

# **Semi-Lazy Learning: Combining Clustering and Classifiers to Build More Accurate Models**

Ian Davidson and Ke Yin

Department of Computer Science, SUNY Albany, New York, 12222, USA

DAVIDSON@CS.ALBANY.EDU, KY7342@CNSVAX.ALBANY.EDU

## **Abstract**

Eager learners such as neural networks, decision trees, and naïve Bayes classifiers construct a single model from the training data before observing any test set instances. In contrast, lazy learners such as K-nearest neighbor consider a test set instance before they generalize beyond the training data. This allows making predictions from only a specific selection of instances most similar to the test set instance which has been shown to outperform eager learners on a number of problems. However, lazy learners require the storing and querying of the entire training data set for each instance which is unsuitable for the large amounts of data typically found in many applications. We introduce and illustrate the benefits of an example of semi-lazy learning that combines clustering and classification models and has the benefits of eager and lazy learners. We propose dividing the instances using cluster-based segmentation and then using an eager learner to build a classification model for each cluster. This has the effect of dividing the instance space into a number of distinct regions and building a local model for each. Our experiments on UCI data sets show clustering the data into segments then building classification models using a variety of eager learners for each segment often results in a greater overall cross-validated accuracy than building a single model or using a pure lazy approach such as K-Nearest-Neighbor. We can also consider our approach to semi-lazy learning as an example of the divide and conquer (DAC) strategy used in many scientific fields to divide a complex problem into a set of simpler problems. Finally, we find that the misclassified instances are more likely to be outliers with respect to the clustering segmentation.

## **1 Introduction and Motivation**

Most machine learning tools are examples of eager learners that build a model of the training data from which to make predictions from before seeing any test set instances. They attempt to build a global model of the phenomenon of interest that is applicable to the entire instance space or in other words a global approximation to the target function [1]. Eager learners can have the advantage of easy comprehension, the time to build complex models and efficient scoring of large test sets. Lazy learners such as K-nearest neighbor wait until they have seen the test set instances before making a prediction. This allows the learner to make predictions based on specific instances that are most similar to the test set instances allowing for a very large number (possibly infinite for a continuous instance space) of local models (or target functions) for different regions of the instance space. The benefit of allowing multiple local models is offset by the cost of storing and querying the training data set for each test set instance which means that lazy learners do not scale well for the large amount of data associated with many applications. Though the use of efficient indexing methods and pre-processing of the data into KD-Trees can reduce some of this computational burden, search and retrieval times can still be large [2].

In this paper we propose a semi-lazy learning approach that readily combines the benefits of eager and lazy learning by first clustering the instances into segments and then building a model for each segment. This effectively divides the instance space into distinct regions and

builds a local model. This approach combines readily available clustering algorithms such as K-Means and eager learners such as naïve Bayes, decision trees, and neural networks as we will demonstrate. We show that building local models allows for improved results over just building a single model using an eager learner or using K-Nearest neighbor learners. We can obtain better results without the penalty of storing and searching the entire training set for every test set instance and have the benefits of using an eager learner.

Our approach can also be considered as an example of the divide and conquer strategy. In most applications, the classification algorithm builds a single model from the entire training set of data. However, many successful industrial applications divide the instances into distinct segments and build models for each [3]. Dividing the instances by their geographic location or time-period are examples of using an obvious DAC strategy. The expectation is that different factors will drive the phenomenon to model depending on the region or time period. For example, if we were to model the phenomenon of rain, we could build a summer and winter models from data collected during those periods. Each model contains predictive patterns specific to a season and may contain general patterns applicable to all seasons. In many problems, an obvious segmentation scheme is not available but we can use clustering to identify appropriate segments. The clustering algorithm only uses the independent attributes and ignores the dependent attribute.

In the next sections we describe clustering, the terminology we use, and explore the reasons for segmenting complex problems. We describe the experimental methodology we plan to use and results. From the results we discuss several insights. Finally, we summarize our findings and describe future directions for this work.

## **2 Clustering and Terminology**

Clustering algorithms attempt to identify intrinsic classes. Most algorithms exclusively assign (known as hard assignment) each instance to only one class. We can consider this as finding the “best” set partition of the instances or partitioning the instance space. Colloquially, the best clustering solution is to find groups of instances so that within a particular group the instances are similar whilst being dissimilar to instances in other groups. In this paper, we use a K-Means clustering algorithm [4] that uses hard assignment.

