Fast Feature Selection using Fractal Dimension

Caetano Traina Jr.[1], Agma Traina[1], Leejay Wu[2], Christos Faloutsos[2]

[1] Department of Computer Science – University of São Paulo at São Carlos – Brazil
{caetano, agma}@icmc.usp.br
[2] Department of Computer Science – Carnegie Mellon University – USA
lw2j@andrew.cmu.edu, christos@cs.cmu.edu

Abstract. Dimensionality curse and dimensionality reduction are two key issues that have retained high interest for data mining, machine learning, multimedia indexing, and clustering. In this paper we present a fast, scalable algorithm to quickly select the most important attributes (dimensions) for a given set of n-dimensional vectors. In contrast to older methods, our method has the following desirable properties: (a) it does not do rotation of attributes, thus leading to easy interpretation of the resulting attributes; (b) it can spot attributes that have either linear or nonlinear correlations; (c) it requires a constant number of passes over the dataset; (d) it gives a good estimate on how many attributes should be kept. The idea is to use the 'fractal' dimension of a dataset as a good approximation of its intrinsic dimension, and to drop attributes that do not affect it. We applied our method on real and synthetic datasets, where it gave fast and correct results.

Categories and Subject Descriptors: Information Systems [**Database Management**]: Database applications

Keywords: Data mining, Feature selection, Intrinsic dimensionality, Multi-scale space mapping

1. INTRODUCTION AND MOTIVATION

When managing the increasing volume of data generated by the organizations, a question that frequently arises is: "*What part of this data is really relevant to be kept?*". Notice that usually the relations of the database have many attributes that are correlated with the others. Attributes that are correlated to others do not introduce any new knowledge, so they can be dropped without losing information.

Attribute selection is a classic goal, as well as battling the "dimensionality curse" [Berchtold et al. 1998; Pagel et al. 2000]. A carefully chosen attribute subset improves performance and efficacy of a variety of algorithms. This is particularly true with redundant data, as many datasets can largely be well-approximated in fewer dimensions. This can also be seen as a way to compress data, as only the attributes that maintain the essential dataset characteristics must be kept [Fayyad 1998].

In this paper we introduce a novel technique that can discover how many attributes are significant to characterize a dataset. We also present a fast, scalable algorithm to quickly select the most significant attributes of a dataset. In contrast to other methods, such as Singular Value Decomposition (SVD) [Faloutsos 1996], our method has the following desirable properties:

[1]This research has been funded by FAPESP (São Paulo State Foundation for Research Support), CNPq(National Council for Scientific and Technological Development) and CAPES (Graduate-Level Courses Personnel Improving Coordination), all from Brazil.
[2]This material is based upon work supported by the National Science Foundation under Grants No. IRI-9625428, DMS-9873442, IIS-9817496, and IIS-9910606. Additional funding was provided by donations from NEC and Intel. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

(1) it does not rotate attributes, leading to easy interpretation of the resulting attributes;
(2) it can spot attributes that have either linear or nonlinear, even non-polynomial correlations;
(3) it is fast, with computational complexity linear on the number of objects in the dataset;
(4) it gives a good estimate on how many attributes we should keep.

The main idea is to use the 'fractal' dimension of the dataset, and to drop attributes that do not affect it. The fractal dimension (D) is relatively unaffected by redundant attributes, and our algorithm can compute it in **linear** time with respect to the number of objects. Thus, we propose a kind of backward-elimination algorithm to take advantage of the fast D computation. This algorithm sequentially removes attributes that contribute minimally to D.

The remainder of the paper is structured as follows. In the next section, we present a brief survey on the related techniques. Section 3 introduces the concepts needed to understand the proposed method. Section 4 presents the fractal dimension algorithm developed as well as the datasets used in the experiments. Section 5 gives the proposed method for attribute selection. Section 6 discusses the experiments and evaluation of the proposed method. Section 7 gives the conclusions of this paper.

2. SURVEY

Numerous attribute selection methods have been studied, including genetic algorithms; sequential feature selection algorithms such as forwards, backwards and bidirectional sequential searches; and feature weighting [Aha and Bankert 1995; Scherf and Brauer 1997; Vafaie and Jong 1993]. A survey on attribute selection using machine learning techniques is presented in [Blum and Langley 1997].

The singular value decomposition (SVD) technique provides a way of reducing the dimensionality of data by generating an ordered set of additional axes [Faloutsos 1996]. However, this is not attribute selection, but instead axis generation as SVD returns vectors that do not need to correspond to the original attributes. These vectors may be inappropriate for assorted situations, such as those involving the presentation of data for human understanding; tasks where accessing additional attributes may be expensive; and when creating a training set to derive a classifier.

A common research challenge in attribute selection methods so far is the exponential growth of computing time required [Blum and Langley 1997]. Indeed the induction methods proposed so far had super-linear or exponential computational complexity [Langley and Sage 1997], as is the case with nearest neighbors, learning decision trees [John et al. 1994; Kira and Rendell 1992], and Bayesian Networks [Singh and Provan 1995]. Notice that these approaches are highly sensitive to both the number of irrelevant or redundant features present in the dataset, and to the size of the dataset, avoiding the use of samples [Langley and Sage 1997].

Fractal dimension has been a useful tool for the analysis of spatial access methods [Belussi and Faloutsos 1995; Kamel and Faloutsos 1994], indexing [Böhm and Kriegel 2000], join selectivity estimation [Faloutsos et al. 2000], and analysis of metric trees [Traina Jr. et al. 2000]. However, to the best of the authors' knowledge, it was never used before to perform attribute selection.

3. FUNDAMENTAL CONCEPTS

The most common way to store data is through tables with as many columns as there are features represented in the data, and as many lines as there are data elements. In this paper we are calling these tables as datasets, the features as attributes, and the data elements (or objects) as points in the space of features. In this way, a dataset is seen as points in an E-dimensional space, where E is the number of attributes. Table I summarizes the symbols used in this paper.

