0	11.5	13.4	18.2	17.3	0

Table 4.14: The COD angle table for ***NB*** on the *lymph* task (in degrees). Average angle is 15.7.

Angle	Seed 1	Seed 2	Seed 3	Seed 4	Seed 5	Seed 6	Seed 7	Seed 8	Seed 9	Seed 10
Seed 1	0	45.4	42.5	35.9	34.1	32.1	38.7	24.6	38.3	47.1
Seed 2	45.4	0	28.1	29.4	36.0	42.8	27.6	31.8	25.8	38.5
Seed 3	42.5	28.1	0	41.9	40.1	46.7	35.7	43.5	42.5	51.0
Seed 4	35.9	29.4	41.9	0	38.9	37.3	40.4	29.7	27.9	41.7
Seed 5	34.1	36.0	40.1	38.9	0	36.0	25.4	35.6	30.0	44.8
Seed 6	32.1	42.8	46.7	37.3	36.0	0	37.9	29.0	40.6	38.9
Seed 7	38.7	27.6	35.7	40.4	25.4	37.9	0	29.9	19.8	37.9
Seed 8	24.6	31.8	43.5	29.7	35.6	29.0	29.9	0	33.6	35.4
Seed 9	38.3	25.8	42.5	27.9	30.0	40.6	19.8	33.6	0	32.1
Seed 10	47.1	38.5	51.0	41.7	44.8	38.9	37.9	35.4	32.1	0

Table 4.15: The COD angle table for ***1NN*** on the *lymph* task (in degrees). Average angle is 36.1.

Angle	Seed 1	Seed 2	Seed 3	Seed 4	Seed 5	Seed 6	Seed 7	Seed 8	Seed 9	Seed 10
Seed 1	0	26.6	21.9	28.7	22.2	26.6	32.7	31.0	26.3	27.9
Seed 2	26.6	0	24.4	23.9	23.1	26.7	22.2	22.2	16.5	26.6
Seed 3	21.9	24.4	0	35.1	28.4	33.3	22.8	24.4	28.4	34.1
Seed 4	28.7	23.9	35.1	0	29.4	28.3	28.9	23.9	18.7	24.4
Seed 5	22.2	23.1	28.4	29.4	0	25.3	28.9	25.3	31.1	30.5
Seed 6	26.6	26.7	33.3	28.3	25.3	0	35.5	22.2	20.9	26.6
Seed 7	32.7	22.2	22.8	28.9	28.9	35.5	0	26.6	26.8	28.7
Seed 8	31.0	22.2	24.4	23.9	25.3	22.2	26.6	0	25.3	26.6
Seed 9	26.3	16.5	28.4	18.7	31.1	20.9	26.8	25.3	0	18.1
Seed 10	27.9	26.6	34.1	24.4	30.5	26.6	28.7	26.6	18.1	0

Table 4.16: The COD angle table for ***5NN*** on the *lymph* task (in degrees). Average angle is 26.4.

Angle	Seed 1	Seed 2	Seed 3	Seed 4	Seed 5	Seed 6	Seed 7	Seed 8	Seed 9	Seed 10
Seed 1	0	68.2	62.2	63.5	48.8	62.9	51.5	57.3	62.4	71.7
Seed 2	68.2	0	56.0	57.7	51.9	74.9	57.4	56.3	64.6	69.4
Seed 3	62.2	56.0	0	52.6	63.1	69.0	56.0	56.7	60.4	54.3
Seed 4	63.5	57.7	52.6	0	49.0	55.7	44.5	62.6	72.5	69.7
Seed 5	48.8	51.9	63.1	49.0	0	60.7	37.6	53.7	62.9	63.4
Seed 6	62.9	74.9	69.0	55.7	60.7	0	50.6	62.5	72.1	73.0
Seed 7	51.5	57.4	56.0	44.5	37.6	50.6	0	58.6	47.4	54.1
Seed 8	57.3	56.3	56.7	62.6	53.7	62.5	58.6	0	67.8	68.7
Seed 9	62.4	64.6	60.4	72.5	62.9	72.1	47.4	67.8	0	55.5
Seed 10	71.7	69.4	54.3	69.7	63.4	73.0	54.1	68.7	55.5	0

Table 4.17: The COD angle table for ***1NNp*** on the *lymph* task (in degrees). Average angle is 59.8.

Angle	Seed 1	Seed 2	Seed 3	Seed 4	Seed 5	Seed 6	Seed 7	Seed 8	Seed 9	Seed 10
Seed 1	0	66.0	59.0	59.9	54.2	66.4	58.2	67.0	56.6	54.3
Seed 2	66.0	0	51.9	47.7	57.5	47.4	51.6	62.9	79.8	52.0
Seed 3	59.0	51.9	0	52.9	42.8	50.1	48.9	54.8	51.0	56.3
Seed 4	59.9	47.7	52.9	0	48.1	58.4	49.7	50.7	61.6	59.4
Seed 5	54.2	57.5	42.8	48.1	0	57.5	50.6	53.4	55.8	58.8
Seed 6	66.4	47.4	50.1	58.4	57.5	0	56.1	57.2	58.1	51.4
Seed 7	58.2	51.6	48.9	49.7	50.6	56.1	0	63.6	53.1	54.7
Seed 8	67.0	62.9	54.8	50.7	53.4	57.2	63.6	0	57.9	60.8
Seed 9	56.6	79.8	51.0	61.6	55.8	58.1	53.1	57.9	0	58.1
Seed 10	54.3	52.0	56.3	59.4	58.8	51.4	54.7	60.8	58.1	0

Table 4.18: The COD angle table for ***5NNp*** on the *lymph* task (in degrees). Average angle is 56.1.

results given here. (The other two behave identically.) The naive Bayes average COD angle value is markedly lower (15.7), suggesting that there would be limited utility in employing an ensemble of naive Bayes hypotheses, compared to the potential in using decision trees. On the other hand, naive Bayes is already performing noticeably better than the decision trees are on *lymph*, according to table 4.5.

Tables 4.12 and 4.13 show similar results for the *MLP* and *SLP* individual experiments on *lymph*. These learning algorithms each achieve a lower error rate (0.18 and 0.17 respectively) and have higher average COD angles (32.5 and 36.4) than the naive Bayes variants, but the angles are not as large as for the decision tree variants.

The nearest neighbor approaches exhibit mixed tendencies. The non-pruning varieties admit similar error rates to naive Bayes (0.2 and 0.19), with somewhat higher variation (average angles between instances of the same algorithm were 36.1 and 26.4). The pruning varieties give much higher results, both in error rate (0.27 and 0.26) and average angle (59.8 and 56.1), which are comparable to the decision tree results. Again, the possibility for improving the performance of the nearest neighbor algorithms through ensemble techniques appears to be greater for the pruning variants, although the ensemble would possibly be mostly making up for lost ground against the non-pruning variants' lower error rate.

To test the hypothesis that greater COD angles imply greater potential for ensemble improvement, a simple voting ensemble was constructed for each of the algorithms discussed above. Ten fold cross-validation was used to measure the results (given in table 4.19). Ten hypotheses were constructed for each ensemble by splitting the training data ten ways and using nine partitions to train each of the ten hypotheses. This is described as a *cross-validated committee* in [PMD96], although in the experiments given here, models needing a hold-out set (i.e. *SLP* and *MLP*) used a portion of the nine partitions of training data given them. These were then combined using a

Alg	Error	Orig.	Angle
DTg	0.28	0.29	42.6
DTr	0.26	0.30	46.7
DTe	0.26	0.26	48.4
MLP	0.19	0.18	32.5
SLP	0.15	0.17	36.4
NB	0.24	0.21	15.7
1NN	0.22	0.20	36.1
5NN	0.20	0.19	26.4
1NNp	0.26	0.27	59.8
5NNp	0.19	0.26	56.1

Table 4.19: The error rate for simple ensembles of the given classifiers, constructed from ten individual hypotheses each with varied sampling. These are compared against the values in table 4.5.

plurality voting scheme to produce the output of the ensemble. The results from table 4.19, and largely confirm the hypothesis. The pruning Nearest Neighbor algorithms show improvement, but not to surpass the non-pruning varieties. Two of the three decision tree variants show improvement. In fact, most algorithms whose angle was greater than 40 show improvement in an ensemble (*DTe* showed no change) and any algorithm whose angle was less than 35 showed degradation.

