

Access to this work was provided by the University of Maryland, Baltimore County (UMBC) ScholarWorks@UMBC digital repository on the Maryland Shared Open Access (MD-SOAR) platform.

Please provide feedback

Please support the ScholarWorks@UMBC repository by emailing [scholarworks-group@umbc.edu](mailto:scholarworks-group@umbc.edu) and telling us what having access to this work means to you and why it's important to you. Thank you.

# An Efficient Method for Discretizing Continuous Attributes

*Kelley M. Engle, University of Maryland Baltimore County, USA*

*Aryya Gangopadhyay, University of Maryland Baltimore County, USA*

---

## ABSTRACT

*In this article the authors present a novel method for finding optimal split points for discretization of continuous attributes. Such a method can be used in many data mining techniques for large databases. The method consists of two major steps. In the first step search space is pruned using a bisecting region method that partitions the search space and returns the point with the highest information gain based on its search. The second step consists of a hill climbing algorithm that starts with the point returned by the first step and greedily searches for an optimal point. The methods were tested using fifteen attributes from two data sets. The results show that the method reduces the number of searches drastically while identifying the optimal or near-optimal split points. On average, there was a 98% reduction in the number of information gain calculations with only 4% reduction in information gain.*

*Keywords:* Continuous Attributes, Decision Tree, Discretization, Golden Search, Hill Climbing, Information Gain

---

## INTRODUCTION

Data mining is an active area of research that has found wide usage for knowledge discovery in databases in application areas ranging from assessment of loan applications to screening satellite images (e.g., Bagui, 2006; Han & Kamber, 2006; Hand et al., 2001; Tzanis et al., 2007; Witten & Frank, 2005). Many data mining algorithms, including decision trees and classification rules, require discretization when predictor attributes are continuous. Dis-

cretization refers to the process of converting the values of a continuous variable in two or more bins, the boundaries of which are referred to as split points. While there are many ways that this binning can be performed, the quality of binning can have a significant impact of the performance of the algorithm.

Discretization can be either static or dynamic. In static discretization all continuous variables are discretized at the outset of the learning algorithm after which the discretized continuous variable is treated like a discrete variable. In dynamic discretization a continuous variable is discretized while the model is being

DOI: 10.4018/jdwm.2010040101

built. Hence, the discretization process may be repeated several times for each continuous attribute over the model building process. Many decision tree-based learning systems such as ID3 (Quinlan, 1986) C4.5 (Quinlan, 1993) and CART (Breiman & Friedman et al., 1998) are based on dynamic discretization. The most commonly used method for discretization (Holte, 1993) is to compare a measure such as information gain for every possible value of a predictor attribute, which we refer to as “brute force” and is clearly not feasible in large data sets. While a number of approaches for discretization of continuous attributes have been developed in the literature, most of them require a significant amount of time for the discretization process or result in loss of predictive accuracy. The only definitive study in this area shows that the optimal split points (referred to as cut points in Fayyad & Irani (1992)) occur at the boundaries where the values of the target or class attribute changes value. However, in very large datasets, the number of such boundary points can also be quite large, which does not solve the problem of having to apply a “brute force” method to the boundary points in trying to find the optimal split point for a continuous attribute.

There are many examples of very large datasets in today’s world where large volumes of routinely collected data require efficient methods for analysis. One such example is the astronomical data in the Sloan Digital Sky Survey (2008) that consists of 9TB of images and 3.6 TB of data collected in an effort to map a quarter of the entire sky. Another example is the Baruch Options Data Warehouse (2008) at the Subotnik Financial Services Center that contains trades, quotes, open interest, and end of day volumes for all options series on US equities and indexes with a total data size of 60TB as of March 2008. Current data mining algorithms need to be scalable to handle such large volumes of data.

In this article we propose a novel approach for finding optimal split points for discretizing continuous attributes. The method is based on univariate parameter optimization and uses a combination of golden section search method

(Press et al., 2007) and gradient descent based on Newton-Raphson method (Hand et al., 2001). The predictive accuracy is measured in terms of information gain. We have shown through empirical tests using a number of datasets of varying sizes that the method would identify the optimal or near optimal solutions and has an extremely high degree of computational efficiency by a drastic reduction in the number of points for which information gain is computed and compared. The rest of the article is organized as follows. We briefly present the related work in the next section, followed by the methodology, experimental results, and conclusions.

## RELATED WORK

There are a number of different methods of discretization for continuous attributes. Dougherty et al. (1995) present three ways of classifying discretization: (1) global vs. local; (2) supervised vs. unsupervised and; (3) static vs. dynamic. Alternatively, Liu et al. (2002), present a hierarchical framework to describe the various discretization methods. Their framework decomposes the methods first by merging vs. splitting and then each of those categories is further broken down into supervised vs. unsupervised.

Global discretization refers to the process by which split points are determined prior to applying a machine learning algorithm. Local discretization is a discretization method that is embedded in an algorithm such as a decision tree and will reiterate numerous times for each attribute (at each level of the decision tree). Table 1 summarizes all the discretization methods discussed in this section by global/local and supervised, unsupervised and hybrid. This table is adapted from Table 1 in Dougherty and Kohavi et al. (1995) – it differs in that we have added an additional category for hybrid methods.

Supervised discretization involves any discretization process that uses the class label as part of the partitioning logic. Unsupervised discretization does not use the class label—these

Table 1. Summary of discretization methods

	Global	Local
<b>Supervised</b>	1RD (Holte 1993) Entropy-based discretization (Fayyad and Irani 1992; Fayyad and Irani 1993; Ting 1994) Hierarchical discretization (Chiu, Cheung et al. 1990) D-2 (Catlett 1991) Entropy with MDL (Pfahring 1995) Predictive Value Maximization (Weiss, Galen et al. 1990) ChiMerge (Kerber 1992) StatDisc (Richeldi and Rossotto 1995) Zeta (Ho and Scott 1997)	C4.5 (Quinlan 1993) Vector Quantization (Kohonen 1989) Entropy-based discretization (Fayyad and Irani 1992; Fayyad and Irani 1993)
<b>Unsupervised</b>	Equal Width Equal Frequency Gaussian Approximation/Uniform/K-tile (Chickering, Meek et al. 2001) Self-Organizing Maps (Vannucci and Colla 2004)	k-means clustering
<b>Hybrid</b>	Adaptive Quantizers (Chan, Batur et al. 1991) Maximal Marginal Entropy (Wong and Chiu 1987) MCC (Van de Merckt 1993)	

\*Adapted from Table 1 in Dougherty, Kohavi et al. (1995), p. 3.

methods such as equal width/equal binning simply partition the predictor attribute into  $k$  number of bins as specified by the user. The supervised and unsupervised discretization methods, as described in literature, will be discussed in more detail in subsequent sections.

Static discretization methods “perform one discretization pass of the data for each feature and determine the value of  $k$  for each feature independent of other features (Dougherty & Kohavi et al., 1995, p. 2).” Alternatively, dynamic discretization methods “conduct a search through the space of possible  $k$  values for all features simultaneously, thereby capturing interdependencies in feature discretization (Dougherty & Kohavi et al., 1995, p. 2).”

There are also a number of discretization methods that could be described as hybrids. They combine both unsupervised and supervised partitioning in determining split points. These methods will be discussed in more detail in a later section.

## Supervised Discretization

Entropy-based discretization can be found in a number of papers including those by Fayyad and Irani (1993), Ting (1994) and Pfahring (1995). This discretization process uses the entropy and information gain calculations to determine the optimal split point for each continuous attribute. There have been a number of papers that have worked to improve the efficiency of entropy-based discretization, including proving that the optimal split point for a continuous attribute will also be at the class label boundary (Fayyad & Irani, 1992) and will further reduce the number of calculations by utilizing the MDL (Minimal Description Length) (Friedman & Goldszmidt, 1996). Some other examples of entropy-based methods include hierarchical discretization, which seeks to maximize the Shannon entropy (Chiu & Cheung et al., 1990), as well as a method by Pfahring (1995), which uses entropy to initially identify split points and then uses a MDL (Minimum Description Length) heuristic to determine the best discretization. The D-2 algorithm by Catlett in 1991 utilizes a number

of different stopping criterion to discover the best split. These stopping criterion include: (1) minimum number of samples in a partition; (2) a maximum number of partitions and; (3) a minimum information gain. Entropy-based discretization can be used locally or globally.