The **BASE ACCURACY** is the 10-fold cross-validated accuracy obtained by building and testing a single model using the entire data set. We obtain the **SEGMENTED ACCURACY** by segmenting the instances into disjoint groups, then building, and testing models specific to a segment using the instances only from that segment. The **SEGMENTED ACCURACY** is the weighted (by cluster size) 10-fold cross-validated accuracy obtained for each segment. We diagrammatically show how we train and test models for each segment in Figure 1. The overall model consists of first dividing the instances according to the clustering model and then applying the appropriate classification model.

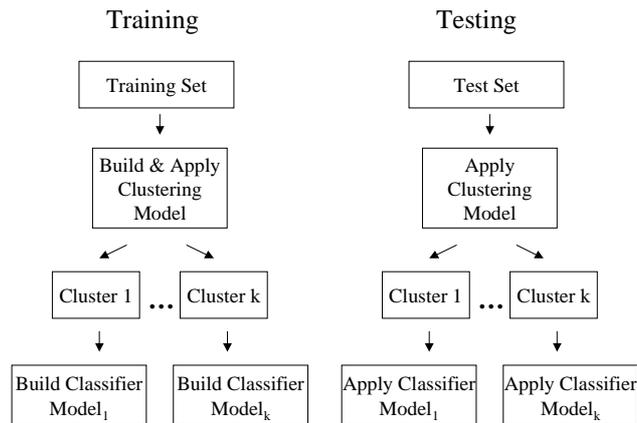


Figure 1. Training and testing using segment specific models. The clustering algorithm or model does not use the dependent variable.

### 3 Why Segment a Problem?

Two main areas of work motivate segmenting instances and then building models. The eager and lazy learning literature from machine learning literature and the divide and conquer approach from the A.I. and computer science communities.

#### 3.1 Lazy Learning: Two or More Models are Better than One

Consider a handwriting digit recognition problem where the aim is to determine the written digit. The data consists of many digits written by a number of different individuals. An eager learner such as a neural network tries to learn a global function to predict each class (digit). However, because different individuals write digits differently (see Figure 2) a single function that models each digit is difficult to produce and eager learners do not fair well on such problems.

Now consider the K-Nearest neighbor the simplest form of lazy learner. Rather than forming a single model, for each test set instance the  $k$  nearest (in terms of the instance space) training set instances are polled and the test set instance is labeled as the majority occurring digit amongst its neighbors. This allows a multitude of simple predictive devices local to parts of the instance space by using a different set of instances for each prediction depending on the test set instance. Lazy learners can outperform eager learners [5] when there exists large within class variations of the concept to learn.



Figure 2: Examples of Differently Drawn Numbers from the UCI Digit Dataset [9].

#### 3.2 Divide and Conquer

The divide and conquer (DAC) strategy is in use in many fields of A.I. and computer science for problems such as combinatorial optimization [6] and machine learning [7]. Though there is other similar heuristics such as separate and conquer (SAC) and reconsider and conquer (RAC) [8], DAC is the most widely used. The premise behind DAC is to divide a complex problem into a number of easier sub-problems. Solving and then combining the sub-solutions provides

an overall solution [6]. For instance, the DAC strategy in the field of combinatorial optimization makes combinatorially large problems tractable. If we are trying to find an optimal tour in a traveling salesman problem of thirty cities in the U.S.A., the number of possible tours is  $30!$ . If we determined the tour could be divided into an east and west coast tours of 15 cities each, with a connecting trip, then we have divided the problem into two sub-problems which together have a potential number of tours of  $2 \cdot 15!$ , a considerable saving.

A common application of the DAC approach in machine learning is to use a combination of simple models to learn a complex concept. Hidden Markov Models (HMM) can be considered an example of the DAC strategy with a complicated merging of the sub-problem solutions. A HMM in an application such as speech recognition is a series of naïve Bayes classifiers whose decisions are merged by a transition matrix [7]. The naïve Bayes classifier by itself would not by itself be able to learn to predict complicated sequences, but when used within a HMM is capable of doing so. The top-down decision tree generating algorithms such as C5.0 are examples of the divide and conquer strategy.

