Systems of Information Granules

Frank Höppner
University of Applied Sciences Braunschweig/Wolfenbüttel
Department of Economics
Robert Koch Platz 10-14
38440 Wolfsburg, Germany
e-mail: f.hoeppner@fh-wolfenbuettel.de

Frank Klawonn
University of Applied Sciences Braunschweig/Wolfenbüttel
Department of Computer Science
Salzdahlumer Str. 46/48
D-38302 Wolfenbuettel, Germany
e-mail: f.klawonn@fh-wolfenbuettel.de

Abstract

Finding a system of information granules corresponds to searching for the right level of abstraction to solve a problem at hand. In this chapter we discuss the purpose of such *partitions* and their desirable properties. The granules can be defined by exploiting expert knowledge, but they must be learned from data in case such knowledge is not available. We review several ways to derive one-dimensional and multidimensional partitions automatically from data with a special focus on utilizing clustering algorithms for this task.

Keywords: partition, clustering, discretization, quantization, granularization

1 Introduction

Collecting large amounts of data has become a routine in business, industry and science nowadays. However, in order to handle large amounts of raw data and extract and present the inherent information, techniques are required that provide compressed and compact representations of the aspects of interest. In

the extreme case a collection of measurements might be represented by a single value, for instance the mean or the median. Although at least one of the crucial characteristics of the data can be covered by a mean or median, such simplified concepts loose by far too much of the information contained in the data to be of practical use in applications.

The two cases – the raw data and a single value in the form of the mean or median – can be viewed as two extremes of granularities to represent the data. By treating all data items as single and separate objects, the finest possible granularity, loss of information is avoided for the price of difficulties in handling the data and the loss of interpretable compressed representations. On the other hand, a mean or median is easy to understand, but looses most of the information contained in the data. The truth lies somewhere in between these extremes. In order to handle, represent, and manage the data in an understandable and efficient way, information granules are introduced. An information granule – or simply granule – is a conceptual notion that has instances expressed in a data set [2, 41, 29]. In the simplest case, a granule is a subset of the data, for instance described by an interval. A typical form of such crisp granules results from a discretization of a continuous variable. Instead of considering exact values, ranges are introduced, i.e. the domain, for instance the interval $[a, b]$ is partitioned into k sub-intervals $[a, x_1)$, $[x_1, x_2)$,..., $[x_{k-1}, b]$ where $a < x_1 < x_2 < \ldots < x_{k-1} < b$.

Although in some cases, there might be a canonical way to choose the boundaries x_i for the intervals, such a discretization often causes problems, since values close to a boundary of one of the intervals, but lying on different sides of the boundary belong to different granules, although their difference might be extremely small. In order to avoid such problems, a reasonable approach is to give up the idea that an object must either belong or not belong to a granule. This leads to granules in the form of fuzzy sets or probability distributions. Also overlapping crisp granules might be considered, leading to rough sets [31], an approach that will not be considered in this chapter.

An information granule is more than just a collection or set of elements. It should be describable by a specific property or concept as in the example of discretization where a granule is defined by the lower and upper bound of the corresponding interval.

The choice of appropriate granules strongly depends on the specific purpose and application. In the fields of business intelligence and data warehouses, online analytical processing (OLAP) [39] exploits granules in a canonical way. OLAP provides views on a data set based on different levels of refinements. Most attributes handled within OLAP are categorical with an additional hierarchical structure. This means that the attribute can take values from a finite, refinable domain. For instance, an attribute representing a location might have different levels of refinement like country, state, region, town. Or an attribute for time might be refinable from year to quarters, months and days. As in this example, for categorical attributes there is often an obvious way to refine or coarsen them,

defining granules directly. This does usually not apply to attributes with a continuous domain. Notions like magnitude are good candidates for granules in the case of continuous attributes. The crucial problem is to define the appropriate granules suited best to the data and to the underlying task or purpose. Therefore, this chapter will focus exclusively on continuous attributes.

The purpose of the granules can be simply to describe important or interesting patterns or substructures in the data. A typical application is subgroup discovery [40], where the aim is to find granules in which a certain class of objects is significantly over- or underrepresented compared to the overall distribution of the class in the data set. However, in most applications it is important to cover more or less the whole data set by granules. In order to avoid redundant information, the granules should not overlap or, at least not too much. In this sense, the granules should roughly represent a partition, not necessarily in the strict mathematical sense.

With this motivation in mind, the chapter is organized as follows. Section 2 provides an overview on the possible contexts and purposes in which granularization is of interest. Section 3 discusses useful and important properties of partitions induced by granules. Section 4 is devoted to a short introduction to cluster analysis, one of the main techniques to construct granules from data in an unsupervised context. Many similarity-driven approaches to granularization, which are described in section 5, rely on clustering methods. Finding granules in a supervised learning context is outlined in section 6 before the summary in the conclusions.

2 Purpose of Granularization

The general problem of finding granules in the context considered here, can be described as follows. Given a data set or a universe of discourse $D \subset \mathbb{R}^p$, how can we cover or partition this domain by a finite set of meaningful concepts?

First of all, the overall purpose of the granularization should be specified. In the case of exploratory data analysis, where one of the main aims is to discover interesting or meaningful structures in the data set, the granules are used for compact descriptions, groupings or segmentations of objects or data. Typically, the granules are found on the basis of a similarity or distance measure. The identification of meaningful substructures without specifying in advance which objects should belong to a certain substructure is called unsupervised learning.

Supervised learning refers to a context, where the prediction of an attribute based on other attributes is the final goal. Regression refers to the case, when the attribute to be predicted is a numerical one, whereas the term classification is used for problems where the attribute to be predicted is of categorical nature. In other words, in supervised learning we want to describe a classification or regression function $f : D \to Y$ where Y is a finite set in the case of classification

and $Y \subseteq \mathbb{R}$ in the case of regression. The granules are involved in the description of the function f. Nevertheless, the task of finding suitable granules is only indirectly supervised in the sense that the granules should support the definition of a suitable function f as much as possible. The granules are found by optimising f. In this sense, the identification of the granules is task-driven, guided by the specific classification or regression problem.

It should be noted that, even in the context of supervised learning, it is sometimes popular to identify granules in an unsupervised manner, ignoring the attribute to be predicted. After the granules are identified in this purely unsupervised way, they are then used for the purpose of supervised learning. However, there is no reason why granules, derived for descriptive purposes, should also perform well in a specific prediction task.

Granules can be defined by a human expert or determined automatically based on an available data set. When the granularization is carried out completely by a human expert, a suitable formal language, model or tool to describe the granules is required. This will not be the focus of this contribution. We will only provide guidelines that should be taken into account when designing granules as well as an overview on techniques to determine granules automatically based on data.

3 Properties of Partitions

As mentioned before, we are not only interested in isolated granules, but in granules that can cover the domain of interest roughly in terms of a partition. There are many ways of organising a partition, some of the aspects that may be taken into account will be addressed in this section. All these aspects or properties influence both, the suitability of the partition for a given task as well as the interpretability, so it depends on the application and the purpose of the granularization (see section 2) which of these properties a partition should possess.