*Zeta*, as presented by Ho and Scott (1997), is a measure of association that is utilized in discretization. It is based upon the *lambda* measurement which measures the strength of association between nominal variables and specifically “measures the proportionate reduction in prediction error that would be obtained by using one variable to predict another, using a modal value prediction strategy in all cases (Ho & Scott, 1997, p. 4).” The *zeta* measure was developed to address deficiencies in the *lambda* measure due to conditions where two variables may have modal values of zero. This was accomplished by pairing each value of the independent variable with the value of the class label (Ho & Scott, 1997).

1RD is a simple supervised method in which it creates a 1-level stump and is bundled with the 1R algorithm by Holte (1993). The Predicative Value Maximization is an algorithm that seeks split points that are most likely to make correct classification decisions (Weiss & Galen et al., 1990). Both 1RD and the Predicative Value Maximization are considered global methods.

Some examples of local supervised methods include: (1) C4.5 (Quinlan, 1993), (2) Vector Quantization (Kohonen, 1989) and, (3) Some entropy-based methods (Fayyad & Irani, 1993). C4.5 is a decision-tree algorithm, which includes discretization of a continuous attribute at each level of the decision tree (Quinlan, 1993) (Quinlan, 1996). Vector Quantization attempts to “partition an N-dimensional space into a *Voronoi Tessellation* and then represent the set of points in each region by the region into which it falls (Dougherty & Kohavi et al., 1995, p. 4).” Because this method creates local regions, it is considered a local method. Entropy-based methods, as described previously, can also be utilized as a local discretization method.

There are also some statistical methods that can be used to discretize continuous attributes.

ChiMerge, by Kerber, can be described as bottom-up approach where the data is initially partitioned and then the method will perform a  $X^2$  test to determine where bins should be merged (Kerber, 1992). Similarly, Stat Disc, by Richeldi and Rossotto (1995), employs a comparable method in that it will initially partition the attribute, but then will apply the  $\Phi$  measure to merge the bins.

## Unsupervised Discretization

Equal width and equal frequency are two of the simplest unsupervised discretization methods. Equal width binning partitions the data into  $k$  number of bins as defined by the user – then the minimum and maximum data value are added together and divided by  $k$ . Equal frequency binning will consider the total number of data points in the continuous attribute and divide it by  $k$ . This method ensures that the bins will have an even number of data points within each partition. The one major limitation to equal frequency binning is that the range can become skewed by outliers (Catlett, 1991).

There have been a number of extensions to the equal width and equal frequency methods. For instance, there are three methods, presented by Chickering and Meek et al. (2001) that utilize the concept of quantile binning: (1) Gaussian approximation, (2) Uniform approximation and, (3) K-tile method. The Gaussian approximation method utilizes an assumed standard distribution and applies the mean and standard deviation to partition the attribute. Likewise, uniform approximation, assumes the continuous attribute is distributed evenly between its minimum and maximum value. The k-tile method is also a quantile approach but uses the empirical distribution function.

Clustering is another method used for unsupervised discretization. An example of clustering was presented by Vannucci and Colla (2004) for application in association rule mining. The continuous attributes are discretized using a SOM (self-organizing map). This method addresses the weaknesses in equal width/equal frequency including: (1) discretization should

account for the original distribution of the attribute; (2) the discretized attributes should not hide patterns and; (3) the intervals should be meaningful to the user (Dougherty & Kohavi et al., 1995).

## Hybrid Discretization

Hybrid discretization refers to those methods that combine supervised and unsupervised methods. For example, adaptive quantizer combines binary equal width partitioning with classification rule measures (Chan & Batur et al., 1991). The method begins by performing a binary equal width split—it then proceeds to run a classification algorithm to collect the predictive accuracies of the splits. The partition with the lowest accuracy is then split again by the binary equal width method. This process will be performed until some minimum predictive accuracy is obtained.

Another hybrid method, called maximal marginal entropy, initially employs the unsupervised method, equal frequency, but then will make modifications to the boundaries based on the entropy calculations. The purpose of these modifications is to decrease the entropy for the final split (Wong & Chiu, 1987).

The last hybrid method to be discussed is the Monothetic Contrast Criteria (MCC), which is a combination of two methods, clustering and entropy-based discretization (Van de Merckt, 1993). The first method utilizes a clustering algorithm and is therefore unsupervised. The second method employs the (supervised) entropy calculation to divide the previously partitioned data.

In this article we propose to extend the current state of the art by describing a method for dynamic discretization in large data sets.

## Scalability Issues in Decision Tree Induction

There is an abundance of research in the area of scalability for various data mining algorithms including association rules (Zaki, 2000), clustering (Ng & Han, 2002), classification (Kamber et

al., 1997), decision trees (Catlett, 1991; Gehrke et al., 1999; Gehrke et al., 2000; Li, 2005) and neural nets (Charles, 2004). However, for the purposes of our research, we are most concerned with scalability approaches for decision tree induction since our method is designed to alleviate the problem of discretizing continuous attributes. The issue of scalability has long been associated with decision tree induction since it is considered an NP-complete problem (Hyafil & Rivest, 1976; Quinlan, 1993). In particular, discretizing continuous attributes at each level of the decision tree makes the building operation quite expensive. In this section we will delve more deeply into the research as it relates directly to decision trees.

The need for scalability in data mining algorithms lies in the theory that working with larger datasets is preferable since the accuracy of any data mining task (including classification) will be improved by using large datasets. This theory has been studied and corroborated by a number of researchers including Catlett (1991) and Chan and Stolfo (1993a, 1993b). Scalability is defined by Ganti and Gehrke et al. (1999) as “given a fixed amount of main memory, its runtime increases linearly with the number of records in the input database” (p. 38).

There have been a number of approaches proposed to address the issue of scalability in data mining algorithms. Some of these methods, such as SLIQ (Mehta & Agrawal et al., 1996), SPRINT (Shafer & Agrawal et al., 1996), Rainforest (Gehrke & Ramakrishnan et al., 2000) and PUBLIC (Rastogi & Shim, 2000), attempt to address scalability issues by directly tackling the problem of main memory issues. Decision tree algorithms operate on the premise that a dataset must be able to fit into main memory. For large datasets, this is often not possible and so the methods mentioned previously address this deficiency.

For instance, SLIQ (Mehta & Agrawal et al., 1996) allows for both continuous and categorical data in its algorithm and uses a pre-sorting approach on disk resident data to address the main memory deficiency issue. As a result this method only sorts once instead of sorting



Table 2. Example of AVC-Group

Age	Buys PC	
	YES	NO
Youth	2	3
Middle-aged	4	0
Senior	3	2

data at each node. One criticism of the SLIQ method is that the performance will suffer if the class list cannot fit into main memory. SPRINT (Shafer & Agrawal et al., 1996), however, removes all main memory issues through the use of hash trees. However, this process may become expensive as the dataset grows due to the complex joins in the tree.

Rainforest (Gehrke & Ramakrishnan et al., 2000) uses AVC (attribute-value-count) sets which will list the counts for attributes into main memory. These AVC lists either be at the group (node) level (from here on called *AVC-group*) or at the individual attribute level (from here on called *individual AVC-sets*). Table 2 gives an example of an AVC-Group node table taken from Gehrke and Ramakrishnan et al. (2000). Gehrke and Ramakrishnan et al. (2000) describes three scenarios based on the amount of main memory: (1) The *AVC-Group* of the root in its entirety will fit into main memory; (2) An *individual AVC-set* can fit into main memory but not the *AVC-group* and; (3) none of the *AVC-Group* or *individual AVC-sets* of the root fit in main memory. A number of algorithms were developed to address the first two scenarios – the combination of which are summarily called Rainforest. For the last scenario the authors propose, in theory, a hybrid version of their work combined with the SPRINT algorithm.

Sampling methods are another avenue for addressing the scalability issues in the induction of decision trees. The basic premise of these methods is that a smaller sample of the original dataset is used to construct the decision tree. Although none of these methods (including our own) are optimal, they are still practical

methods when dealing with very large datasets such as the astronomy data described earlier which is now in the range of 15+ terabytes. As championed by Chaudhuri in 1998, sampling methods should not be ignored in the quest for scalable data mining solutions.

One such sampling method, BOAT (Gehrke & Ganti et al., 1999), utilizes a bootstrapping approach where the dataset is partitioned into smaller subsets that are able to fit into main memory. A number of decision trees are created on each of these subsamples – from  $t_1$  to  $t_i$  where  $i$  is equal to the number of subsets created. Each of the trees ( $t_1, \dots, t_i$ ) are analyzed and the final tree ( $T'$ ) constructed is usually a close approximation of the optimal tree.

Breiman and Friedman et al. (1998) discuss a number of sampling methods including *bootstrapping* and *one-shot sampling*. Many of these simple random sampling methods, as empirically tested by (Catlett, 1991), have significantly reduced accuracy rates. Windowing, developed by (Quinlan, 1983), is a sampling method designed to significantly decrease the time associated with decision tree induction by examining small samples from the training data (usually 10%-20%). The method will only add remaining records if the tree does not correctly classify. This method, in particular, is very susceptible to noisy data.

