

Parameter Optimized, Vertical, Nearest-Neighbor-Vote and Boundary-Based Classification

William Perrizo

Dept. of Computer Science
North Dakota State University
P.O. Box 5164,

Fargo, ND 58105, USA.

1-701-231-7248

william.perrizo@ndsu.edu

Amal Perera

Dept. of Computer Science
North Dakota State University
P.O. Box 5164,

Fargo, ND 58105, USA.

1-701-866-6433

amal.perera@ndsu.edu

ABSTRACT

In this paper, we describe a reliable high performance classification system that includes a Nearest Neighbor Vote based classification and a Local Decision Boundary based classification combined with an evolutionary algorithm for parameter optimization and a vertical data structure (Predicate Tree or P-tree¹) for processing efficiency.

Categories and Subject Descriptors

H 2.8 [Database Applications]: Data mining

General Terms

Algorithms, Performance, Classification, Design, Reliability.

Keywords

Predicate tree, total variation, nearest neighbor, boundary.

1. INTRODUCTION

Computer Aided Detection (CAD) applications are very interesting data mining problems. Typical CAD training data sets are large and extremely unbalanced between positive and negative classes. When searching for descriptive features that can characterize the target medical situations, researchers often deploy a large set of experimental features, which may introduce irrelevant and redundant features. Labeling is often noisy as labels may be created without corresponding biopsies or other independent confirmations. In the absence of CAD decision support, labeling made by humans typically relies on a relatively small number of features, due to naturally limitations in the number of independent features that can be reasonably integrated by human decision makers.

In order to achieve clinical acceptance, CAD systems have to meet extremely high performance thresholds to provide

value to physicians in their day-to-day practice. Pulmonary embolism (PE) is a condition that occurs when an artery in the lung becomes blocked. In most cases, the blockage is caused by one or more blood clots that travel to the lungs from another part of your body. The final goal in a PE CAD system is to be able to identify the sick patients from the available descriptive features. This paper describes an approach that was successfully used to submit the winning entry for the Negative Predictive Value (NPV) task of ACM 2006 KDD Cup data mining competition. All three tasks were related to developing error tolerant PE CAD systems[15]. According to the competition description one of the most useful applications for CAD would be a system with very high Negative Predictive Value. In other words, a CAD system with zero positive candidates for a given patient, that would increase the confidence that the patient was indeed free from any condition that requires medical attention. In a very real sense, this is considered as the “Holy Grail” of a PE CAD system. This task was further complicated by the relatively few negative cases in the provided training set for the competition.

In this CAD classifier, variant forms of Nearest Neighbor Vote based classification and Local Decision Boundary based classification were combined with an evolutionary algorithm for parameter optimization and a vertical data structure (Predicate Tree or P-tree) for efficient processing. Because of the inability to conduct exhaustive search to find the best possible parameter combination, a Genetic Algorithm (GA) heuristic search was employed to optimize parameters. Each iteration of the genetic algorithm requires multiple evaluations of the classification algorithms with suggested sets of adaptive parameters. Even for small data sets, this results in a large number of training database scans to arrive at an acceptable solution. The use of a vertical data structure (P-tree) and an efficient reduction functional (Total Variation) to identify neighbors, provide acceptable computation times. Each

¹ United States Patent Number US 6,941,303 B2, Sept. 6, 2005.

component of the proposed solution and the adaptive parameters optimized by the GA are discussed in subsequent sections of this paper.

2. BACKGROUND

2.1 Similarity and Attribute Relevance

Similarity models that consider all attributes as equal, such as the classical K-Nearest-Neighbor classification (KNN) work well when all attributes are similar in their relevance to the classification task. This is, however, often not the case. The problem is particularly pronounced for situations with a large number of machine generated attributes with some known to be irrelevant. Many solutions have been proposed, that weight dimensions according to their relevance to the classification problem. The weighting can be derived as part of the algorithm [4]. In an alternative strategy the attribute dimensions are scaled, using an evolutionary algorithm to optimize the classification accuracy of a separate algorithm, such as KNN [20]. Our algorithm is similar to the second approach, which is generally slower but more accurate.

