XXXIII CONFERENZA ITALIANA DI SCIENZE REGIONALI

HIGH-DIMENSIONALITY IN SPATIAL DATA MINING: A SURVEY OF SPATIAL DATA STRUCTURES

G. Arbia[1], M. Tabasso[2]

SOMMARIO

Il presente lavoro pone l'attenzione su alcune tecniche di Spatial Data Mining e Machine Learning presenti in letteratura, considerando il fenomeno della "curse of dimensionality" che caratterizza gli spatial databases e che rende inefficace i processi di pattern recognition e di knowledge discovery.
In particolare si considera la classe degli "space-partitioning trees" (PCA trees, dyadic trees, k-d trees, RP-trees) e una versione del CART (Classification and Regression Tree) basata su un indice di eterogeneità spaziale.
Infine, vengono analizzati alcuni algoritmi di clustering, quali lo "spectral clustering" e "algoritmi basati sulla densità" (DBSCAN) mettendo in evidenza la relazione con gli spatial partitioning trees in termini di efficienza e complessità computazionale.

[1] University "Cattolica del Sacro Cuore", Faculty of Economics- Via F. Vito, 1-00168 Rome - email: giuseppe.arbia@rm.unicatt.it

[2] University "Sapienza", Doctoral School of Economics, Department of Economic and Social Analyses- Piazzale A. Moro, 5-00161 Rome email :myriam.tabasso@uniroma1.it

1. INTRODUCTION

Datamining and Knowledge discovery has become an active research area in the commercial and scientific communities. Dealing with huge volume of data is of primary concern in knowledge discovery from spatial data. Spatial data are multi-sourced, multi-typed, multi-scaled, heterogeneous, dynamic. Therefore, knowledge mining in spatial data is much more complicated than general data mining and knowledge discovery. Spatial data mining and knowledge discovery allow the efficient extraction of hidden, implicit, interesting, previously unknown, potentially useful, ultimately understandable, spatial or non-spatial knowledge (rules, regularities, patterns, constraints) from incomplete, noisy, fuzzy, random and practical data in large spatial databases. Basic tasks of spatial data mining are:

- **classification**: intends to discover in data hypersurfaces that can classify spatial objects into pre-specified classes; this piece of knowledge can be a separating function that can classify objects into classes according similarities in characteristics. It can also be a set of classification rules stipulating how different spatial objects can be assigned to pre-specified classes;
- **association rules**: the knowledge task aims at the identification of patterns of spatial association of certain phenomena: spatial dependence of data, local and global issue of spatial association or relationships;
- **clustering**: is the basic task by which structures can be discovered as clusters in spatial data. The general interest is to discover the partitioning or covering of space by clusters representing a particular spatial structure;
- **process** is to discover underlying processes that generate time series manifesting the dynamics of certain spatial data: trends, cycles or irregular occurrences.

In this context, we will deal a fundamental issue that regards the high dimensionality in spatial data: the "*curse of dimensionality*". This term originally introduced by Bellman (Bellman, 1961) is nowadays commonly used in many fields to refer to challenges posed by high dimensionality of data space.

In data mining, in large datasets with many potential predictor variables, this expression describes the problem that increases as more variables are added to a model. In particular, spatial datasets are characterized by large size (e.g., millions of observations), high dimensionality (e.g., hundreds of variables), and complexity (e.g.,heterogeneous data sources, space–time dynamics, multivariate connections, explicit and implicit spatial relations and interactions).

In many cases the data present a *low intrinsic dimension* and in this paper we present how a nonparametric statistical methods are adaptive to the intrinsic dimension of the data. The rest of paper is organized as follows. In the Section 2, we present the class of space-partitioning trees

and the theoretical support of these methods. In the Section 3 we briefly review the Classification and Regression Trees (CART) and we discuss the particular version of CART based on spatial heterogeneity index. Section 4 presents the discussion about some clustering algorithms: spectral clustering and density based algorithm for discovering clusters (DBSCAN). Furthermore, we describe the relationship with RP-trees in terms of efficiency and computational complexity. Finally, section 5 provides the conclusions and future works.

2. SPATIAL PARTITIONING TREE

In this section we present the class of spatial partitioning trees and its interaction with the Nearest Neighbor searching.

A *spatial partitioning tree* recursively divides space into increasingly fine partitions. There are many types of spatial partitioning trees, such as k-d trees, dyadic trees, PCA trees, random projection trees (RP-Trees). These spatial structures depend on different split coordinate is choosen at each stage. The most populare data structure is k-d tree, which partitions $\mathbb{R}^D$ into hyperrectangular cells. It is built in a recursive manner, splitting along one coordinate direction at a time (Figure 1). The succession of splits corresponds to a binary tree whose leaves contain the individual cells in $\mathbb{R}^D$. These trees are among the most widely-used spatial partitionings in machine learning and statistics. To understand their application, consider Figure 1, and suppose that the dots are points in a database, while the cross is a query point q.

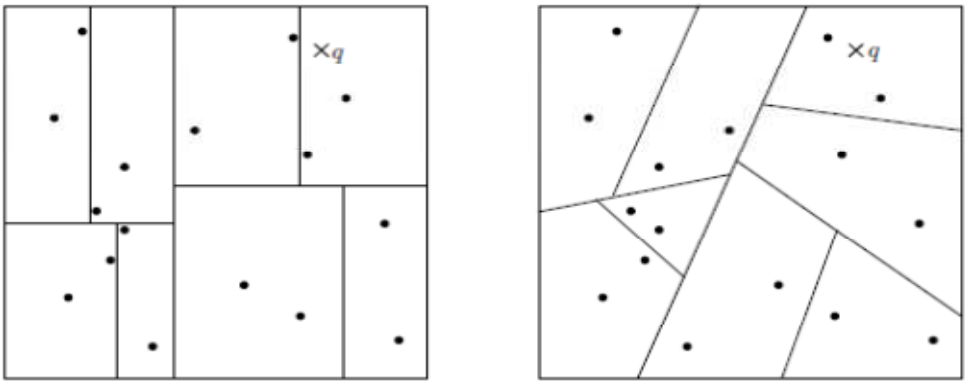

Figure 1: Left: A spatial partitioning of R^2 induced by a k-d tree with three levels. The dots are data points; the cross marks a query point q. Right: Partitioning induced by an RP tree (S.Dasgupta and Y.Freud, 2008).

The cell containing q, henceforth denoted cell (q), can quickly be identified by moving q down the tree. If the diameter of cell (q) is small (where the diameter is taken to mean the distance

between the furthest pair of data points in the cell), then the points in it can be expected to have similar properties, for instance similar labels. In classification, q is assigned the majority label in its cell, or the label of its nearest neighbor in the cell. In regression, q is assigned the average response value in its cell. In vector quantization, q is replaced by the mean of the data points in the cell. Naturally, the statistical theory around k-d trees is centered on the rate at which the diameter of individual cells drops as you move down the tree. For k-d trees, these cell diameters can shrink very slowly when the data is high-dimensional. For D-dimensional data, it may require D levels of the tree in order to halve the cell diameter. Thus the k-d trees suffer from the same curse of dimensionality as other nonparametric statistical methods. Recent works in machine learning have showed that a lot of data which superficially lie in a very high-dimensional space $\mathbb{R}^D$ actually have low intrinsic dimension, in the sense of lying close to a manifold of dimension $d \ll D$ (S.Dasgupta and Y.Freud, 2008). The k-d trees doesn't adapt to intrinsic low dimensional structure, so we review some variants of k-d trees that possess this property.
Spatial partition trees conform to a simple template:

Procedure `PartitionTree`(*dataset* $A \subset \mathcal{X}$)

if $|A| \leq MinSize$ **then**
  return leaf
else
  $(A_{\text{left}}, A_{\text{right}}) \leftarrow SplitAccordingToSomeRule(A)$
  $LeftTree \leftarrow PartitionTree(A_{\text{left}})$
  $RightTree \leftarrow PartitionTree(A_{\text{right}})$
return $(LeftTree, RightTree)$

Figure 2: Spatial partition tree (N. Verma, S. Kpotufe and S. Dasgupta, 2009).