Catlett (1991) actually developed two methods – the first was a discretization method call D-2 that employed a static discretization process to speed up the induction of decision trees. The second method, called peepholing, is a sampling method that has two functions: (1) shortlisting and; (2) blinkering. The purpose of shortlisting is to eliminate attributes from

the evaluation criterion (such as information gain). Once an attribute is scratched, it no longer needs to be considered at any of the lower levels of the tree induction process. *Blinkering* uses a sampling technique on the remaining attributes to reduce the search space through the use of a pair of numbers that represent the left and right *blinkers*. These *blinkers* are initially set to infinitely large numbers to cover the entire range of the attribute but will be narrowed over time. The samples used in this process are referred to as *peeps* and this method is similar to windowing. Catlett's criticism of windowing (Quinlan, 1983) was that the performance decreases when data is "noisy." Although by the author's own admission, the *blinkering* method will only work on attributes that have one peak (i.e., well-behaved data).

Kamber and Winstone et al. (1997) proposed a heuristic-based 3-step approach to addressing scalability issues: (1) *attribute-oriented induction*; (2) *relevance analysis* and; (3) *multi-level mining*. With *attribute-oriented induction*, each continuous would be discretized prior to the decision tree induction through the use of *concept hierarchies*. These *concept hierarchies* must be supplied by domain experts, database administrators or using the database schema itself. *Relevance analysis* involves removing irrelevant or redundant attributes prior to the decision tree induction using a normalized version of information gain called the uncertainty coefficient. This process will run into the same efficiency issue with attributes that have a large number of distinct values and as a result it is suggested by the authors to simply remove such attributes if no *concept hierarchy* can be determined. This indiscriminate removal of attributes could lead to a decision tree that is far from optimal. The third step, *multi-level mining*, does not necessarily relate to efficiency but allows for decision trees to be induced at different levels of abstraction based on the concept hierarchy.

From the statistics community, two related algorithms were developed to address scalability issues in CART. The first algorithm, FACT (Loh & Vanichsetakul, 1988) calculates an ANOVA

F-statistic for each attribute at each node and thereby choosing the attribute with the largest F-statistic. LDA (linear discriminant analysis) is then applied to find the actual split. In 1997, Loh and Shih presented QUEST which performs similarly to FACT except that it yields only binary splits.

## METHODOLOGY

Our problem of finding the optimal split point is similar to that of the univariate parameter optimization problem (Hand et al., 2001) of finding the value of a parameter  $s$  so as to maximize the information gain function  $\Psi(s)$ . For continuous functions the local optima can be detected by applying differential calculus in a straight-forward manner. However, since we are dealing with a discrete set of points which is generated by an unknown function, optimization by taking derivatives is not possible. Our method for finding the optimal split point for continuous attributes is a two step process. The first step, which we call the Bisecting region Method (BRM), is based on the golden section search in one dimension (Press et al., 2007), and the second step is based on the gradient descent method applied to Newton-Raphson method for finding local optimum. In the first step we sort the parameter  $s$  and start with a bracket  $[s_1, s_2]$  that contains the optimum of the function  $\Psi(s)$ , where  $s_1$  and  $s_2$  are the lower and upper bounds of  $s$ . We check the midpoint  $s_m$  such that  $s_1 < s_m < s_2$ . If  $\Psi(s_m) > \Psi(s_1)$  and  $\Psi(s_m) > \Psi(s_2)$  then a local maximum exists between  $s_1$  and  $s_2$ . If that is not the case, then we create brackets  $[s_m, s_2]$  and  $[s_1, s_m]$  and repeat the above process. This step would find the maximum if the function  $\Psi(s)$  is well-behaved and free from local fluctuations. However, most information gain functions may not have these properties, hence we need a second step to avoid stopping at a suboptimal point.

The second step in our methodology, called Hill Climbing Algorithm (HCA), works on the following principles based on Newton-Raphson method for gradient descent. The challenge is



to find a close enough starting point so that the number of calculations can be minimized. We rely on the first step of our methodology (BRM) for identifying the starting point. Since we cannot derive the first and second derivative for the unknown functional form for an attribute with an unknown information gain function, we use the gradient descent approach to approximate the Newton-Raphson method. Instead of using the first derivative we can use the gradient information to determine the correct direction to move. Since we have applied our method to one attribute at a time, there are only two directions to choose from. In our methodology we choose the direction of positive ascent. If the information gain increases in both directions then both directions are traversed. The algorithm stops when the gradient becomes zero, which represents a summit. If the traversal was bidirectional, the higher summit point is returned as the maximum.

In the following section we first describe the algorithms for the two steps in our methodology, followed by some illustrative examples.

## Algorithms

Let us assume that  $f$  is an attribute for which we are trying to determine the optimal split point for a given set of sorted values of  $f$ :  $S = \{x_p, \dots, x_n\}$ , where  $x_i < x_{i+1}$ . Our goal is to find out the point  $x^*$  that corresponds to the maximum information gain without exhaustively computing the information gain values for all points in  $S$ . Information gain is calculated as the difference in entropies before and after taking the attribute  $f$  into account, where entropy is calculated as

$H(X) = -\sum p_x \log p_x$  (Liu et al., 2002). At the outset we do not know the information gain for any of the points in  $S$ . Our first algorithm, called Bisecting Region Method (BRM), computes the information gain for three points in set  $S$ : the mid-point ( $mid$ ) between the lower and upper bounds of  $S$ , the mid-point of the right region, called  $mid\_right$ , which is calculated by averaging  $mid$  and the upper bound ( $max$ ), and that of the left region, called  $mid\_left$ , which is

calculated by averaging the lower bound ( $min$ ) and  $mid$ . If  $mid\_right$  is the point with the highest information gain the algorithm performs search on the region to the right of  $mid$ . Similarly, if  $mid\_left$  has the highest information gain, the algorithm proceeds with a search to the left of  $mid$ . Otherwise the algorithm stops and returns  $mid$ . At this point, the second algorithm, hill climbing algorithm (HCA) is invoked with the return value of BRM as an input parameter. HCA starts by searching both to the left and right of the initial point ( $s_i$ ) that is passed to it as a parameter by BRM. If the point to the left (and/or right) of  $s_i$  has a higher information gain than that of  $s_i$ , HCA continues its search. Otherwise it returns the point at which it has encountered the maximum information gain. The BRM and HCA algorithms are shown in Figures 1 and 2 respectively. The recursive function calls are shown for clarity and simplicity of presentation and can be implemented as iterations rather than recursions.

One of the features of the BRM algorithm is that it may cover only a small fraction of the points in  $S$ . While this reduces the computational time, it may not provide enough cover for  $S$ . We added a heuristic to force the BRM algorithm to go through more calculations by introducing two additional variables,  $SUM1$  and  $SUM2$ , where  $SUM1$  is the sum of the information gains of the  $min$  and  $mid\_left$ , and  $SUM2$  is the sum of the information gains of the  $mid\_right$  and  $max$ . This additional heuristic helps in identifying points of maximum information gains when they tend to appear on the left or right edges of the data. The methods are compared with detailed experimental results in Section **Results**.

## Datasets Description

The data for these experiments consist of: (1) *Forest cover* dataset<sup>1</sup> and; (2) *Ionosphere* dataset<sup>2</sup>. *Forest cover* was obtained by the U.S. Forest Service (USFS) and contains 54 attributes to capture information such as soil type, elevation, horizontal and vertical distances to water sources and more. It is a classification dataset where the variable *cover\_type* is the

Figure 1. Bisecting region method

**Algorithm 1: Bisecting Region Method**

**Input:**  $S$  = Sorted values of feature  $f$ , lower\_bound, upper\_bound of  $S$   
**Output:**  $s^* \in S$ , where  $s^*$  represents the point with the highest information gain found.

```

{
  mid  $\leftarrow$  (lower_bound+upper_bound)/2
  mid_left  $\leftarrow$  (lower_bound+mid)/2
  mid_right  $\leftarrow$  (mid+upper_bound)/2
  COMPUTE information gain  $g()$  for mid, mid_left, mid_right
  IF  $g(\text{mid}) > g(\text{mid\_left})$  and  $g(\text{mid}) > g(\text{mid\_right})$ 
    Return mid( $X$ )
  ELSE IF  $g(\text{mid\_left}) > g(\text{mid})$ 
    BRM (( $S_l \mid \forall s_i \in S_l, \text{lower\_bound} \leq s_i \leq \text{mid}$ ), lower_bound, mid) // Shift to region on left
  ELSE
    BRM (( $S_u \mid \forall s_j \in S_u, \text{mid} \leq s_j \leq \text{upper\_bound}$ ), mid, upper_bound) // Shift to region on right
  END IF

  Exceptions:
  IF  $g(\text{mid\_left}) = g(\text{mid}) = g(\text{mid\_right})$  Return mid
  IF  $g(\text{mid\_left}) = g(\text{mid\_right}) > g(\text{mid})$  Return mid_left or mid_right (random choice)
}
```

class label that designates the type of forest cover present – such as Spruce/Fir, Aspen, Douglas-Fir etc. *Forest cover* has a total of 581,012 records and for the purposes of these experiments we are using the 10 continuous attributes (the remainder of the attributes are binary variables). *Forest cover* was chosen not only due to the total number of records but also due to the high cardinality present in many of the attributes such as *h\_dist\_roadways* and *h\_dist\_fire\_points*. This high cardinality was desired in order to test the efficacy of the BRM/HCA method against continuous attributes with a large number of distinct values. The number of distinct values and the ranges for both datasets are shown in Table 3.