Chapter 5

Contribution and Future Work

The results in section 4 provide several interesting insights into the behavior of the learning algorithms surveyed by providing a way of measuring observable behavioral differences. If one were only to observe the error rates (or accuracies) of the various models in table 4.5 (such as naive Bayes and *MLP* on *lymph*), one might conclude that the two learning algorithms exhibit roughly the same accuracies and variation. Yet, by observing the actual *behavior* differences, it is discovered that *MLP* has roughly twice as much behavioral variation as naive Bayes.

The COD metric has shown good evidence that the different learning algorithms surveyed have significant differences in their behavior, and that the behavior of hypotheses produced by different learning algorithms (at least of those surveyed) are likely to exhibit markedly diverse behaviors, beyond what would generally be seen by using one algorithm to generate multiple hypotheses.

There are other insights waiting to be gleaned through future application of COD at different levels. For example, investigation is planned to determine the significance of initial parameters, sampling, and presentation order for neural networks. Knowing which sources of variability cause behavior differences will allow researchers and

practitioners to focus efforts more effectively on relevant issues. Also, further investigations into the utility of COD in improving the effectiveness of ensemble approaches can be made, both in determining where an ensemble would be potentially useful and in selecting which particular models to include in an ensemble.

Refinements might be made on the COD process. Improvements in accuracy might be made if a distribution on the behavior of each output pattern could be used, rather than a single classification. By observing the behavior of a learning algorithm as multiple parameters are allowed to vary, the behavior of a learning algorithm could be more thoroughly characterized for comparison. Additionally, the effect of many classes and underrepresented classes on the COD measurement can be explored, both to refine the measure and to gain insight into the effect of skew and many classes on algorithms examined.

COD can be used to explore the coverage of P_I, pointing out where coverage is thin and potential improvements can be made either by discovering new learning algorithms or by combining existing learning algorithms to improve coverage. Hybrid models and ensemble work can be guided by COD metrics to indicate where combinations of models have potential to increase the coverage of P_I or where new learning algorithms might be discovered.

Also, P_I is a fuzzy concept and the sampling of it employed for this project is imperfect at best. Certainly an effective sampling of P_I ought to extend beyond the Irvine Machine Learning Repository. Employment of COD on more machine learning tasks is a significant undertaking, but one likely to be fruitful in expanding knowledge of where new learning algorithms may be discovered.

Bibliography

[AE99] A. Adejumo and Andries Engelbrecht. A comparative study of neural network active learning algorithms. In Vladimir B. Bajic and Daohang Sha, editors, *Proceedings of the International Conference on Artificial Intelligence*, pages 32–35, Durban, South Africa, 1999.

[Aha92] David W. Aha. Generalizing from case studies: A case study. In *Proceedings of the Ninth International Conference on Machine Learning, ICML'2000*, pages 1–10, Aberdeen, Scotland, 1992. Morgan Kaufmann.

[AM99] Tim L. Andersen and Tony R. Martinez. The little neuron that could. In *Proceedings of the IEEE International Joint Conference on Neural Networks*, 1999.

[Bro92] Carla E. Brodley. Dynamic automatic model selection. Technical Report UM-CS-1992-030, University of Massachusetts, February 1992.

[CBM96] Rich Caruana, Shumeet Baluja, and Tom M. Mitchell. Using the future to "sort out" the present: Rankprop and multitask learning for medical risk evaluation. In David S. Touretzky, Michael C. Mozer, and Michael E. Hasselmo, editors, *Advances in Neural Information Processing Systems*, volume 8, pages 959–965. The MIT Press, 1996.

[CH67] Thomas M. Cover and Peter Hart. Nearest neighbor pattern classification. *IEEE Transactions on Information Theory*, 13:21–27, 1967.

[DHS01] Richard O. Duda, Peter E. Hart, and David G. Stork. *Pattern Classification, Second Edition*. John Wiley & Sons, Inc., 2001.

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

[Lan95] Ken Lang. NewsWeeder: learning to filter netnews. In *Proceedings of the 12th International Conference on Machine Learning*, pages 331–339. Morgan Kaufmann publishers Inc.: San Mateo, California, 1995.

[Mit80] Tom M. Mitchell. The need for biases in learning generalizations. Technical Report CBM-TR-117, Rutgers Computer Science Department, New Brunswick, New Jersey, May 1980.

[Mit97] Tom M. Mitchell. *Machine Learning*. WCB/McGraw-Hill, 1997.

[MM96] Christopher J. Merz and Patrick M. Murphy. UCI repository of machine learning databases. Available at http://www.ics.uci.edu/ mlearn/MLRepository.html, 1996.

[OM99] David Opitz and Richard Maclin. Popular ensemble methods: An empirical study. *Journal of Artificial Intelligence Research*, 11:169–198, 1999.

[PBGC00] Bernhard Pfahringer, Hilan Bensusan, and Christophe Giraud-Carrier. Meta-learning by landmarking various learning algorithms. In *Proceedings of the Seventeenth International Conference on Machine Learning,*

ICML'2000, pages 743–750, San Francisco, California, 2000. Morgan Kaufmann.

[PMD96] Bambang Parmanto, Paul W. Munro, and Howard R. Doyle. Improving committee diagnosis with resampling techniques. In David S. Touretzky, Michael C. Mozer, and Michael E. Hasselmo, editors, *Advances in Neural Information Processing Systems*, volume 8, pages 882–888. The MIT Press, 1996.

[Qui86] J. Ross Quinlan. Induction of decision trees. *Machine Learning*, 1(1):81–106, 1986.

[Qui93] J. Ross Quinlan. *C4.5: Programs For Machine Learning*. Morgan Kaufmann Publishers, Inc., 1993.

[RHW86] David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. Learning internal representations by error propagation. *Parallel distributed processing: explorations in the microstructure of cognition, vol. 1: foundations*, pages 318–362, 1986.

[Ros58] Frank Rosenblatt. The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, 65:386–408, 1958.

[SJW93] Wolfram Schiffmann, Merten Joost, and Randolf Werner. Comparison of optimized backpropagation algorithms. In Michel Verleysen, editor, *Proceedings of the European Symposium on Artificial Neural Networks*, pages 97–104, Brussels, Belgium, 1993.

[WM95] David H. Wolpert and William G. Macready. No free lunch theorems for search. Technical Report SFI-TR-95-02-010, Santa Fe Institute, Santa Fe, New Mexico, 1995.

[Zhe93] Zijian Zheng. A benchmark for classifier learning. Technical Report TR474, Basser Department of Computer Science, N.S.W Australia 2006, 1993.

[ZM00] Xinchuan Zeng and Tony R. Martinez. Distribution-balanced stratified cross-validation for accuracy estimation. *Journal of Experimental and Theoretical Artificial Intelligence*, 12(1):1–12, 2000.

Appendix A

Individual distance tables

COD	***DTg***	***DTr***	***DTe***	***MLP***	***SLP***	***NB***	***NBi***	***NBs***	***1NN***	***5NN***	***1NNp***	***5NNp***
DTg	0	0.17	0.00	0.35	0.32	0.31	0.31	0.31	0.40	0.32	0.44	0.39
DTr	0.17	0	0.17	0.37	0.33	0.33	0.33	0.33	0.39	0.33	0.45	0.41
DTe	0.00	0.17	0	0.35	0.32	0.31	0.31	0.31	0.40	0.32	0.44	0.39
MLP	0.35	0.37	0.35	0	0.07	0.08	0.08	0.08	0.48	0.25	0.43	0.36
SLP	0.32	0.33	0.32	0.07	0	0.05	0.05	0.05	0.45	0.22	0.39	0.33
NB	0.31	0.33	0.31	0.08	0.05	0	0.00	0.00	0.44	0.22	0.40	0.32
NBi	0.31	0.33	0.31	0.08	0.05	0.00	0	0.00	0.44	0.22	0.40	0.32
NBs	0.31	0.33	0.31	0.08	0.05	0.00	0.00	0	0.44	0.22	0.40	0.32
1NN	0.40	0.39	0.40	0.48	0.45	0.44	0.44	0.44	0	0.38	0.32	0.44
5NN	0.32	0.33	0.32	0.25	0.22	0.22	0.22	0.22	0.38	0	0.39	0.33
1NNp	0.44	0.45	0.44	0.43	0.39	0.40	0.40	0.40	0.32	0.39	0	0.41
5NNp	0.39	0.41	0.39	0.36	0.33	0.32	0.32	0.32	0.44	0.33	0.41	0

Table A.1: The table of COD metrics for the *balanc* task. Average: 0.30.