Different types of trees are distinguished by their splitting criteria (N. Verma, S. Kpotufe and S. Dasgupta, 2009) :

- **Dyadic tree**: pick a coordinate direction and splits the data at the midpoint along that direction. One generally cycles through all the coordinates as one moves down the tree.
- **k-d tree (k-dimensioanal tree)**: pick a coordinate direction and splits the data at the median along that direction. One often chooses the coordinate with largest spread.
- **Random Projection (RP) tree**: split the data at the median along a random direction chosen from the surface of the unit sphere.

- **Principal Direction (PD or PCA) tree**: split at the median along the principal eigenvector of the covariance matrix.
- **Two Means (2M) tree**: pick the direction spanned by the centroids of the 2-means solution, and split the data as per the cluster assignment.

2.1 Notion of intrinsic dimension and diameter

In this subsection, we present a theoretical support of these methods: notion of ***intrinsic dimension*** and ***diameter***. Let X denote the space in which data lie and it is a subset of $\mathbb{R}^D$ and the metric of interest is Euclidean distance. There are many different definitions of intrinsic dimension: covering dimension, Assouad (or doubling) dimension, manifold dimension, local covariance dimension.

1. **Covering dimension**: a subset $X \subset \mathbb{R}^D$ has covering dimension d if there is a constant $C > 0$ such that for any ε, X has an ε-cover of size $C(1/\varepsilon)^d$.
2. **Manifold dimension:** a d-dimensional manifold is a subset $\mathbf{M} \subset \mathbb{R}^D$ such that for each $x \in M$, there is an open neighborhood around x, N(x) and a diffeomorphism $f : N(x) \rightarrow \mathbb{R}^d$.
3. **Assouad (doubling) dimension**: a subset $X \subset \mathbb{R}^D$ has doubling dimension d if for any (Euclidean) ball $B \subset \mathbb{R}^D$, $X \cap B$ can be covered by 2^d balls of half the radius.

   These notions don't capture the intrinsic dimensionality of data, especially for the empirical verification: given a sample of points drawn from an underlying distribution P, it is not easy to check whether P is concentrated near a low –dimenisonal manifold or near a set of low Assouad dimension. We introduce a notion of local covariance dimension.
4. **Local covariance dimension**: a subset $X \subset \mathbb{R}^D$ has covariance dimension (d, ε, r) if its restriction to any ball of radius r has covariace matrix whose largest d eigenvalues satisfy:

$$\lambda_1^2 + ... + \lambda_d^2 \geq (1-\varepsilon)(\lambda_1^2 + ... + \lambda_D^2).$$

where $(\lambda_1^2 + ... + \lambda_D^2)$ denote the eigenvalues of the covariace matrix; there are the variances in each of the eingenvalues directions. Therefore, we say that a subset $X \subset \mathbb{R}^D$ has local covariance dimension (d, ε, r) if neighborhoods of radius r have $(1-\varepsilon)$ fraction of their variance concentrated in a d-dimensional subspace.

Intuitively, the local covariance condition lies somewhere between manifold dimension and assouad dimension, although it is more general in that it merely requires points to be close to a locally flat set.

Based on the training set, we will construct a partition A of X and we will built a piecewise –

constant estimator f on the cells on this partition. It standard to decompose the error of estimator in two parts:

- **bias** ≡ how much does f vary within a single cell?
- **variance** ≡ what is the error in estimating the mean value of f within a cell?

The bias can controlled by making sure cells are small. The variance can be controlled by making sure cells are large enough that they contain many data points.
Traditionally the analysis of bias is based on physical diameters of cells $A \in \boldsymbol{A}$,

$$\Delta(A) \doteq \max_{x,x'} \|x - x'\|$$

This kind of diameter is hard to analyze for general convex cells, so we consider more flexible notions of diameter that measure the diameter of data within the cell.
For any cell A, we have two types of *data diameter:* the maximum distance between data points in A, denoted $\Delta(A)$, and the average interpoint distance among data in A, denoted $\Delta_a(A)$ (Figure 3).

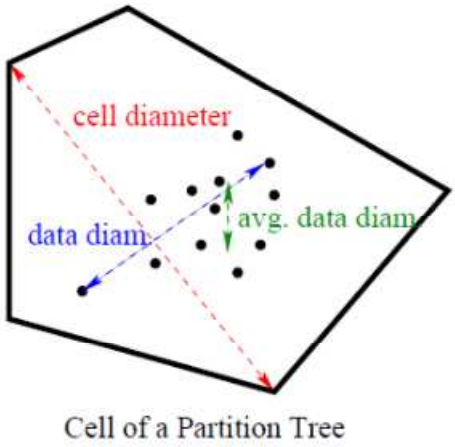

Figure 3: Various Notions of Diameter (N. Verma, S. Kpotufe and S. Dasgupta, 2009).

Let $X = \{X_1, \ldots, X_n\}$ be a data set drawn from $\mathcal{X}$, and let μ be the empirical distribution that assigns equal weight to each of these points. Consider a partition of $\mathcal{X}$ into a collection of cells $A \in \boldsymbol{A}$. For each such cell A, we can considere its maximul (data) diameter or its average (data) diameter; these are respectively,

$$\Delta(A) \doteq \max_{x,x' \in A \cap X} \|x - x'\|$$
$$\Delta_a(A) \doteq \frac{1}{(n\mu(A))} \left(\sum_{x,x' \in A \cap X} \|x - x'\|^2\right)^{1/2}$$

For a collection $\boldsymbol{A}$ of disjoint subset of $\mathcal{X}$, we use the following notion of average data diameter: (all over cells $A \in \boldsymbol{A}$).

$$\Delta(\boldsymbol{A}) \doteq \left(\frac{\sum_{A \in \mathbf{A}} \mu(A)\Delta^2(A)}{\sum_{A \in \mathbf{A}} \mu(A)}\right)^{1/2}$$

$$\Delta_a(\boldsymbol{A}) \doteq \left(\frac{\sum_{A \in \mathbf{A}} \mu(A){\Delta_a}^2(A)}{\sum_{A \in \mathbf{A}} \mu(A)}\right)^{1/2}$$

2.2 Splitting rules

In what follows, we assume the data lie in $\mathbb{R}^D$ and we review spatial data structures built by recursive binary splits. They differ only in the nature of the split, which we present in a subroutine called CHOOSERULE. The core tree-building algorithm is called MAKETREE, and takes as input a data set $S \subset \mathbb{R}^D$.

procedure MAKETREE (S)
 if |S| < *MinSize*[3] return (*Leaf*)
 Rule ← CHOOSERULE(S)
 LeftTree← MAKETREE ($\{x \in S : Rule(x) = true\}$)
 RightTree← MAKETREE ($\{x \in S : Rule(x) = false\}$)
 return ([*Rule, LeftTree, RightTree])*

The k-d tree CHOOSERULE picks a coordinate direction (typically the coordinate with largest spread) and then splits the data on its median value for that coordinate.

procedure CHOOSERULE (S)
 comment: **k-d tree version**
choose a coordinate direction *i*
$Rule(x) \coloneqq x_i \leq median(\{z_i : z \in S\})$
return *(Rule)*

A natural way to try building a manifold-adaptive spatial data structure is to split each cell along its principal component direction.

[3] Stopping criteria can be set with the follows arguments: *MinSize*, for the minimum node size, *maxdepth* for the maximum tree depth.

procedure CHOOSERULE (S)

comment: **PCA/PD tree version**

let *u* be the principal eigenvector of the covariace of S

$Rule(x) \coloneqq x \cdot u \leq median\,(\{z \cdot u : z \in S\})$

return *(Rule)*

This method is adapted to low intrinsic dimension, but it presents two significant drawbacks in practise. First, estimating the principal eigenvector requires a significat amount of data (about $1/2^k$ fraction of data winds up at a cell at level k of the tree). Second, when the extrinsic dimension is high the amount of memory and computation to compute data vectors and eigenvectors becomes the dominant part of the computation. As each node in the tree is likely to have a different eigenvector this severely limits the feasible tree depth.

The random projection trees overcomes these problem, but maintaing the adaptivity to low intrinsic dimension.

The RPTree chooses a direction uniformly at random from the unite sphere S^{D-1} and splits the data into two roughly equal-sized sets using a hyperplane orthognal to this direction.