*Ionosphere* is another classification dataset that contains 34 continuous attributes and one class label that classifies the ionosphere as either ‘good’ or ‘bad.’ The data was collected by radar and consists of ‘pulse numbers’ which correspond “to the complex values returned by the function resulting from the complex electromagnetic signal (Asuncion & Newman, 2007).” The total number of records is 351 and the cardinality of each continuous attribute is in the range of 250. The predictability of the

cardinality is due to the consistent range across each attribute since each pulse has a min of -1 and max of 1.

### Illustrative Examples for the BRM/HCA Method

Two illustrations are presented to portray the process involved in discovering the global optimum using the BRM/HCA method<sup>3</sup>. We have chosen the following attributes to illustrate the BRM/HCA’s capability of ascertaining the global optimum: (1) *slope* and (2) *hillshade\_noon*, both from the dataset *forest cover*.

Figure 3 illustrates how the BRM/HCA method will determine the global optimum - this example utilizes the attribute *slope* from the *forest cover* dataset. The BRM will shift to the left after evaluating the three points: (1) *mid-point* (50); (2) *mid-left* (25) and; (3) *mid-right* (75). The BRM will then continue for another calculation as it calculates the information gain for the new *mid-left* and *mid-right* (denoted by 25(2) and 75(2) in Figure 3). At this point, since the stopping criterion has been met, the BRM stops and the high point is sent to the HCA to finish its ascent to the global optimum indicated by the star sign.

Figure 2. Hill climbing algorithm

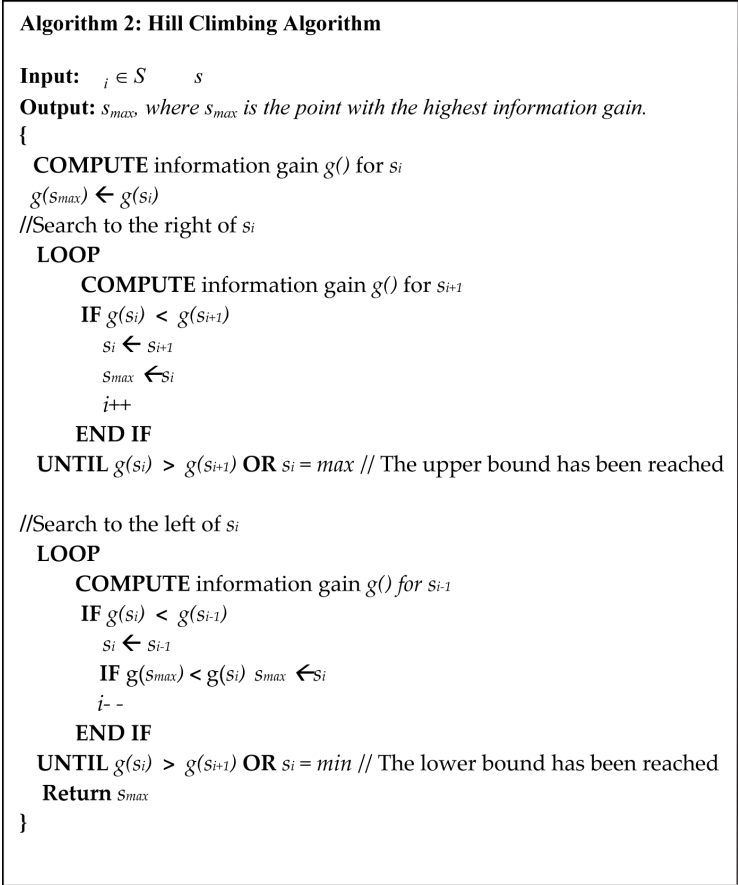


Figure 4 illustrates how the BRM/HCA method will determine the global optimum utilizing the attribute *hillshade\_noon* from the forest cover dataset. The BRM will shift to the right after evaluating the three points: (1) *mid-point* (50); (2) *mid-left* (25) and; (3) *mid-right* (75). The BRM will then continue for one more calculation as it calculates the information gain for the new *mid-left* and *mid-right* (denoted by 25(2) and 75(2) in Figure 4). At this point, since the stopping criterion has been met, the BRM stops and the high point is sent to the HCA to finish its ascent to the global optimum.

**Illustrative Examples for the BRM/HCA Method Utilizing the SUM Extension**

Figure 5 illustrates how the bisecting region method with the hill-climbing algorithm traverses a particular attribute to find the global optimum, in this case, *attr22* from the *ionosphere* dataset. There is an extra function as part of the SUM extension – called *firstpass*. The *firstpass* function is called once, prior to calling the BRM. Its main purpose is to attempt to *force* the BRM algorithm to iterate at least twice and aids in discovering global optima at the boundaries of an attribute.

Table 3. Range and discrete values by attribute

DATASET	Attribute	Range	NBR of DISCRETE VALUES
Forest Cover	Aspect	360	361
Forest Cover	Elevation	1,999	1,978
Forest Cover	Slope	66	67
Forest Cover	H_dist_roadways	7,117	5,785
Forest Cover	H_dist_hydrology	1,397	551
Forest Cover	V_dist_hydrology	774	700
Forest Cover	H_dist_fire_points	7,173	5,827
Forest Cover	Hillshade_9am	254	207
Forest Cover	Hillshade_noon	254	185
Forest Cover	Hillshade_3pm	254	255
Ionosphere	Attr1	2	219
Ionosphere	Attr6	2	260
Ionosphere	Attr20	2	265
Ionosphere	Attr22	2	264
Ionosphere	Attr32	2	263

The grey circles represent the initial calculation of the information gain for the *min*, *mid-left*, *mid-point*, *mid-right* and *max* data points. The information gain from the *min* and *mid-left* split points are then summed as depicted by *SUM1* in the figure. Respectively,

the information gain for the *mid-right* and *max* split points are summed, which is indicated by *SUM2*. At this point the *mid-point* is compared with *SUM1* and *SUM2*. In this particular case, *SUM1* is greater than *mid-point* and *SUM2* and

Figure 3. Example of finding global optimal

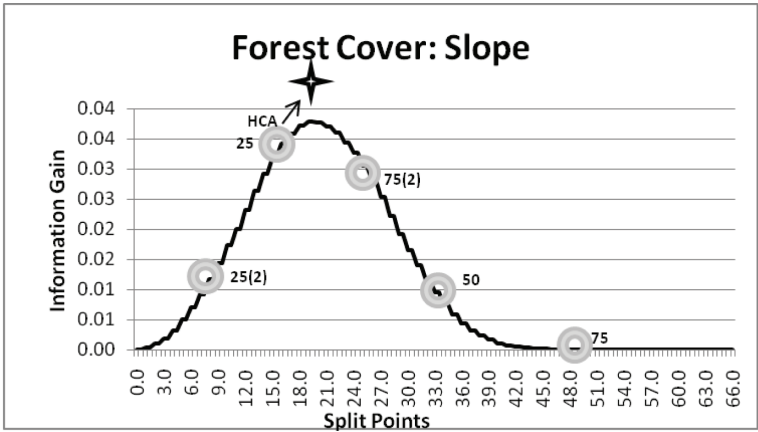
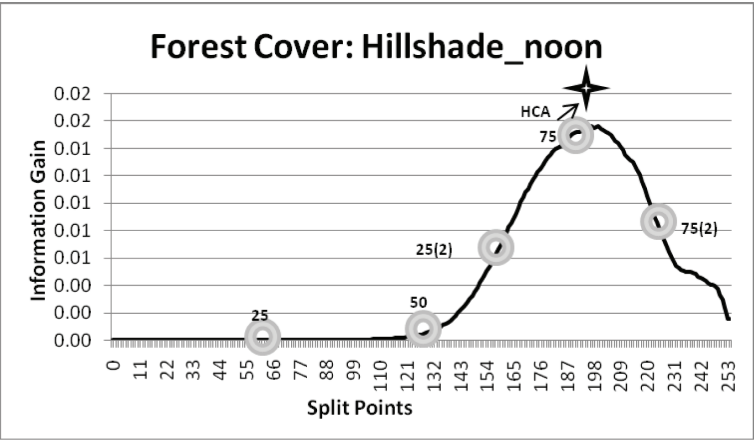


Figure 4. Example of finding global optimal



therefore the stopping criteria have not been met for the bisecting region method.