Genetic Algorithms (GA) have been effectively used in data analysis and pattern recognition [19][22][10][14]. An important aspect of a GA is pattern recognition. Two major approaches are evident in the literature. One approach is to apply a GA directly as a classifier to find the decision boundary in N dimensional feature space [3]. The other approach is to use a GA as an optimization tool for resetting the parameters in other classifiers. Researchers have used GAs in feature selection [9][25][17] and also to find an optimal set of feature weights that improve classification accuracy.

Preliminary prediction trials using all attributes indicated a lack of convergence for the evolutionary algorithms with all the attributes in the training data set. Inability to optimize a global cost function through attribute scaling in a large collection of attributes is documented [7]. It is also, important to consider that current detections used as the training data set are done by humans. It is likely that only a few of the many attributes are actually used. This is due to the known fact that the human mind has very limited capacity for managing multiple simultaneous contexts.

In an effort to reduce the number of attributes in this task, information gain with respect to class was employed, along with several vertical greedy attribute selection techniques as a preprocessing step. It is also important to note that the vertical representation facilitates efficient attribute relevance in situations with a large number of attributes. Attribute relevance for a particular attribute with respect to the class attribute can be achieved by only reading those two data columns compared to reading all the long rows of data in a horizontal approach.

2.2 P-tree Data Structure

The input data was converted to vertical Predicate-trees or P-trees. P-trees are a lossless, compressed, and data-mining-ready vertical data structures. P-trees are used for fast computation of counts and for masking specific phenomena. This vertical data representation consists of set structures representing the data column-by-column rather than row-by row (horizontal relational data). Predicate-trees are one choice of vertical data representation, which can be used for data mining instead of the more common sets of relational records. A P-tree is a lossless, compressed, and data-mining ready data structure. This data structure has been successfully applied in data mining applications ranging from Classification and Clustering with K-Nearest Neighbor, to Classification with Decision Tree Induction, to Association Rule Mining [11][6][19][1][21][2][12][18][23].

A basic P-tree represents one attribute bit that is reorganized into a tree structure by recursive sub-division, while recording the predicate truth value for each division. Each level of the tree contains truth-bits that represent subtrees and can then be used for phenomena masking and fast computation of counts. This construction is continued recursively down each tree path until downward closure is reached. E.g., if the predicate is “purely 1 bits”, downward closure is reached when purity is reached (either purely 1 bits or purely 0 bits). In this case, a tree branch is terminated when a sub-division is reached that is entirely pure (which may or may not be at the leaf level). These basic P-trees and their complements are combined using boolean algebra operators such as AND(&) OR() and NOT(ˆ) to produce mask P-trees for individual values, individual tuples, value intervals, tuple rectangles, or any other attribute pattern. The root count of any P-tree will indicate the occurrence count of that pattern. The P-tree data structure provides a structure for counting patterns in an efficient, highly scalable manner.

2.3 Total Variation Based Nearest Neighbor Pruning

The total variation of a training data set about a given data point can be used as an index to arrive at a good set of potential neighbors [1][2]. We call these reduced sets of potential candidate neighbors functional contours (where the functional is Total Variation about the point). Total variation is just one of many potentially useful functionals for fast pruning of non-neighbors. The total variation for each data point in a data set can be computed rapidly using vertical P-trees. For each given new sample the total variation is calculated and the data points within a total variation plus or minus a certain distance (epsilon) is identified as potential nearest neighbors. Subsequently the actual Euclidian distance for each sample point is calculated to find the true neighbors. This approach avoids

the requirement of computing all Euclidian distances between the sample and all the other points in the training data, in arriving at the Nearest Neighbor set.