We report an illustration of the RP-Tree algorithm (Figure 4) to understand better this procedure.

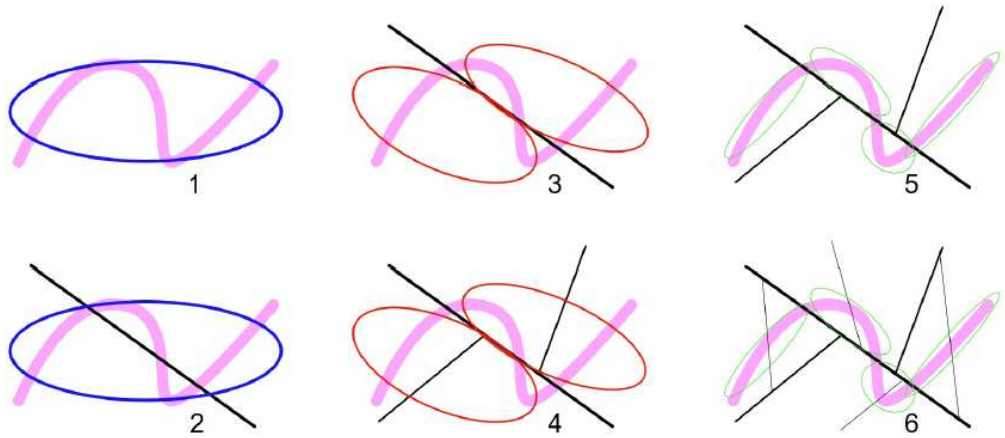

Figure 4: An illustration of the RP-Tree algorithm. 1: The full data set and the PCA ellipse that approximates it. 2: The first level split. 3: The two PCA ellipses corresponding to the two cells after the first split. 4: The two splits in the second level. 5: The four PCA ellipses for the cells at the third level. 6: The four splits at the third level. As the cells get smaller, their individual PCAs reveal 1D manifold structure. Note: the ellipses are for comparison only; the RP tree algorithm does not look at them (Y. Freund, S. Dasgupta, M. Kabra and N. Verma, 2007).

The RP-tree algorithm presents several advantages: the data requirements are reduced: only medians need to be estimated, not principal eigenvectos; the tree depth is more feasible: we use only k projection vectors, as opposed to 2^k with a PCA tree; the amount of memory and computation requirement for the training set is reduced because we can compute the k projection values before building the tree ad we can replace each high dimensional data point with its k projection values (in general we have $10 \leq k \leq 20$).

We describe two variants of RP-Tree, called respectvely *RPTree-Max* , *RPTree-Mean.* They are both adpative to intrinsic dimension (S. Dasgupta, Y. Freund,2008).

procedure CHOOSERULE (S)
comment: RPTree-Max version
choose a random unit direction v $\in R^D$
pick any x $\in$ S; let y $\in$ S be the farthest point from it
choose δ uniformly at random in $[-1, 1] \cdot 6\|x - y\|/\sqrt{D}$
$Rule(x) \coloneqq x \cdot v \leq median(\{z \cdot v : z \in S\}) + \delta)$
return (*Rule*)

procedure CHOOSERULE (S)
comment: RPTree-Mean version
if $\Delta^2(S) \leq c \cdot \Delta_a^2(S)$

$$\textbf{then} \begin{cases} \text{choose a random unit direction v} \\ Rule(x) \coloneqq x \cdot v \leq median(\{z \cdot v : z \in S\}) \end{cases}$$

$$\textbf{else} \{Rule(x) \coloneqq \|x - mean(S)\| \leq median \|z - mean(S)\| : z \in S\}$$

return (*Rule*)

In the RPTree-Mean version, in the first type of split, the data in a cell are projected into a random direction, instead the second type of split is based on the distance from the mean of the cell.

Now, we investigate the decrease in average diameter resulting from a given split depends just on the eignspectrum of the data in the local neighbourhood.

In the RP-Tree the data diameter of the cell decreases at a rate that depends only on the intrinsic dimension of the data, not D: "*Let d be the intrinsic dimension of data falling in a partitcular cell C of an RP tree. Then all cells O(d) levels below C have at most half data diameter of C "* (N. Verma, S. Kpotufe and S. Dasgupta, 2009).

For the RP-tree we remind two main results: suppose an RP tree is built from a data set $S \subset \mathbb{R}^D$, not necessarily finite. If the tree has k levels, then it partitions the space into 2^k cells. We define the radius of a cell $C \subset \mathbb{R}^D$ to be the smallest $r > 0$ such that $S \cap C \subset B(x, r)$ for some $x \in C$. The following theorem gives an upper bound on the rate at which the radius of cells in an RPTree-Max decreases as one moves down the tree.

THEOREM 1 (Dasgputa and Freud, 2008): *There is a constant c_1 with the following property. Suppose an RPTree-Max is built using data set $S \subset \mathbb{R}^D$. Pick any cell C in the RP tree; suppose that $S \cap C$ has Assouad dimension $\leq d$. Then with probability at least 1/2 (over the randomization in constructiong the subtree rooted at C), for every descendent C' which is more than $c_1\, d \log d$ levels below C, we have radius $(C') \leq$ radius $(C)/2$.*

The next theorem gives a result for RPTree-Mean that based on two different splits: *by distance* and *by projection*.

THEOREM 2 (Dasgputa and Freud, 2008): *There are constants $0 < c_1, c_2, c_3 < 1$ with the following property. Suppose an RPTree-Mean is built using data set $S \subset \mathbb{R}^D$. Consider any cell C of radius r such that $S \cap C$ has local coviariance dimension (d,ϵ,r) where $\epsilon < c_1$. Pick a point $x \in S \cap C$ at random, and let C' be the cell contains it at the next level down.*

- *If C is split by distance,* $\mathbb{E}\,[\,\Delta\,(S \cap C')] \leq c_2\,\Delta\,(S \cap C)$.
- *If C is split by projection, then* $\mathbb{E}\,[\,\Delta_a^2\,(S \cap C')] \leq (1 - (c_3/d))\,\Delta_a^2\,(S \cap C)$.

In both cases, the expectation is over the randomization in splitting C and the choice of $x \in S \cap C$. For PCAtree, the diameter decrease after a split depends on the local spectrum of the data. Let A be the current cell being split, and suppose the covariance matrix of data in A has eigenvalues $\lambda_1 \geq \ldots \geq \lambda_D$. If the covariance dimension of A is (d, ϵ), define

$$k \doteq \frac{1}{\lambda_1}\sum_{i=1}^{d}\lambda_i \qquad (1)$$

By definition, $k \leq d$.

The diameter decrease after the split depends on k^2, the worst case beign when the data distribution in the cell has heavy tails; in the absence of heavy tails (condition (2)) it depends just on k. This condition holds for any logconcave distribution (such as Gaussian or uniform distribution), for istance.

Proposition 1 (N. Verma, S. Kpotufe and S. Dasgupta, 2009): *There exist constants 0 <* c_1 , c_2 *<1 with the following property. Suppose* $\Delta^2(A) \leq c \cdot \Delta_a^2(A)$, *so that A is split by projection into* **A***={A₁, A₂} using PDtree split. If A*∩**X** has covariance dimension (d, c_1) then

$$\Delta_a^2(\boldsymbol{A}) < (1 - c_2/k^2)\, \Delta_a^2(A)$$

where k *is defined in (1).*
If in addition the empirical distribution on A ∩ **X** *satisfies (for any s* ∈ ℝ *and some* $c_0 \geq 1$*)*

$$\mathbb{E}_A [\, (X \cdot v - s)^2 \,] \leq c_0\, (\mathbb{E}_A [X \cdot v - s])^2 \qquad (2)$$

We obtain a faster decrease where

$$\Delta_a^2(\boldsymbol{A}) < (1 - c_2/k)\, \Delta_a^2(A)$$

For 2Mtree, the rule split by projection is based on the difference between the mean over the bisection of A, the direction v*= mean(A₁) –mean(A₂)* , the threshold is represent by the half point between the two means.
The 2-means cost can be written as

$$\sum_{i\in[2]} \sum_{x\in A_i \cap X} \|x - mean\ (A_i)\|^2 = \frac{n}{2} \Delta_a^2(\boldsymbol{A})$$

Thus, the 2Mtree (assuming an exact solver) minimizes$\Delta_a^2(\boldsymbol{A})$. It decreases diameter at least as fast RPtree and PDtree.