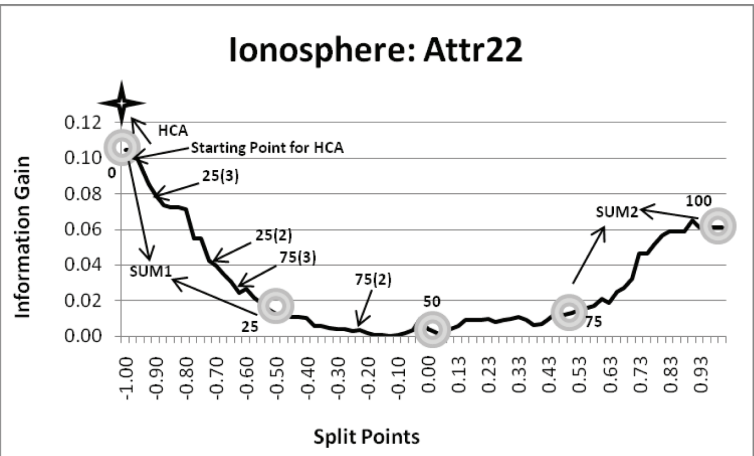
The processing will continue by shifting to the left, whereby the new *mid-left* and *mid-right* points will be determined – this is analogous to the shifting logic in the BRM. The *mid-left* point will become the new *mid-point* and based on this the new *mid-left* and *mid-right* points (denoted in the figure with 25(2) and 75(2)) are established. The information gain from these points are compared and as is visually obvious

in the figure, the *mid-left* point (shown in the chart as 25(2)) is the highest.

This process consecutively continues shifting to the left as it ascends towards the global optimum (due to space constraints all calculated data points are not shown in the figure). The BRM, for this particular attribute, actually concludes when the information gains for all three split points (*mid-point*, *mid-left*, and *mid-right*) are equal.

The final split point, which will become the starting point for the HCA, is in this case the

Figure 5. Example of finding global optimal with SUM extension



*mid-point* from the final round of processing of the BRM (indicated in Figure 5 as the starting point for the HCA). The HCA, will first search to the left and find that the information gain for the next data point is lower. It then proceeds to the right for two calculations until it begins its descent. At this time, the split point with the highest information gain, as returned by the BRM/HCA, is chosen as the final split point as denoted by the star in the figure.

Figure 6 illustrates another instance of how the BRM/HCA method will determine the global optimum. The example utilizes the attribute *Attr32* from the *ionosphere* dataset. As denoted in the previous example, the grey circles represent the first pass. The BRM will shift to the right after evaluating the three points: (1) *mid-point*; (2) *SUM1* and; (3) *SUM2*. The BRM will then continue to the right for three more calculations as it calculates the information gain for the new *mid-left* and *mid-right* split points (not all split points are indicated on Figure 6 due to space constraints).

The last calculation of the BRM resulted in equivalent information gain values for the *mid-left*, *mid-point* and *mid-right* values therefore the stopping criterion has been met and the *mid-point* split point is selected as the starting point for the HCA. The HCA examines the splits to the left and right of the starting point and correctly determines that the starting point is the global optimum.

## Experiments

The algorithms were implemented using PL/SQL on an Oracle 10g server.

### Brute Force Algorithm

Prior to comparing the results of the BRM/HCA with the *brute force* method, an explanation of the *brute force* algorithm will be briefly discussed. The *brute force* algorithm functions similarly to the discretization performed in Quinlan's C4.5 algorithm (Quinlan, 1993). In the *brute force* method, essentially the algorithm performs a sequential read of the sorted attribute.

Each time the discrete value in the attribute record changes, a new split point is determined as the midpoint between the two values. If  $n$  is the number of discrete values for a particular attribute, then the number of calculations for the *brute force* method will be  $n - 1$ .

## RESULTS

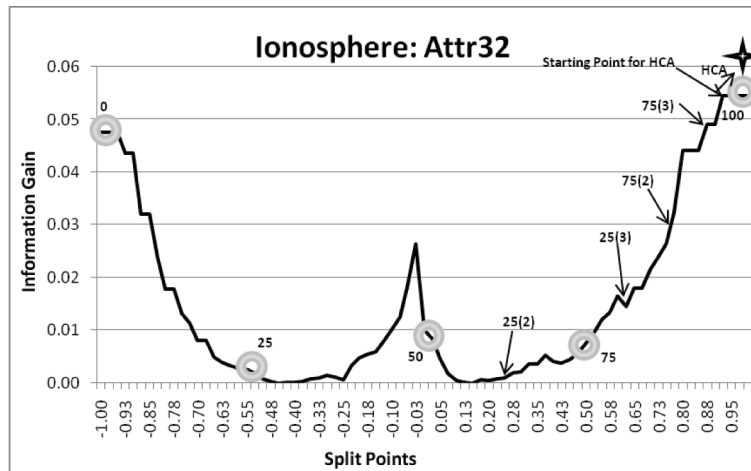
Figures 7-10 show the performance in terms of accuracy and efficiency for all three methods—*brute force*, *BRM/HCA* and *BRM/HCA with SUM* extension—and their relative differences. Figures 7 and 8 specifically detail the accuracy of both the *BRM/HCA* and *BRM/HCA with SUM* methods when compared to the benchmark (*brute force*). It is obvious from these two charts that the information gain for each of these methods is comparable to the *brute force*. The *BRM/HCA with SUM* performs somewhat better in the *ionosphere* dataset. Overall, in *ionosphere*, the *BRM/HCA* method was able to discover the global optimum 80% of the time. For the *forest cover* dataset, the global optimum was discovered 60% of the time. However, it is important to note that the other (local) optima discovered in both *forest cover* and *ionosphere* had information gain values that were not significantly different from the global.

Figures 9 and 10 illustrate the dramatic differences in the number of information gain calculations for all the methods. It was necessary to stack these values in order to show the relative differences between the methods. For *forest cover*, it is palpable how the *BRM/HCA* and *BRM/HCA with SUM* methods are dramatically more efficient than the *brute force* method. The *BRM/HCA* method is most efficient in datasets where a large range and high cardinality are present.

For *forest cover*, processing time became an important factor to consider due to the large number of records combined with widespread ranges in some of the attributes. On average, the processing time for the *brute force* algorithm was over 7 minutes. The average process time for the *BRM/HCA* was just under 33 seconds



Figure 6. Example of finding global optimal using SUM extension



(see Figure 11 for processing time listings by attribute/method). The times collected were conducted using an Oracle 10g server – obviously if run in main memory the times would be significantly faster. However, large data sets might not fit into main memory, and hence we chose to use a database management system. Also, the purpose of reporting these times is to show the *relative* differences rather than absolute processing time. For each individual attribute in *forest cover*, there is a significant amount of variation, particularly with the brute force algorithm. This is due to three factors: (1) the number of records in the dataset; (2) the range associated with the individual attribute and; (3) the number of discrete values for each attribute. For reference, the information for the range and number of discrete values for both *forest cover* and *ionosphere* can be found in the previous section in Table 3.

The results from the empirical tests comparing the *brute force* to the bisecting region method—with the *hill-climbing algorithm*—demonstrate that the BRM/HCA method and its SUM extension can find the (near) optimal split point and do so with noticeably less calculations. To report accuracy first, the average reduction (over both methods) in information

gain was 4% with a 98% reduction in the number of calculations.

## The Issue of Cost

The computational complexity of the brute force algorithm for discretization is  $O(n)$  where  $n$  is the number of distinct values per an attribute, which can be very large in applications such as the Sloan Digital Sky Survey data or the Baruch Options data warehouse discussed in the Introduction section of this article. The brute force algorithm – albeit optimal – is not a feasible solution for such datasets. The problem becomes worse in decision tree induction, where discretization is required at every node (i.e., local discretization as in C4.5) and this operation is therefore multiplied by the number of levels produced by the decision tree (though  $n$  is decreasing at each level).