COD	***DTg***	***DTr***	***DTe***	***MLP***	***SLP***	***NB***	***NBi***	***NBs***	***1NN***	***5NN***	***1NNp***	***5NNp***
DTg	0	0.28	0.00	0.33	0.33	0.46	0.37	0.47	0.36	0.32	0.40	0.39
DTr	0.28	0	0.28	0.34	0.35	0.44	0.37	0.46	0.35	0.32	0.40	0.39
DTe	0.00	0.28	0	0.33	0.33	0.46	0.37	0.47	0.36	0.32	0.40	0.39
MLP	0.33	0.34	0.33	0	0.15	0.41	0.30	0.45	0.35	0.27	0.40	0.36
SLP	0.33	0.35	0.33	0.15	0	0.44	0.30	0.47	0.38	0.28	0.41	0.37
NB	0.46	0.44	0.46	0.41	0.44	0	0.42	0.34	0.43	0.46	0.46	0.46
NBi	0.37	0.37	0.37	0.30	0.30	0.42	0	0.44	0.37	0.29	0.42	0.38
NBs	0.47	0.46	0.47	0.45	0.47	0.34	0.44	0	0.41	0.44	0.45	0.45
1NN	0.36	0.35	0.36	0.35	0.38	0.43	0.37	0.41	0	0.24	0.20	0.37
5NN	0.32	0.32	0.32	0.27	0.28	0.46	0.29	0.44	0.24	0	0.32	0.29
1NNp	0.40	0.40	0.40	0.40	0.41	0.46	0.42	0.45	0.20	0.32	0	0.40
5NNp	0.39	0.39	0.39	0.36	0.37	0.46	0.38	0.45	0.37	0.29	0.40	0

Table A.2: The table of COD metrics for the *bupa* task. Average: 0.37.

COD	***DTg***	***DTr***	***DTe***	***MLP***	***SLP***	***NB***	***NBi***	***NBs***	***1NN***	***5NN***	***1NNp***	***5NNp***
DTg	0	0.10	0.00	0.14	0.14	0.15	0.15	0.66	0.15	0.15	0.17	0.18
DTr	0.10	0	0.10	0.13	0.13	0.16	0.15	0.66	0.15	0.15	0.18	0.19
DTe	0.00	0.10	0	0.14	0.14	0.15	0.15	0.66	0.15	0.15	0.17	0.18
MLP	0.14	0.13	0.14	0	0.04	0.14	0.15	0.73	0.11	0.09	0.15	0.15
SLP	0.14	0.13	0.14	0.04	0	0.15	0.15	0.72	0.12	0.10	0.16	0.15
NB	0.15	0.16	0.15	0.14	0.15	0	0.12	0.77	0.15	0.15	0.17	0.18
NBi	0.15	0.15	0.15	0.15	0.15	0.12	0	0.74	0.15	0.16	0.17	0.17
NBs	0.66	0.66	0.66	0.73	0.72	0.77	0.74	0	0.74	0.77	0.73	0.73
1NN	0.15	0.15	0.15	0.11	0.12	0.15	0.15	0.74	0	0.09	0.09	0.15
5NN	0.15	0.15	0.15	0.09	0.10	0.15	0.16	0.77	0.09	0	0.15	0.14
1NNp	0.17	0.18	0.17	0.15	0.16	0.17	0.17	0.73	0.09	0.15	0	0.17
5NNp	0.18	0.19	0.18	0.15	0.15	0.18	0.17	0.73	0.15	0.14	0.17	0

Table A.3: The table of COD metrics for the *iono* task. Average: 0.24.

COD	***DTg***	***DTr***	***DTe***	***MLP***	***SLP***	***NB***	***NBi***	***NBs***	***1NN***	***5NN***	***1NNp***	***5NNp***
DTg	0	0.03	0.00	0.05	0.09	0.04	0.07	0.10	0.03	0.03	0.06	0.06
DTr	0.03	0	0.03	0.05	0.09	0.04	0.08	0.11	0.03	0.04	0.07	0.07
DTe	0.00	0.03	0	0.05	0.09	0.04	0.07	0.10	0.03	0.03	0.06	0.06
MLP	0.05	0.05	0.05	0	0.08	0.05	0.07	0.11	0.05	0.04	0.08	0.07
SLP	0.09	0.09	0.09	0.08	0	0.06	0.09	0.13	0.08	0.09	0.11	0.12
NB	0.04	0.04	0.04	0.05	0.06	0	0.06	0.10	0.03	0.04	0.06	0.07
NBi	0.07	0.08	0.07	0.07	0.09	0.06	0	0.12	0.07	0.07	0.09	0.10
NBs	0.10	0.11	0.10	0.11	0.13	0.10	0.12	0	0.10	0.10	0.13	0.12
1NN	0.03	0.03	0.03	0.05	0.08	0.03	0.07	0.10	0	0.02	0.05	0.06
5NN	0.03	0.04	0.03	0.04	0.09	0.04	0.07	0.10	0.02	0	0.06	0.05
1NNp	0.06	0.07	0.06	0.08	0.11	0.06	0.09	0.13	0.05	0.06	0	0.09
5NNp	0.06	0.07	0.06	0.07	0.12	0.07	0.10	0.12	0.06	0.05	0.09	0

Table A.4: The table of COD metrics for the *iris* task. Average: 0.07.

COD	***DTg***	***DTr***	***DTe***	***MLP***	***SLP***	***NB***	***NBi***	***NBs***	***1NN***	***5NN***	***1NNp***	***5NNp***
DTg	0	0.25	0.24	0.28	0.27	0.27	0.27	0.27	0.32	0.28	0.34	0.33
DTr	0.25	0	0.30	0.28	0.29	0.26	0.26	0.26	0.31	0.28	0.35	0.34
DTe	0.24	0.30	0	0.24	0.24	0.25	0.25	0.25	0.26	0.22	0.31	0.28
MLP	0.28	0.28	0.24	0	0.10	0.14	0.14	0.14	0.19	0.15	0.26	0.22
SLP	0.27	0.29	0.24	0.10	0	0.16	0.16	0.16	0.20	0.17	0.25	0.23
NB	0.27	0.26	0.25	0.14	0.16	0	0.00	0.00	0.21	0.17	0.27	0.24
NBi	0.27	0.26	0.25	0.14	0.16	0.00	0	0.00	0.21	0.17	0.27	0.24
NBs	0.27	0.26	0.25	0.14	0.16	0.00	0.00	0	0.21	0.17	0.27	0.24
1NN	0.32	0.31	0.26	0.19	0.20	0.21	0.21	0.21	0	0.14	0.21	0.24
5NN	0.28	0.28	0.22	0.15	0.17	0.17	0.17	0.17	0.14	0	0.23	0.19
1NNp	0.34	0.35	0.31	0.26	0.25	0.27	0.27	0.27	0.21	0.23	0	0.29
5NNp	0.33	0.34	0.28	0.22	0.23	0.24	0.24	0.24	0.24	0.19	0.29	0

Table A.5: The table of COD metrics for the *lymph* task. Average: 0.23.