3. CLASSIFICATION

3.1 Closed Nearest Neighbor Classification

Most of the classification algorithms applicable to pattern recognition problems are based on the k-nearest neighbor (KNN) rule[23]. Each training data vector is labeled by the class it belongs to and is treated as a reference vector. During classification k nearest reference vectors to the unknown (query) vector X are found, and the class of vector X is determined by a ‘majority rule’. Restricting the Nearest Neighbor set to k can lead to wrong classifications if there are many ties for the kth neighbor. In a classical KNN approach the nearest neighborhood is not closed due to algorithmic complexities of finding all neighbors within a given distance using one horizontal pass of the dataset. Using P-tree vertical structures, all these ties can be included at no additional processing cost using the ‘closed k Nearest Neighbor’ set[11]. Optimized attribute relevance weighted Euclidian distances were used to find the nearest neighbor set. The weight values are based on the parameter optimization step. Rather than allowing a uniform vote for each neighbor Gaussian weighted votes were used. Gaussian weighting is based on the distance from the sample to the voting training point. A training sample closer to the test sample will have a higher influence on the final classification outcome compared to a sample that is further away from the test sample based on its’ distance to the sample. The vote contribution is also further adjusted based on the overall count for each class. The contribution to each class vote from ‘xth’ epsilon neighbor in class ‘c’ for test sample ‘s’ is calculated as follows.

$$Vote(x_c) = e^{-(\sigma^2 d(x_c, s)^2)} \times ClassPercent_c \times VoteFact_c$$

In the above equation d(x,s) indicates the weighted Euclidian distance from sample ‘s’ to neighbor x. The parameters ‘VoteFact’, ‘sigma’ and ‘epsilon’ are optimized by evolutionary algorithms. Values used for the final classification is described in a later section of this paper. These values will be based on the training data set and the cost function of the particular task and cannot be assumed to be global for all situations. ‘ClassPercent’ is a constant based on the proportion of training samples in each class which will enable to balance any major difference in the number of items in each class.

3.2 Boundary based Classification

In contrast to the traditional boundary based approach (e.g., Support Vector Machines) of finding a hyper plane that

separates the two classes for the entire data set (albeit, likely in a much higher dimension translate), we find the direction of the sample with respect to a local boundary segment between the two classes for a given sample’s nearest neighborhoods only. The same epsilon nearest neighbors identified in the Nearest Neighbor voting step is used to compute the local boundary between the classes. The justification is that all smooth boundaries can be approximated to any level of accuracy by a piecewise linear boundary. For the two classes we identify the vector of medians for each class (-ve and +ve) and the Euclidian spatial mid point between these two medians. The inner product of the vector that is formed by the midpoint and the sample and the vector that is formed by the midpoint and the positive (class label 1) median is used for the classification. The following diagram shows an example for two dimensional data.

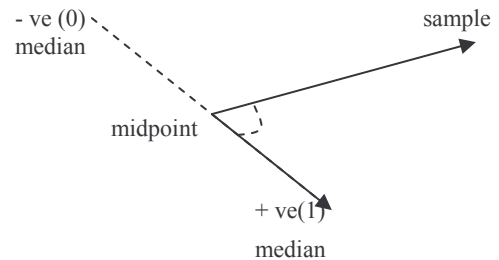


Figure 1 Inner Product Computation

The basic intuition for the above classification decision is, if the class is positive the inner product will be positive. The magnitude of the Inner Product will indicate the confidence of classification. With higher values of the Inner Product classification confidence will increase. If the sample lies on the perpendicular bisector that goes through the mid point (on the boundary) it will return a 0 that indicates a situation where it is unable to classify the given sample with the available training samples. Rather than using 0 as the ultimate decision threshold we are also able to increase the confidence of the positive or negative prediction by using a non 0 threshold that is optimized by the evolutionary algorithm. This is useful in a situation where a certain type of class identification is valued more than the other class and in situations with a high level of noise in the training set. In this specific medical CAD application it is more important to be able to identify candidates who do not suffer from PEs with a higher confidence. Evaluation cost function penalizes false negatives with respect to the computation of the Negative Predictive Value (NPV) Other less expensive centroids such as the mean, weighted mean were used in the initial trials and were found to be not as good as the use of the median. Even in a local neighborhood outliers with respect to each class could exist and the adverse effect from those