3.1 Degree of Uncertainty

The traditional notion of a partition is the following: Given a data set $D = \{x_1, .., x_n\}$ and a number of subsets C_i, $i = 1..c$, the system of sets $\{C_i \,|\, i = 1..c\}$ is called a partition, if the following properties hold: the elements of a partition are non-empty, pairwise disjoint and cover the whole data set.

$$\forall i \in \{1, .., c\} \qquad C_i \neq \emptyset \tag{1}$$

$$\bigcup_{i=1..n} C_i = D \tag{2}$$

$$\forall i, j \in \{1, .., c\} \quad C_i \cap C_j = \emptyset \tag{3}$$

Any element of D belongs to exactly one element of the partition, there is no ambiguity about the data-granule relationship. Although convenient from a mathe-

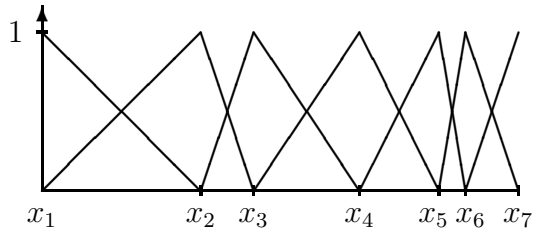

Figure 1: A typical fuzzy partition.

matical point of view, many concepts in the real world are not that strict. Sometimes an object may belong to two granules equally likely. In this case, condition (3) is removed and we speak of an *overlapping partition*.

As soon as uncertainty comes into play, one may want to express the degree of uncertainty in a more expressive way. We reformulate the above mentioned properties by a characteristic function $\chi_{C_i} : D \to \{0, 1\}$, where $\chi_{C_i}(x_j) = 1 \Leftrightarrow x_j \in C_i$. Rather than considering a binary decision $\chi_{C_i}(x_j) \in \{0, 1\}$ we then consider the belongingness to C_i as a matter of degree – as a fuzzy set determined by a membership function $\mu_{C_i} : D \to [0, 1]$ with the unit interval as its range.

The definition of a *fuzzy partition* is not as obvious as the generalisation of the notion of a crisp set to the concept of a fuzzy set. In most practical applications, where fuzzy sets are used as granules to describe a domain, these fuzzy sets satisfy the condition that they are disjoint (cf. (3)) with respect to the Łukasiewicz t-norm. The Łukasiewicz t-norm is defined by $\alpha \odot \beta = \max\{\alpha + \beta - 1, 0\}$ and can be viewed as one of many possible choices for a $[0, 1]$-valued conjunction or intersection operation. Two fuzzy sets μ_1 and μ_2 are disjoint with respect to the Łukasiewicz t-norm, if $\mu_1(x) + \mu_2(x) \leq 1$ holds for all x in the considered domain.

The covering property (2) is very often satisfied with respect to the dual t-conorm to the Łukasiewicz t-norm, the bounded sum defined by $\alpha \oplus \beta = \min\{\alpha + \beta, 1\}$. In other words, the sum of membership degrees should add up to at least one for all elements in the considered domain.

Figure 1 shows a typical fuzzy partition that satisfies the disjointness property with respect to the Łukasiewicz t-norm and the covering property with respect to the bounded sum. These two conditions will be satisfied by any fuzzy partition on an interval where the membership degrees of two neighbouring fuzzy sets always add up to one and at most the supports of two fuzzy sets overlap in every point.

Fuzzy partitions can either be defined on the domain or on the considered data. In the latter case, the membership degrees of the data objects $x_1, .., x_n$ to the granules $C_1, .., C_c$ can be defined by a membership matrix $U = [u_{i,j}]_{j=1..n, i=1..c}$. The above mentioned disjointness and covering conditions translate in this case to the condition $\sum_{i=1}^{c} u_{i,j} = 1$ for all $j = 1, .., n$. This equation can also be used to replace the two conditions (2) and (3) for crisp partitions on finite domains,

where only $u_{i,j} \in \{0, 1\}$ is allowed. In this case we assume $u_{i,j} = 1 \Leftrightarrow x_j \in C_i$.

There might be various reasons for introducing uncertainty in the partition. Uncertainty may already be contained in the domain, for instance in the form of noisy measurements or vague concepts that are used to describe the granules. Another reason might be to avoid sharp boundaries between the granules, when the belongingness of data to granules is used in further processing. For instance, when each granule is associated with a function in terms of a local model, then switching from one granule to another one will result in discontinuous behaviour in the output function unless the corresponding functions fit to each other at the boundaries of the granules. In order to avoid this effect and achieve smooth switching between the local models, membership degrees can be taken into account to combine the local models according to the membership degrees.

The condition $\sum_{i=1}^{c} u_{i,j} = 1$ can be seen as a probabilistic constraint for the partition. The membership degree $u_{i,j}$ could be interpreted as the probability that object x_j belongs to the granule c_i. However, even if this probabilistic interpretation is intended, granules are seldomly interpreted in terms of probability distributions. Mixture models [27] are a very popular probabilistic approach in clustering and classification. But the resulting distributions might look as shown in figure 2. The two Gaussian distributions might be used for classification in order two distinguish between two classes. Both distributions might have the same a priori probability 0.5. In this case an object in the form of a real number would be assigned to the Gaussian distribution that yields the higher likelihood. This means that values far away from zero will be assigned to the flatter Gaussian distribution drawn with a dotted line, and values closer to zero will be assigned to the other Gaussian distribution. It is obvious that these two Gaussian distributions do not represent useful granules, since they overlap too much. Therefore, we will not consider probability distributions as granules in this chapter.

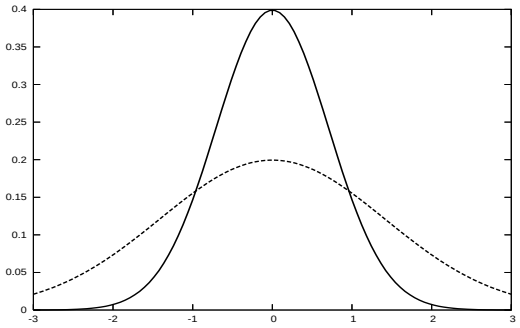

Figure 2: Two Gaussian distributions.

Even though the probabilistic constraint is also very common for fuzzy partitions, it can be dropped and be generalised to a possibilistic framework [5]. In this case, additional steps have to be taken into account in order to guarantee a partition-like set of granules.

3.2 Multidimensional and Hierarchical Granules

The considered domain for which the granules have to be defined is usually a multidimensional space, typically a subset of $\mathbb{R}^p$. A crucial problem is the description of multidimensional granules. A very intuitive approach consists in defining granules or partitions for the single dimensions and then combine these granules to describe multidimensional granules. In this case an implicit independence assumption is made. Independence should not be interpreted in the sense of probability theory, but in more general terms. When the granules are based on similarity concepts, independence means that there is no interaction between the similarities. In order to understand this effect, let us consider the two simple examples in figure 3. The aim is in both cases a classification problem where the circles should be separated from the squares. In both cases, looking at the projections to the single dimensions will result in a complete loss of information for the separation of the two classes circle and square. Nevertheless, the left example can still be treated easily by defining suitable granules on the single dimensions. When each of the dimensions is partitioned into two granules as indicated by the dotted lines, then it is easy to use these one-dimensional granules to define four two-dimensional granules that will either contain only circles or only squares.