COD	***DTg***	***DTr***	***DTe***	***MLP***	***SLP***	***NB***	***NBi***	***NBs***	***1NN***	***5NN***	***1NNp***	***5NNp***
DTg	0	0.24	0.00	0.23	0.24	0.32	0.24	0.30	0.24	0.23	0.25	0.26
DTr	0.24	0	0.24	0.24	0.25	0.32	0.27	0.32	0.24	0.25	0.27	0.28
DTe	0.00	0.24	0	0.23	0.24	0.32	0.24	0.30	0.24	0.23	0.25	0.26
MLP	0.23	0.24	0.23	0	0.09	0.24	0.19	0.26	0.17	0.16	0.20	0.21
SLP	0.24	0.25	0.24	0.09	0	0.23	0.21	0.28	0.20	0.19	0.23	0.23
NB	0.32	0.32	0.32	0.24	0.23	0	0.22	0.32	0.28	0.28	0.31	0.30
NBi	0.24	0.27	0.24	0.19	0.21	0.22	0	0.20	0.24	0.23	0.25	0.26
NBs	0.30	0.32	0.30	0.26	0.28	0.32	0.20	0	0.32	0.32	0.32	0.33
1NN	0.24	0.24	0.24	0.17	0.20	0.28	0.24	0.32	0	0.14	0.14	0.21
5NN	0.23	0.25	0.23	0.16	0.19	0.28	0.23	0.32	0.14	0	0.17	0.18
1NNp	0.25	0.27	0.25	0.20	0.23	0.31	0.25	0.32	0.14	0.17	0	0.23
5NNp	0.26	0.28	0.26	0.21	0.23	0.30	0.26	0.33	0.21	0.18	0.23	0

Table A.6: The table of COD metrics for the *musk1* task. Average: 0.24.

COD	***DTg***	***DTr***	***DTe***	***MLP***	***SLP***	***NB***	***NBi***	***NBs***	***1NN***	***5NN***	***1NNp***	***5NNp***
DTg	0	0.06	0.00	0.06	0.08	0.06	0.06	0.09	0.07	0.08	0.10	0.10
DTr	0.06	0	0.06	0.09	0.10	0.08	0.09	0.11	0.08	0.10	0.09	0.11
DTe	0.00	0.06	0	0.06	0.08	0.06	0.06	0.09	0.07	0.08	0.10	0.10
MLP	0.06	0.09	0.06	0	0.04	0.04	0.05	0.08	0.07	0.07	0.10	0.09
SLP	0.08	0.10	0.08	0.04	0	0.06	0.05	0.08	0.08	0.05	0.11	0.10
NB	0.06	0.08	0.06	0.04	0.06	0	0.04	0.08	0.06	0.06	0.09	0.09
NBi	0.06	0.09	0.06	0.05	0.05	0.04	0	0.07	0.05	0.05	0.09	0.08
NBs	0.09	0.11	0.09	0.08	0.08	0.08	0.07	0	0.10	0.08	0.13	0.12
1NN	0.07	0.08	0.07	0.07	0.08	0.06	0.05	0.10	0	0.06	0.06	0.09
5NN	0.08	0.10	0.08	0.07	0.05	0.06	0.05	0.08	0.06	0	0.10	0.08
1NNp	0.10	0.09	0.10	0.10	0.11	0.09	0.09	0.13	0.06	0.10	0	0.11
5NNp	0.10	0.11	0.10	0.09	0.10	0.09	0.08	0.12	0.09	0.08	0.11	0

Table A.7: The table of COD metrics for the *newthy* task. Average: 0.08.

COD	***DTg***	***DTr***	***DTe***	***MLP***	***SLP***	***NB***	***NBi***	***NBs***	***1NN***	***5NN***	***1NNp***	***5NNp***
DTg	0	0.26	0.00	0.24	0.24	0.25	0.25	0.30	0.31	0.26	0.36	0.33
DTr	0.26	0	0.26	0.24	0.23	0.24	0.24	0.30	0.29	0.26	0.35	0.32
DTe	0.00	0.26	0	0.24	0.24	0.25	0.25	0.30	0.31	0.26	0.36	0.33
MLP	0.24	0.24	0.24	0	0.07	0.11	0.15	0.23	0.26	0.18	0.33	0.28
SLP	0.24	0.23	0.24	0.07	0	0.09	0.15	0.21	0.26	0.17	0.32	0.28
NB	0.25	0.24	0.25	0.11	0.09	0	0.14	0.22	0.27	0.19	0.33	0.29
NBi	0.25	0.24	0.25	0.15	0.15	0.14	0	0.24	0.26	0.20	0.34	0.28
NBs	0.30	0.30	0.30	0.23	0.21	0.22	0.24	0	0.31	0.27	0.36	0.34
1NN	0.31	0.29	0.31	0.26	0.26	0.27	0.26	0.31	0	0.23	0.18	0.31
5NN	0.26	0.26	0.26	0.18	0.17	0.19	0.20	0.27	0.23	0	0.30	0.23
1NNp	0.36	0.35	0.36	0.33	0.32	0.33	0.34	0.36	0.18	0.30	0	0.34
5NNp	0.33	0.32	0.33	0.28	0.28	0.29	0.28	0.34	0.31	0.23	0.34	0

Table A.8: The table of COD metrics for the *pima* task. Average: 0.25.

COD	***DTg***	***DTr***	***DTe***	***MLP***	***SLP***	***NB***	***NBi***	***NBs***	***1NN***	***5NN***	***1NNp***	***5NNp***
DTg	0	0.04	0.00	0.04	0.08	0.21	0.09	0.86	0.05	0.07	0.08	0.10
DTr	0.04	0	0.04	0.05	0.08	0.21	0.09	0.86	0.06	0.07	0.08	0.10
DTe	0.00	0.04	0	0.04	0.08	0.21	0.09	0.86	0.05	0.07	0.08	0.10
MLP	0.04	0.05	0.04	0	0.06	0.20	0.08	0.86	0.05	0.07	0.08	0.10
SLP	0.08	0.08	0.08	0.06	0	0.22	0.09	0.85	0.08	0.09	0.10	0.12
NB	0.21	0.21	0.21	0.20	0.22	0	0.20	0.81	0.21	0.21	0.22	0.22
NBi	0.09	0.09	0.09	0.08	0.09	0.20	0	0.86	0.09	0.10	0.12	0.13
NBs	0.86	0.86	0.86	0.86	0.85	0.81	0.86	0	0.85	0.85	0.86	0.86
1NN	0.05	0.06	0.05	0.05	0.08	0.21	0.09	0.85	0	0.04	0.04	0.08
5NN	0.07	0.07	0.07	0.07	0.09	0.21	0.10	0.85	0.04	0	0.07	0.08
1NNp	0.08	0.08	0.08	0.08	0.10	0.22	0.12	0.86	0.04	0.07	0	0.10
5NNp	0.10	0.10	0.10	0.10	0.12	0.22	0.13	0.86	0.08	0.08	0.10	0

Table A.9: The table of COD metrics for the *segm* task. Average: 0.23.

COD	***DTg***	***DTr***	***DTe***	***MLP***	***SLP***	***NB***	***NBi***	***NBs***	***1NN***	***5NN***	***1NNp***	***5NNp***
DTg	0	0.20	0.00	0.28	0.28	0.36	0.28	0.41	0.29	0.30	0.31	0.35
DTr	0.20	0	0.20	0.28	0.28	0.36	0.30	0.39	0.28	0.31	0.31	0.37
DTe	0.00	0.20	0	0.28	0.28	0.36	0.28	0.41	0.29	0.30	0.31	0.35
MLP	0.28	0.28	0.28	0	0.10	0.25	0.21	0.36	0.28	0.26	0.31	0.32
SLP	0.28	0.28	0.28	0.10	0	0.26	0.23	0.38	0.28	0.28	0.30	0.32
NB	0.36	0.36	0.36	0.25	0.26	0	0.26	0.39	0.36	0.36	0.36	0.38
NBi	0.28	0.30	0.28	0.21	0.23	0.26	0	0.30	0.28	0.28	0.31	0.35
NBs	0.41	0.39	0.41	0.36	0.38	0.39	0.30	0	0.36	0.36	0.37	0.42
1NN	0.29	0.28	0.29	0.28	0.28	0.36	0.28	0.36	0	0.14	0.12	0.28
5NN	0.30	0.31	0.30	0.26	0.28	0.36	0.28	0.36	0.14	0	0.19	0.23
1NNp	0.31	0.31	0.31	0.31	0.30	0.36	0.31	0.37	0.12	0.19	0	0.30
5NNp	0.35	0.37	0.35	0.32	0.32	0.38	0.35	0.42	0.28	0.23	0.30	0

Table A.10: The table of COD metrics for the *sonar* task. Average: 0.30.