## 4 Experimental Methodology

Our first set of experiments focus on often used smaller data sets available from the UCI collection to get a clear understanding on the effect of the approach on accuracy and variance. The second set of experiments show the approaches usefulness on larger more complex data sets from the same collection.

### 4.1 Smaller Data Sets

The BREAST CANCER (BC), IRIS (I), DNA (D), VOTE (V), PIMA (P), HYPOTHYROID (H) and CHURN (C) data sets available from the UCI collection [9] will be the basis of this part of the empirical study.

For each data set we compare the cross-validated accuracy of a single model built and tested from the entire data set against models built and tested specifically for each segment. Every record is assigned entirely to only one segment, meaning no records are removed. We segment the instances using a weighted K-Means clustering algorithm trying  $k=1 \dots 10$  and stop the algorithm when the change in distortion is less than  $10^{-7}$ . We present the best results and the value of  $k$  used to achieve them. The predictive models are built using C5.0 with the default settings or a naïve Bayes classifier with uniform priors. We also present the ten fold cross-validated accuracies for the K-nearest neighbor algorithm for the value of  $k$  that obtained the best results.

Our aim is to determine if the approach leads to statistically significant increases in the accuracy. To get an estimate of the classifiers accuracy we perform 10 fold cross validation 20 times and compute the mean and standard deviation of the cross-validated accuracy. In our experiments if the model for a segment did not obtain an accuracy greater than the best guess model then the best guess model is used in it's place. The best guess model is to always predict the most populous class.

We are particularly interested in knowing if the misclassified instances are more likely to be outliers with respect to the clustering scheme. To investigate this we group the instances into

four intervals depending on their normalized distances to their respective cluster's center. We determine what proportion of all misclassifications fall in each interval. Finally, we use a recent visualization technique [10] to visually display where the outliers occur.

## 4.2 Larger Data Sets

To test our approach on large data sets we focus on the PEN-BASED RECOGNITION OF HANDWRITTEN DIGITS (over ten thousand instances, sixteen continuous attributes, ten classes) and ISOLET SPOKEN LETTER RECOGNITION (over seven thousand instances, six hundred and seventeen attributes and twenty six classes) data sets from the UCI repository. The problem in the DIGIT data set consists of forty people writing the digits zero through nine with the aim to identify the written digit. The ISOLET data set consists of 150 individuals who spoke each letter of the alphabet twice. The aim is to identify the spoken letter of the alphabet. As neural networks have been very accurate in these two data sets we will use them as the base learner. The neural network is an example of a fully connected feed-forward network that is trained using the back propagation (BP) algorithm. As before, we build a model for each cluster. It would appear on the surface that both these data sets would benefit from local models to capture the variations in the way that individuals write digits and speak letters. We present the ten fold cross-validated accuracies for the K-nearest neighbor algorithm to test this hypothesis and for comparison.

## 5 Experimental Results

### 5.1 Smaller Data Sets

Table 1 and Table 2 show the BASE and SEGMENTED ACCURACY for C5.0 decision trees, Table 3 and Table 4 for naïve Bayes classifiers and Table 5 the results for the K-Nearest neighbor algorithm.

	<b>BC</b>	<b>I</b>	<b>H</b>	<b>P</b>	<b>C</b>	<b>D</b>	<b>V</b>
<b>X-Validated Mean (%)</b>	93.6	94.3	99.2	74.5	94.8	94.2	95.3
<b>Standard<sup>1</sup> Deviation(%)</b>	0.54	0.38	0.01	0.10	0.02	0.04	0.04
<b>MSE</b>	0.30	0.15	0.001	0.08	0.003	0.01	0.004

Table 1. BASE ACCURACY for various data sets using the C5.0 algorithm.