We are especially interested in datasets comprising complex data, usually composed of numerical attributes. Features extracted from images are well-known examples of high-dimensional datasets,

Table I. Definition of Symbols

Symbols	Definitions
E	Embedding dimension (Euclidean dimensionality)
D	Fractal dimension (intrinsic dimensionality)
N	Number of points in the dataset
Cr, i	Count ('occupancies') of points in the i-th grid cell of side r
r	Side of a grid cell
$S(r)$	Total occupancies for a specific grid cell side r
R	Number of side sizes r to plot $S(r)$

which are used in content-based image retrieval systems. For these datasets, it is difficult to choose the set of attributes that can be assigned as keys of the dataset. In this way, if one is interested in creating an index structure for the dataset the whole set of attributes needs to be considered. This leads to the previously mentioned dimensionality curse.

3.1 'Embedding' and 'intrinsic dimensionality'

Our objective in this paper is to find a subset of the attributes that can be discarded when creating indexes or applying data mining techniques, without compromising the results. Attributes that can be calculated from others are immediate candidates to discard, if the way to calculate them is known. However, in general, their correlations are not known. Thus, our objective is to detecting correlations between attributes in a dataset, and how many redundant attributes the dataset has, even if we cannot represent the correlation expression. This leads to the definition of the embedding and intrinsic dimensions.

Definition 3.1. The embedding dimension E of a dataset is the dimension of its address space. In other words, it is the number of attributes of the dataset. The dataset can represent a spatial object that has a dimension lower than the space where it is embedded. For example, a line has an intrinsic dimensionality one, regardless if it is in a higher dimensional space.

Definition 3.2. The intrinsic dimension D of a dataset is the dimension of the spatial object represented by the dataset, regardless of the space where it is embedded.

Conceptually, if a dataset has all of its variables independent from the others, then its intrinsic dimension is the embedding dimension ($D = E$). However, whenever there is a correlation between two or more variables, the intrinsic dimensionality of the dataset is reduced accordingly. For example, each polynomial correlation (linear, quadratic, etc.) reduces the intrinsic dimension by a unit. Other types of correlations can reduce the intrinsic dimension by a different amount, even a fractional amount, as will be shown later.

Usually the embedding dimensionality of the dataset hides the actual characteristics of the dataset, and in general correlations between the variables in real datasets are not known and even the existence of correlations is not known either. This motivated us to look for a technique that allows one to find the intrinsic dimension of the dataset even when the existence of correlations is not identified. Knowing its intrinsic dimension, it is possible to decide how many attributes are in fact required to characterize a dataset.

3.2 Fractals and Fractal Dimension

A fractal dataset is known by its characteristic of being self-similar. This means that the dataset has roughly the same properties for a wide variation in scale or size, i.e., parts of any size of the fractal are similar (exactly or statistically) to the whole fractal. This idea is illustrated in Figure 1, which shows the first three steps to build the Sierpinski triangle, a well-known point-set fractal. The Sierpinski triangle is constructed from an equilateral triangle ABC, excluding its middle triangle A'B'C' and recursively repeating this procedure for each of the resulting smaller triangles. The Sierpinski triangle is generated after infinite iterations of this procedure. The Sierpinski triangle has an infinite perimeter, so it is not a 1-dimensional object. And it has no area, so it is not a 2-dimensional object either. In fact, it has an intrinsic dimension, equal to $log(3)/log(2) = 1.58$ [Schroeder 1991]. For a real set of points, we measure the fractal dimension with the box-count plot, which is the basis of the algorithm to be proposed in Section 4.

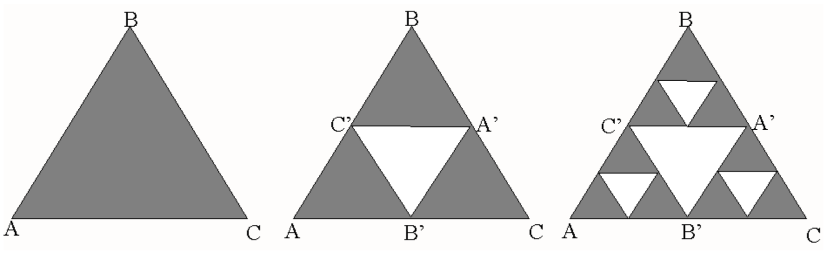


Fig. 1. Recursive construction of the Sierpinski triangle.

Definition 3.3. **(Correlation Fractal dimension)**: Given a dataset presenting self-similarity in the range of scales $[r_1, r_2]$, its Correlation Fractal dimension D_2 for this range is measured as

$$D_2 \equiv \frac{\partial \log \sum_i C_{r,i}^2}{\partial \log r}, \; r \in [r_1, r_2]$$

As shown in [Belussi and Faloutsos 1995], the correlation fractal dimension corresponds to the intrinsic dimension of the dataset. Thus, from now on, we will use D_2 as the intrinsic dimension D.

Observation 3.4. - The fractal dimension of an Euclidean object corresponds to its Euclidean dimension and it is always an integer number.

For example, a set of points along lines, circumferences and all standard curves have $D = 1$; planes, circle disks, squares and surfaces have $D = 2$; Euclidean volumes have $D = 3$, and so on. Indeed, a line segment in any n-dimensional space will always have $D = 1$, as well as a square will always have $D = 2$, even if the points are in a higher-dimensional space.

Observation 3.5. - The fractal dimension of a dataset cannot be greater than its embedding dimension.

Many real datasets are fractals [Traina Jr. et al. 1999b; Schroeder 1991]. Thus, for these datasets we can take the advantage of working with their correlation fractal dimension as their intrinsic dimension D.

Observation 3.6. - The intrinsic dimensionality gives a lower bound of the number of attributes needed to keep the essential characteristics of the dataset.