COD	***DTg***	***DTr***	***DTe***	***MLP***	***SLP***	***NB***	***NBi***	***NBs***	***1NN***	***5NN***	***1NNp***	***5NNp***
DTg	0	0.25	0.00	0.27	0.28	0.54	0.39	0.48	0.32	0.33	0.36	0.38
DTr	0.25	0	0.25	0.28	0.29	0.55	0.39	0.49	0.34	0.34	0.38	0.38
DTe	0.00	0.25	0	0.27	0.28	0.54	0.39	0.48	0.32	0.33	0.36	0.38
MLP	0.27	0.28	0.27	0	0.17	0.52	0.37	0.46	0.32	0.31	0.36	0.37
SLP	0.28	0.29	0.28	0.17	0	0.52	0.37	0.46	0.33	0.32	0.37	0.38
NB	0.54	0.55	0.54	0.52	0.52	0	0.35	0.36	0.55	0.53	0.58	0.57
NBi	0.39	0.39	0.39	0.37	0.37	0.35	0	0.34	0.38	0.34	0.43	0.42
NBs	0.48	0.49	0.48	0.46	0.46	0.36	0.34	0	0.47	0.46	0.50	0.50
1NN	0.32	0.34	0.32	0.32	0.33	0.55	0.38	0.47	0	0.19	0.17	0.29
5NN	0.33	0.34	0.33	0.31	0.32	0.53	0.34	0.46	0.19	0	0.28	0.26
1NNp	0.36	0.38	0.36	0.36	0.37	0.58	0.43	0.50	0.17	0.28	0	0.31
5NNp	0.38	0.38	0.38	0.37	0.38	0.57	0.42	0.50	0.29	0.26	0.31	0

Table A.11: The table of COD metrics for the *stvehi* task. Average: 0.37.

COD	***DTg***	***DTr***	***DTe***	***MLP***	***SLP***	***NB***	***NBi***	***NBs***	***1NN***	***5NN***	***1NNp***	***5NNp***
DTg	0	0.24	0.00	0.27	0.51	0.40	0.46	0.42	0.20	0.23	0.22	0.29
DTr	0.24	0	0.24	0.27	0.49	0.40	0.47	0.43	0.21	0.24	0.23	0.30
DTe	0.00	0.24	0	0.27	0.51	0.40	0.46	0.42	0.20	0.23	0.22	0.29
MLP	0.27	0.27	0.27	0	0.41	0.34	0.47	0.39	0.15	0.17	0.18	0.26
SLP	0.51	0.49	0.51	0.41	0	0.43	0.59	0.55	0.46	0.46	0.47	0.50
NB	0.40	0.40	0.40	0.34	0.43	0	0.52	0.42	0.35	0.36	0.36	0.39
NBi	0.46	0.47	0.46	0.47	0.59	0.52	0	0.49	0.42	0.44	0.43	0.47
NBs	0.42	0.43	0.42	0.39	0.55	0.42	0.49	0	0.36	0.38	0.37	0.41
1NN	0.20	0.21	0.20	0.15	0.46	0.35	0.42	0.36	0	0.08	0.05	0.17
5NN	0.23	0.24	0.23	0.17	0.46	0.36	0.44	0.38	0.08	0	0.09	0.18
1NNp	0.22	0.23	0.22	0.18	0.47	0.36	0.43	0.37	0.05	0.09	0	0.18
5NNp	0.29	0.30	0.29	0.26	0.50	0.39	0.47	0.41	0.17	0.18	0.18	0

Table A.12: The table of COD metrics for the *vowel* task. Average: 0.34.

COD	***DTg***	***DTr***	***DTe***	***MLP***	***SLP***	***NB***	***NBi***	***NBs***	***1NN***	***5NN***	***1NNp***	***5NNp***
DTg	0	0.08	0.00	0.08	0.07	0.07	0.10	0.18	0.26	0.31	0.30	0.33
DTr	0.08	0	0.08	0.09	0.09	0.08	0.11	0.18	0.26	0.30	0.31	0.34
DTe	0.00	0.08	0	0.08	0.07	0.07	0.10	0.18	0.26	0.31	0.30	0.33
MLP	0.08	0.09	0.08	0	0.02	0.04	0.07	0.17	0.25	0.29	0.29	0.32
SLP	0.07	0.09	0.07	0.02	0	0.04	0.07	0.17	0.25	0.29	0.29	0.32
NB	0.07	0.08	0.07	0.04	0.04	0	0.05	0.16	0.24	0.29	0.29	0.32
NBi	0.10	0.11	0.10	0.07	0.07	0.05	0	0.15	0.27	0.31	0.31	0.34
NBs	0.18	0.18	0.18	0.17	0.17	0.16	0.15	0	0.30	0.36	0.35	0.39
1NN	0.26	0.26	0.26	0.25	0.25	0.24	0.27	0.30	0	0.16	0.11	0.23
5NN	0.31	0.30	0.31	0.29	0.29	0.29	0.31	0.36	0.16	0	0.23	0.20
1NNp	0.30	0.31	0.30	0.29	0.29	0.29	0.31	0.35	0.11	0.23	0	0.24
5NNp	0.33	0.34	0.33	0.32	0.32	0.32	0.34	0.39	0.23	0.20	0.24	0

Table A.13: The table of COD metrics for the *wine* task. Average: 0.20.

COD	***DTg***	***DTr***	***DTe***	***MLP***	***SLP***	***NB***	***NBi***	***NBs***	***1NN***	***5NN***	***1NNp***	***5NNp***
DTg	0	0.07	0.09	0.11	0.09	0.11	0.11	0.11	0.06	0.08	0.08	0.09
DTr	0.07	0	0.02	0.08	0.08	0.07	0.07	0.07	0.03	0.04	0.06	0.07
DTe	0.09	0.02	0	0.08	0.08	0.07	0.07	0.07	0.05	0.05	0.07	0.08
MLP	0.11	0.08	0.08	0	0.06	0.11	0.11	0.11	0.07	0.06	0.09	0.11
SLP	0.09	0.08	0.08	0.06	0	0.11	0.11	0.11	0.06	0.06	0.08	0.09
NB	0.11	0.07	0.07	0.11	0.11	0	0.00	0.00	0.08	0.09	0.10	0.10
NBi	0.11	0.07	0.07	0.11	0.11	0.00	0	0.00	0.08	0.09	0.10	0.10
NBs	0.11	0.07	0.07	0.11	0.11	0.00	0.00	0	0.08	0.09	0.10	0.10
1NN	0.06	0.03	0.05	0.07	0.06	0.08	0.08	0.08	0	0.02	0.04	0.06
5NN	0.08	0.04	0.05	0.06	0.06	0.09	0.09	0.09	0.02	0	0.05	0.06
1NNp	0.08	0.06	0.07	0.09	0.08	0.10	0.10	0.10	0.04	0.05	0	0.07
5NNp	0.09	0.07	0.08	0.11	0.09	0.10	0.10	0.10	0.06	0.06	0.07	0

Table A.14: The table of COD metrics for the *zoo* task. Average: 0.08.