outliers is minimized by the use of the median as the centroid. It is also important to note that the identification of the local median for a given class can be easily achieved using the vertical data structure P-tree with a few AND operations on the selected nearest neighbor set. The following figure shows the efficient in-place computation of the median using P-trees for a given attribute without having to copy and do a partial sort.

```

Input: P0, P1, ..., Pn-1, C, N
Output: MEDIAN (P) {
  MEDIAN=0; SoFar=0;
  T = C & N ; CNT = COUNT(T)
  for i = n-1 to 0 step -1
    if (SoFar+COUNT(T&Pi') ≤ CNT/2){
      SoFar = SoFar + COUNT( T & Pi')
      T = T & Pi ;
      MEDIAN = MEDIAN + 2i
    } else T = T & Pi'
  return MEDIAN }

```

Figure 2 Algorithm to find Median

In the above, for a given attribute P, P_i will refer to the ith bit of attribute P. COUNT(P_i) will refer to the count of 1 bits in the ith bit slice. C and N refer to the Predicate tree that represents the particular class and the local neighbor set respectively.

3.3 Final Classification Rule

As described in the previous section for the NPV task predicting the negative ('0') class is more important than predicting the positive class ('1'). In other words we want to be absolutely certain when we clear a patient of having any potential harmful conditions that may require medical attention. The final classification rule was based on a simple rule that combined classification by the Closed Nearest Neighbor Vote and the Local Class Boundary, as follows.

```

If(NNClass=0) & ( BoundryBasedClass=0 )
  Class = 0
else class = 1

```

Figure 3 Final Classification Rule

As it is evident in the above rule a candidate is designated as a negative ('0') if and only if both classifiers agree. Attempts in using boosting with optimized weights for the

respective classifiers did not yield any significant improvements compared to the above simple rule.

4. PARAMETER OPTIMIZATION

The multiple parameters involved in the two classification algorithms were optimized via an evolutionary algorithm. The most important aspect of an evolutionary algorithm is the evaluation or fitness function, to guide the evolutionary algorithm towards a optimized solution. The evaluation function was based on the following.

```

If( NPV ≥ 0.5 )
  SolutionFitness = NPV+TN
else SolutionFitness = NPV

```

Figure 4 Evaluation (Cost) Function

Negative Predictive Value NPV and True Negatives (TN) are calculated based on the task specification for KDDCup06 [15]. NPV is calculated by TN/(TN+FN). The above fitness function encourages solutions with high TN, provided that NPV was within a threshold value. Although the task specified threshold for NPV was 1.0, with the very low number of negative cases it was hard to expect multiple solutions that meet the actual NPV threshold and also maintained a high TN level. In a GA, collections of quality solutions in each generation potentially influence the set of solutions in the next generation.

Since the training data set was small, patient level bootstrapping was used to validate solutions. Bootstrap implies taking out one sample from the training set for testing and repeating until all samples were used for testing. In this specific task which involves multiple candidates for the same patient we removed all candidates from the particular training set for a given patient when used as a test sample.

The overall approach of the solution is summarized in the following diagram. As described in the above figure, attribute relevance is carried out on the training data to identify the top M attributes. Those attributes are subsequently used to optimize the parameter space, with a standard Genetic Algorithm (GA) [8]. The set of weights on the parameters represented the solution space that was encoded for the GA. Multiple iterations with the feedback from the solution evaluation is used to arrive at the best solution. Finally the optimized parameters are used on the combined classification algorithm to classify the unknown samples.