This observation states that the minimum number of attributes to be retained is equal to the ceiling function on $\lceil D \rceil$.

4. THE FRACTAL DIMENSION ALGORITHM

This section presents an algorithm to compute the fractal dimension D of any given set of points in any E-dimensional space. A practical way to estimate D from a spatial dataset is using the box-counting approach [Schroeder 1991]. Theoretically, this method gives a close approximation of the fractal dimension, and our experiments showed that it indeed does [Traina Jr. et al. 2000; 1999a]. One of the best published algorithm to calculate D of a dataset is an $O(Nlog(N))$ algorithm, where N is the number of points in the dataset [Belussi and Faloutsos 1995]. However, we developed a new, very fast, $O(N)$ algorithm to implement it, which is presented as follows.

Consider the address space of a point-set in an E-dimensional space, and impose an E-grid with grid-cells of side size r. Focusing on the i-th cell, let $C_{r,i}$ be the count ('occupancies') of points in each cell. Then, compute the value $S(r) = \sum_i C_{r,i}^2$. The fractal dimension is the derivative of $log(S(r))$ with respect to the logarithm of the radius. As we assume self-similar datasets, we expect this derivative results in a constant value. Thus, we can obtain the fractal dimension D of a dataset plotting $S(r)$ in log-log scales for different values of the radius r, and calculating the slope of the resulting line.

It is needed to process $S(r)$ for a quantity R of values of r, so we can achieve a suitable statistical approximation of the line. To avoid reading the dataset again for each value of the radius, we propose to create a multi-level grid structure, where each level has a radius the half of the size of the previous level ($r = 1, 1/2, 1/4, 1/8$, etc.). Each level of the structure corresponds to a different radius, so the depth of the structure is equal to the number of points in the resulting graph. The structure is created in main memory, so the number of points in the graph is limited by the amount of main memory available. If this graph is linear for a suitable range of radii, the dataset is a fractal and its fractal dimension D is the slope of the fitting line of this graph.

The proposed algorithm is linear on the number of points in the dataset. The computational complexity of the algorithm is $O(N * E * R)$, where N is the number of objects in the dataset, E is the embedding dimensionality, and R is the number of points used to plot the $S(r)$ function. This shows that the algorithm is scalable to datasets of any size.

For each given cell side r, only the cells having at least one already processed point are maintained, counting the sum of occupancies $C_{r,i}$ of this cell. In this way, each new point is directly associated to a cell in each level, without the need to be compared with the previously read points. Figure 2 shows the structure used in the algorithm for 2- and 3-dimensional datasets.

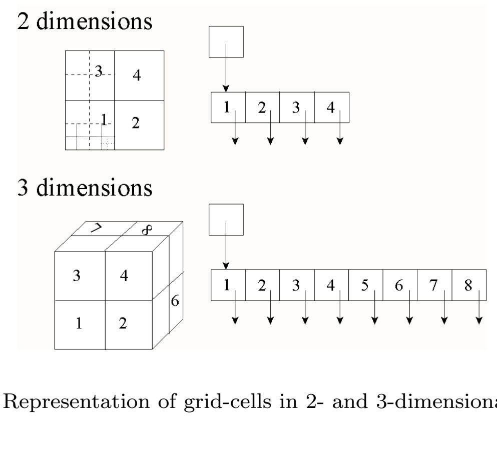


Fig. 2. Representation of grid-cells in 2- and 3-dimensional spaces.

The largest cell side of the space of points generates 2^n cells. In the next level, each cell is split into other 2^n cells, and so on. Given that the position of each cell in the space is always known, each cell is represented by: the sum of occupancies $C_{r,i}$ in this cell, and the pointers to the cells in the next level covered by this cell (see Figure 2). This structure is a kind of a multidimensional "quad-tree" (oct-tree for a 3D space, or E-dim-tree). Figure 3 shows an example of this structure for a dataset with five points in three levels in a 2-dimensional space.

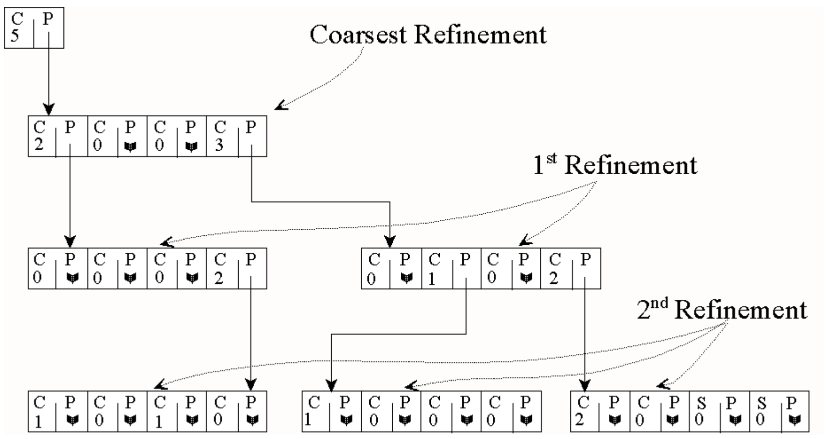


Fig. 3. Example of the data structure used for calculating the Sum of Occupancies of a dataset with 5 points (with three levels of resolution).

Notice that new cells are added to the structure on demand. Thus, only cells occupied by at least one point are created ($C_{r,i} > 0$). The algorithm processes the points set only once, so it is indeed very fast. Algorithm 1 summarizes this computation process.