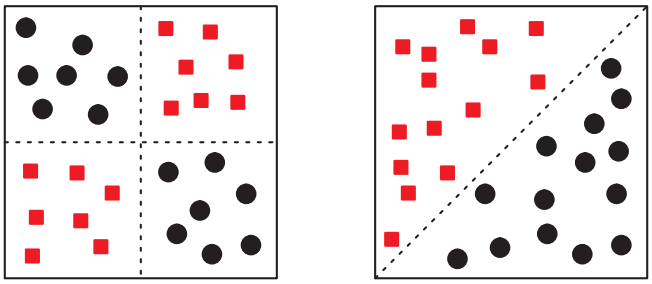

Figure 3: Cases for two-dimensional granules.

However, the right example in figure 3 is difficult to treat with combinations of one-dimensional granules. Of course, with a sufficient number of one-dimensional granules the dotted separating line can be approximated with combinations of one-dimensional granules. But this is a typical case where an independent consideration of the single dimension does not provide a helpful insight to the two-dimensional problem.

Another problem in granularization is the number of granules or how coarse or how fine the granularization should be chosen. In principle, it is possible to consider independently different levels of granularization. However, this will make it more difficult to switch between the different levels of granularization. Therefore, it is recommended to choose hierarchical granules or partitions, when different levels of granularization are required. Such hierarchical partitions might be derived in a canonical way as for instance in the case of OLAP applications or they might be found by hierarchical clustering techniques [20].

Another popular approach is the concept of *rough sets*, where granules are organized hierarchically [31, 37]. Having the notion of *granule inclusion* available, the human way of focussing and generalizing can be emulated quite naturally. The inclusion in a lower approximation of a granule is defined formally as the satisfaction of additional constraints. Approaches to learn hierarchical granules are often quite similar to hierarchical clustering algorithms.

4 Brief Introduction to Clustering

The problem of clustering is that of finding a partition that captures the similarity among data objects by grouping them accordingly in the partition. Data objects within a group (element of the partition) should be similar, data objects from different groups should be dissimilar. It is immediately clear, that clustering requires an explicit notion of similarity, which is often provided by means of a distance (or dissimilarity) function $d : D \times D \to \mathbb{R}_+$ or matrix $\mathcal{D} \in \mathbb{R}_+^{n \times n}$, $|D| = n$. A small value indicates that two objects are similar, a large value that they are dissimilar.

type	approach	algorithm
linkage	iteratively merge data objects with their resp. closest cluster	hierarchical clustering [38, 20]
density	identify connected regions where the data density exceeds some thresholds	DBScan [8], Clique [1]
objective function	perform clustering by minimizing an objective function that, e.g., minimizes the sum of within-cluster distances	c-Means [28], FCM [6, 3]

Table 1: Types of clustering algorithms

There are several types of clustering algorithms and many algorithms of each type available in the literature (table 1 shows only some representatives). The choice of the clustering algorithm depends in the first place on how the dissimilarity information is provided. If only a dissimilarity matrix is given, relational clustering algorithms have to be used. Usually, this approach can only be used if the entities we are going to cluster are only a few, because the space (and time) complexity is at least quadratic (the distance matrix contains a distance value for each pair of entities). Whenever the data is numerical in nature, dissimilarity *functions* such as the Euclidean distance can be utilized and there is no need for storing n^2 distances explicitly. Most clustering algorithms assume that such a distance function is given.

Here, we briefly discuss only the c-means and fuzzy c-means clustering algorithm, for the other techniques we refer to the literature [19, 18, 20, 17]. The

c-means algorithm [28] partitions the available data into c groups by finding c *prototypical data objects* that best represent the whole data set. Given a prototype, all data objects that are closer to this prototype than to any other, form the cluster or group of this prototype. The algorithm iteratively performs the following two steps: First, for all data objects, find the closest prototype (which are initialized randomly at the beginning). Second, adjust each of the prototypes to better represent the group by moving it to the centre of gravity of the group. If we encode the belongingness of data object x_j to prototype p_i in a binary variable $u_{i,j}$, which is 1 if and only if x_j is closest to p_i (0 otherwise), then the c-means algorithm actually minimizes

$$J_1(U, P; X) = \sum_{j=1}^{n} \sum_{i=1}^{c} u_{i,j} ||x_j - p_i||^2 \tag{4}$$

by alternatingly minimizing J with respect to $U = [u_{i,j}]$ and $P = (p_1, .., p_c)$.

The algorithm considers the belongingness of a data object to a prototype as a binary decision, giving it the flavour of a combinatorial problem. The fuzzy c-means variant of c-means [6, 3] turns the problem into a continuous one by making the cluster membership a matter of degree ($u_{i,j} \in [0, 1]$ rather than $u_{i,j} \in \{0, 1\}$, cf. section 3.1). Despite the relaxation, the minimization of (4) would still lead to crisp membership degrees, therefore an exponent m for the membership degrees is introduced, the so-called *fuzzifier*. (For a detailed discussion of the effect of the fuzzifier and alternative approaches, see [21].) Thus, the fuzzy c-means algorithm minimizes

$$J(U, P; X) = \sum_{j=1}^{n} \sum_{i=1}^{c} u_{i,j}^m ||x_j - p_i||^2 \tag{5}$$

subject to the constraints $\sum_{i=1}^{c} u_{i,j} = 1$ and $\sum_{j=1}^{n} u_{i,j} > 0$. The necessary conditions for a minimum of (5) w.r.t. p_i (assuming $u_{i,j}$ to be constant) yields the same as in the c-means algorithm: the prototypes have to be shifted into the (weighted) centre of all data points in the group:

$$\forall 1 \leq i \leq c: \qquad p_i = \frac{\sum_{j=1}^{n} u_{ij}^m x_j}{\sum_{j=1}^{n} u_{ij}^m} \tag{6}$$

The minimization w.r.t. $u_{i,j}$ (assuming p_i to be constant) delivers:

$$u_{ij} = \begin{cases} \frac{1}{\sum_{l=1}^{c} \left(\frac{\|x_j - p_i\|^2}{\|x_j - p_l\|^2} \right)^{\frac{1}{m-1}}} & \text{in case } I_j = \emptyset \\ \frac{1}{|I_j|} & \text{in case } I_j \neq \emptyset, i \in I_j \\ 0 & \text{in case } I_j \neq \emptyset, i \notin I_j \end{cases} \tag{7}$$

where $I_j = \{k \in \mathbb{N}_{\leq c} \,|\, x_j = p_k\}$. The fuzzy c-means (FCM) algorithm is depicted in figure 4. For a more detailed discussion of FCM and examples we refer to the literature, e.g. [17].

choose $m > 1$ (typically $m = 2$)
choose termination threshold $\varepsilon > 0$
initialize prototypes p_i (randomly)
repeat
  update memberships using (7)
  update prototypes using (6)
until change in memberships drops below ε

Figure 4: The FCM algorithm.

If a hierarchical partition is desired, it can be obtained quite easily by recursively applying a clustering algorithm to each of its clusters. In the case of a fuzzy partition the objective function-based clustering algorithms can be modified easily to consider the degree of belongingness to the clusters by an additional weight (see e.g. [11]). In [32] granules are represented by hyperboxes and these granules are then themselves clustered in a hierarchical fashion.

Almost all clustering algorithms deliver a partition regardless of the data set provided, that is, c-means yields c prototypes even in case of a uniform distribution without any cluster substructure. Thus, once a partition has been obtained by a clustering algorithm, the question is whether it represents significant structural information or just random fluctuations. In the literature, one can find many validity measures, see e.g. [12], that address global properties of the partition (e.g. degree of ambiguity) or local properties of each individual cluster (e.g. separation, compactness). Validity measures are also used to determine the number of clusters in c-means or fuzzy c-means clustering.