The BRM/HCA discovers the global optimum in well-behaved datasets. However, when data is noisy the method may stop after discovering a local optimum. This is also true for other sampling methods such as Catlett's (1991) peephaling. However, for the empirical tests conducted, the local optima discovered were very close to the global. For example, the attribute *v\_dist\_hydrology* had an optimal split where the information gain was equal to .0079.

Figure 7. Accuracy comparison by method for forest cover

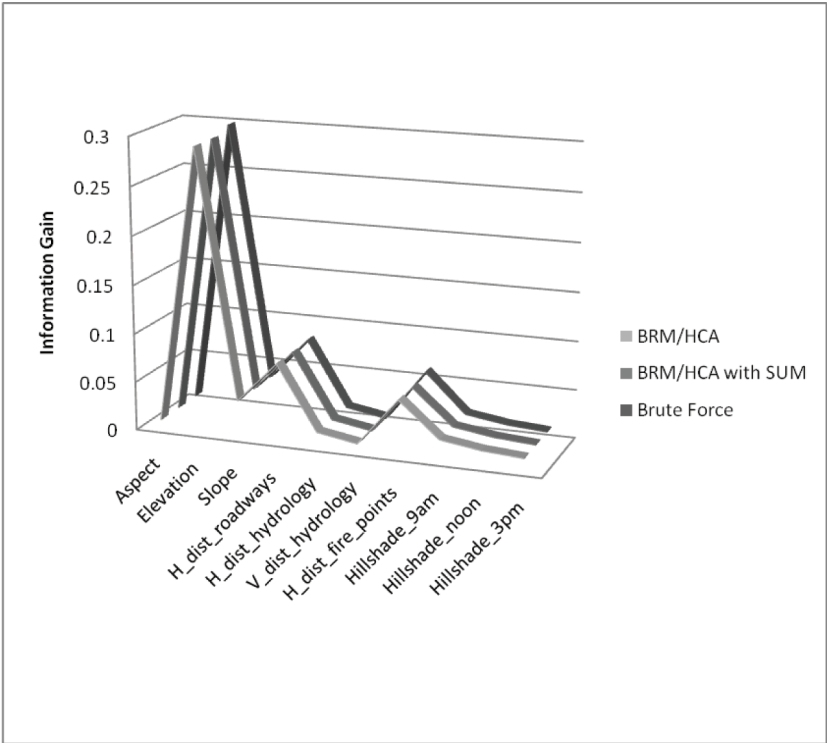


Figure 8. Accuracy comparison by method for ionosphere

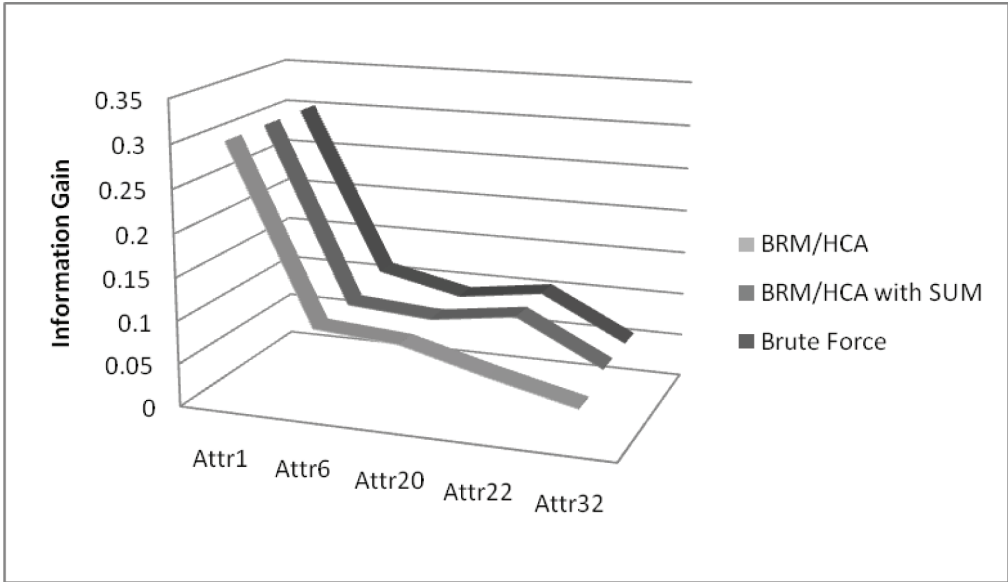
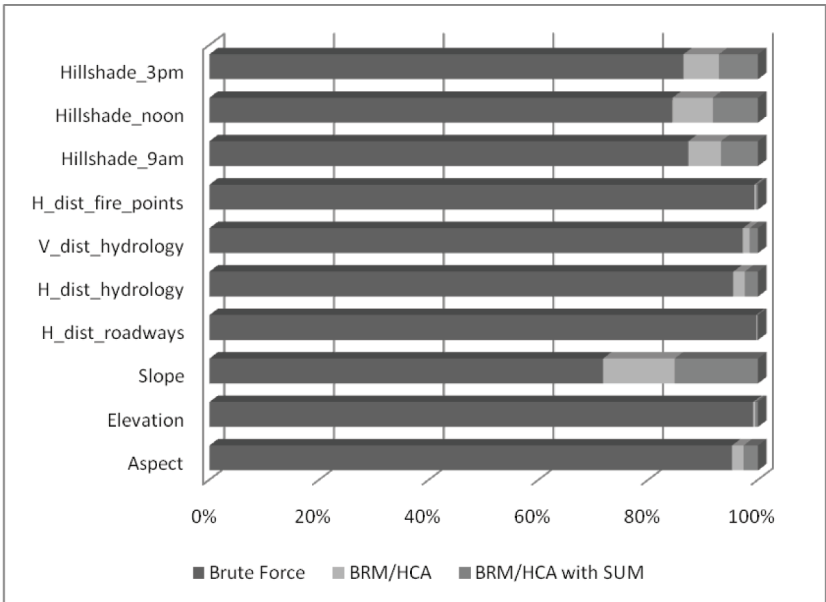


Figure 9. Efficiency comparison by method for forest cover



The BRM/HCA method find a split point where the information gain was equal to .0078 – or 2% less than the (global) optimal split. Figure 12 shows the local optimum discovered for

v\_dist\_hydrology by the BRM/HCA method in relation to the global optimum. The BRM/HCA method in this case, has in fact, stopped at a local optimum.

Figure 10. Efficiency comparison by method for ionosphere

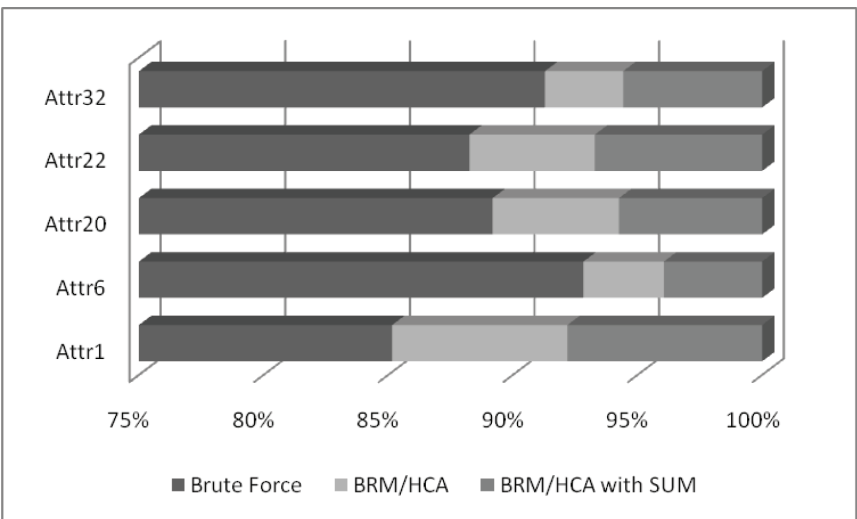
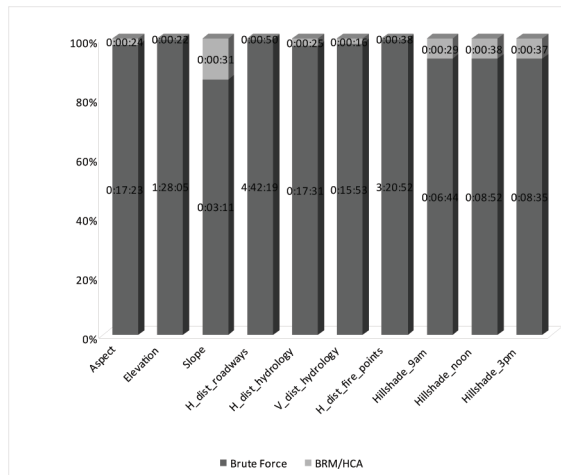


Figure 11. Processing time comparison (hours:minutes:seconds)



## Comparison with Fayyad and Irani's Class Boundary Changes

Fayyad and Irani (1992) established that the optimal split point for any attribute will always occur where the class boundary label changes. Consequently, significantly less information gain calculations need to be performed since only where there is a change in the class label does the information gain need to be examined. In regards to efficiency, the BRM/HCA method, for the datasets used in this empirical validation, outperforms the class label boundary changes. In Figure 13, we have compared our results from the BRM/HCA method as well as the SUM extension with Fayyad and Irani's theory using the *ionosphere* dataset. The comparison indicates that the BRM/HCA method is superior in terms of efficiency due to the fact that our method is examining approximately 75% fewer number of potential split points than the class boundary changes approach.