Algorithm 1 Compute the fractal dimension D of a dataset A (box-count approach).

```
Require: Normalized dataset $A$ ($N$ rows, with $E$ dimensions/attributes each)
Ensure: Fractal dimension $D$
1: for each desirable grid-size $r = 1/2^j, j = 1, 2, ..., l$ do
2:    for each point of the dataset do
3:       Decide which grid cell it falls in (say, the $i$-th cell)
4:       Increment the count $C_i$ (occupancy')
5:    end for
6:    Compute the sum of occupancies $S(r) = \sum C_i^2$
7: end for
8: Print the values of $log(r)$ and $log(S(r))$, generating a plot;
9: Return the slope of the linear part of the plot (linear regression) as the fractal dimension $D$ of
   the dataset $A$.
```

As the grid side increases, the number of pointers to empty cells increases as well. Thus, for high-dimensional datasets it is worthwhile to keep the cells as linked lists instead of arrays. We implemented this structure as an object in C++, using an array for datasets with the embedding dimension less or equal three, and using a linked list for datasets with higher dimensionality.

4.1 Datasets used in the experiments

We used synthetic and real datasets to evaluate our method. Figure 4 shows a mapping in a 3-dimensional space of the higher-dimensional datasets used in the experiments. This mapping was done through the *FastMap* algorithm [Faloutsos and Lin 1995]. We used two synthetic datasets built

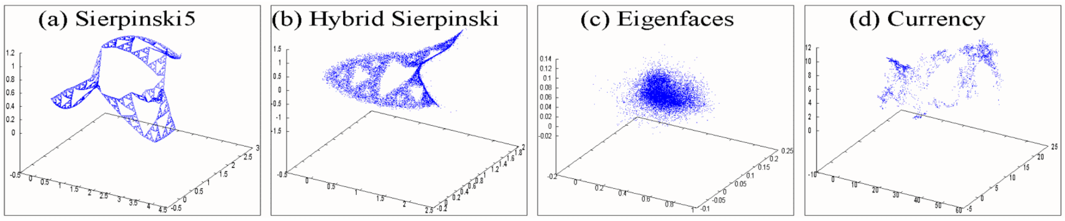


Fig. 4. Three-dimensional mappings of the datasets used in the experiments of the proposed method ("FDR").

over a Sierpinski triangle (9,841 points in a 2-dimensional space), adding three more attributes to the dataset in order to test our method. The synthetic datasets are:

Sierpinski. (see Figure 4(a)) - The original 2D points of the original dataset (x, y) became 5D points $(a = x, b = y, c = a + b, d = a^2 + b^2, e = a^2 - b^2)$. The three latest coordinates included in the dataset are strongly correlated with the two first coordinates. Thus, the fractal dimension (1.68) of the new dataset is close to the fractal dimension of the original Sierpinski triangle.

Hybrid5. (see Figure 4(b)) - The original 2D points of the Sierpinski triangle (x, y) became 5D points $(a = x, b = y, c = f(a, b), d = random_1, e = random_2)$. As the two latest coordinates include random noise to the dataset, the fractal dimension of the dataset is equal to 3.62, basically the dimensionality of the Sierpinski (1.58) plus the dimensionality of a square in 2D (2.00). The third variable ('c') depends non-linearly on the others. It is obtained by the algorithm during the Sierpinski triangle generation.

It is also important to assess the algorithm's behavior for real data. Thus, two real datasets were also employed to evaluate our proposed method:

Eigenfaces. (see Figure 4(d)) - a dataset of 11,900 face vectors given by the Informedia project [Wactlar et al. 1996] at Carnegie Mellon University. Each face was processed with the *eigenfaces* method [Turk and Pentland 1991], resulting in 16-dimensional vectors.

Currency. (see Figure 4(c)) - This is a 6-dimensional dataset, presenting the normalized exchange rate of currencies based on Canadian Dollar. The data was collected from 01/02/87 until 01/28/97. This resulted in $N = 2,561$ measurements made on working days. Each attribute corresponds to a currency (a = Hong Kong Dollar, b = Japanese Yen, c = American Dollar, d = German Mark, e = French Franc, f =British Pound).

Looking at Figure 5, we can see that Observation 3.4 indeed holds for these datasets. As it can be seen, the correlation fractal dimension gives the fractal dimension of the datasets, regardless of their embedding dimension. Figure 6 shows the correlation fractal dimension of the real datasets used in this paper.

5. THE ATTRIBUTE SELECTION ALGORITHM

5.1 Intuition

In this Section we present an approach to quickly discard some attributes (dimensions) from the original dataset, taking advantage of the fractal dimension concept. We stated in Observation 3.5 that the fractal dimension D of a dataset cannot exceed its embedding dimensionality E. Moreover, there are at least $\lceil D \rceil$ attributes that cannot be determined from the others. Since $D \leq E$, there are at least

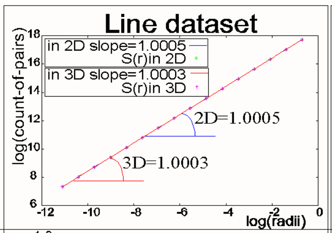
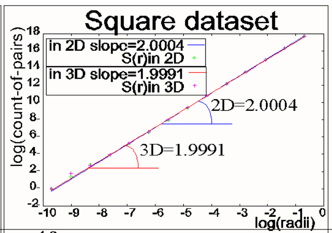
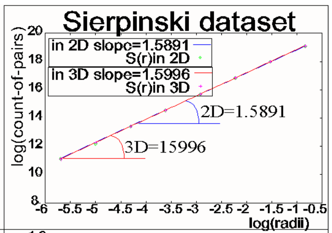
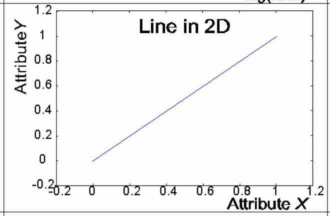
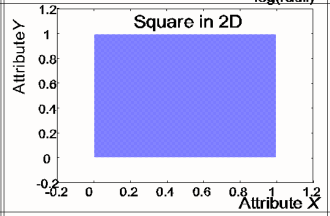
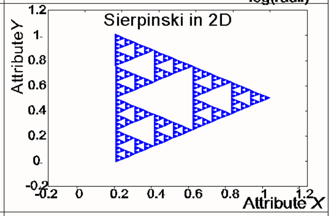
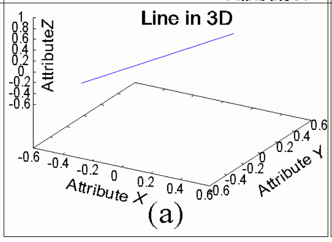
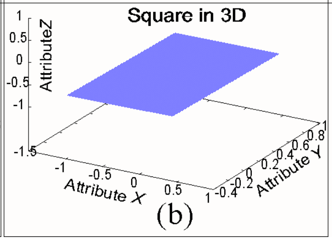
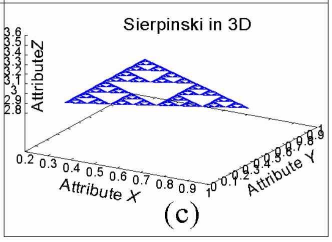
$S(r)$ and slope (FD) for the dataset in 2D and 3D			
Visualization of the dataset in 2D			
Visualization of the dataset in 3D			