Appendix B

Individual expected distance tables

ECOD	***DTg***	***DTr***	***DTe***	***MLP***	***SLP***	***NB***	***NBi***	***NBs***	***1NN***	***5NN***	***1NNp***	***5NNp***
DTg	0	0.50	0.49	0.37	0.39	0.39	0.39	0.39	0.54	0.46	0.52	0.50
DTr	0.50	0	0.50	0.38	0.40	0.41	0.41	0.41	0.54	0.47	0.53	0.50
DTe	0.49	0.50	0	0.37	0.39	0.39	0.39	0.39	0.54	0.46	0.52	0.50
MLP	0.37	0.38	0.37	0	0.11	0.12	0.12	0.12	0.49	0.28	0.44	0.38
SLP	0.39	0.40	0.39	0.11	0	0.16	0.16	0.16	0.49	0.31	0.45	0.39
NB	0.39	0.41	0.39	0.12	0.16	0	0.17	0.17	0.50	0.31	0.46	0.40
NBi	0.39	0.41	0.39	0.12	0.16	0.17	0	0.17	0.50	0.31	0.46	0.40
NBs	0.39	0.41	0.39	0.12	0.16	0.17	0.17	0	0.50	0.31	0.46	0.40
1NN	0.54	0.54	0.54	0.49	0.49	0.50	0.50	0.50	0	0.52	0.55	0.54
5NN	0.46	0.47	0.46	0.28	0.31	0.31	0.31	0.31	0.52	0	0.50	0.46
1NNp	0.52	0.53	0.52	0.44	0.45	0.46	0.46	0.46	0.55	0.50	0	0.52
5NNp	0.50	0.50	0.50	0.38	0.39	0.40	0.40	0.40	0.54	0.46	0.52	0

Table B.1: The Expected table of COD metrics for the *balanc* task. Average: 0.40.

ECOD	***DTg***	***DTr***	***DTe***	***MLP***	***SLP***	***NB***	***NBi***	***NBs***	***1NN***	***5NN***	***1NNp***	***5NNp***
DTg	0	0.46	0.45	0.45	0.44	0.48	0.46	0.48	0.47	0.45	0.48	0.47
DTr	0.46	0	0.46	0.45	0.45	0.48	0.46	0.48	0.47	0.46	0.48	0.48
DTe	0.45	0.46	0	0.45	0.44	0.48	0.46	0.48	0.47	0.45	0.48	0.47
MLP	0.45	0.45	0.45	0	0.43	0.48	0.45	0.48	0.46	0.44	0.47	0.47
SLP	0.44	0.45	0.44	0.43	0	0.47	0.45	0.47	0.46	0.44	0.47	0.47
NB	0.48	0.48	0.48	0.48	0.47	0	0.48	0.49	0.49	0.48	0.49	0.49
NBi	0.46	0.46	0.46	0.45	0.45	0.48	0	0.48	0.47	0.46	0.48	0.48
NBs	0.48	0.48	0.48	0.48	0.47	0.49	0.48	0	0.49	0.48	0.49	0.49
1NN	0.47	0.47	0.47	0.46	0.46	0.49	0.47	0.49	0	0.46	0.48	0.48
5NN	0.45	0.46	0.45	0.44	0.44	0.48	0.46	0.48	0.46	0	0.48	0.47
1NNp	0.48	0.48	0.48	0.47	0.47	0.49	0.48	0.49	0.48	0.48	0	0.49
5NNp	0.47	0.48	0.47	0.47	0.47	0.49	0.48	0.49	0.48	0.47	0.49	0

Table B.2: The Expected table of COD metrics for the *bupa* task. Average: 0.47.

ECOD	***DTg***	***DTr***	***DTe***	***MLP***	***SLP***	***NB***	***NBi***	***NBs***	***1NN***	***5NN***	***1NNp***	***5NNp***
DTg	0	0.20	0.20	0.22	0.21	0.23	0.22	0.61	0.22	0.24	0.23	0.24
DTr	0.20	0	0.20	0.22	0.22	0.23	0.22	0.61	0.22	0.24	0.24	0.24
DTe	0.20	0.20	0	0.22	0.21	0.23	0.22	0.61	0.22	0.24	0.23	0.24
MLP	0.22	0.22	0.22	0	0.23	0.24	0.24	0.60	0.23	0.25	0.25	0.26
SLP	0.21	0.22	0.21	0.23	0	0.24	0.24	0.60	0.23	0.25	0.25	0.25
NB	0.23	0.23	0.23	0.24	0.24	0	0.25	0.60	0.24	0.26	0.26	0.27
NBi	0.22	0.22	0.22	0.24	0.24	0.25	0	0.60	0.24	0.26	0.25	0.26
NBs	0.61	0.61	0.61	0.60	0.60	0.60	0.60	0	0.60	0.60	0.60	0.59
1NN	0.22	0.22	0.22	0.23	0.23	0.24	0.24	0.60	0	0.25	0.25	0.26
5NN	0.24	0.24	0.24	0.25	0.25	0.26	0.26	0.60	0.25	0	0.27	0.27
1NNp	0.23	0.24	0.23	0.25	0.25	0.26	0.25	0.60	0.25	0.27	0	0.27
5NNp	0.24	0.24	0.24	0.26	0.25	0.27	0.26	0.59	0.26	0.27	0.27	0

Table B.3: The Expected table of COD metrics for the *iono* task. Average: 0.30.

ECOD	***DTg***	***DTr***	***DTe***	***MLP***	***SLP***	***NB***	***NBi***	***NBs***	***1NN***	***5NN***	***1NNp***	***5NNp***
DTg	0	0.12	0.11	0.12	0.15	0.10	0.14	0.17	0.10	0.09	0.12	0.11
DTr	0.12	0	0.12	0.13	0.15	0.11	0.15	0.18	0.10	0.10	0.13	0.12
DTe	0.11	0.12	0	0.12	0.15	0.10	0.14	0.17	0.10	0.09	0.12	0.11
MLP	0.12	0.13	0.12	0	0.15	0.11	0.15	0.18	0.10	0.10	0.13	0.12
SLP	0.15	0.15	0.15	0.15	0	0.14	0.17	0.20	0.13	0.13	0.16	0.15
NB	0.10	0.11	0.10	0.11	0.14	0	0.13	0.16	0.09	0.08	0.11	0.11
NBi	0.14	0.15	0.14	0.15	0.17	0.13	0	0.20	0.13	0.12	0.15	0.14
NBs	0.17	0.18	0.17	0.18	0.20	0.16	0.20	0	0.16	0.15	0.18	0.18
1NN	0.10	0.10	0.10	0.10	0.13	0.09	0.13	0.16	0	0.07	0.11	0.10
5NN	0.09	0.10	0.09	0.10	0.13	0.08	0.12	0.15	0.07	0	0.10	0.09
1NNp	0.12	0.13	0.12	0.13	0.16	0.11	0.15	0.18	0.11	0.10	0	0.13
5NNp	0.11	0.12	0.11	0.12	0.15	0.11	0.14	0.18	0.10	0.09	0.13	0

Table B.4: The Expected table of COD metrics for the *iris* task. Average: 0.13.

ECOD	***DTg***	***DTr***	***DTe***	***MLP***	***SLP***	***NB***	***NBi***	***NBs***	***1NN***	***5NN***	***1NNp***	***5NNp***
DTg	0	0.43	0.41	0.37	0.37	0.39	0.39	0.39	0.38	0.38	0.42	0.41
DTr	0.43	0	0.42	0.38	0.38	0.40	0.40	0.40	0.39	0.39	0.42	0.42
DTe	0.41	0.42	0	0.36	0.35	0.37	0.37	0.37	0.37	0.36	0.41	0.40
MLP	0.37	0.38	0.36	0	0.30	0.32	0.32	0.32	0.32	0.31	0.36	0.35
SLP	0.37	0.38	0.35	0.30	0	0.31	0.31	0.31	0.31	0.30	0.36	0.35
NB	0.39	0.40	0.37	0.32	0.31	0	0.34	0.34	0.33	0.33	0.38	0.37
NBi	0.39	0.40	0.37	0.32	0.31	0.34	0	0.34	0.33	0.33	0.38	0.37
NBs	0.39	0.40	0.37	0.32	0.31	0.34	0.34	0	0.33	0.33	0.38	0.37
1NN	0.38	0.39	0.37	0.32	0.31	0.33	0.33	0.33	0	0.32	0.37	0.36
5NN	0.38	0.39	0.36	0.31	0.30	0.33	0.33	0.33	0.32	0	0.37	0.36
1NNp	0.42	0.42	0.41	0.36	0.36	0.38	0.38	0.38	0.37	0.37	0	0.40
5NNp	0.41	0.42	0.40	0.35	0.35	0.37	0.37	0.37	0.36	0.36	0.40	0

Table B.5: The Expected table of COD metrics for the *lymph* task. Average: 0.36.