Proposition 2 (N. Verma, S. Kpotufe and S. Dasgupta, 2009): *Suppose* $\Delta^2(A) \leq c \cdot \Delta_a^2(A)$, *so that A is split by projection into A={A₁, A₂} using 2Mtree split. There exist constants 0 <* c_1 , c_2 *<1 with the following property. Assume A*∩X has covariance dimension *(d,* c_1*). We then have*

$$\Delta_a^2(\boldsymbol{A}) < (1 - c_2/d')\, \Delta_a^2(A)$$

where $d' \leq min\{d, k^2\}$ *for general distribution, and* d' *is at most k for distribution satisfy (2).*

The diameter decrease parameters d, k^2, k, d' in the previous proposition are a function of the covariance dimension of the data in the cell A being split. The covariance dimensions of the cell may over the course of the split implying that the decrease rates may vary.

The axis-parallel splitting rules (k-d tree or dyadic tree) do not always adapt to data that is intrisically low-dimensional: a data set in $\mathbb{R}^D$ that has Assouad dimension O (logD), k-d trees (and dyadic trees) require D levels to halve the data diameter.

The adaptivity of axis-parallel rules to covariance dimension is unclear, but the decrease diameter depends on D. The following proposition states that it takes at most *O* (*D log D*) (log(1/ε)) levels to decrease average diameter to an ε fraction of the origianl data diameter.

Proposition 3 (N. Verma, S. Kpotufe and S. Dasgupta, 2009): Suppose a partition tree is built using either k-d tree or dyadic tree by cycling through the coordinates. Let A_l be the partition of $\mathcal{X}$ defined by the nodes at level l. Then we have

$$\Delta_a^2(\boldsymbol{A}_l) \leq \Delta^2(\boldsymbol{A}_l) \leq \frac{D}{2^{\lfloor l/D \rfloor}} \Delta^2(\mathcal{X})$$

On the other hand, the irregular splitting rules (RP, PD, 2M trees) always adapt to intrinsic dimension. They therefore tend to perform better on real world tasks.

We summarize the diameter decrease rate (k) for the types of spatial partitioning analyzed in this section (smallest k such that data diameter is haveld every k levels):

- RPTree: $k \leq O(d)$
- PD Tree: $k \leq O\left(\sum \frac{\lambda_i}{\lambda_1}\right)^2$
- 2M Tree: $k \leq O\left(min\left\{ d, \left(\sum \frac{\lambda_i}{\lambda_1}\right)^2 \right\}\right)$
- Dyadic/ Kd Tree: $k \leq O$ (*D log D*)

It is important to underline that the RP-Tree plays a crucial role in algorithms for fast approximate nearest neighbor searches (see section 2.3) and clustering (see section 4.3).

2.3. Approximation Nearest Neighbor Algorithm

A large part of the benefit of RP trees comes from the use of random unit directions, which is rather like running k-dtrees with a preprocessing step in which the data are projected into a random low-dimensional subspace. In fact, a recent experimental study of nearest neighbor algorithms observes that a similar pre-processing step improves the performance of nearest neighbor schemes based on spatial data structures. K-nearest neighbor computation has been widely used in spatial databases.

The k-nearest-neighbor searching problem is to find the *k* nearest points in a dataset $X \subset R^D$ containing *n* points to a query point $q \in R^D$, usually under the Euclidean distance. It has applications in a wide range of real-world settings, in particular pattern recognition, machine learning and database querying.

In presence of high dimensionality the problem is redefined in terms of *approximate searching*. One approach is based on the (1+ε) approximate k-nearest-neighbor searching problem, which

returns points whose distance from the query is no more than (1+ ε) times the distance of the true k^{th} nearest-neighbor.

The most widely used algorithm for nearest-neighbor search is the kd-tree (Freidman et al., 1977), which works well for exact nearest neighbor search in low dimensional data, but quickly loses its effectiveness as dimensionality increases.

To solve this question, one approach is based on random projection trees that allow to deal with the high-dimensional datasets. In the (1+ ε) NN search algorithm, the random projection as a pre-processing step: project the datapoint to a subspace of lower dimension and then do apply the hybrib sp-tree search (for more details see *Liu T. et al., 2005*).

3. SPATIAL CART

We briefly recall some general background on Classification and Regression Tree (CART).

Classification and regression tree has been an important data mining methodology for the analysis of large data sets via binary partitioning procedure (Breiman et al, 1984). It consists in recursive division of N cases on which a response variable and a set of predictiors are observed. Such a partitioning procedure is know as regression tree when the response variable is continuously valued and as a classification tree when the response variable is categorical. A classification tree procedure provides not only a classification rule for new cases of unknow class, but also an analysis of the dependence structure in large data sets.

Figure 5 depicts a simple tree structure with tree layers of nodes.

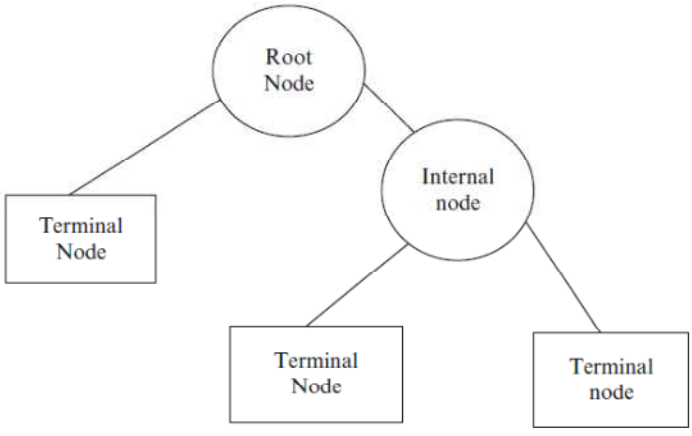

Figure 5: A simple tree structure (Y. Leung, 2010).

The root node contains the entire learning sample and the other nodes correspond to subgroups of the learning sample. The two subgroups in the left and right offspring nodes are disjoint, and their union comprises the subgroups for the parent node. A critical step of the tree-based technique is to determine the split from one parent node to two offspring nodes.

Let (X,Y) be a multivariate random variable where X is the predictor vector ($X_1, \dots, X_m, \dots X_M$) where $X_1, \dots, X_m, \dots, X_M$ (m=1,..., M) can be a mixture of ordered and categorical variable: and Y is the criterion variable taking values in the set of prior classes $\Gamma = \{1, \dots, j, \dots, J\}$.
Four elements are needed in the classification tree growing procedure:

1. A set of binary questions of the form { $is\ X \in A$? }.
2. A goodness of split criterion $\Delta i\ (s|t)$ that can be evaluated for any split s of any node t.
3. A splitting termination rule.
4. A rule for assigning every terminal node to a class.

For each ordered variable X_m, all questions in a set of binary questions are of the form { is $X_m \leq$ c?} for c ranging over (-∞, ∞).
If X_m is categorical taking values, say, in $\{b_1, b_2, \dots b_u\}$, then all questions in a set of binary questions are questions of the form $\{is\ X \in s?\}$, as s ranges over all nontrivial subset of $\{b_1, b_2, \dots b_u\}$.
The set of binary questions generates a set Q of splits s of every node t. For those cases in t answering "yes" to a question will go to the left descendant node $\{t_L\}$ and those answering "no" will go to the right descendant node $\{t_R\}$.
The goodness of split is measured by an impurity function defined for each node. Intuitively, we want each leaf node to be "pure", that is, one class dominates.

Definition 1 (Breiman et al., 1984): *An **impurity function** is a function ∅ defined on the set all J-tuples of numbers (p_1,..., p_J) satisfying $p_j \geq 0$, j=1, . .., J, $\sum_j p_j = 1$ with the properties*

(i) ∅ is a maximum only at the point $(\frac{1}{j}, \frac{1}{j}, \dots, \frac{1}{j})$

(ii) ∅ achieves its minimum only at the point (1,0 , …, 0), (0,1,… ,0), …, (0,0,..,0,1)

(iii) ∅ is a symmetric function of p_1,..., p_J.