There are two important properties of the fuzzy c-means algorithm: (1) it derives fuzzy partitions (that is, allows overlapping granules, which is much more realistic in many applications) and (2) the possibility of modifying the objective function allows us to force FCM quite easily to consider additional aspects in the resulting partition. If the purpose of granularization is purely descriptive in nature, granularization and clustering become almost undistinguishable. However, as we have seen in section 2, quite often the partition must also fit in a certain task or application and/or must be easily interpretable. The fact that FCM can be tailored to specific applications by changing the objective function makes FCM a good candidate also for task-driven granularization purposes.

5 Similarity-Induced Partitions

In this section we consider the problem of deriving a system of granules in an unsupervised setting, e.g., the only information available to develop the granules is the data itself, we cannot utilize any feedback in form of a classification or approximation error to guide the process. If such information is available, section

6 should be consulted. In an unsupervised context we cannot measure the fitness for a specific purpose, so the granules should at least be representative for the data from which the granules have been induced.

5.1 One-Dimensional Granules with Crisp Boundaries

Let us first consider the case of a single numerical attribute. It is common practice in many data mining applications, that the set of values is discretized in an equi-width or equi-frequency fashion, thereby obtaining a reduced number of c intervals rather than (up to) n individual values. In the equi-width approach, all intervals share the same width. To define such a partition we only need to know the minimum and maximum value and set the number of intervals c a priori. Such a set of intervals hardly qualifies as a system of granules, because it does not reflect the underlying data distribution. In the equi-frequency approach each granule covers the same number of data objects: having sorted all instances we introduce a new interval every $\frac{n}{c}$ values. Figure 5 shows the effect of both approaches in an example data set of 20 values. An equi-width partition ($c = 5$) is shown in subfigure (a), the intervals do not reflect the data distribution, the fourth interval does not contain any instances. Subfigure (b) shows the case of an equi-frequency partition ($c = 4$). Again, the partition does not characterize the underlying data: The values near the middle are separated into two different intervals although they form a compact group of similar values. In contrast, the results of clustering algorithms lead to much more meaningful partitions, as shown in subfigures (c) and (d). Density-based approaches identify those instances that do not reach a desired data density as outliers, therefore in subfigure (c) the granules do not cover the complete range of values. Clustering algorithms that identify prototypical values, e.g. c-means, can be used to define granules, too, by introducing boundaries half-way between the prototypes, as shown in subfigure (d).

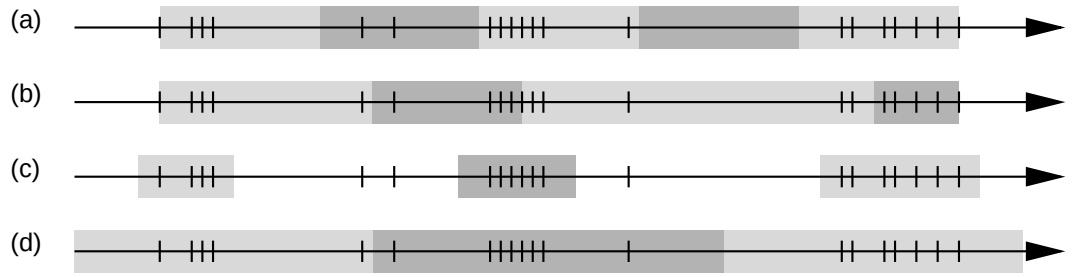

Figure 5: Interval-partitions (shaded regions) for a numerical attribute obtained through (a) equi-width partitioning, (b) equi-frequency partitioning, (c) density-based clustering, (d) prototype-based clustering.

If the attribute under consideration is of a categorical scale, there is no in-

formation whatsoever to group the data by *similarity* – except in the case that an expert provides additional information by means of a similarity matrix. If such a matrix is given, relational clustering algorithms such as the single-linkage hierarchical clustering algorithm [38] can be used to obtain a partition. If the scale is ordinal, we have two options: either we provide a distance matrix and use relational clustering or transform the attribute into a numerical one by assigning numbers to the ordinal values and proceed as before with numerical values.

5.2 One-Dimensional Granules with Fuzzy Boundaries

The systems of information granules shown in figure 5(c) and (d) also illustrate the unsatisfying consideration of outliers by crisp partitions. The two instances between the leftmost partitions may be considered as outliers (not belonging to any granule) or as equally poor representatives of both of the leftmost granules. It may, however, easily happen that these two instances are assigned to different granules if a crisp assignment is required (as in subfigure 5(d)). The uncertainty in the assignment to a granule is better captured when fuzzy memberships are used, because then a smaller membership degree indicates a less confident assignment to a granule.

The most simple way to turn the interval granules from figure 5(d) into fuzzy granules is by replacing the c-means algorithm with the fuzzy c-means algorithm. The fuzzy membership functions of FCM are given analytically by (7) and the only parameters are the cluster prototypes. Once the prototypes are determined, the membership functions can serve as information granules. The left column of figure 6 shows the fuzzy granules for three different degrees of fuzziness. The fuzzier the granules get, the faster the membership functions approach the value $\frac{1}{c}$ at the boundaries of the domain. This is because the farther we are away from the prototypes, the less certain we can be that a value belongs to a particular prototype. On the other hand, if we regard the introduction of fuzzy memberships as a *fuzzification of the crisp intervals* in figure 5(d), the degradation near the borders seems counterintuitive[1]. From that point of view, granules like those on the right hand side of figure 6 are preferable.

This kind of membership function can be obtained by a modification of the FCM objective function. The motivation for the modification proposed in [15] is to avoid the unexpected local maxima (cf. bottom left plot in figure 6) and the completely fuzzy membership degrees near the borders by rewarding crisp membership degrees. Only where we have to switch between the cores of the sets, fuzziness is pertained (cf. right hand side of figure 6). We introduce a number of parameters $a_j \in \mathbb{R}_{\geq 0}$, $1 \leq j \leq n$, and consider the following modified

[1] In case of a single attribute, this problem affects two granules at most, however, this is no longer true when seeking for granules in high-dimensional spaces (section 5.3).