Fig. 5. Fractal dimension of synthetic datasets embedded in 2- and 3-dimensional spaces. (a) Line; (b) Square; (3) Sierpinski triangle.

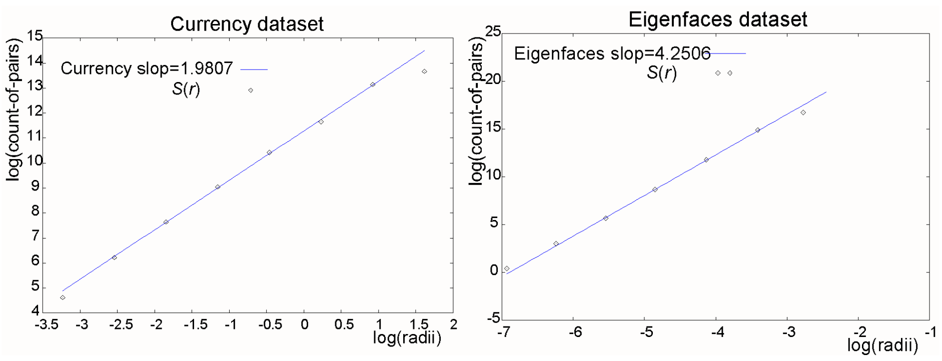


Fig. 6. Fractal dimension of real datasets. (a) Currency dataset; (b) Eigenfaces dataset.

$E - \lceil D \rceil$ attributes that can be correlated with the others. Correlated attributes contribute to increase the complexity of any treatment that the dataset must be submitted to, such as spatial indexing in a database, and knowledge retrieval in data mining processes, without adding new information. Moreover, the correlated attributes can be re-obtained from the other attributes. Hence, whenever it is possible, such attributes should be detected and dropped from the dataset.

Definition 5.1. - (Partial fractal dimension $- pD$) : Given a dataset A with E attributes, the Partial fractal dimension is obtained evaluating the correlation fractal dimension of the dataset excluding one or more attributes from the dataset.

Figure 7(a) illustrates the intuition behind our approach. This is the ‘Quarter-circle’ dataset, whose points are in $E = 2$ dimensions, but has fractal dimension $D = 1$. Notice that the two attributes x and y are nonlinearly correlated, because $y = \sqrt{1 - x^2}$. Also notice that the traditional dimensionality

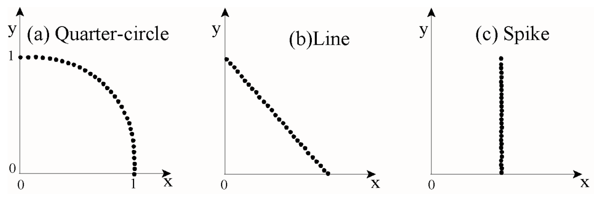


Fig. 7. Example of points sets in $E = 2$-dimensional space. (a) Quarter-circle; (b) Line; (c) Spike.

reduction method, SVD, only works well for linear correlations. Even without knowing how the correlation is expressed, knowing that the fractal dimension $D \approx 1$ gives a hint that likely the two attributes are correlated. Thus, the points projected on one axes (say x) probably will preserve the original distances. The fractal dimension of the projected points reveals how well preserved the intrinsic properties of the dataset are. In this specific case, the pD for x is $pD = 0.9$, which means that the mode of the dataset was kept after projection. Consider also Figure 7(b) and 7(c) presenting the 'Line' and 'Spike' examples respectively. Again, our approach correctly flag attributes x and y for omission, but it will not allow to drop the attribute y in the 'Spike' picture.

5.2 The Proposed Algorithm - "FDR"

We propose the *Fractal Dimension Reduction* (FDR) algorithm, which uses the backward elimination of attributes approach. The proposed idea is to calculate the correlation fractal dimension of the whole dataset, and also to calculate its pD dropping one of its E attributes at a time. Thus, it will result in E partial fractal dimensions. The process continues selecting the attribute that leads to the minimum difference in the pD for the whole dataset. If this difference is within a small threshold, we can be confident that this attribute contributes almost nothing to the overall characteristics of the dataset. Therefore, this attribute can be dropped from the list of important attributes that characterizes the dataset. The threshold depends on how precise the resulting dataset needs to preserve the characteristics of the original dataset. As a rule of thumb, and considering that each polynomial correlation drops the fractal dimension by one unit, the threshold could be set around the same value allowed for the error in fitting the line that retrieves the fractal dimension. In our experiments, we used a threshold of 0.015, allowing an error of 1.5% to fit the line.

The algorithm is iterative, i.e., using the resulting set of attributes, it repeat the previous reduction steps, until there are no more attributes to be dropped without changing the previous partial fractal dimension more than a fixed threshold.