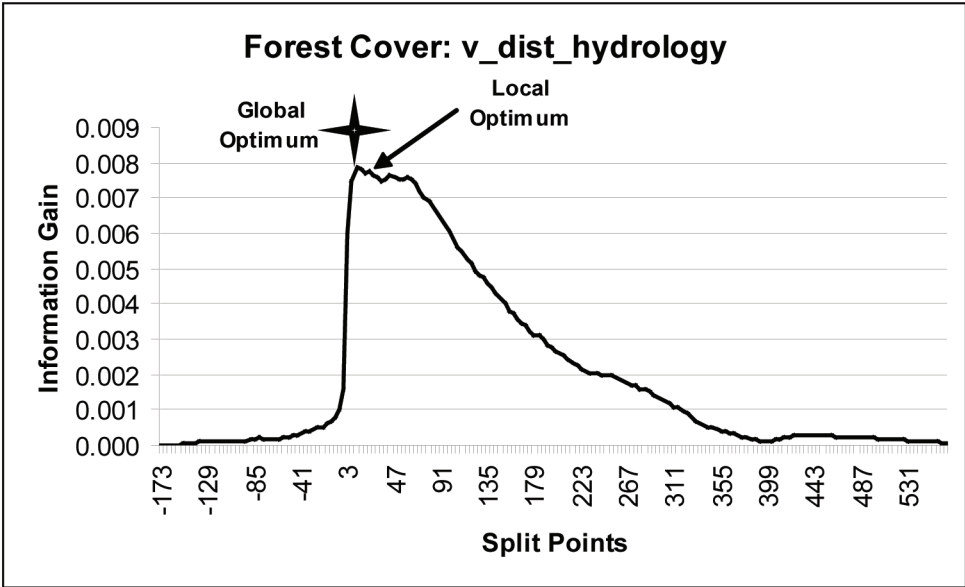
## CONCLUSION

In this article we describe a new method (BRM/HCA) for finding optimal split points in continuous attributes. The BRM/HCA method has

been empirically tested to find the global (or near-global) optima in a number of datasets although we have only presented a few of those results in this article. The necessity for the addition of the SUM extension was determined through empirical testing. However, finding global optima at the boundaries should be a rare occurrence and could indicate that the attribute has poor predictive power. Such an attribute should most likely not be included in a decision tree algorithm. Consequently, the SUM extension could be used to find such anomalies and exclude them from a decision tree algorithm. For general purposes, the SUM extension (as opposed to just the BRM/HCA) should be used for discretization procedures since it is able to more accurately identify the global optimum and it adds only a minimal number of calculations.

A limitation of the BRM/HCA method is that it does not guarantee the discovery of the global optimum due to the fact that it does not cover the entire range of attribute values. A possible improvement of this method could be to allow the user to specify a certain percentage of coverage of the attribute. One such method that is currently under development is to develop a functional form from a sample of the data points. This method would require an

Figure 12. Example of BRM/HCA finding local optimum

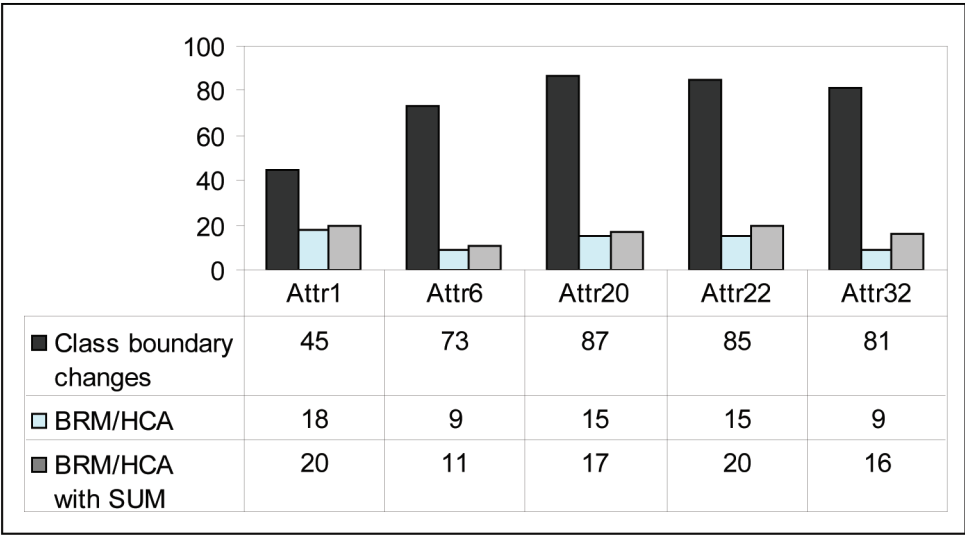


initial sampling of the attribute, from which a functional form for the information gain could be derived. We plan to investigate the effect of

sample size on the accuracy of the functional form derived.

In closing, the purpose of this article was to present a novel entropy-based discretization

Figure 13. Comparison of class boundary changes with BRM/HCA



method—the *bisecting region method with the hill-climbing algorithm*—and empirically validate its ability to find the (near) optimal split point with significantly fewer number of information gain calculations. This method could be a viable candidate for dynamic discretization especially for large datasets where computational efficiency becomes a critical factor to consider. The results from this empirical study show that the BRM/HCA can in fact find the optimal or near optimal split point when compared to the brute force method with only an average 4% reduction in information gain. In conclusion, on average, there was a 98% reduction in the number of information gain calculations when compared to the *brute force* method.

## REFERENCES

- Asuncion, A., & Newman, D. J. (2007). *UCI machine learning repository*. Retrieved from <http://www.ics.uci.edu/~mllearn/MLRepository.html>
- Bagui, S. (2006). An approach to mining crime patterns. *International Journal of Data Warehousing and Mining*, 2(1), 50–80.
- Baruch Options Data Warehouse. (2008). Subotnik Financial Services Center. Retrieved
- Breiman, L., Friedman, J. H., Olshen, R. A., & Stone, C. J. (1998). *MegaMind: machine learning on very large databases*. Sydney, Australia: University of Sydney.
- Catlett, J. (1991). *On changing continuous attributes into ordered discrete attributes*. Paper presented at the Proceedings of the European Working Session on Learning, Berlin, Germany.
- Chan, C. C., Batur, C., & Srinivasan, A. (1991). *Determination of quantization intervals in rule based model for dynamic systems*. Paper presented at the Proceedings of the IEEE Conference on Systems, Man, and Cybernetics, Charlottesville, VA.
- Chan, P. K., & Stolfo, S. J. (1993). *Experiments on multistrategy learning by metalearning*. Paper presented at the Proceedings of the Second International Conference on Information and Knowledge Management, Washington, D.C.
- Chan, P. K., & Stolfo, S. J. (1993). *Metalearning for multistrategy and parallel learning*. Paper presented at the Proceedings of Second International Workshop on Multistrategy Learning.
- Charles, F., & Cavazza, M. (2004). *Exploring the scalability of character-based storytelling*. Paper presented at the Third International Joint Conference on Autonomous Agents and Multiagent Systems.
- Chaudhuri, S. (1998). Data mining and database systems: where is the intersection? *A Quarterly Bulletin of the Computer Society of the IEEE Technical Committee on Data Engineering*, 21(1), 4–8.
- Chickering, D. M., Meek, C., & Rounthwaite, R. (2001). *Efficient Determination of Dynamic Split Points in a Decision Tree*. Paper presented at the First IEEE International Conference on Data Mining, San Jose, CA.
- Chiu, D. K. Y., Cheung, B., & Wong, A. K. C. (1990). Information synthesis based on hierarchical entropy discretization. *Journal of Experimental & Theoretical Artificial Intelligence*, 2, 117–129. doi:10.1080/09528139008953718
- Dougherty, J., Kohavi, R., & Sahami, M. (1995). *Supervised and unsupervised discretization of continuous features*. Paper presented at the Proceedings of the 12th International Conference on Machine Learning, Tahoe City, CA.
- Fayyad, U. M., & Irani, K. B. (1992). On the handling of continuous-valued attributes in decision tree generation. *Machine Learning*, 8, 87–102.
- Fayyad, U. M., & Irani, K. B. (1993). *Multi-interval discretization of continuous-valued attributes for classification learning*. Paper presented at the Proceedings of the 13th International Joint Conference on Artificial Intelligence.
- Friedman, N., & Goldszmidt, M. (1996). *Discretizing continuous attributes while learning Bayesian networks*. Paper presented at the Proceedings of the Thirteenth International Conference on Machine Learning, San Francisco, CA.
- Ganti, V., Gehrke, J., & Ramakrishnan, R. (1999). Mining very large databases. *Computer*, 32(8), 38–45. doi:10.1109/2.781633
- Gehrke, J., Ganti, V., Ramakrishnan, R., & Loh, W. Y. (1999). *BOAT - optimistic decision tree construction*. Paper presented at the 1999 ACM SIGMOD International Conference on Management of Data, Philadelphia, PA.