ECOD	***DTg***	***DTr***	***DTe***	***MLP***	***SLP***	***NB***	***NBi***	***NBs***	***1NN***	***5NN***	***1NNp***	***5NNp***
DTg	0	0.34	0.32	0.28	0.31	0.36	0.31	0.34	0.29	0.29	0.30	0.32
DTr	0.34	0	0.34	0.29	0.32	0.37	0.32	0.35	0.31	0.30	0.32	0.33
DTe	0.32	0.34	0	0.28	0.31	0.36	0.31	0.34	0.29	0.29	0.30	0.32
MLP	0.28	0.29	0.28	0	0.26	0.32	0.26	0.30	0.24	0.23	0.25	0.27
SLP	0.31	0.32	0.31	0.26	0	0.35	0.30	0.32	0.28	0.27	0.29	0.30
NB	0.36	0.37	0.36	0.32	0.35	0	0.35	0.37	0.34	0.33	0.34	0.36
NBi	0.31	0.32	0.31	0.26	0.30	0.35	0	0.33	0.28	0.27	0.29	0.30
NBs	0.34	0.35	0.34	0.30	0.32	0.37	0.33	0	0.31	0.30	0.32	0.33
1NN	0.29	0.31	0.29	0.24	0.28	0.34	0.28	0.31	0	0.25	0.27	0.29
5NN	0.29	0.30	0.29	0.23	0.27	0.33	0.27	0.30	0.25	0	0.26	0.28
1NNp	0.30	0.32	0.30	0.25	0.29	0.34	0.29	0.32	0.27	0.26	0	0.30
5NNp	0.32	0.33	0.32	0.27	0.30	0.36	0.30	0.33	0.29	0.28	0.30	0

Table B.6: The Expected table of COD metrics for the *musk1* task. Average: 0.31.

ECOD	***DTg***	***DTr***	***DTe***	***MLP***	***SLP***	***NB***	***NBi***	***NBs***	***1NN***	***5NN***	***1NNp***	***5NNp***
DTg	0	0.13	0.12	0.11	0.13	0.09	0.10	0.14	0.12	0.13	0.14	0.15
DTr	0.13	0	0.13	0.12	0.14	0.11	0.11	0.15	0.13	0.14	0.15	0.16
DTe	0.12	0.13	0	0.11	0.13	0.09	0.10	0.14	0.12	0.13	0.14	0.15
MLP	0.11	0.12	0.11	0	0.12	0.09	0.10	0.14	0.11	0.13	0.14	0.14
SLP	0.13	0.14	0.13	0.12	0	0.10	0.11	0.15	0.13	0.14	0.15	0.16
NB	0.09	0.11	0.09	0.09	0.10	0	0.08	0.12	0.09	0.11	0.12	0.13
NBi	0.10	0.11	0.10	0.10	0.11	0.08	0	0.13	0.10	0.12	0.13	0.13
NBs	0.14	0.15	0.14	0.14	0.15	0.12	0.13	0	0.14	0.16	0.17	0.17
1NN	0.12	0.13	0.12	0.11	0.13	0.09	0.10	0.14	0	0.13	0.14	0.15
5NN	0.13	0.14	0.13	0.13	0.14	0.11	0.12	0.16	0.13	0	0.16	0.16
1NNp	0.14	0.15	0.14	0.14	0.15	0.12	0.13	0.17	0.14	0.16	0	0.17
5NNp	0.15	0.16	0.15	0.14	0.16	0.13	0.13	0.17	0.15	0.16	0.17	0

Table B.7: The Expected table of COD metrics for the *newthy* task. Average: 0.13.

ECOD	***DTg***	***DTr***	***DTe***	***MLP***	***SLP***	***NB***	***NBi***	***NBs***	***1NN***	***5NN***	***1NNp***	***5NNp***
DTg	0	0.42	0.41	0.39	0.39	0.39	0.39	0.41	0.43	0.41	0.45	0.43
DTr	0.42	0	0.42	0.40	0.39	0.40	0.39	0.42	0.43	0.41	0.45	0.43
DTe	0.41	0.42	0	0.39	0.39	0.39	0.39	0.41	0.43	0.41	0.45	0.43
MLP	0.39	0.40	0.39	0	0.36	0.37	0.36	0.40	0.41	0.39	0.44	0.42
SLP	0.39	0.39	0.39	0.36	0	0.36	0.36	0.39	0.41	0.38	0.43	0.41
NB	0.39	0.40	0.39	0.37	0.36	0	0.37	0.40	0.41	0.39	0.44	0.42
NBi	0.39	0.39	0.39	0.36	0.36	0.37	0	0.39	0.41	0.39	0.44	0.41
NBs	0.41	0.42	0.41	0.40	0.39	0.40	0.39	0	0.43	0.41	0.45	0.43
1NN	0.43	0.43	0.43	0.41	0.41	0.41	0.41	0.43	0	0.42	0.46	0.44
5NN	0.41	0.41	0.41	0.39	0.38	0.39	0.39	0.41	0.42	0	0.45	0.43
1NNp	0.45	0.45	0.45	0.44	0.43	0.44	0.44	0.45	0.46	0.45	0	0.46
5NNp	0.43	0.43	0.43	0.42	0.41	0.42	0.41	0.43	0.44	0.43	0.46	0

Table B.8: The Expected table of COD metrics for the *pima* task. Average: 0.41.

ECOD	***DTg***	***DTr***	***DTe***	***MLP***	***SLP***	***NB***	***NBi***	***NBs***	***1NN***	***5NN***	***1NNp***	***5NNp***
DTg	0	0.07	0.07	0.07	0.10	0.23	0.11	0.86	0.07	0.09	0.09	0.12
DTr	0.07	0	0.07	0.07	0.11	0.23	0.12	0.86	0.08	0.09	0.10	0.12
DTe	0.07	0.07	0	0.07	0.10	0.23	0.11	0.86	0.07	0.09	0.09	0.12
MLP	0.07	0.07	0.07	0	0.10	0.23	0.11	0.86	0.07	0.09	0.09	0.12
SLP	0.10	0.11	0.10	0.10	0	0.26	0.15	0.86	0.11	0.13	0.13	0.15
NB	0.23	0.23	0.23	0.23	0.26	0	0.27	0.86	0.23	0.25	0.25	0.27
NBi	0.11	0.12	0.11	0.11	0.15	0.27	0	0.86	0.12	0.14	0.14	0.16
NBs	0.86	0.86	0.86	0.86	0.86	0.86	0.86	0	0.86	0.86	0.86	0.86
1NN	0.07	0.08	0.07	0.07	0.11	0.23	0.12	0.86	0	0.09	0.10	0.12
5NN	0.09	0.09	0.09	0.09	0.13	0.25	0.14	0.86	0.09	0	0.12	0.14
1NNp	0.09	0.10	0.09	0.09	0.13	0.25	0.14	0.86	0.10	0.12	0	0.14
5NNp	0.12	0.12	0.12	0.12	0.15	0.27	0.16	0.86	0.12	0.14	0.14	0

Table B.9: The Expected table of COD metrics for the *segm* task. Average: 0.25.