	<b>BC</b>	<b>I</b>	<b>H</b>	<b>P</b>	<b>C</b>	<b>D</b>	<b>V</b>
<b>X-Validated Mean (%)</b>	97.5	96.5	99.5	74.8	94.8	93.9	94.7
<b>Standard<sup>1</sup> Deviation(%)</b>	0.19	0.55	0.00	0.08	0.03	0.02	0.03
<b>MSE</b>	0.04	0.31	0.000	0.07	0.004	0.004	0.004
<b>Lift / Room For Lift (%)</b>	61	39	37.5	1	1	-4	-12
<b>Significant Difference<sup>2</sup></b>	Yes	Yes	Yes	No	No	No	Yes
<b>Value of k</b>	3	2	2	3	5	4	3

Table 2. SEGMENTED ACCURACY for various data sets using C5.0 algorithm. The data sets are ordered left to right in decreasing order of lift that segmentation provides.

<sup>1</sup> The accuracy standard deviation for all folds over all 20 experiments.

<sup>2</sup> Test of statistical significance between the BASE ACCURACY and SEGMENTED ACCURACY at 95% confidence level

	<b>BC</b>	<b>I</b>	<b>H</b>	<b>P</b>	<b>C</b>	<b>D</b>	<b>V</b>
<b>X-Validated Mean (%)</b>	97.1	92.1	98.4	75.5	87.2	95.6	91.2
<b>Standard<sup>1</sup> Deviation(%)</b>	0.630	2.100	0.220	1.560	0.470	0.360	1.430
<b>MSE</b>	0.001	0.006	0.000	0.060	0.016	0.002	0.008

Table 3. BASE ACCURACY for various data sets using the naïve Bayes algorithm.

	<b>BC</b>	<b>I</b>	<b>H</b>	<b>P</b>	<b>C</b>	<b>D</b>	<b>V</b>
<b>X-Validated Mean (%)</b>	96.2	95.9	98.4	76.4	93.3	95.3	96.4
<b>Standard<sup>1</sup> Deviation(%)</b>	1.140	2.380	0.320	2.65	0.650	0.620	1.710
<b>MSE</b>	0.002	0.003	0.000	0.056	0.008	0.002	0.002
<b>Lift / Room For Lift (%)</b>	-34	48	-1	4	48	-8	59
<b>Significant Difference<sup>2</sup></b>	Yes	Yes	No	No	Yes	No	Yes
<b>Value of k</b>	3	2	2	3	5	4	3

Table 4. SEGMENTED ACCURACY for various data sets using the naïve Bayes algorithm.

	<b>BC</b>	<b>I</b>	<b>H</b>	<b>P</b>	<b>C</b>	<b>D</b>	<b>V</b>
<b>X-Validated Mean (%)</b>	94	87.2	94.3	70.2	85.6	92.2	91.2
<b>Standard<sup>1</sup> Deviation(%)</b>	0.63	2.1	0.22	1.56	0.37	0.36	1.43
<b>MSE</b>	0.36	1.68	0.33	7.75	2.07	0.61	0.79
<b>Value of k</b>	12	5	7	4	6	6	3

Table 5. BASE ACCURACY for various data sets using the K-Nearest Neighbor algorithm.

Table 6 and Table 7 show how many test set instance mis-classifications fall into the various distance-to-cluster-center intervals. Figure 3 visualizes misclassifications for a Breast Cancer test data set using a recent visualization technique [10]. This technique places the cluster centers in three-dimensional space so that similar clusters are adjacent and different clusters far apart. This involves mapping the distances between the cluster centers that occur in the original higher-dimensional instance space into the lower three dimensional space. The instances are then placed amongst the cluster centers to reflect their distance to the cluster centers.

	<b>BC</b>	<b>I</b>	<b>H</b>	<b>P</b>	<b>C</b>	<b>D</b>	<b>V</b>
<b>25% of Closest Observations</b>	0	0	2	13	11	21	23
<b>26-50% of Closest Observations</b>	0	0	3	16	12	24	23
<b>51-75% of Closest Observations</b>	32	27	31	36	32	23	29
<b>76-100% of Closest Observations</b>	68	73	64	35	35	32	25

Table 6. The percentage of all instance misclassifications occurring in a distance-to-cluster-center interval for the C5.0 decision tree algorithm.

	BC	I	H	P	C	D	V
<b>25% of Closest Observations</b>	14	0	16	14	2	21	1
<b>26-50% of Closest Observations</b>	15	0	17	17	7	23	1
<b>51-75% of Closest Observations</b>	31	23	22	32	25	22	32
<b>76-100% of Closest Observations</b>	40	77	45	37	66	34	66

Table 7. The percentage of all instance misclassifications occurring in a distance-to-cluster-center interval for the naïve Bayes algorithm.