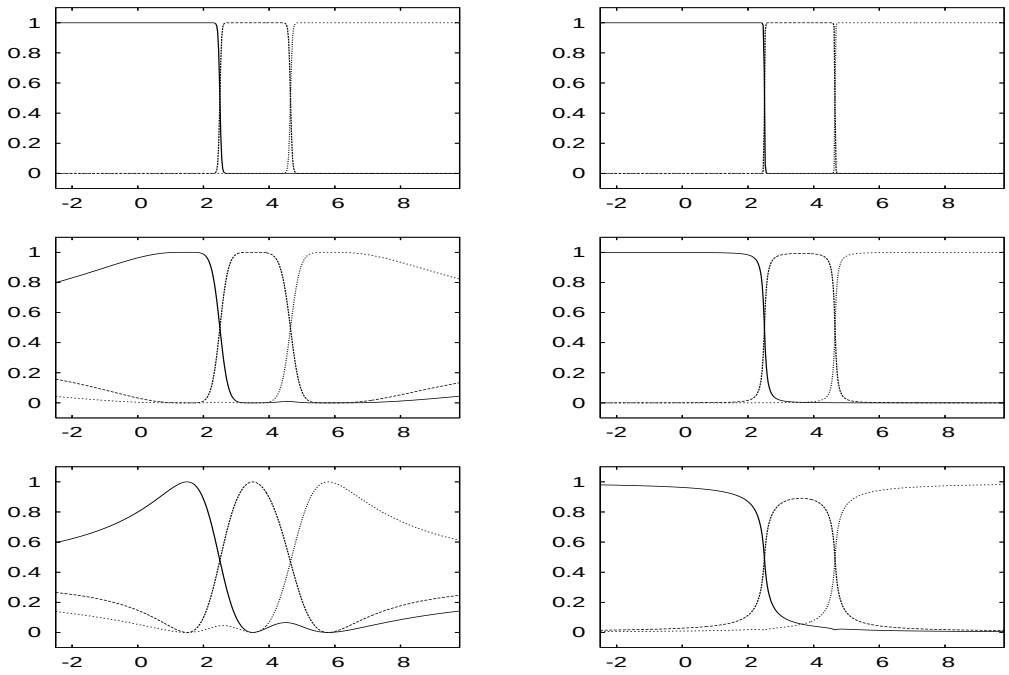

Figure 6: Fuzzy membership functions. From top to bottom the fuzziness is increased (m resp. η), left column: original FCM, right column: modified FCM.

objective function:

$$J(U,P;X,A) = \sum_{j=1}^{n}\sum_{i=1}^{c} u_{i,j}^2 d^2(x_j,p_i) - \sum_{j=1}^{n} a_j \sum_{i=1}^{c}\left(u_{i,j} - \frac{1}{2}\right)^2 \tag{8}$$

Only the second term is new in the objective function, the first term is identical to the objective function of FCM with $m = 2$. If a data object x_j is clearly assigned to one prototype p_i, then we have $u_{i,j} = 1$ and $u_{k,j} = 0$ for all other $k \neq i$. For all these cases, the second term evaluates to $-\frac{a_j}{4}$. If the membership degrees become more fuzzy, the second term increases. Since we seek to minimize (8), this modification rewards crisp membership degrees. The maximal reward we can give to obtain positive membership degrees is then

$$a_j = \min d_{*,j}^2 = \min\{\, d_{i,j}^2 \,|\, i \in \{1,..,c\}\,\} - \eta \tag{9}$$

where $\eta > 0$ is a constant that takes a similar role as the fuzzifier. The resulting membership functions are

$$u_{i,j} = \frac{1}{\sum_{k=1}^{c} \frac{d_{i,j}^2 - \min d_{*,j}^2}{d_{k,j}^2 - \min d_{*,j}^2}} \tag{10}$$

Since the notion of a cluster prototype itself is not affect by this modification, the prototype update equations remain the same. The resulting algorithm finds the

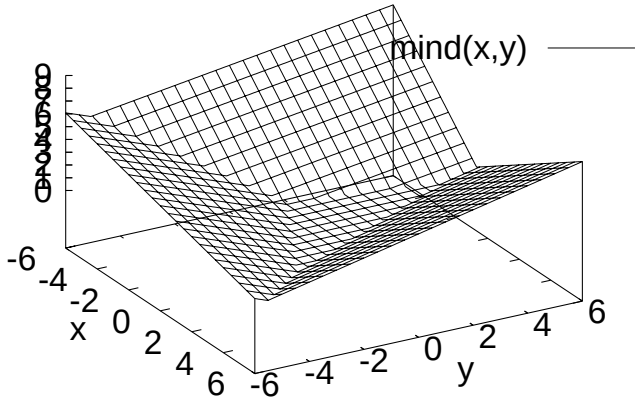

Figure 7: Distance to Voronoi cell.

fuzzy partition by performing a three-stage alternating optimization, consisting of

1. the calculation of the membership degrees via (10), assuming prototypes and a_j as being constant,

2. the calculation of the prototypes via (6), assuming memberships and a_j as being constant,

3. the calculation of the a_j via (9), assuming prototypes and memberships as being constant.

Figure 6 illustrates the resulting membership functions in the one-dimensional case. Interestingly, even the multidimensional membership functions can be interpreted in a very intuitive manner. Given a set of prototypes, the Voronoi cell of a prototype p is the set of all points that are closer to p than to any other prototype. It turns out [15] that the distance term $d_{i,j}^2 - \min d_{*,j}^2$ in the membership functions corresponds to a (scaled) distance to the Voronoi cell of the respective prototype. This is illustrated for the 2D-case in figure 7. The fact that the original FCM uses a *squared* Euclidean distance to the prototype, whereas this modification uses a (scaled, but *unsquared*) distance to the Voronoi cell of the prototype, makes it less sensitive to outliers.

If the application requires the same number of instances per granules, just as with the equi-frequency partitions, FCM can also be biased towards uniformly sized clusters [22].

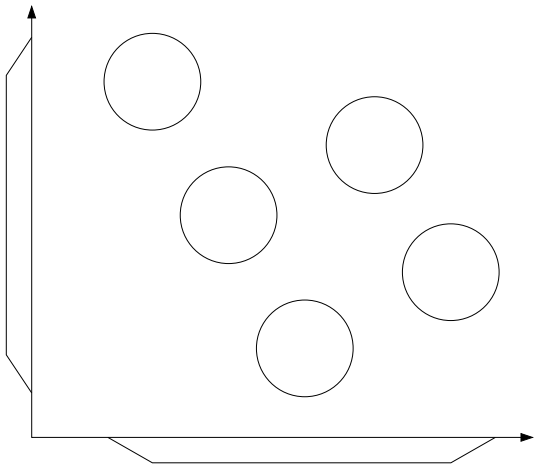

Figure 8: Granularization of single attributes does not provide a good starting point for granularization in the two-dimensional space.

5.3 Multi-Dimensional Granules

If we have to deal with multiple variables, they may be granularized individually and independently as described in the previous section. Will the granules obtained from the marginal data distributions automatically be qualified to compose the granules in the higher-dimensional space? Figure 8 shows that this is not true in general (see also section 3.2). The circles in the two-dimensional plane shall represent regions of similar data, each circle may be considered as a granule we seek to discover. In this particular example, the data regions slightly overlap in the projection on each individual variable, such that an almost uniform marginal distribution is perceived (indicated along the axes). Given such a uniform distribution, we will obtain arbitrary one-dimensional granules and it is not very likely, that we can describe the two-dimensional granules by a cross-product of one-dimensional granules. In this particular case, a clustering-based partition may even detect that there are no subclusters at all and thus a single granule will be sufficient. Although we have seen that the equi-width partitioning (section 5.1) hardly qualifies as a system of granules, in this particular example equi-width partitioning may actually outperform a clustering-based partition.

The example suggests to invert the approach: rather than developing the granules in the low-dimensional space and trying to approximate the high-dimensional granules by a product of low-dimensional granules, we could find the high-dimensional granules and extract low-dimensional granules from them. This can be easily done by applying clustering algorithms directly to the high-dimensional space[2]. The disadvantage of this approach is that different multidimensional granules may lead to *similar* projections in the one-dimensional space. These

[2]We are simplifying at this point, because many clustering algorithms depend themselves on the set of attributes. If different subsets of attributes are used, different clusters will be found.