If there are two or more correlated attributes, algorithm FDR will sequentially drop attributes using this correlation, until only the number of attributes that corresponds to independent attributes remain. For example, if there are three attributes $\{a, b, c\}$, where the third is a function of the previous two, e.g., $c = a + b$, any of the three attributes can be dropped, because the others can be used to derive the dropped one. However, if there is no other correlation linking the remaining attributes, then no other attribute could be dropped without mischaracterizing the dataset. Algorithm 2 presents the algorithm FDR, used to generate the attribute classification, which are presented ordered by their significance. That is, the first attribute to be dropped is the least important attribute, the second attribute dropped is the second least important one and so on. Based on the algorithm, the following observation can be made.

Observation 5.2. - The most independent attributes are saved to the end of the process.

Due to the algorithm construction, the process can stop earlier, when the minimum number of attributes is achieved.

Algorithm 2 Fractal dimensionality reduction (FDR)

```
Require: Dataset $A$
Ensure: A list of attributes in the reverse order of their importance
 1: Compute the fractal dimension $D$ of the whole dataset;
 2: Initially set all attributes of the dataset as significant;
 3: Set the whole fractal dimension as the current $D$;
 4: while there are significant attributes do
 5:    for every significant attribute $i$ do
 6:       Compute the partial fractal dimensions $pD_i$ using all significant attributes excluding attribute
          $i$;
 7:    end for
 8:    Sort the partial fractal dimensions $pD_i$ and select the attribute $a$ that leads to the minimum
       difference (current $D - pD_i$);
 9:    Set the $pD_i$ obtained removing attribute $a$ as the current $D$;
10:    Output attribute $a$ as irrelevant and remove it from the set of significant attributes;
11: end while
```

6. EXPERIMENTS AND EVALUATION

We did experiments to answer the following questions:

(1) How scalable are the proposed algorithms?
(2) How many attributes should be kept in order to reduce the dimensionality of the dataset?

The following sections will clarify these points. The experiments and measurements were taken on a 450MHz Pentium II machine with 128 Mbytes of RAM under Windows NT4.0. All the proposed algorithms were implemented in C++.

6.1 Scalability of the proposed method

The algorithm developed to obtain the correlation fractal dimension is linear over the number of points in the dataset, i.e., $O(N)$. As the embedding dimensionality E of the dataset is a constant and the number of grid sizes R is fixed as a parameter for the algorithm, then the complexity of our algorithm is $O(N * R * E)$. However, R is typically set to 20 (the value used in all of our experiments) and E is a small value, typically much smaller than the number of points in the datasets, which can be in order of hundreds of thousands or more. Figure 8 shows the wall-clock time required to get the fractal dimension against the size of the dataset. The datasets have a varying number of points in 2, 4, 8 and 16-dimensional spaces. There was generated 20 grid sizes for each dataset. Figure 8 shows that the execution time of this algorithm is linear on the number of points in the dataset.

The algorithm developed to select the attributes of a dataset by their significance is very fast. Instead of the super-linear time over the size of the dataset (N) being analyzed, as it is needed by the machine learning techniques [Blum and Langley 1997], our FDR algorithm is linear on N (number of objects) and quadratic on the embedding dimensionality E of the dataset. Table II shows the wall-clock time needed to generate the classification of the attributes for the datasets we presented in this paper. Table II also summarizes the meaningful information of the datasets.

6.2 Dimensionality reduction using fractal dimension

Figure 9 presents the graphs generated by the FDR algorithm on the test datasets. Figure 9(a) shows the graph of the pD of the ‘Sierpinski5’ dataset when its attributes are sequentially dropped. From this plot, it can be seen that just two attributes are enough to characterize this dataset. Our algorithm

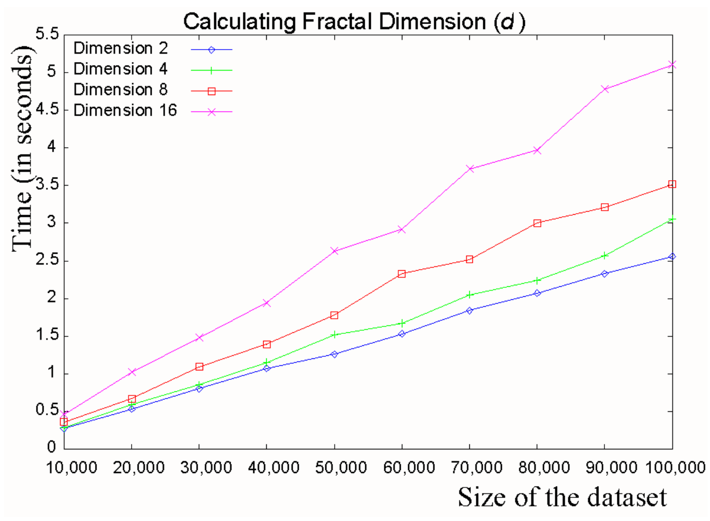


Fig. 8. Wall-clock time (in seconds) needed to obtain the fractal dimension of varying sized datasets. The curves show the datasets with 2, 4, 8 and 16 dimensions.

Table II. Wall-clock time (in seconds) spent to run the backward-selection algorithm on the datasets presented. A summarization of the datasets is also given.

Dataset	Number of points – N	Embedding dimensionality – E	Intrinsic dimensionality – D	Time (in seconds)
Sierpinski5	9,841	5	1.597	6.24
Hybrid5	9,841	5	3.627	7.03
Eigenfaces	11,900	16	4.250	132.82
Currency	2,561	6	1.980	2.54

drops $c = a + b, e = a^2 - b^2$ and a attributes, holding b and $d = a^2 + b^2$, with a resulting partial fractal dimension $pD = 1.568$ (versus a whole $pD = 1.597$). Notice that knowing b and $d = a^2 + b^2$, the other attributes can be recalculated.