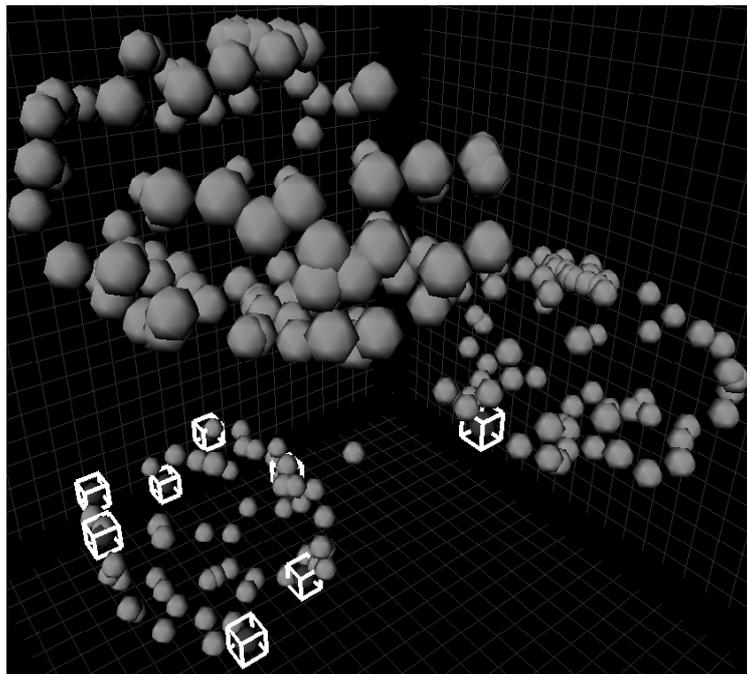


Figure 3. Visualization of the misclassifications (colored darker and bounded by a box) for a breast cancer test set using a C5.0 classifier. Each instance is a sphere; the distance an instance is from the cluster center represents its Euclidean distance to that cluster center.

### 5.2 Larger Data Sets

Table 8 shows the performances comparison of different learners: K-Nearest Neighbor (KNN), Neural Network (NN) and our approach to semi-lazy learning on some large data sets where local models should be useful.

Data set - Measure	KNN (%)	NN (%)	Semi-lazy (%)
<b>Digit – Mean Accuracy</b>	93.3	93.2	96.9
<b>Digit – Std. Deviation</b>	1.3	2.8	1.4
<b>Isolet – Mean Accuracy</b>	79.1	95.5	93.2
<b>Isolet – Std. Deviation</b>	4.0	1.0	1.4

Table 8. Average accuracy across all ten folds and standard deviation for large data sets obtained using K-Nearest Neighbor, Neural Networks, and Semi-lazy learning

## 6 Experimental Discussion

In this section, we discuss several insights derived from our experiments.

We found that decision tree, naïve Bayes classifiers and neural networks benefited from building local models for each segment. Segmenting the population using K-Means clustering and then building prediction models using either decision trees or naïve Bayes classifiers overall provides better results for a variety of data sets. For only one data set was the BASE ACCURACY significantly greater than the SEGMENTED ACCURACY. When using decision trees the BREAST CANCER, IRIS and HYPOTHYROID data sets benefited from segmentation whilst the VOTE data set did not benefit from segmentation. When using naïve Bayes classifiers the IRIS, CHURN and VOTE data sets had significantly better SEGMENTED than BASE ACCURACIES and one data set had worse (BREAST CANCER). Interestingly, the significantly better results for each classification technique did not occur for the same data sets indicating that best way to divide the instance space depends on the base learner.

When using the neural network on large data sets we found that a significant increase in accuracy occurred for the DIGIT data set but not the ISOLET data set. Given the poor results of the K-Nearest Neighbor algorithm on the ISOLET data set this is to be expected as it appears that localized models do not provide any advantage.