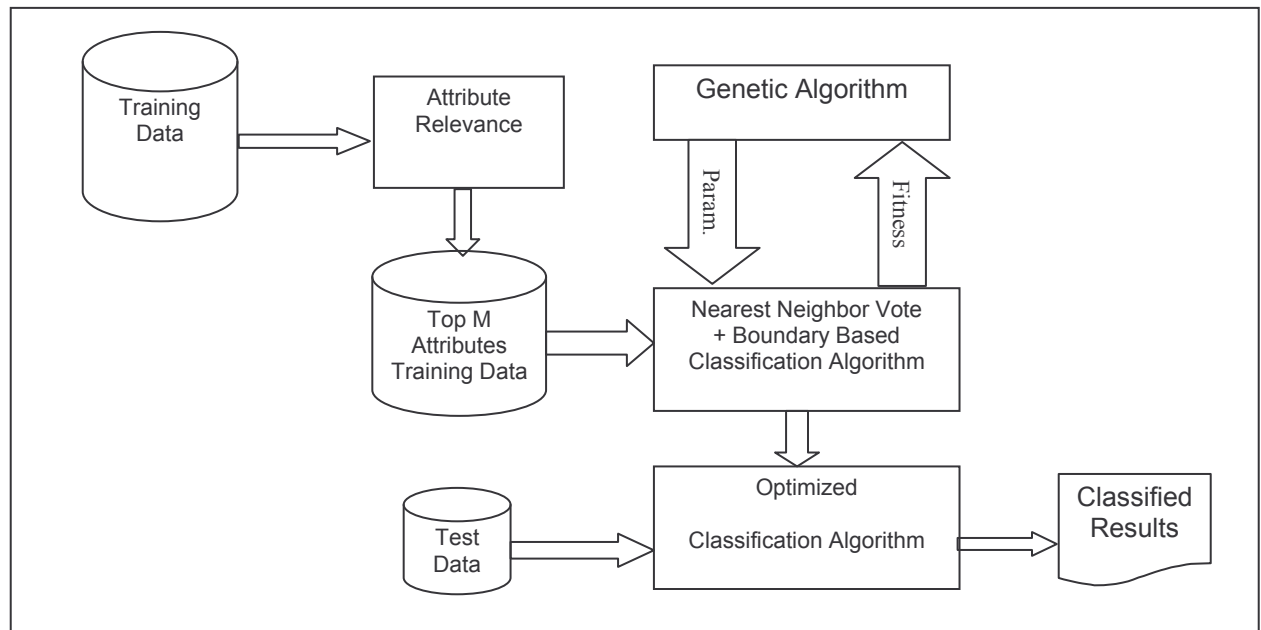


Figure 5 Overview of the Classification Approach

Following table describes the optimized parameters for NPV task(3).

Parameter	Value	Range
Epsilon	0.5	0.1 to 3.2
VoteFactor _c	1.0 ₀ , 0.75 ₁	0.0 to 2.0
Inner Product Threshold	0.23	- 0.6 to 0.27

Table 1 Optimized Parameters

In the above table ‘epsilon’ indicates the radius of the closed nearest neighbor set used for both the NN classification and the local boundary based classification. ‘VoteFactor’ and ‘Inner Product Threshold’ was used to influence one class over the other class in the NN classification and the Local Boundary Based Classification respectively. Both values chosen by the optimized solution indicate a bias towards the negative class as expected from the heavy penalty on having too many false negatives.