Figure 9(b) presents the same plot of the pD for the 'Hybrid5' dataset when its attributes are sequentially dropped. Looking at this plot, it can be seen that four attributes are needed to characterize this dataset. Just the $c = f(a + b)$ attribute can be dropped, as every other contributes with a significant portion of D. This is correct, as the attributes a and b correspond to the original Sierpinski triangle points, and the attributes d and e depend on random numbers, which are independent variables and cannot be obtained from the other attributes. Also, as 'Hybrid5' dataset has $D = 3.62$, it is expected that four attributes should remain.

Figure 9(c) shows the plot of the pD for the 'Eigenfaces' dataset when its attributes are sequentially dropped. Looking at this plot, we can see that, from the original 16 attributes, just five are enough to characterize this dataset $\{b, d, f, a, e\}$. The resulting partial fractal dimension with five attributes is 3.815, and the whole partial fractal dimension is 4.207 (that is, eleven attributes contribute only 0.392 to the whole fractal dimension).

Figure 9(d) shows the plot of the pD of the 'Currency' dataset when its attributes are sequentially dropped. It shows that the Hong Kong Dollar is the only currency that can be immediately dropped. This is correct, as we know that the Hong Kong Dollar is linked to the American Dollar, so there is some strong correlation between both currencies. The other currencies have more independent behaviors, as their contribution to the whole fractal dimension is a value between 0.16 and 0.68.

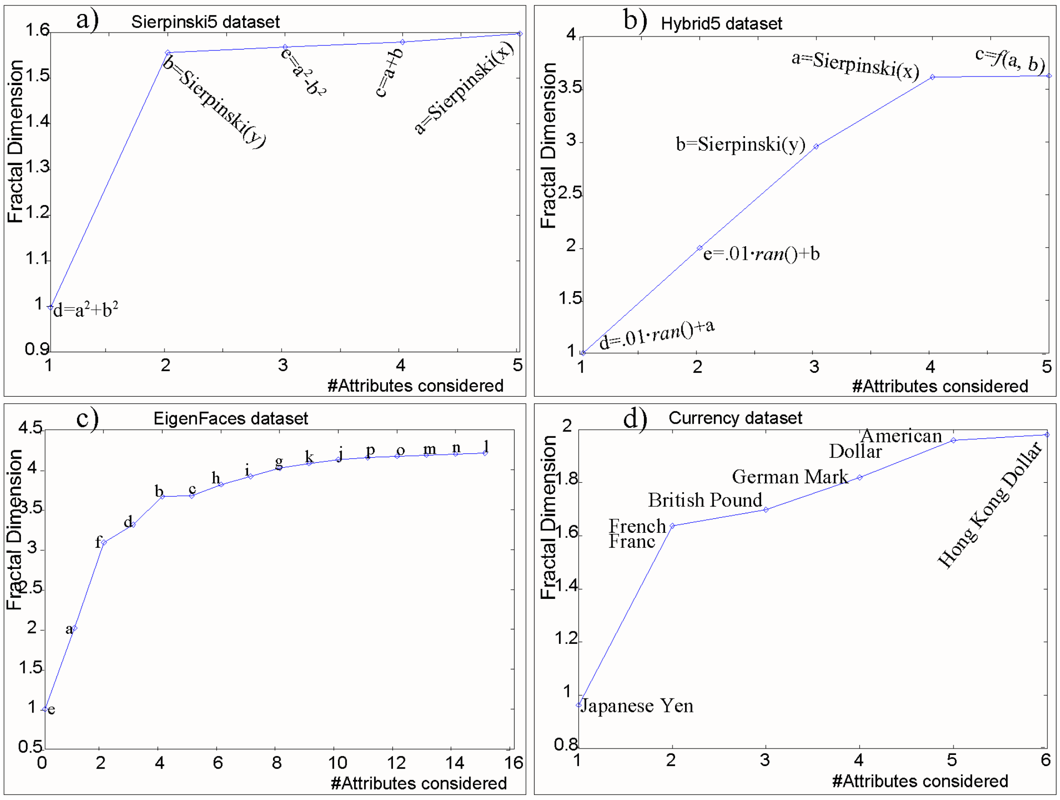


Fig. 9. Plots of the number of points dropped versus the partial fractal dimensions of the following datasets: (a) Sierpinski; (b) Hybrid5; (c) Eigenfaces; (d) Currency.

6.3 Discussion

Intuitively, the attribute selection could be performed in backward or forward direction. If there are only polynomial correlations between the attributes, both backward or forward selection works well. However, when there is a fractal correlation between the attributes (such as the x and y coordinates in the Sierpinski triangle), the experiments showed that the backward selection works better.

The fractal dimension D is a guide to know when to stop the backward selection algorithm FDR. Indeed, $\lceil D \rceil$ is the minimum number of attributes that must be in the resulting set. This is due to the fact that $\lceil D \rceil$ attributes are enough to preserve the essential characteristics of the dataset.

7. CONCLUSIONS

The main contribution of this paper is the proposal of a novel approach in feature selection and dimensionality reduction, using the concept of fractal dimension. This approach leads to a method to reduce the dimensionality of spatial datasets with the following properties:

—It can detect the hidden correlations existing in the dataset, spotting how many attributes strongly affect the behavior of the dataset regarding index and retrieval operations;

—It can show the attributes that have nonlinear and even non-polynomial correlations, where the traditional SVD method fails;

—It provides a small subset of attributes that can represent the whole dataset.

—It is scalable on the number N of elements in the dataset - $O(N)$. This is a striking advantage over methods from Machine Learning field [Blum and Langley 1997], which are super-linear on the number of objects N.

—It can be applied to high-dimensional datasets as well.

—It does not rotate the address space of the dataset. Thus, it leads to easy interpretation of the resulting attributes.