Definition 2 (Breiman et al., 1984): *Given an impurity function ∅, define the **impurity measure** $i(t)$ of any node t as*

$$i(t) = \emptyset\ p(1|t), p(2|t), \dots, p(J|t)$$

If a split s of a note t sends a proportion p_R of the data cases in t to $t_R$and proportion p_L to t_L , define the ***decrease in impurity*** to be

$$\Delta\ i\ (s,t) = i(t) - p_R\ i(t_R) - p_L\ i(t_L)$$

where p_R and p_L are the proportions of the samples in node t that go to the right node t_R and the left node t_L respectively.

Definition 3 (Breiman et al., 1984): *the **Tree impurity** I (T) is defined by*

$$I(T) = \sum_{t \in \tilde{T}} I(t) = \sum_{t \in \tilde{T}} i(t)\, p(t)$$

where $\tilde{T}$ denote the set of terminal node.

The most popular splitting rules are the Entropy and the Gini index.

The **Entropy index** is

$$H(t) = \sum_j p(j|t) \log\{ p(j|t)\}$$

The **Gini index** is

$$D(t) = \sum_{j=1}^{J} p(j|t)\, p(i|t) = 1 - \sum_{j=1}^{J} p^2(j|t)$$

Both indices are equal to 0 when there is only class present in leaf t and maximum when all classes are present equal probabilities.

Let be $(X^1, \dots, X^p, Y)$ an independent sample of random variables, where $X^k$is the explanatory variables and Y is categorical variable to explained. The final tree overfits the available data and the **prediction error** $\boldsymbol{R(T) = P\{ T\,(X^1, \dots, X^p) \neq Y\}}$ is typically large. In designing a classification tree, the ultimate goal is to produce from the data a tree T whose probability of prediction error $R(T)$ is as small as possible. Thus, in second stage the tree T is "pruned" to produce a subtree T' whose expected performance is superior to $R(T)$.

When the data are independent samples the proportion $p(j|t)$ is estimated by $\hat{p}\,(j|t) = n_{jt}/n_t$, where n_{jt} is the number of samples in leaf t that are in class *j*, and n_t is the total number of samples in leaf *t*.

Definition 4 (L. Bel, D. Allard, J.M. Laurent, R. Cheddadi, A. Bar-Hen, 2009) The ***empirical risk*** is

$$\hat{R}(T) = \frac{1}{n} \sum_{\alpha=1}^{n} \mathbb{I}\{ T\,(X_\alpha^1, \dots, X_\alpha^p) \neq Y_\alpha\}$$

where $\mathbb{I}\,(\cdot)$ is the indicator function and n the total number of samples.

In the case of spatial data, the assumption related to samples independence is not acceptable because these data generally exhibit strong dependence due to their possible proximity.

We present two approaches to adapt CART to the case of spatially dependent samples. The first one considers the irregularity of the sampling by weighting the data according their spatial pattern. The idea is to "decluster" the data based on the Kriging method.

A second approach that uses spatial estimates of the quantities involved in the construction of the discriminant rule (L. Bel, D. Allard, J.M. Laurent, R. Cheddadi, A. Bar-Hen, 2009).

Let be $\{ X^1(s_\alpha), \dots, X^p(s_\alpha), Y(s_\alpha)\}$, $\alpha = 1, \dots, n$ the samples that are originated from random fields $\{X^1(\cdot), \dots, X^p(\cdot), Y(\cdot)\}$ on some domain $\mathcal{D} \epsilon \mathbb{R}^2$ and explicity take into account the dependence structure on these fields.

The basic idea of the first approach is to weight the samples such that clustered data have less weight than sparse data. In particular we have:

$$\hat{p}(j|t) = \sum_{\alpha \epsilon t}^{n} \mathbb{I}\{Y(s_\alpha) = i\}$$

and

$$\hat{R}(T) = \sum_{\alpha=1}^{n} w_\alpha \, \mathbb{I}\{T(X^1(s_\alpha), \dots, X^p(s_\alpha)) \neq Y(s_\alpha)\}, \text{ s.t. } \sum_{\alpha=1}^{n} w_\alpha = 1$$

For determining these wieghts we consider the method related to geostatistics (L. Bel, D. Allard, J.M. Laurent, R. Cheddadi, A. Bar-Hen, 2009).

If the covariance function $C(\cdot)$ of a random field $Z(\cdot)$ is know, the best linear unbiased predictor of a regional average on a 2d domain $\mathcal{D}$ is the so called **Kriging of a regional average** (or kriging of the mean if $\mathcal{D} \rightarrow \mathbb{R}^2$) . Let us denote $Z_\mathcal{D}$ the average $Z(\cdot)$ over $\mathcal{D}$. The kriging of $Z_\mathcal{D}$ is the quantity $\widehat{Z_\mathcal{D}} = \sum_\alpha w_\alpha \, Z(s_\alpha)$, such that $\mathbb{E}\left(\widehat{Z_\mathcal{D}} - Z_\mathcal{D}\right) = 0$ and $var\left(\widehat{Z_\mathcal{D}} - Z_\mathcal{D}\right)$ is minimum. It can be shown (Wackernagel, 2003) that the vector $W = (w_1, \dots, w_n)^T$ is the solution of the system

$$\begin{pmatrix} \boldsymbol{C} & \mathbf{1} \\ \mathbf{1}^T & 0 \end{pmatrix} = \begin{pmatrix} W \\ v \end{pmatrix} \begin{pmatrix} C_\mathcal{D} \\ 1 \end{pmatrix}$$

where **C** is the matrix whose α, β element is $C(s_\alpha, s_\beta)$, $C_\mathcal{D}$ is the vector with elements

$$C(s_\alpha, \mathcal{D}) = \frac{\mathbf{1}}{|\mathcal{D}|} \int_\mathcal{D} C(s_\alpha, s) \, ds$$

1 is a vector of ones of length n and v is the Lagrange parameter associated with the unbiasedness condition. To evaluate the last integral we define a grid G on $\mathcal{D}$ and use the following approximation

$$C(s_\alpha, \mathcal{D}) = \frac{\mathbf{1}}{|\mathcal{D}|} \int_\mathcal{D} C(s_\alpha, s) \, ds \;\cong\; \frac{\mathbf{1}}{n_G} \sum_{s_\beta \in G} C(s_\alpha, s_\beta).$$

This method consists in applying the kriging paradigm to the regional average of $Y(\cdot)$ and use the resulting kriging weights in the CART algorithm.

The solution of the adpated kriging system is

$$\min_W var\,(W^T Y - Y_\mathcal{D}) \quad \text{with } \mathbf{1}^T W = 1 \text{ and } w_\alpha \geq 0$$

where $Y = (Y_1, \dots, Y_n)^T$.

Finally, this approach allowes to reduce the bias of the regression tree by taking into account the spatial redundancy of the data.

Instead of simply introducing weights, a second approach consist in deriving spatial estimates for all quantities involved the algorithm: proportions in leaves, Gini index and empirical risk.

In the case of two classes, only one parameter describers the proportion of each class and the variance of the indicator of, say class 1, and hence

$$D = 2p(1-p) = 2\sigma^2$$

There are several possibilities for extending the definition of Gini index to spatial data but we report the approach followed by (L. Bel, D. Allard, J.M. Laurent, R. Cheddadi, A. Bar-Hen, 2009).

Consider a leaf t of the tree T. The theoretical proportion $p(j|t)$ of class j in t is the conditional probability $P\,(Y = j|X \in B_t) = \mathbb{E}\{\mathbb{I}\,(Y = j|X \in B_t)\}$, where B_t is thee subdomain of $\mathbb{R}^p$ corresponding to the leaf t. It can thus be estimates by kriging the spatial average of the variable $\mathbb{I}\,\{(Y(\cdot) = j|X\,(\cdot) \in B_t\}$ over the domain D_t defined as the set of location $s \in D$ such that $\left(X^1\,(s), \dots, X^p\,(s)\right) \in B_t$.