The accuracy increase for decision trees, naïve Bayes classifiers and neural network did not increase the variance unduly and for unstable learners it decreased. We found that the semi-lazy learning approach yields a lower learner variance in five of the seven data sets for decision trees. This is expected as decision trees are notoriously unstable estimators and techniques such as bagging improve accuracy by increasing their stability [11]. In four of the seven data sets the overall MSE was also reduced. Naïve Bayes classifiers are more stable to variations in the training set and as expected the application of our approach yields a higher variance, but of the seven data sets, five (IRIS, PIMA, HYPOTHYROID, CHURN and VOTE) had a lower MSE. This approach can reduce the bias and variance for unstable learners and reduces the bias of stable learners. Regardless of the stability of the learner, MSE is reduced.

Finally, we investigated if misclassified instances more likely to be outliers with respect to the clustering scheme. Table 6 and Table 7 shows that when the segmentation scheme yields a SEGMENTATION ACCURACY greater than the BASE ACCURACY, overwhelmingly the misclassifications are outliers. Figure 3 shows that the misclassified instances are located on the outer edges of the clusters. Furthermore, their placement is often between the cluster centers indicating that they do not belong strongly to any one cluster.

## 7 Related Work

In this section we survey related work to our own. Lazy learning has been the focus of considerable attention in machine learning literature. Pure K-Nearest neighbor approaches have been popular, as have alternatives such as locally weighted regressions [12] where the “vote” of an instance depends on its distance to the test set instance. These approaches however will not scale well as the training instances need to be stored and queried repetitively for each test set instance. Lazy decision trees (LazyDT) [13], lazy naïve Bayesian classifiers [14] are examples of lazy learning where a model is built after seeing the test set instance. Though these

approaches can improve on the accuracy of model-less lazy learners they are even more time consuming to apply.

Perhaps the closest related work in principle to our approach to semi-lazy learning is radial basis functions [15] a form of neural network that consists of many local kernel functions which are gated together to form one model. Each hidden layer node is a Gaussian kernel and represents some part of the instance space. A key decision in radial basis functions is how many kernels to use. It is common in smaller data sets to have a kernel function for each instance [1] but this does not scale well. An alternative is to use mixture modeling (soft assignment clustering) to find a region of the instance space for each kernel. This has the effect of separately setting the weights of the input layer using a mixture modeler and the output layer weights using a typical neural network training algorithm.

Even though it was known that lazy learners scale poorly since the computational cost is directly proportional to the number of test and training set instance [1] and less lazy learners have been mentioned [16] little work has explored combining eager and lazy learners to form a hybrid approach. Our approach to semi-lazy learner can overcome the scalability problem by partitioning the instance space and building local models in advance of obtaining the test set instances thus avoiding storing and searching the training set. However, our approach inherits the advantage of pure lazy learners that a prediction only uses the most pertinent training data.

## **8 Conclusion and Future Work**

Eager and lazy learners both have desirable properties. Eager learners can in the case of naïve Bayes classifiers and decision trees provide an interpretable model and can quickly score large test sets. However, eager learners do not perform well when multiple localized models best approximate the general phenomenon to model. In this situation lazy learners such as the K-Nearest neighbor algorithm can outperform eager learners as the instances to base predictions on are not chosen until the test set instance is seen. However, lazy learners require the storage and search of the entire training set for each and every test instance which does not lend itself to the large amount of data typically found in many applications. We postulate semi-lazy learning and discuss one such approach that segments the instances using clustering and then building a model for each segment. By dividing the training data into segments, we build a local model for a particular region of the instance space. This can be considered a divide and conquer strategy when an obvious segmentation scheme is not available. Divide and conquer strategies have been used extensively and successfully in many areas of science.

We tried in total seven smaller UCI data sets and used both naïve Bayes and C5.0 decision tree classifiers. Of these data sets for both techniques, three had statistically significant increases in accuracies, three had no significant difference, and one was significantly worse. The approach also worked for larger data sets with neural networks such as the DIGIT data set where localized models are desirable but not for the ISOLET data sets where the K-Nearest neighbor technique performed poorly.

From our empirical studies we found that the approach yields significantly better results in many cases for the three types of base classifier tried. The improved accuracy is often better than any of the base classifier's accuracy and the accuracy obtained by the K-Nearest neighbor

approach showing that the improvement is practically significant. For unstable learners such as decision trees, the increase in accuracy comes with a decrease in variance and MSE. However, when applied to stable learners such as naïve Bayes classifiers an increase in variance occurs, but in five cases, the MSE is reduced. Finally, when using this approach the misclassifications that occur are overwhelmingly outliers.