Other contributions are:

—The detailed design of the single pass algorithm to compute the correlation fractal dimension of any spatial dataset. This algorithm is $O(N)$, thus scaling up for arbitrarily sized datasets. This algorithm works in main memory, but the amount of memory available limits only the resolution of results, and not the size or dimension of the dataset.

—The quick backward attribute reduction algorithm. As it uses the quick algorithm to calculate the fractal dimension, it is also linear on the size of the dataset. Moreover, it can quickly compute the meaningful attributes (seconds), in contrast to current methods that take hours or days to give answers.

—Experiments on synthetic and real datasets, showing the effectiveness and speed of the results.

REFERENCES

Aha, D. W. and Bankert, R. L. A comparative evaluation of sequential feature selection algorithms. In *Proceedings of the Fifth Intl. Workshop on Artificial Intelligence and Statistics.* Ft. Lauderdale, FL, 1995.

Belussi, A. and Faloutsos, C. Estimating the selectivity of spatial queries using the 'correlation' fractal dimension. In *Proceedings of the 21th Intl. Conf. on Very Large Data Bases.* Morgan Kaufmann, Zurich, Switzerland, pp. 299–310, 1995.

Berchtold, S., Böhm, C., and Kriegel, H.-P. The pyramid-tree: Breaking the curse of dimensionality. In *Proceedings of the ACM SIGMOD Intl. Conf. on Management of Data.* ACM Press, Seattle, Washington, pp. 142–153, 1998.

Blum, A. L. and Langley, P. Selection of relevant features and examples in machine learning. *Artificial Inteligence* vol. 97, pp. 245–271, 1997.

Böhm, C. and Kriegel, H.-P. Dynamically optimizing high-dimensional index structures. In *Proceedings of the 7th Intl. Conf. on Extending Database Technology.* Lecture Notes in Computer Science, vol. 1777. Springer, Konstanz, Germany, pp. 36–50, 2000.

Faloutsos, C. *Searching Multimedia Databases by Content.* Advances in Database Systems. Kluwer Academic Publishers, Boston, 1996.

Faloutsos, C. and Lin, K.-I. Fastmap: A fast algorithm for indexing, data-mining and visualization of traditional and multimedia datasets. In *Proceedings of the ACM SIGMOD Intl. Conf. on Management of Data.* ACM Press, San Jose, California, pp. 163–174, 1995.

Faloutsos, C., Seeger, B., Traina, A. J. M., and Traina Jr., C. Spatial join selectivity using power laws. In *Proceedings of the ACM SIGMOD Intl. Conf. on Management of Data.* ACM Press, Dallas, Texas, USA, pp. 177–188, 2000.

Fayyad, U. Mining databases: Towards algorithms for knowledge discovery. *Bulletin of the IEEE Technical committee on Data Engineering* 21 (1): 39–48, 1998.

John, G. H., Kohavi, R., and K.Pfleger. Irrelevant features and the subset selection problem. In *Proceedings of the 11' Intl. Conf. on Machine Learning.* Morgan Kaufmann, New Brunswick, NJ, USA, pp. 121–129, 1994.

Kamel, I. and Faloutsos, C. Hilbert r-tree: An improved r-tree using fractals. In *Proceedings of the 20th Intl.Conf. on Very Large Data Bases.* Morgan Kaufmann, Santiago de Chile, Chile, pp. 500–509, 1994.

Kira, K. and Rendell, L. L. A practical approach fo feature selection. In *Proceedings of the 9th Intl. Conf. on Machine Learning.* Morgan Kaufmann, Aberdeen Scotland, pp. 249–256, 1992.

Langley, P. and Sage, S. Scaling to domains with many irrelavant features. In *Computational learning theory and natural learning systems*, R. Greiner (Ed.). Vol. 4. MIT Press, Cambridge, MA, 1997.

Pagel, B.-U., Korn, F., and Faloutsos, C. Deflating the dimensionality curse using multiple fractal dimensions. In *Proceedings of the 16th Intl. Conf. on Data Engineering.* IEEE Computer Society, San Diego, CA - USA, pp. 589–598, 2000.

Scherf, M. and Brauer, W. Feature selection by means of a feature weighting approach. Tech. rep., Technische Universität München, 1997.

Schroeder, M. *Fractals, Chaos, Power Laws.* W.H. Freeman and Company, New York, 1991.

Singh, M. and Provan, G. M. A comparison of induction algorithms for selective and non-selective bayesian classifiers. In *Proceedings of the 12th Intl. Conf. on Machine Learning.* Morgan Kaufmann, Lake Tahoe, CA, USA, pp. 497–505, 1995.

Traina Jr., C., Traina, A. J. M., and Faloutsos, C. Distance exponent: A new concept for selectivity estimation in metric trees. Tech. Rep. CMU-CS-99-110, Carnegie Mellon University – School of Computer Science. March 1, 1999, 1999a.

Traina Jr., C., Traina, A. J. M., and Faloutsos, C. Fastmapdb user's manual, 1999b.

Traina Jr., C., Traina, A. J. M., and Faloutsos, C. Distance exponent: a new concept for selectivity estimation in metric trees. In *IEEE Intl. Conf. on Data Engineering (ICDE).* IEEE CS Press, San Diego - CA, pp. 195, 2000.

Turk, M. and Pentland, A. Eigenfaces for recognition. *Journal of Cognitive Neuroscience* 3 (1): 71–86, 1991.

Vafaie, H. and Jong, K. A. D. Robust feature selection algorithms. In *Proceedings of the Intl. Conf. on Tools with AI.* IEEE Computer Society Press, Boston, MA, 1993.

Wactlar, H. D., Kanade, T., Smith, M. A., and Stevens, S. M. Intelligent access to digital video: Informedia project. *IEEE Computer* 29 (3): 46–52, 1996.