Applying the kriging approach on the estimation of $p(j|t)$ leads to

$$\widehat{p}\,(j|t) = \sum_{\alpha: X(s_\alpha) \in B_t}^{n} \lambda_\alpha \mathbb{I}\,\{\,Y(s_\alpha) = i\}$$

where λ_α is the solution of the system of n_t equations with $\alpha: X(s_\alpha) \in B_t$:

$$\sum_{\beta: X(s_\beta) \in B_t} \lambda_\beta C_j(s_\alpha, s_\beta) = \frac{1}{|D_t|} \int_{D_t} C(s_\alpha, s)\, ds \qquad (3)$$

under the constraint $\sum_\alpha \lambda_\alpha = 1$. In the above equations, $C_j(s, s')$ is the covariance function of $\mathbb{I}\,\{(Y(s) = j|X(s) \in B_t\}$.

The Gini index is then computed from the estimated proportions:

$$\widehat{D}_t = 1 - \sum_i \widehat{p}\,(i|t)^2 \qquad (4)$$

In this setting, the empirical risk is also estimated by kriging the spatial average of the variable $\mathbb{I}\,\{\,T\,(X(\cdot) \neq Y(\cdot)\}$ on D. Notice that the domain D_t is not known, it is approximated by the convex hull of the sample points lying in D_t and the integral is computed on the points of the grid G falling within that convex hull.

Proposition 4 (L. Bel, D. Allard, J.M. Laurent, R. Cheddadi, A. Bar-Hen, 2009).
For the model described above, let us denote D the population Gini index in $\mathcal{D}$, and $\widehat{D}$ its estimate computed from(4). Let us further denote n ($\mathcal{D}$) the number of samples in $\mathcal{D}$. Assume that the density of samples tends to a strictly positive quantity as the domain increases: $\frac{n(\mathcal{D})}{|\mathcal{D}|} \to \lambda > 0$ *as* $\mathcal{D} \to \mathbb{R}^2$. *Then, as* $\mathcal{D} \to \mathbb{R}^2$,

$$\mathbb{E}[\widehat{\mathrm{D}}] \to \mathrm{D}$$

Proof. The estimated proportion $\hat{p}_j$ are obtained from the solution of the kriging equation (3)

$$\hat{p}_j = Y^T \Lambda_j, \qquad \Lambda_j = C_j^{-1}\left(C_{j,\mathcal{D}} + \frac{1-\mathbf{1}\boldsymbol{C}_j^{-1}C_{j,\mathcal{D}}}{\mathbf{1}^T\boldsymbol{C}_j^{-1}\mathbf{1}}\mathbf{1}\right)$$

where $\boldsymbol{C}_j$ and $C_{j,\mathcal{D}}$ are the matrix that corresponding respectively to the left-hand side and right-hand side of 5. It is easy to show that $\mathbb{E}(\hat{p}_j) = p_j$ and $(\hat{p}_j{}^2) = \Lambda_j{}^T \boldsymbol{C}_j \Lambda_j$. Hence, we have:

$$[\widehat{\mathrm{D}}] = \mathrm{D} - 2\sum_{\mathrm{j}} \Lambda_j{}^T \boldsymbol{C}_j \Lambda_j \qquad (5)$$

After straightforward developments, each term of the sum in (5) is seen to be equal to twice $C_{\mathcal{D},j}{}^T \boldsymbol{C}_j^{-1} C_{\mathcal{D},j} + \{1 - (\mathbf{1}^T \boldsymbol{C}_j^{-1} C_{\mathcal{D},j})^2\}/\ \mathbf{1}^T \boldsymbol{C}_j^{-1}\mathbf{1}$. But, as $\mathcal{D} \to \mathbb{R}^2$, each element of the vector $C_{\mathcal{D},j} \to 0$ by the ergodic assumption; and $\mathbf{1}^T \boldsymbol{C}_j^{-1}\mathbf{1} \to \infty$ as $\mathcal{D} \to \mathbb{R}^2$ because the number of samples in $\mathcal{D}$ tends to infinity. Hence $\mathbb{E}[\widehat{\mathrm{D}}] \to D$ as $\mathcal{D} \to \mathbb{R}^2$.

4. A REVIEW SOME OF THE EXISTING ALGORITHMS FOR CLUSTERING HIGH-DIMENSIONALTY DATA: DENSITY-BASED ALGORITHMS FOR CLUSTERS DISCOVERING (DBSCAN) AND SPECTRAL CLUSTERING

4.1 A Density based notion of clusters

The key idea of a density-based cluster is that for each point of a cluster its neighborhood for some given radius has to contain at least a minimum number of points, i.e. the "density" in the Eps-neighborhood of points has to exceed some threshold (Ester et al. 1996). This idea is illustrated by the sample sets of points depicted in figure 6.

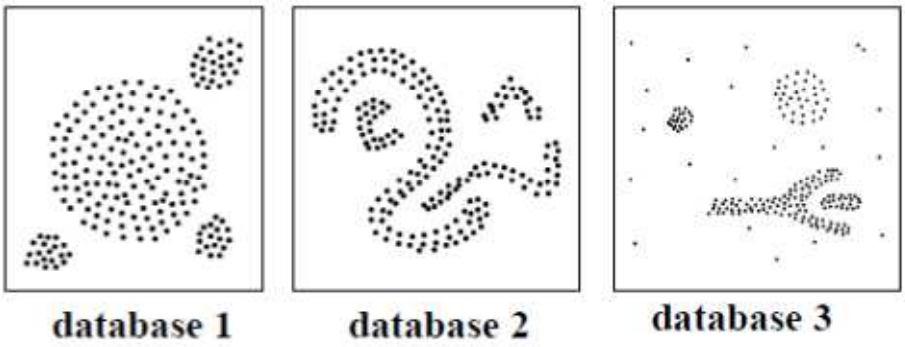

Figure 6: Simple databases (M. Ester, H.-P. Kriegel, J. Sander,1996)

In these examples, we can easily and unambiguously detect clusters of points and noise points not belonging to any of those clusters, mainly because we have a typical density of points inside the clusters which is considerably higher than outside of the clusters. Furthermore, the density within the areas of noise is lower than the density in any of the clusters.

In the following, we formalize the intuitive notion of "clusters" and "noise" in a database D of point of k-dimensional space S.

The shape of a neighborhood is determined by the choice of a distance function for two points p and q, denote by $dist(p,q)$. DBSCAN is a density based algorithm which discovers clusters with arbitrary shape and with minimal number of input parameters. The input parameters required for this algorithm is the radius of the cluster (Eps) and minimum points required inside the cluster (MinPts).

The following definitions are the key concepts of the DBSCAN algorithm.

Definition 5 (M. Ester, H.-P. Kriegel, J. Sander,1996) : *The **Eps-neighborhood** of a point p, denote by N_{Eps} (p) is defined by* $\mathrm{N_{Eps}}$ (p)= $\{q \in D | dist\,(p,q) \leq Eps\}$.

A naïve approach could require for each point in a cluster that there are at least a minimum number (*MinPts*) of points in an Eps-neighborhood of that point. This approach fails because there are two kinds of points in the cluster, the points which is inside the cluster (***core points***), and points on the border of the cluster (***border points***). In general, for every point p in a cluster C there is a point q in C so that p is inside of the Eps-neighborhood of q and N_{Eps} contains at least *MinPt*s points. This concept is elaborated in the following definition.

Definition 6 (M. Ester, H.-P. Kriegel, J. Sander, 1996): A point p is ***directly density-reachable*** from a point q with respect to Eps, MinPts if

1) $p \in \mathrm{N_{Eps}}$ (q) and
2) $[\mathrm{N_{Eps}}$ (q)] ≥ Minpts (***core point condition***).

Obviously, directly density-reachable is symmetric for pairs of core points. In general, however, it is not symmetric if one core point and one border point are involved.
Figure 7 shows the asymmetric case.