From these insights, it seems worthwhile to simultaneously build both the clustering and classification models and we plan to explore this option in the future. Furthermore, since misclassifications tend to be outliers, it would be interesting to further investigate the idea of building models only from the inliers (that is remove the outliers).

We have inherently created an ensemble of predictors and we will conduct empirical comparisons against similar techniques such as bagging. We have shown our approach works for hard divisive clustering approaches such as K-Means and believe this will also occur for probabilistic approaches such as mixture modeling.

We would like to determine if the approach yields overall less complex models and compare its computation time to producing a single model. This would require careful consideration of the measure of complexity to factor in the complexity of performing the segmentation. Finally, we would also like to explore further and propose a measure not unlike the BIC (Bayesian Information Criterion) or AIC (Akaike Information Criterion) [17] to determine the best model space (value of  $k$ ) to search that will yield the greatest SEGMENTED ACCURACY improvement.

## References

---

- [1] Mitchell, T., *Machine Learning*, MacGraw Hill, 1997.
- [2] Sample N., Haines M., Arnold M., Purcell T., Hybrid Search for Optimization of High-Dimensional k-d Trees, 5th WSES/IEEE World Multiconference on Circuits, Systems, Communications & Computers, 2001.
- [3] Tuason, N. and Parekh, R. "Mining Operational Databases to Predict Short-Term Defection Among Insured Households." *Proceedings of the 2000 Advanced Research Techniques Forum (ART'2000)*, Monterey, CA. Jun 4-7, 2000
- [4] MacQueen, J., Some Methods for classification and analysis of multivariate observations, *Proceedings of the Fifty Berkeley Symposium on Mathematics, Statistics and Probability*, volume 1, pages 281-296, 1967.
- [5] Alimoglu F., Combining Multiple Classifiers for Pen-Based Handwritten Digit Recognition, MSc Thesis, Institute of Graduate Studies in Science and Engineering, Bogazici University, 1996.
- [6] Horowitz E. and Sahni S., *Fundamentals of Computer Algorithms*, Computer Science Press, Inc, 1978.
- [7] Dietterich T. G., "The Divide-and-Conquer Manifesto", *Algorithmic Learning Theory*, pp. 13-26, 2000.
- [8] Bostrom H., "Covering vs Divide and Conquer For Top Down Induction of Logic Programs", *Proceedings of the Fourteen International Joint Conference on Artificial Intelligence*, Morgan Kaufman, pp 1194-1200, 1995.
- [9] Merz C, Murphy P., *Machine learning Repository* [<http://www.ics.uci.edu/~mllearn/MLRepository.html>]. Irvine, CA: University of California, Department of Information and Computer Science. 1998.
- [10] Davidson, I., "Visualizing Clustering Results", *SIAM International Conference on Data Mining*, Arlington Virginia, 2002.
- [11] Breiman L., Bagging predictors, *Machine Learning*, 26(2):123-140, 1996
- [12] Cleveland W. and Devlin S., *Locally-weighted regression: An approach to regression analysis by local fitting*. *Journal of the American Statistical Association*, 83:596--610, 1988.
- [13] Friedman J., Kohavi R., Yun Y., Lazy Decision Trees. *Proceedings of the Thirteenth National Conference on Artificial Intelligence and the Eighth Innovative Applications of Artificial Intelligence Conference*, 1996.
- [14] Zheng Z., Webb G., Ting K., (1999). Lazy Bayesian Rules: A Lazy Semi-Naive Bayesian Learning Technique Competitive to Boosting Decision Trees. *Proc. 16th International Conf. on Machine Learning*.
- [15] Powell, M., Radial Basis Functions for multivariate interpolation: A Review, In *Algorithms for Approximations*, Oxford: Clarendon Press, 1987.
- [16] Atkeson C., Moore A., Schaal S.,(1996). Locally Weighted Learning. *Artificial Intelligence Review*.
- [17] Akaike, H., "On entropy maximization principle", In P R Krishnaiah, editor, *Applications of Statistics*, pages 27-41. North Holland, Amsterdam, 1977.