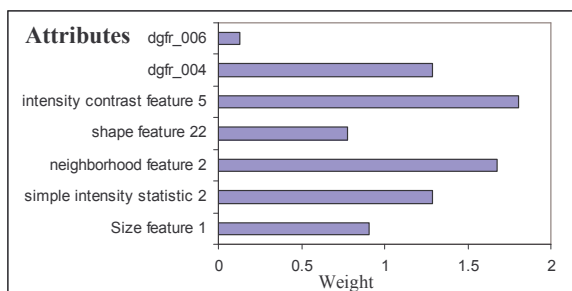


Figure 6 Optimized Attribute Weights

The above graph shows optimized weights for the selected attributes for the NPV task(3). Attribute names are based on the definitions given with the data[15]. Attributes with zero weights at the end of the optimization run were omitted from the above graph. These parameters optimized with the training data were able to achieve a 100 % NPV with acceptable true negative percentages for the given test data set. In other words eliminate all false negatives while predicting candidates who are not sick. The values in the above graph and the table were used for the final NPV classification task is based on the training data set and the cost function and cannot be assumed to be global for all situations. For a given training data set once the values are optimized they can be repeatedly used on unknown test data until the training data set is changed.

5. Related Work

P-tree is one form of data representation for vertical data mining. Most of the vertical data representations were developed in an attempt to do scalable Association Rule Mining by using an alternative compressed data structure. Initial ARM work as it is with any other data mining approaches designed algorithms to use horizontal data structures. In most cases the data was always available in horizontal format. With the increase of the dataset sizes researchers started to look for alternative scalable approaches. One of the earliest ARM work to suggest vertical data representation is MaxClique, which uses a vertical transaction id list instead of a typical horizontal item list [24]. Later, VIPER attempts to improve most of the problems in previous vertical data representation

approaches by suggesting a compressed bit vector in place of the transaction id list [24]. In VIPER a novel encoding scheme is used to achieve run length like compression. Runs of '1's and '0's are divided into two groups of size W_0 and W_1 and each full group is represented by setting one bit to 1. The last partial group holds the count of either 1's or 0's with a 0 bit in between the full groups and the partial counts. The length of the count field is fixed to enable the identification of '1' and '0' blocks that alternate. This work suggests the use of $W_0 = 256$ and $W_1 = 1$ expecting large runs of 0 blocks. It is argued that for support levels of less than 0.02% the data representation does not compress compared to the original data representation as a transaction id list [16]. VIPER also converts the compressed bit vectors back to a transaction id list in memory to do the computations such as set intersection to find the support of two item lists. The required AND operation is done as a classical database join.

Another major development in the area of vertical data mining with compressed data is suggested in Diffsets for scalable vertical Association Rule Mining [16]. Though it cannot be easily generalized as a compressed vertical data structure for other data mining applications we present it here for completion. Diffsets only keep the differences of the transaction id lists from the generating frequent pattern. When constructing frequent pattern item lists by combining lists, the algorithm only keeps track of the transaction id differences from the generating patterns as Diffsets. This works since each pattern class is totally independent, in the sense that it has a list of all possible items, and their transaction id lists, that can be combined with each other to produce all frequent patterns sharing a pattern class prefix. Diffsets achieve major reduction in size up to 10^{-6} for dense data sets. But for sparser data sets it is larger than the original vertical data and suggests the use of original transaction id lists or Diffsets based on data set characteristics [16]. Diffsets do not use any bit vectors and intersection is done as a join operation.

P-tree is a compressed vertical data structure that can be used on any type of data mining application with any type of data to achieve scalability with respect to data set size in cardinality and dimensionality. P-tree based algorithms use compressed vertical bit vectors to represent data. The same bit vectors in the stored format are used to do computations such as AND and OR operations. P-tree data structure also provides other advantages that enable algorithms to efficiently increase the quality of the results. In this work we were able to use a GA that requires multiple evaluations for parameter optimization due to computational efficiency provided by the P-tree vertical data structure.

6. CONCLUSIONS AND FUTURE WORK

In this paper we show a successful approach to a CAD classifier. The approach takes into consideration typical CAD training data set characteristics such as large and extremely unbalanced positive and negative classes, noisy class labels, large number of machine generated descriptive features with irrelevant and redundant features. The approach also takes into consideration the requirement to have extremely high performance thresholds in proposed systems in order to achieve clinical acceptance. The approach described in this paper was successfully used to submit the winning entry for the NPV task at the ACM 2006 KDD Cup data mining competition tasked to develop an error tolerant CAD system.