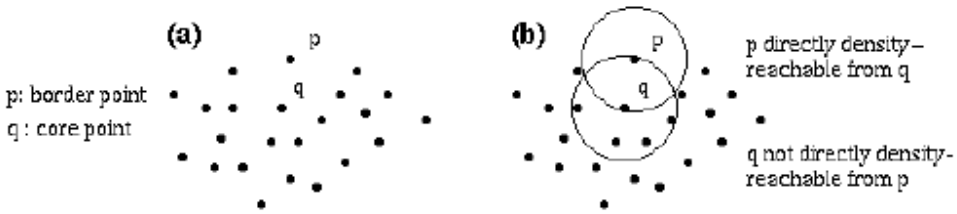

Figure 7: core points and border points (M. Ester, H.-P. Kriegel, J. Sander,1996)

Definition 7 (M. Ester, H.-P. Kriegel, J. Sander, 1996): A point p is ***density-reachable*** from a point q with respect to Eps and MinPts if there is a chain of points $p_1, \ldots, p_n$, $p_1 = q$, $p_n =$ p such that p_{i+1} is directly density-reachable from p_i.
Density-reachability is a canonical extension of direct density-reachability. This relation is transitive, but it is not symmetric.

Definition 8 (M. Ester, H.-P. Kriegel, J. Sander, 1996): A point p is ***density-connected*** to a point q with respect to Eps and MinPts if there is a point o such that both, p and q are density-reachable from o with respect to Eps and MinPts.
This relation is a symmetric relation and for density reachable points it is also reflexive.

Definition 9 (M. Ester, H.-P. Kriegel, J. Sander, 1996): Let *D* be a database of points. A ***cluster*** C with respect to Eps and MinPts is a non-empty subset of *D* satisfying the following conditions:

1) ∀ p, q: if p ∈ C and q is density-reachable from p wrt. Eps and MinPts, then q∈ C (**Maximality**);
2) ∀ p, q if p ∈ C : p is density-connected to q wrt. Eps and MinPts (**Connectivity**).

Definition 10 (M. Ester, H.-P. Kriegel, J. Sander, 1996): Let $C_1, \ldots, C_k$ be the clusters of the database D with respect to parameters Eps_i and $MinPts_i$, i=1, …, k. Then we define the ***noise*** as the set of points in the database D not belonging to any cluster C_i, i.e. noise={p ∈ D | ∀ i: p ∉ C_i).

4.2 The Algorithm

In this subsection, we describe the algorithm DBSCAN (Density Based Spatial Clustering of Applications with noise) that is designed to discover the spatial data clusters with noise. The steps involved in this algorithm are as follows: (i) select an arbitrary point; (ii) retrieve all points density-reachable from p with respect to *Eps* and *MinPts*; (iii) if p is a core point, a cluster is formed; (iv) if p is a border point, no points are density reachable from p and DBSCAN visits the next point of the database; (v) continue the process until all the points have been processed.
DBSCAN requires two input parameters (Minimum points and radius) and supports the user in finding an approximate value for it using *k-dist graph*.
Let d be the distance of a point p to its k-th nearest neighbor, then the d-neighborhood of p contains exactly k+1 points for almost all points p. The d-neighborhood of p contains more than k+1 points only if several points have exactly the same distance d from p which is quite unlikely.
The *k- dist* approach looks at the behavior of the distance from a point to its *kth* nearest neighbor. If *k* is not larger than the cluster size, the value of *k - dist* is small for points that belong to the same cluster. The *k- dist* for points not in the cluster is relatively large. The idea is to pick a value of *k* to be the *MinPts*. The following steps are performed to find the value of *k*:

- Compute the *k- dist*, (distance to its *kth* nearest neighbor) for each of the data points.
- Sort *k- dist* measures in increasing order.
- Plot the sorted *k-dist* values: this graph is called the sorted *k-dist graph*. We expect to see a sharp change at the value of k-*dist (threshold point)* that corresponds to a suitable value of *Eps*. If we select this distance as the *Eps* parameter and take value of *k* as the *MinPts* parameter, then points for which k-dist is less than *Eps* will be labeled as core points, while other points will be labeled or border points.

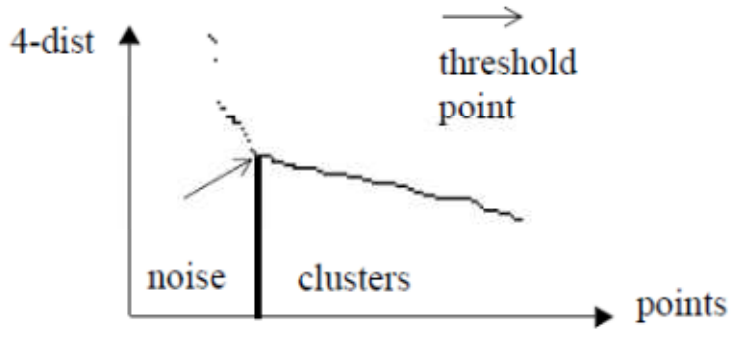

Figura 8: sorted 4-dist grapgh for sample database 3 (M. Ester, H.-P. Kriegel, J. Sander,1996)

Finally we observe that DBSCAN is very robust to outliers but it very sensible to the choise of values of these parameters (Eps, MinPts) and it is highly affected by the distance measure used in finding the distance between two points.
The algorithm fails to identify clusters if density varies and if the data set is too sparse.
It is very important underline the improvement performance of DBSCAN when we use spatial structures as Kd-trees.
The use of the Kd-tree data structure enables efficient computation of the k-nearest neighbours (k-NN) of a pattern point, particularly for large data. (Sushmita Mitra, Jay Nandy, 2011).
The basic time complexity of the DBSCAN algorithm is *O(m*time to find points in the Eps-neighborhood)* where m is the number of points. In the worst case time complexity of DBSCAN algorithm is $O(m^2)$. However, in low dimensional data, this time complexity can be reduced to O(m log m) using Kd- trees, that allow efficient retrieval of all points within a given distance of a specific point. The space requirement of DBSCAN, even high-dimensional data, is *O(m)* because it is only necessary to keep a small amount of data for each point i.e. the cluster label and the identification of each point as a core, or noise point (Pang-Ning Tan, Michael Steinbach and Vipin Kumar, 2005).

4.3 Spectral CLustering

In the current subsection we briefly introduce spectral clustering methodololghy and we present the fast spectral clustering based on RP-Tree, used in this case as a local data reduction step.
Clustering is a fondamental problem in data mining, statistical machine learning and scientific discovery. In particular in spatial data mining, a wide variety of methods have been devoloped to solve spatial clustering problem (Han et al., 2001).
Some clustering methods are strongly lied to Euclidean geometry, making explicit or implicit assumptions that clusters form convex regions in Euclidean space, spectral methods are more flexible and capturing a wider range of geometry.
Given a set of n data points $x_1, \dots x_n$, with each $x_i \in \mathbb{R}^d$, we define an *affinity graph* $\mathcal{G} = (V, E)$ as an undirected graph in which the i^{th} vertex corresponds to the data point x_i. For each edge $(i, j) \in E$, it is associated a weight a_{ij} that encodes the affinity (or similarity) of the data points x_i and x_j. The matrix $A = (a_{ij})_{i,j=1}^{n}$ is the *affinity matrix.*
The goal of the spectral clustering is to partition the data into m disjont classes such that each x_i belongs to one and only one class. Different spectral clustering formalize this partitioining problem in different way. We report the *normalized cuts* (Ncut) formulation **(**Donghui Yan, Ling Huang, Michael I. Jordan, 2009).

Define $W = (V_1, V_2) = \sum_{i \in V_1, j \in V_2} a_{ij}$ for two (possibly overlapping) subsets V_1 and V_2 of V. Let $V = (V_1, .., V_m)$ denote a partition of V, and consider the following optimization criterion:

Ncut= $\sum_{j=1}^{m} \frac{W(V_j, V) - W(V_j, V_j)}{W(V_j, V)}$, where the numerator in the j^{th} term is equal to the sum of the affinities on edges leaving the subset V_j and the denominator is equal to the total degree of the subset V_j. Minimizing the sum of such terms thus aims at finding a partition in which edges with large affinities tend to stay within the individual subset V_j and in which the size of the V_j are balanced.