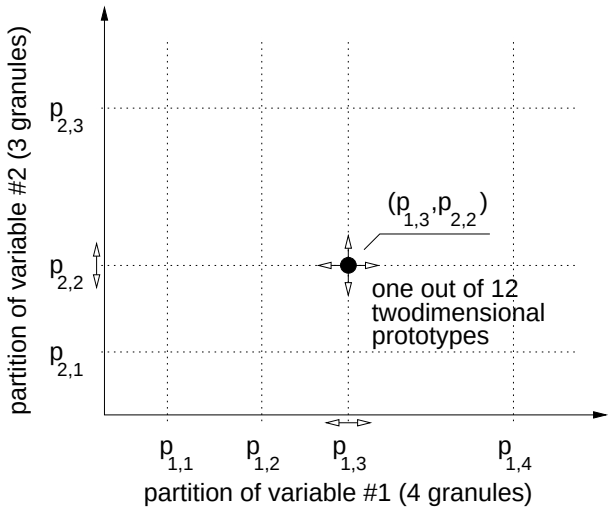

Figure 9: Illustration of the notation used in this section: $d = 2$ dimensions, first dimension consists of $c_1 = 4$ granules, second of $c_2 = 3$ granules. The multidimensional granules are composed out of one-dimensional granules and are denoted by a tuple.

heavily overlapping granules should be identified and merged to obtain an understandable one-dimensional granularization (e.g. [36]). In such an approach, where only the projection of multidimensional granules will be processed further, it is advantageous to support merging by appropriate clustering algorithms that align the clusters along the main axes only [24] or seek for clusters in the shape of hyperboxes [32].

An alternative solution to this problem can also be found in a modified FCM algorithm [16]: rather than seeking for c independent, multidimensional granules, we may look for one-dimensional partitions of c_i granules each, such that the multidimensional space is tesselated into $\prod_i c_i$ granules. The prototypes are no longer optimized individually, but the grid of regularly distributed prototypes is optimized as a whole. During optimization the aim is to find optimal multidimensional granules, but in order to adjust the multidimensional granules we are limited to the modification of the one-dimensional partitions.

Given d input variables, suppose we divide the domain of variable v_i into c_i granules, induced by representatives $p_{i,j}$, $i \in \{1, ..., d\}$, $j \in \{1, ..., c_i\}$ as illustrated by the example in figure 9. We construct multidimensional granules by combining the one-dimensional granules $p_{i,j}$ (one for each dimension i) and denote them by a tuple $(p_{1,i_1}, p_{2,i_2}, .., p_{d,i_d})$ where $i_k \in \{1, .., c_k\}$.

The granules of interest are the multidimensional ones, so our cluster prototypes live in the multidimensional space and their total number is $c = \prod_{k=1}^{d} c_k$. But in contrast to the traditional FCM algorithm, where each of the prototypes is optimized individually, our prototypes are constrained to lie on the grid defined

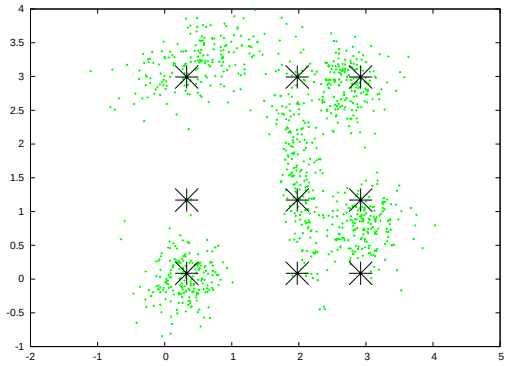

Figure 10: Clustering with 9 prototypes that share their coordinates (3 x-values and 3 y-values).

by the one-dimensional partitions. So the set of prototypes is given by

$$P = \{(p_{1,i_1}, p_{2,i_2}, \cdots, p_{d,i_d})^\top \mid i_k \in \{1, .., c_k\}\}$$

The standard FCM objective function remains unchanged, but has to take the construction of the prototypes into account:

$$J = \sum_{j=1}^{n} \sum_{i_1=1}^{c_1} \sum_{i_2=1}^{c_2} \ldots \sum_{i_d=1}^{c_d} u^m_{(i_1,i_2,..,i_d),j} \left\| \begin{pmatrix} x_1 - p_{1,i_1} \\ x_2 - p_{2,i_2} \\ \vdots \\ x_k - p_{k,i_k} \end{pmatrix} \right\|^2 \tag{11}$$

(still subject to the same constraints on the membership degrees as before). The multidimensional granules are *parameterized* by the one-dimensional ones, so to optimize this objective function we have to solve for each of the $p_{i,j}$ individually. That is, the large set of $c = \prod_{k=1}^{d} c_k$ multidimensional granules is *parameterized* by a much smaller set of $\sum_{k=1}^{d} c_k$ parameters. The necessary conditions for a minimum of the objective function (11) are given by [16]:

$$p_{l,r} = \frac{\sum_{j=1}^{n} \sum_{(i_1,i_2,\ldots,i_k),i_l=r} u^m_{(i_1,i_2,\ldots,i_k),j} \cdot x_l}{\sum_{j=1}^{n} \sum_{(i_1,i_2,\ldots,i_k),i_l=r} u^m_{(i_1,i_2,\ldots,i_k),j}} \tag{12}$$

Since the derivation of the necessary condition arrives at a unique solution, convergence of the iterative alternating optimization scheme is guaranteed [14]. Figure 10 shows the resulting cluster centres for a two-dimensional data sets: the six parameters (three positions on both axes) define nine prototypes, forming a regular grid that approximates the regions of high data density appropriately.

6 Task-Driven Partitions

If the purpose of granularization is not only descriptive, the incorporation of application specific information, i.e. the tailoring of the granules towards the application at hand, always leads to better results. Additional information might be a class label in the case of a classification tasks or an output value or error for regression tasks.

There are two ways to exploit such additional information in the granularization. One way is to use this information to minimize an error like the misclassification rate or the mean squared error directly by choosing or modifying the granules. Another possibility is to use other measures that are related to the actual goals, but do not guarantee an optimal choice with respect to the goal.

6.1 One-Dimensional Granules

There is a large variety of techniques to find granules for supervised learning from which we can only present a selection.

A typical example for a granularization based on an indirect strategy is the way in which decision trees are usually constructed. In order to build a classifier, decision trees partition the domains of the single attributes in order to construct the classification rules. We briefly recall a method to partition the domain of a numerical attribute proposed by Elomaa and Rousu [7], an extension of the algorithm described by Fayyad and Irani [9] for finding a partition splitting the domain into two sets only. The partition of the domain of the numerical attribute will be based on another categorical attribute that we intend to predict later on.

We consider a single numerical attribute j whose domain should be partitioned into a predefined number t of intervals. Therefore, $t-1$ cut points $T_1, \ldots, T_{t-1}$ have to specified. These cut points should be chosen in such a way that the entropy of the partition is minimized. Assume that T_0 and T_t are the left and right boundary of the domain, respectively.

Assume, we have n data objects and n_i $(i = 1, \ldots, t)$ of these fall into the interval $[T_{i-1}, T_i]$. Let k_ℓ denote the number of the n_i data that belong to class ℓ. The entropy in this interval is given by

$$E_i \;=\; -\sum_{\ell=1}^{c} \frac{k_\ell}{n_i} \cdot \log\left(\frac{k_\ell}{n_i}\right). \tag{13}$$

The entropy of the partition induced by these cut points is simply the weighted sum of the single entropies

$$E \;=\; \sum_{i=1}^{t} \frac{n_i}{n} \cdot E_i. \tag{14}$$

The weights are the probability or fractions for the corresponding intervals. The aim is to choose the cut points in such a way that (14) is minimized.