In this CAD classifier, variant forms of Nearest Neighbor Vote based classification and Local Decision Boundary based classification were combined with a genetic algorithm for parameter optimization and a vertical data structure (Predicate Tree or P-tree) for efficient processing. The use of a vertical data structure (P-tree) and an efficient reduction functional (Total Variation) to identify neighbors, provide acceptable computation times for this approach.

The final optimized predictor can be used as a quick decision support system. If new medical data is made available the optimization process can be repeated offline to arrive at a new optimized classifier. Further research and testing is required to produce a CAD system that meets clinical acceptance to provide value to physicians in their day-to-day practice.

7. ACKNOWLEDGMENTS

We acknowledge Dr. Terran Lane for the design and conduct of the ACM 2006 KDD Cup data mining competition. His valuable insights and suggestions regarding the data set and the application environment are appreciated.

8. REFERENCES

- [1] Abidin T., and Perrizo W.. SMART-TV: A Fast and Scalable Nearest Neighbor Based Classifier for Data Mining. Proceedings of the 21st ACM Symposium on Applied Computing , Dijon, France, April 2006.
- [2] Abidin, T., Perera, A., Serazi, M., Perrizo, W., Vertical Set Square Distance: A Fast and Scalable Technique to Compute Total Variation in Large Datasets, CATA-2005 New Orleans, 2005. Q. Ding, M. Khan, A. Roy, and W. Perrizo, The P-tree Algebra, Proceedings of the ACM Sym. on App. Comp., pp. 426-431, 2002.
- [3] Bandyopadhyay, S., and Muthy, C.A., Pattern Classification Using Genetic Algorithms. Pattern Recognition Letters, Vol. 16, (1995) 801-808.