The equation above is intractable and it can be rewriting as normalized quadratic form involving indicator vectors that then replaced with real-valued vector, resulting in a generalized eigenvector problem. The problem is redefined in terms of the (normalized) *graph Laplacian L* of *A* as follows:

$$L = D^{-1/2}(D - A)D^{-1/2} = I - D^{-1/2} A D^{-1/2} = I - L^0$$

where $D = diag(d_1, \dots d_n)$ with $d_i = \sum_{j=1}^{n} a_{ij}$, $i = 1, \dots, n$ and where the final equality defines L^0. Ncut is based on the eigenvectors of this normalized graph Laplacian.

A specific example of a spectral cluster algorithm based on Gaussian Kernel as the pairwise affinities is defined by:

Algorithm 1 Spectral Clustering (Donghui Yan, Ling Huang, Michael I. Jordan, 2009)

1. Compute the affinity matrix A with elements : $a_{ij} = \exp\left(-\frac{\|x_i - x_j\|^2}{2\sigma^2}\right)$ $i, j = 1, \dots, n$
2. Compute the diagonal degree matrix D with elements: $d_i = \sum_{j=1}^{n} a_{ij}$
3. Compute the normalized Laplacian matrix: $L = D^{-1/2}(D - A)D^{-1/2}$
4. Find the second eigenvector v_2 of L
5. Obtain the two partitions using v_2: $S = \{i : (v_2)_i > 0\}, \bar{S} = \{i : (v_2)_i \leq 0\}$

We observe that we have:

- **Input:** n data points $\{x_i\}_{i,j=1}^{n}$, $x_i \in \mathbb{R}^d$
- **Ouput**: Bipartition S and $\bar{S}$ of the input data

Vector quantitization is the problem of choosing a set of representative points that best represent a data set in sense of minimizing a distortion measure.

When we use the RP-tree as a local distortion-minimizing transformation, the algorithm is called "**RP-tree-based approximate spectral clustering**" (**RASP**) and it is obtained by:

Algorithm 2 RASP (Donghui Yan, Ling Huang, Michael I. Jordan, 2009)

1. Build an h-level random projection tree on $x_1, \ldots, x_n$; compute the centers of mass $y_1, \ldots y_k$ of the data points in the leaf cells as the k representative points.
2. Run a spectral clustering algorithm on $y_1, \ldots y_k$ to obtain an m-way cluster membership for each of y_i.
3. Recover the cluster membership for each x_i by looking up the cluster membership of the corresponding centroid y_j.

In the algorithm above we observe that we have:

- **Input:** n data points $\{x_i\}_{i,j=1}^{n}$, number of representative points k
- **Ouput**: m-way partition of input data

The total computational cost of this method is O(k^3) + O(hn) , where the O(hn) term arises from the cost of the bulding the h-level random projection tree.

The vector quantization error is calculated by the average *squared* Euclidean distance between a vector in the set and the representative vector to which it is mapped. This error is closely related (in fact, proportional) to the *average diameter* of cells, that is, the average squared distance between pairs of points in a cell. We remind that in RP-Tree the diameter of the cells for a vector quantization construction method depends on the intrinsic dimension, rather than the extrinsic dimension of the data.

The quantization error of the RP tree is characterized in terms of local covariance dimension as showed in the Theorem 2 (for more details view Section 2).

Finally, the latter Theorem shows that the vector quantization error of RP tree behaves as $e^{-O\left(\frac{h}{d}\right)}$ with h the depth of the tree and d the intrinsic dimension of the data. Thus the quantization error can be made small as the tree depth as grows.

6. CONCLUSIONS AND FUTURE WORKS

In this paper we have presented a briefly introduction to the notions of intrinsic dimension and diameter to analyze the class of space partitioning trees. Furthermore, we have examinated a framework of classification and regression trees and we have reviewed a particular version of CART based on the blend of geostatistic methodology such as Kriging and heterogeneity measure such as Gini index. We have showed the crucial role of space partitioning trees, that are more used in Machine Learning, to implement some spatial clustering methods and nearest neighbor algorithms. In future works, we would extend this version of CART in economic fields and show how it can supported the statistic or econometric spatial models.

Finally, we would investigate other tecniques of data mining such as grid based-methods, support

vector machine, random forest, boosting and neural networks, to study the improvement of the accuracy of classification or regression methods, taking account the complexity and the peculiarities that characterize the various types of spatial data (point data, continuous data, regular and irregular areas).

REFERENCES

Bel L., Allard D., Laurent J.M., Cheddadi R., Bar-Hen A. , CART algorithm for spatial data: Application to environmental and ecological data, *Computational Statistics & Data Analysis, Volume 53, Issue 8, 15 June 2009: 3082-3093*

Bellman Richard E., Adaptive Control Processes, A Guided Tour, *Princeton University Press, 1961*

Breiman L., Friedman J. H, Olshen R. A., Stone C. J., Classification and Regression Trees, *Wadsworth, Belmont, 1984*

Dasgupta S. and Freund Y., Random projection trees and low dimensional manifolds, *ACM Symposium on Theory of Computing (STOC), 2008*

Donghui Yan, Ling Huang, Michael I. Jordan: Fast approximate spectral clustering, KDD 2009: *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining (2009): 907-916*

Ester M., Kriegel H.-P., Sander J., Xu X., A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. *Proc. 2nd Int. Conf. on Knowledge Discovery and Data Mining, Portland, 1996*

Freund Y., Dasgupta S., Kabra M., N. Verma, Learning the structure of manifolds using random projections, *Neural Information Processing Systems (NIPS), 2007*

Friedman, J. H., Bentley, J. L., and Finkel, R. A. An algorithm for finding best matches in logarithmic expected time. *ACM Trans. Mathematical Software, 3(3):209–226, 1977*

Han J., Kamber M., Tung A. K. H., Spatial Clustering Methods in Data Mining: A Survey, *Geographic Data Mining and Knowledge Discovery, Taylor and Francis, 2001*

Y. Leung, Knowledge Discovery in Spatial Data, *Springer-Verlag Berlin Heidelberg, 2010*

Liu T., Moore A., Gray A., Yang K. , An investigation of practical approximate nearest neighbor algorithms, *Neural Information Processing Systems, 2005*

Pang-Ning Tan, Michael Steinbach and Vipin Kumar, Introduction to Data Mining, *Pearson Addison Wesley, 2005*

Sushmita Mitra, Jay NandyKDDClus, A Simple Method for Multi-Density Clustering, *Proceedings of the International Workshop on Soft Computing Applications and Knowledge Discovery (SCAKD 2011), Moscow, Russia, June 25, 2011- Vol. 758 : 72-76*

Verma N., Kpotufe S., Dasgupta S., Which spatial partition trees are adaptive to intrinsic dimension? *Uncertainty in Artificial Intelligence (UAI), 2009*

ABSTRACT

The widespread use of information technology, remote sensing and GIS (Geographic Information System) in several fields, is leading to a considerable usage of geo-spatial data. One of research study related to exploring of data with large volume Spatial Data Mining. It combines different disciplines: databases technology, artificial intelligence, machine learning, probability and statistics, visualization, information science, pattern recognition. Spatial Data Mining (SDM) is the process of discovering interesting and previously unknown, but potentially useful and reliable patterns from large spatial datasets. This process is more complex than conventional Data mining because of the complexities inherent in spatial data. The main difference between data mining and spatial data mining is that in spatial data mining tasks we use not only non-spatial attributes (as it is usual in datamining in non-spatial data), but also spatial attributes. Spatial database represent, store and manipulate spatial data such as points, lines, areas, surfaces and hypervolumes in multidimensional space. The existence of high dimensional datasets in spatial datamining affects pattern recognition and knowledge discovery, because the search space usually grows exponentially when the number of dimensions increases. This issue is known as "the curse of dimensionality". In this paper we present an overview of the methods in the literature to overcome this problem based on the general principle of "recursive decomposition". In particular we focus on two directions: 1) the class of space partitioning tree that includes PCA trees, k-d trees, dyadic trees, random projection trees (RP-tree); 2) a spatial version of classification and regression trees (CART). We investigate the curse of dimensionality inherent to these methods and their behavior when high dimensional datasets have low intrinsic dimension. Finally we review some of the existing algorithms for clustering high-dimensional data like spectral clustering and density-based algorithms for clusters discovering (DBSCAN) and we show the relationship with RP-tree in terms of efficiency and computational complexity.