Sorting the data with respect to the values of the jth attribute, it was proved in [7] that it is sufficient to consider boundary points only for finding an optimal partition. If the class label changes directly before or after a value, this value becomes a boundary point. The following example illustrates the concept of boundary points that are marked by lines.

value:	0	1	2	2	3	4	4	5	**5**	6	7	7	8	9	10	10	11
class:	c	c	a	a	a	b	b	a	c	c	c	c	c	b	a	a	a

Note that a boundary point occurs after 4 and also before 6 in the attribute j. Since at least one object with value 5 in attribute j belongs to class a and at least one object with value 6 in attribute j belongs to another class, namely class c, it is necessary to introduce a boundary point after the second value 5 printed in bold face style.

Once the boundary points are computed, the optimal partition minimizing (14) for a fixed number t of intervals can be constructed. A recursive search can be carried out to find the best partition among the $\binom{b}{t-1}$ possible partitions. Of course, a complete search can only be carried out, when $\binom{b}{t-1}$ is reasonably small. Otherwise, heuristic strategies as proposed in [25] might be used to find a suboptimal solution.

Similar ideas can be applied to regression problems. Instead of entropy, the variance in the numerical attribute to be predicted, can be taken into account.

Instead of using entropy also other measures are possible. The minimum description length principle (MDL) [34, 35] is proposed in [10] to find good partitions. MDL is a technique for choosing the proper complexity of a model for a given data set. The underlying idea is as follows. The aim is to encode the given data set in such a way that a minimum amount of bits is required to transfer the data over a channel. When a model to represent or approximate the data, this model is used to compress the data. The total amount of bits to be transferred are the compressed data as well as the description of the model. A very complex model might lead to a very compact representation of the data, resulting in a very effective compression. However, the amount of bits needed to transfer the complex model as well might result in a larger total number of bits to be transferred. On the other hand, a very simple model will only need a few bits to be transferred, but it will usually not provide a good compression for the data.

So far, the granularization or partition was considered to be crisp. But the same strategies can be applied to obtain fuzzy partitions [25]. Once the cut points for the crisp partition are determined, a suitable fuzzy partition can be defined as shown in figure 11. In [26] an MDL-based approach to find good fuzzy partitions is introduced.

MDL- and entropy-driven techniques for granularization are indirect approaches in the sense that they do not explicitly aim to optimise the actual objective function like the misclassification rate or the mean square error. There

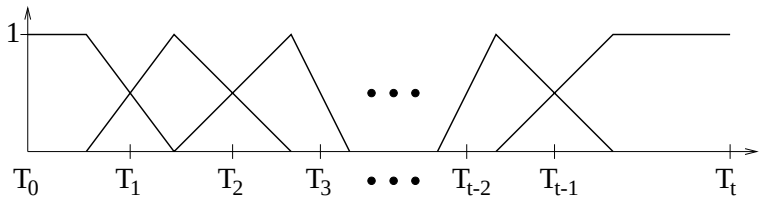

Figure 11: A fuzzy partition induced by cut points.

are numerous specific techniques to optimise partitions in order to minimize an objective function like the classification or the mean square error directly. These techniques include local linear regression models, gradient descent methods, neural networks or evolutionary algorithms, especially in combination with fuzzy systems [4, 30]. However, to discuss the granularization for such models in detail cannot be within the scope of this contribution, since the methods strongly depend on the underlying model.

6.2 Multi-Dimensional Granules

The techniques mentioned in the previous section focus on granularizations of single attributes. The indirect methods consider the single attributes in an isolated manner. The methods that try to minimize an error measure for the prediction directly take the interaction of different attributes into account, although they still focus on partitions for the single attributes. Although multidimensional granules are more flexible, they are usually difficult to interpret. Therefore, most of the techniques focussing on multidimensional dependencies still try to maintain granules for single attributes, but try to construct them based on their interaction with respect to the variable to be predicted. In [25] techniques are proposed that construct partitions on single variables based on entropy minimization where the interaction of the attributes is taken into account in order to find the best partitions or to simplify partitions. Other methods try to find a multidimensional grid [23] in order to find simple granules.

We have seen in section 5 that clustering-based methods are well-suited to find systems of granules. Such unsupervised techniques can also be altered to reflect additional information. This is for example helpful in case of partially labeled data, e.g. when class information is only available for some data objects (see e.g. [33]). The clustering techniques can be modified to additionally reflect either class label information or the approximation error in a regression task. In both cases, the objective function of the FCM algorithm is supplemented with an additional penalty term that promotes *pure* granules [13] (classification problem) or granules with a *good fit* [16] (regression problem). After that, we have no longer a clustering algorithm. Since both, partitioning and classification/regression error, are considered in the *same* objective function, an interdependency between the

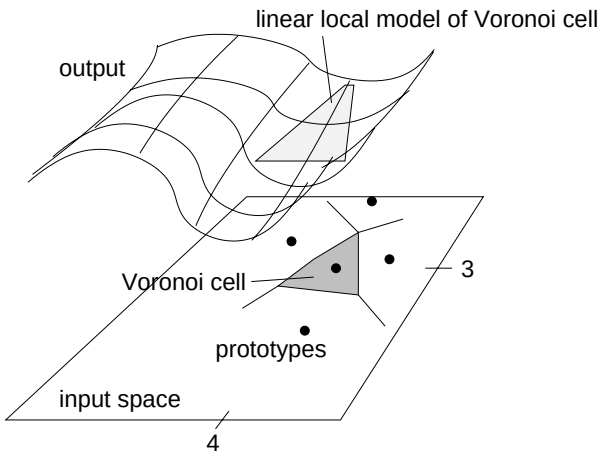

Figure 12: Interaction between partitioning task (Voronoi cell) and regression task (local model).

granules and the models is established. A poor classification or fit (cf. figure 12) can be resolved in the next iteration step by a better model *and/or* by modified granules. Therefore, the partitioning of the input space indirectly influences the classification or regression error and vice versa. To ease the interpretation of the granules, the prototypes may additionally be organized on a regular grid, as discussed in section 5.3, leading to Voronoi cells in the shape of hyperboxes.

7 Conclusions

Finding useful information granules is closely related to finding good partitions, where a partition should not be understood in a strict mathematical sense. The methods for defining and finding the partitions and the kind of partitions strongly depend on the context in which the granules will be used. This chapter has provided an overview on different purposes for the use as well as various techniques for the construction of granules. The choice of the granularization depends highly on the specific application.

Nowadays, data is usually collected and stored in the finest granularity available to prevent loss of any information. This is, however, completely different from the way humans perceive and understand the world. Depending on the level of abstraction, humans focus on the data at different resolutions or granularities. It is very often the case that the structure inherent in the data becomes evident only at a specific level of abstraction. Exploiting *the right* granularity for an application at hand is therefore essential in human problem solving and consequently in the design and implementation of intelligent systems. For the case of data mining it is quite often not the choice of the learning algorithm but the choice of (knowledge-based or data-driven) granularization that decides about

success or failure.

References

[1] R. Agrawal, J. Gehrke, D. Gunopulos, and P. Raghavan. Automatic subspace clustering of high-dimensional data for data mining applications. In *Proc. of ACM SIGMOD Int. Conf. on Management of Data*, pages 94–105, 1998.