ECOD	***DTg***	***DTr***	***DTe***	***MLP***	***SLP***	***NB***	***NBi***	***NBs***	***1NN***	***5NN***	***1NNp***	***5NNp***
DTg	0	0.39	0.38	0.37	0.37	0.41	0.39	0.43	0.35	0.36	0.37	0.41
DTr	0.39	0	0.39	0.37	0.37	0.41	0.39	0.44	0.35	0.37	0.37	0.41
DTe	0.38	0.39	0	0.37	0.37	0.41	0.39	0.43	0.35	0.36	0.37	0.41
MLP	0.37	0.37	0.37	0	0.35	0.40	0.37	0.43	0.33	0.34	0.35	0.39
SLP	0.37	0.37	0.37	0.35	0	0.40	0.38	0.43	0.33	0.35	0.35	0.40
NB	0.41	0.41	0.41	0.40	0.40	0	0.42	0.45	0.39	0.40	0.40	0.43
NBi	0.39	0.39	0.39	0.37	0.38	0.42	0	0.44	0.36	0.37	0.37	0.41
NBs	0.43	0.44	0.43	0.43	0.43	0.45	0.44	0	0.42	0.42	0.43	0.45
1NN	0.35	0.35	0.35	0.33	0.33	0.39	0.36	0.42	0	0.32	0.33	0.38
5NN	0.36	0.37	0.36	0.34	0.35	0.40	0.37	0.42	0.32	0	0.34	0.39
1NNp	0.37	0.37	0.37	0.35	0.35	0.40	0.37	0.43	0.33	0.34	0	0.39
5NNp	0.41	0.41	0.41	0.39	0.40	0.43	0.41	0.45	0.38	0.39	0.39	0

Table B.10: The Expected table of COD metrics for the *sonar* task. Average: 0.39.

ECOD	***DTg***	***DTr***	***DTe***	***MLP***	***SLP***	***NB***	***NBi***	***NBs***	***1NN***	***5NN***	***1NNp***	***5NNp***
DTg	0	0.46	0.45	0.39	0.41	0.62	0.52	0.57	0.50	0.49	0.52	0.52
DTr	0.46	0	0.46	0.40	0.43	0.62	0.53	0.58	0.51	0.50	0.53	0.53
DTe	0.45	0.46	0	0.39	0.41	0.62	0.52	0.57	0.50	0.49	0.52	0.52
MLP	0.39	0.40	0.39	0	0.35	0.59	0.48	0.54	0.45	0.45	0.48	0.47
SLP	0.41	0.43	0.41	0.35	0	0.60	0.49	0.55	0.47	0.47	0.50	0.49
NB	0.62	0.62	0.62	0.59	0.60	0	0.65	0.67	0.64	0.64	0.65	0.65
NBi	0.52	0.53	0.52	0.48	0.49	0.65	0	0.61	0.56	0.56	0.58	0.57
NBs	0.57	0.58	0.57	0.54	0.55	0.67	0.61	0	0.60	0.60	0.61	0.61
1NN	0.50	0.51	0.50	0.45	0.47	0.64	0.56	0.60	0	0.53	0.56	0.55
5NN	0.49	0.50	0.49	0.45	0.47	0.64	0.56	0.60	0.53	0	0.56	0.55
1NNp	0.52	0.53	0.52	0.48	0.50	0.65	0.58	0.61	0.56	0.56	0	0.57
5NNp	0.52	0.53	0.52	0.47	0.49	0.65	0.57	0.61	0.55	0.55	0.57	0

Table B.11: The Expected table of COD metrics for the *stvehi* task. Average: 0.53.

ECOD	***DTg***	***DTr***	***DTe***	***MLP***	***SLP***	***NB***	***NBi***	***NBs***	***1NN***	***5NN***	***1NNp***	***5NNp***
DTg	0	0.36	0.35	0.32	0.56	0.47	0.53	0.48	0.21	0.27	0.24	0.34
DTr	0.36	0	0.36	0.33	0.57	0.48	0.53	0.48	0.22	0.28	0.25	0.35
DTe	0.35	0.36	0	0.32	0.56	0.47	0.53	0.48	0.21	0.27	0.24	0.34
MLP	0.32	0.33	0.32	0	0.54	0.44	0.50	0.45	0.17	0.23	0.20	0.30
SLP	0.56	0.57	0.56	0.54	0	0.63	0.67	0.64	0.47	0.51	0.49	0.55
NB	0.47	0.48	0.47	0.44	0.63	0	0.61	0.57	0.36	0.40	0.38	0.46
NBi	0.53	0.53	0.53	0.50	0.67	0.61	0	0.61	0.43	0.47	0.45	0.52
NBs	0.48	0.48	0.48	0.45	0.64	0.57	0.61	0	0.36	0.41	0.39	0.46
1NN	0.21	0.22	0.21	0.17	0.47	0.36	0.43	0.36	0	0.10	0.07	0.19
5NN	0.27	0.28	0.27	0.23	0.51	0.40	0.47	0.41	0.10	0	0.14	0.25
1NNp	0.24	0.25	0.24	0.20	0.49	0.38	0.45	0.39	0.07	0.14	0	0.23
5NNp	0.34	0.35	0.34	0.30	0.55	0.46	0.52	0.46	0.19	0.25	0.23	0

Table B.12: The Expected table of COD metrics for the *vowel* task. Average: 0.39.

ECOD	***DTg***	***DTr***	***DTe***	***MLP***	***SLP***	***NB***	***NBi***	***NBs***	***1NN***	***5NN***	***1NNp***	***5NNp***
DTg	0	0.16	0.14	0.10	0.10	0.10	0.13	0.22	0.30	0.33	0.33	0.36
DTr	0.16	0	0.16	0.12	0.12	0.11	0.15	0.23	0.31	0.34	0.34	0.37
DTe	0.14	0.16	0	0.10	0.10	0.10	0.13	0.22	0.30	0.33	0.33	0.36
MLP	0.10	0.12	0.10	0	0.06	0.05	0.09	0.18	0.27	0.31	0.31	0.34
SLP	0.10	0.12	0.10	0.06	0	0.06	0.09	0.18	0.27	0.31	0.31	0.34
NB	0.10	0.11	0.10	0.05	0.06	0	0.09	0.18	0.26	0.30	0.30	0.34
NBi	0.13	0.15	0.13	0.09	0.09	0.09	0	0.21	0.29	0.32	0.33	0.36
NBs	0.22	0.23	0.22	0.18	0.18	0.18	0.21	0	0.35	0.38	0.38	0.40
1NN	0.30	0.31	0.30	0.27	0.27	0.26	0.29	0.35	0	0.43	0.43	0.45
5NN	0.33	0.34	0.33	0.31	0.31	0.30	0.32	0.38	0.43	0	0.45	0.47
1NNp	0.33	0.34	0.33	0.31	0.31	0.30	0.33	0.38	0.43	0.45	0	0.47
5NNp	0.36	0.37	0.36	0.34	0.34	0.34	0.36	0.40	0.45	0.47	0.47	0

Table B.13: The Expected table of COD metrics for the *wine* task. Average: 0.26.

ECOD	***DTg***	***DTr***	***DTe***	***MLP***	***SLP***	***NB***	***NBi***	***NBs***	***1NN***	***5NN***	***1NNp***	***5NNp***
DTg	0	0.07	0.10	0.12	0.09	0.11	0.11	0.11	0.07	0.08	0.08	0.10
DTr	0.07	0	0.10	0.12	0.10	0.11	0.11	0.11	0.07	0.08	0.08	0.10
DTe	0.10	0.10	0	0.14	0.12	0.14	0.14	0.14	0.09	0.11	0.11	0.13
MLP	0.12	0.12	0.14	0	0.14	0.16	0.16	0.16	0.12	0.13	0.13	0.15
SLP	0.09	0.10	0.12	0.14	0	0.13	0.13	0.13	0.09	0.10	0.10	0.12
NB	0.11	0.11	0.14	0.16	0.13	0	0.15	0.15	0.11	0.12	0.12	0.14
NBi	0.11	0.11	0.14	0.16	0.13	0.15	0	0.15	0.11	0.12	0.12	0.14
NBs	0.11	0.11	0.14	0.16	0.13	0.15	0.15	0	0.11	0.12	0.12	0.14
1NN	0.07	0.07	0.09	0.12	0.09	0.11	0.11	0.11	0	0.08	0.08	0.10
5NN	0.08	0.08	0.11	0.13	0.10	0.12	0.12	0.12	0.08	0	0.09	0.11
1NNp	0.08	0.08	0.11	0.13	0.10	0.12	0.12	0.12	0.08	0.09	0	0.11
5NNp	0.10	0.10	0.13	0.15	0.12	0.14	0.14	0.14	0.10	0.11	0.11	0

Table B.14: The Expected table of COD metrics for the *zoo* task. Average: 0.11.