- [4] Cost, S. and Salzberg, S., A weighted nearest neighbor algorithm for learning with symbolic features, *Machine Learning*, 10, 57-78, 1993.
- [5] DataSURG, P-tree Application Programming Interface Documentation, North Dakota State University. <http://midas.cs.ndsu.nodak.edu/~datasurg/ptree/>
- [6] Ding, Q., Ding, Q., Perrizo, W., "ARM on RSI Using Ptrees," *Pacific-Asia KDD Conf.*, pp. 66-79, Taipei, May 2002.
- [7] Duch W, Grudzi 'N.K., and Diercksen G., *Neural Minimal Distance Methods.*, World Congress of Computational intelligence, May 1998, Anchorage, Alaska, IJCNN'98 Proceedings, pp. 1299-1304.
- [8] Goldberg, D.E., *Genetic Algorithms in Search Optimization, and Machine Learning*, Addison Wesley, 1989.
- [9] Guerra-Salcedo C., and Whitley D., Feature Selection mechanisms for ensemble creation: a genetic search perspective, *Data Mining with Evolutionary Algorithms: Research Directions – Papers from the AAAI Workshop*, 13-17. Technical Report WS-99-06. AAAI Press (1999).
- [10] Jain, A. K.; Zongker, D. Feature Selection: Evaluation, Application, and Small Sample Performance. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, Vol. 19, No. 2, February (1997)
- [11] Khan M., Ding Q., Perrizo W., k-Nearest Neighbor Classification on Spatial Data Streams Using P-trees, *Advances in KDD*, Springer Lecture Notes in Artificial Intelligence, LNAI 2336, 2002, pp 517-528.
- [12] Khan, M., Ding, Q., and Perrizo, W., K-Nearest Neighbor Classification of Spatial Data Streams using P-trees, *Proceedings of the PAKDD*, pp. 517-528, 2002.
- [13] Krishnaiah, P.R., and Kanal L.N., *Handbook of statistics 2: classification, pattern recognition and reduction of dimensionality*. North Holland, Amsterdam 1982.
- [14] Kuncheva, L.I., and Jain, L.C.: *Designing Classifier Fusion Systems by Genetic Algorithms*. *IEEE Transaction on Evolutionary Computation*, Vol. 33 (2000) 351-373.
- [15] Lane, T., *ACM Knowledge Discovery and Data Mining Cup 2006*, <http://www.kdd2006.com/kddcup.html>
- [16] M.J. Zaki, K. Gouda, *Fast Vertical Mining Using Diffsets*, Special Interest Group in Knowledge discovery and Data Mining (SIGKDD), Washington DC, August 2003.
- [17] Martin-Bautista M.J., and Vila M.A.: *A survey of genetic feature selection in mining issues*. *Proceeding Congress on Evolutionary Computation (CEC-99)*, Washington D.C., July (1999) 1314-1321.
- [18] Perera, A., Abidin T., Serazi, M. Hamer, G., Perrizo, W., *Vertical Set Square Distance Based Clustering without Prior Knowledge of K*, 14th International Conference on Intelligent and Adaptive Systems and Software Engineering (IASSE'05), Toronto, Canada, 2004.
- [19] Perera, A., Denton A., Kotala P., Jockheck W., Valdivia W., Perrizo W., *P-tree Classification of Yeast Gene Deletion Data*, *SIGKDD Explorations*, Volume 4, Issue 2, December 2002.
- [20] Punch, W.F. Goodman, E.D., Pei, M., Chia-Shun, L., Hovland, P., and Enbody, R., *Further research on feature selection and classification using genetic algorithms*, *Proc. of the Fifth Int. Conf. on Genetic Algorithms*, pp 557-564, San Mateo, CA, 1993.
- [21] Rahal, I. and Perrizo, W., *An Optimized Approach for KNN Text Categorization using P-Trees*. *Proceedings of ACM Symposium on Applied Computing*, pp. 613-617, 2004.
- [22] Raymer, M.L. Punch, W.F., Goodman, E.D., Kuhn, L.A., and Jain, A.K.: *Dimensionality Reduction Using Genetic Algorithms*. *IEEE Transactions on Evolutionary Computation*, Vol. 4, (2000) 164-171
- [23] Serazi, M. Perera, A., Ding, Q., Malakhov, V., Rahal, I., Pan, F., Ren, D., Wu, W., and Perrizo, W., *ACM SIGMOD*, Paris, France, June 2004.
- [24] Shenoy, P. Haritsa, J. R., Sudarshan, S., *Turbo-charging Vertical Mining of Large Databases*, *International Conference in Management of Data*, May 2000.
- [25] Vafaie, H. and De Jong, K.: *Robust feature Selection algorithms*. *Proceeding of IEEE International Conference on Tools with AI*, Boston, Mass., USA. November. (1993) 356-363.

About the authors:

William Perrizo is the data mining Guru at North Dakota State University with a US patent on his novel data mining data structure P-tree and several other patents pending. He is a senior Professor in Computer Science. He leads the Data Systems Users and Research Group (DataSURG) involved in pioneering innovative research on scalable data mining using vertical data structures in the Computer Science Department at NDSU. He has multiple publications in prestigious journals, books and conferences. He is currently working on a book on vertical data mining. Under his direction DataSURG has deployed DataMIME™, a vertical suite of data mining tools for distributed data mining. DataSURG research has been supported by NSF, NASA, DARPA, and GSA. He is also involved in leading collaborative data mining research at the Center for High Performance Computing (CHPC) at NDSU. (www.cs.ndsu.edu/~perrizo)

Amal Perera is a doctoral candidate in computer science at NDSU. His research interests include scalable data mining with vertical data structures and software engineering. He is a major contributor to the design, implementation, and deployment of DataMIME™. He is also a lecturer at the Department of Computer Science and Engineering at the University of Moratuwa, Sri Lanka, on study leave to complete his doctorate at NDSU. (www.cs.ndsu.edu/~perera)