[2] A. Bargiela and W. Pedrycz. *Granular Computing: An Introduction*. Kluwer, Dordrecht, 2002.

[3] J. C. Bezdek. A convergence theorem for the fuzzy ISODATA clustering algorithms. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 2(1):1–8, Jan. 1980.

[4] O. Cordón, F. Gomide, F. Herrera, F. Hoffmann, and L. Magdalena. Ten years of genetic fuzzy systems: Current framework and new trends. *Fuzzy Sets and Systems*, 141:5–31, 2004.

[5] D. Dubois and H. Prade. Possibility theory, probability theory and multiple-valued logics: A clarification. *Annals of Mathematics and Artificial Intelligence*, 32:35–66, 2001.

[6] J. Dunn. A fuzzy relative of the ISODATA process and its use in detecting compact, well-separated clusters. *Journal of Cybernetics*, 3(3):32–57, 1973.

[7] T. Elomaa and J. Rousu. Finding optimal multi-splits for numerical attributes in decision tree learning. technical report NC-TR-96-041, Department of Computer Science, Royal Holloway University of London, 1996.

[8] M. Ester, H.-P. Kriegel, J. Sander, and X. Xiaowei. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proc. of the 2nd ACM SIGKDD Int. Conf. on Knowl. Discovery and Data Mining*, pages 226–331, Portland, Oregon, 1996.

[9] U. Fayyad and K. Irani. On the handling of continues-valued attributes in decision tree generation. *Machine Learning*, 8:87–102, 2001.

[10] U. M. Fayyad and K. B. Irani. Multi-interval discretization of continuous-valued attributes for classification learning. In *Proc. of the 13th Int. Joint Conf. on Artifial Intelligence*, pages 1022–1027, Chambery, France, Aug. 1993.

[11] A. B. Geva. Feature extraction and state identification in biomedical signals using hierarchical fuzzy clustering. *Medical & Biological Engineering & Computing*, 36:608–614, Sept. 1998.

[12] M. Halkidi, Y. Batistakis, and M. Vazirgiannis. On clustering validation techniques. *Journal of Intelligent Information Systems*, 17(2):107–145, 2001.

[13] F. Höppner. Objective function-based discretization. In *Proc. 29th Annual Conf. of the Gesellschaft für Klassifikation*, Studies in Classification, Data Analysis, and Knowledge Organization, pages 438–445. Springer, 2006.

[14] F. Höppner and F. Klawonn. A contribution to convergence theory of fuzzy c-means and derivatives. *IEEE Trans. on Fuzzy Systems*, 11(5):682–694, Oct. 2003.

[15] F. Höppner and F. Klawonn. Improved fuzzy partitions for fuzzy regression models. *International Journal of Approximate Reasoning*, 32:85–102, 2003.

[16] F. Höppner and F. Klawonn. Learning fuzzy systems – an objective-function approach. *Mathware & Soft Computing*, 11(5):143–162, 2004.

[17] F. Höppner, F. Klawonn, R. Kruse, and T. A. Runkler. *Fuzzy Cluster Analysis*. John Wiley & Sons, Chichester, England, 1999.

[18] A. K. Jain and R. C. Dubes. *Algorithms for Clustering Data*. Prentice-Hall, 1988.

[19] A. K. Jain, N. M. Murty, and P. J. Flynn. Data clustering: A review. *ACM Computing Surveys*, 31(3):264–323, Sept. 1999.

[20] L. Kaufman and P. Rousseeuw. *Finding Groups in Data: An Introduction to Cluster Analysis*. Wiley, Chichester, 2005.

[21] F. Klawonn and F. Höppner. What is fuzzy about fuzzy clustering? – understanding and improving the concept of the fuzzifier. In *Advances in Intelligent Data Analysis*, pages 254–264. Springer, 2003.

[22] F. Klawonn and F. Höppner. Equi-sized, homogeneous partitioning. In *Proc. of Int. Conf. on Knowledge-Based Intelligent Engineering Systems & Allied Technologies*, volume 4252 of *LNCS*, pages 70–77. Springer, 2006.

[23] F. Klawonn and A. Keller. Fuzzy clustering with evolutionary algorithms. *Intern. Journ. of Intelligent Systems*, 13:975–991, 1998.

[24] F. Klawonn and R. Kruse. Constructing a fuzzy controller from data. *Fuzzy Sets and Systems*, 85:177–193, 1997.

[25] F. Klawonn and D. Nauck. Automatically determine initial fuzzy partitions for neuro-fuzzy classifiers. In *2006 IEEE International Conference on Fuzzy Systems*, pages 7894–7900, Vancouver, 2006. IEEE.

[26] A. Klose. *Partially Supervised Learning of Fuzzy Classification Rules*. Ph.d., Otto-von Guerecke University Magdeburg, 2004.

[27] G. McLachlan and D. Peel. *Finite Mixture Models*. Wiley, Chichester, 2000.

[28] J. B. McQueen. Some methods of classification and analysis of multivariate observations. In *Proc. of 5th Berkeley Symp. on Mathematical Statistics and Probability*, pages 281–297, 1967.

[29] J. Mill and A. Inoue. Granularization of machine learning. In *Proc. Eleventh International Conference on Information Processing and Management of Uncertainty in Knowledge-based Systems (IPMU 2006)*, pages 1907–1912, Paris, 2006. Éditions E.D.K.

[30] D. Nauck, F. Klawonn, and R. Kruse. *Neuro-Fuzzy Systems*. Wiley, Chichester, 1997.

[31] Z. Pawlak. *Rough Sets: Theoretical Aspects of Reasoning About Data*. Kluwer, Dordrecht, 1991.

[32] W. Pedrycz and A. Bargiela. Granular clustering: A granular signature of data. *IEEE Trans. on Systems, Man, and Cybernetics, Part B*, 32(2):212–224, Apr. 2002.

[33] W. Pedrycz and J. Waletzky. Fuzzy clustering with partial supervision. *IEEE Trans. on Systems, Man, and Cybernetics, Part B*, 27(5):787–795, Oct. 1997.

[34] J. Rissanen. A universal prior for integers and estimation by minimum description length. *Annals of Statistics*, 11:416–431, 1983.

[35] J. Rissanen. *Stochastic Complexity and Statistical Inquiry*. World Scientific, Singapore, 1989.

[36] M. Setnes, R. Babuska, U. Kaymak, and H. R. van Nauta Lemke. Similarity measures in fuzzy rule base simplification. *IEEE Trans. on Systems, Man, and Cybernetics, Part B*, 28(3):376–386, June 1998.

[37] A. Skowron, J. Stepaniuk, J. Peters, and R. Swiniarski. Calculi of approximation spaces. *Fundamenta Informaticae*, 72:363–378, 2006.

[38] P. Sneath. The application of computers to taxonomy. *J. Gen. Microbiol.*, 17:201–206, 1957.

[39] E. Thomsen. *OLAP Solutions: Building Multidimensional Information Systems*. Wiley, Chichester, 2nd edition, 2002.

[40] S. Wrobel. An algorithm for multi-relational discovery of subgroups. In *Proc. of the 1st Europ. Conf. on Principles of Data Mining and Knowl. Discovery*, pages 78–87, 1997.

[41] L. Zadeh. Toward a theory of fuzzy information granulation and its centrality in human reasoning and fuzzy logic. *Fuzzy Sets and Systems*, 90:111–127, 1997.