- Gehrke, J., Ramakrishnan, R., & Ganti, V. (2000). Rainforest - A framework for fast decision tree construction of large datasets. *Data Mining and Knowledge Discovery*, 4, 127-162. doi:10.1023/A:1009839829793
- Han, J., & Kamber, M. (2006). *Data Mining: Concepts and Techniques* (2nd ed.). Amsterdam: Morgan Kaufman.
- Hand, D., Mannila, H., & Smyth, P. (2001). *Principles of Data Mining*. Cambridge, MA: MIT Press.
- Ho, K. M., & Scott, P. D. (1997). *Zeta: A global method for discretization of continuous variables*. Paper presented at the KDD97: 3rd International Conference of Knowledge Discovery and Data Mining, Newport Beach, CA.
- Holte, R. C. (1993). Very simple classification rules perform well on most commonly used datasets. *Machine Learning*, 11, 63-90. doi:10.1023/A:1022631118932
- Hyafil, L., & Rivest, R. L. (1976). Constructing optimal binary decision trees is NP-complete. *Information Processing Letters*, 5(1), 15-17. doi:10.1016/0020-0190(76)90095-8
- June 20, 2008 from <http://optionsdata.baruch.cuny.edu/>
- Kamber, M., Winstone, L., Gong, W., Cheng, S., & Han, J. (1997). *Generalization and decision tree induction: efficient classification in data mining*. Paper presented at the RIDE '97.
- Kerber, R. (1992). *Chimerge: Discretization of numeric attributes*. Paper presented at the Proceedings of the Tenth National Conference on Artificial Intelligence.
- Kohonen, T. (1989). *Self-Organization and Associative Memory*. Berlin, Germany: Springer-Verlag.
- Liu, H., Hussain, F., Tan, C. L., & Dash, M. (2002). Discretization: An enabling technique. *Data Mining and Knowledge Discovery*, 6, 393-423. doi:10.1023/A:1016304305535
- Loh, W.-Y., & Shih, Y.-S. (1997). Split selection methods for classification trees. *Statistica Sinica*, 7, 815-840.
- Loh, W.-Y., & Vanichsetakul, N. (1988). Tree-structured classification via generalized discriminant analysis. *Journal of the American Statistical Association*, 83(403), 715-725. doi:10.2307/2289295
- Mehta, M., Agrawal, R., & Rissanen, J. (1996, 1996). *SLIQ: A Fast Scalable Classifier for Data Mining*. Paper presented at the EDBT, Avignon, France.
- Ng, R. T., & Han, J. (2002). CLARANS: A Method for Clustering Objects for Spatial Data Mining. *IEEE Transactions on Knowledge and Data Engineering*, 14(5), 1003-1016. doi:10.1109/TKDE.2002.1033770
- Pfahring, B. (1995). *Compression-based discretization of continuous attributes*. Paper presented at the Proceedings of the Twelfth International Conference on Machine Learning.
- Press, W. H. (2007). *Numerical recipes: the art of scientific computing* (3rd ed.). New York: Cambridge University Press.
- Quinlan, J. R. (1983). Learning efficient classification procedures and their application to chess endgames. In R. Michalski, T. Carbonell & T. Mitchell (Eds.), *Machine Learning: an AI approach*. Los Altos, CA: Morgan Kaufmann.
- Quinlan, J. R. (1986). Induction of decision trees. *Machine Learning*, 1(81-106), 81-106.
- Quinlan, J. R. (1993). *C4.5: Programs for Machine Learning*. Los Altos, CA: Morgan Kaufman.
- Quinlan, J. R. (1996). Improved use of continuous attributes in C4.5. *Journal of Artificial Intelligence Research*, 4, 77-90.
- Rastogi, R., & Shim, K. (2000). Public: A decision tree classifier that integrates building and pruning. *Data Mining and Knowledge Discovery*, 4(4), 315-344. doi:10.1023/A:1009887311454
- Richeldi, M., & Rossotto, M. (1995). Class-driven statistical discretization of continuous attributes. In N. Lavrac & S. Wrobel (Eds.), *Lecture notes in Artificial Intelligence* (Vol. 914, pp. 335-338). Berlin, Heidelberg, New York: Springer-Verlag.
- Shafer, J., Agrawal, R., & Mehta, M. (1996). *SPRINT: A scalable parallel classifier for data mining*. Paper presented at the Proceedings of the 22nd International Conference of Very Large Databases, Mumbai (Bombay), India.
- Sloan Digital Sky Survey. (2008). *The 5th data release of the Sloan digital sky survey*. Retrieved June 20, 2008 from <http://cas.sdss.org/dr5/en>
- Tan, P.-N., et al. (2005) *Introduction to Data Mining*. Boston, MA: Addison-Wesley Longman Publishing Co., Inc.

Ting, K. M. (1994). Discretization of continuous-valued attributes and instance-based learning (Vol. 491) (Tech. Rep.). Australia: University of Sydney.

Tzanis, G., & Berberidis, C. (2007). Mining for Mutually Exclusive Items in Transaction Databases. *International Journal of Data Warehousing and Mining*, 3(3), 45–59.

Van de Merckt, T. (1993). *Decision trees in numerical attribute spaces*. Paper presented at the 13th International Joint Conference on Artificial Intelligence.

Vannucci, M., & Colla, V. (2004, April 28-30). *Meaningful discretization of continuous features for association rules mining by means of a SOM*. Paper presented at the European Symposium on Artificial Neural Networks, Bruges, Belgium.

Weiss, S. M., Galen, R. S., & Tadepalli, P. V. (1990). Maximizing the predicative value of production rules. *Artificial Intelligence*, 45, 47–71. doi:10.1016/0004-3702(90)90037-Z

Witten, I. H., & Frank, E. (2005). *Data Mining: Practical Machine Learning Tools and Techniques*. San Francisco: Elsevier, Inc.

Wong, A. K. C., & Chiu, D. K. Y. (1987). Synthesizing statistical knowledge from incomplete mixed-mode data. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 9, 796–805. doi:10.1109/TPAMI.1987.4767986

Zaki, M. J. (2000). Scalable Algorithms for Association Mining. *IEEE Transactions on Knowledge and Data Engineering*, 12(3), 372–390. doi:10.1109/69.846291

## ENDNOTES

- <sup>1</sup> Forest cover dataset was obtained from UCI KDD Archive at <http://kdd.ics.uci.edu/>.
- <sup>2</sup> Ionosphere dataset was obtained from UCI Machine Learning Repository at <http://mllearn.ics.uci.edu/MLRepository.html>.
- <sup>3</sup> This method was implemented using PL/SQL on an Oracle 10g server.

*Kelley M. Engle received the master's degree in Information Systems from Pennsylvania State University in 2002. She is a PhD student in the Department of Information Systems at the University of Maryland Baltimore County (UMBC), Maryland. Her research interests center around the study of data mining and artificial intelligence, especially the area of healthcare informatics.*

*Aryya Gangopadhyay is a Professor of Information Systems at the University of Maryland Baltimore County (UMBC). He has a PhD in Computer Information Systems from Rutgers University. His research interests include privacy preserving data mining, data cube navigation, and core and applied research on data mining. He has co-authored and edited three books, many book chapters, and numerous papers in peer-reviewed journals. His research has been funded by the National Science Foundation, U.S. Department of Education, and various Government and commercial entities.*