KNOWLEDGE-BASED CLUSTERING

From Data to Information Granules

Witold Pedrycz

Department of Electrical and Computer Engineering
University of Alberta
Edmonton, Canada

and

Systems Research Institute
Polish Academy of Sciences
Warsaw, Poland
KNOWLEDGE-BASED CLUSTERING
To Ewa, Adam, and Barbara
## Contents

<table>
<thead>
<tr>
<th>Chapter</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>Foreword</strong></td>
<td>xiii</td>
</tr>
<tr>
<td></td>
<td><strong>Preface</strong></td>
<td>xv</td>
</tr>
<tr>
<td>1</td>
<td><strong>1 Clustering and Fuzzy Clustering</strong></td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>1.1 Introduction</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>1.2 Basic Notions and Notation</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>1.2.1 Types of Data</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>1.2.2 Distance and Similarity</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>1.3 Main Categories of Clustering Algorithms</td>
<td>6</td>
</tr>
<tr>
<td></td>
<td>1.3.1 Hierarchical Clustering</td>
<td>6</td>
</tr>
<tr>
<td></td>
<td>1.3.2 Objective Function-Based Clustering</td>
<td>8</td>
</tr>
<tr>
<td></td>
<td>1.4 Clustering and Classification</td>
<td>10</td>
</tr>
<tr>
<td></td>
<td>1.5 Fuzzy Clustering</td>
<td>11</td>
</tr>
<tr>
<td></td>
<td>1.6 Cluster Validity</td>
<td>18</td>
</tr>
<tr>
<td></td>
<td>1.7 Extensions of Objective Function-Based Fuzzy Clustering</td>
<td>19</td>
</tr>
<tr>
<td></td>
<td>1.7.1 Augmented Geometry of Fuzzy Clusters: Fuzzy C Varieties</td>
<td>19</td>
</tr>
<tr>
<td></td>
<td>1.7.2 Possibilistic Clustering</td>
<td>20</td>
</tr>
<tr>
<td></td>
<td>1.7.3 Noise Clustering</td>
<td>22</td>
</tr>
<tr>
<td></td>
<td>1.8 Self-Organizing Maps and Fuzzy Objective Function-Based Clustering</td>
<td>23</td>
</tr>
<tr>
<td></td>
<td>1.9 Conclusions</td>
<td>25</td>
</tr>
<tr>
<td></td>
<td>References</td>
<td>26</td>
</tr>
<tr>
<td>2</td>
<td><strong>2 Computing with Granular Information: Fuzzy Sets and Fuzzy Relations</strong></td>
<td>28</td>
</tr>
<tr>
<td></td>
<td>2.1 A Paradigm of Granular Computing: Information Granules and Their Processing</td>
<td>28</td>
</tr>
<tr>
<td></td>
<td>2.2 Fuzzy Sets as Human-Centric Information Granules</td>
<td>31</td>
</tr>
<tr>
<td></td>
<td>2.3 Operations on Fuzzy Sets</td>
<td>32</td>
</tr>
<tr>
<td></td>
<td>2.4 Fuzzy Relations</td>
<td>33</td>
</tr>
<tr>
<td></td>
<td>2.5 Comparison of Two Fuzzy Sets</td>
<td>35</td>
</tr>
<tr>
<td></td>
<td>2.6 Generalizations of Fuzzy Sets</td>
<td>37</td>
</tr>
<tr>
<td></td>
<td>2.7 Shadowed Sets</td>
<td>38</td>
</tr>
<tr>
<td>Chapter</td>
<td>Title</td>
<td>Page</td>
</tr>
<tr>
<td>---------</td>
<td>----------------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>2.8</td>
<td>Rough Sets</td>
<td>44</td>
</tr>
<tr>
<td>2.9</td>
<td>Granular Computing and Distributed Processing</td>
<td>46</td>
</tr>
<tr>
<td>2.10</td>
<td>Conclusions</td>
<td>47</td>
</tr>
<tr>
<td></td>
<td>References</td>
<td>47</td>
</tr>
<tr>
<td>3</td>
<td>Logic-Oriented Neurocomputing</td>
<td>50</td>
</tr>
<tr>
<td>3.1</td>
<td>Introduction</td>
<td>50</td>
</tr>
<tr>
<td>3.2</td>
<td>Main Categories of Fuzzy Neurons</td>
<td>51</td>
</tr>
<tr>
<td>3.2.1</td>
<td>Aggregative Neurons</td>
<td>52</td>
</tr>
<tr>
<td>3.2.2</td>
<td>Referential (Reference) Neurons</td>
<td>55</td>
</tr>
<tr>
<td>3.3</td>
<td>Architectures of Logic Networks</td>
<td>59</td>
</tr>
<tr>
<td>3.4</td>
<td>Interpretation Aspects of the Networks</td>
<td>61</td>
</tr>
<tr>
<td>3.5</td>
<td>Granular Interfaces of Logic Processing</td>
<td>62</td>
</tr>
<tr>
<td>3.6</td>
<td>Conclusions</td>
<td>64</td>
</tr>
<tr>
<td></td>
<td>References</td>
<td>64</td>
</tr>
<tr>
<td>4</td>
<td>Conditional Fuzzy Clustering</td>
<td>66</td>
</tr>
<tr>
<td>4.1</td>
<td>Introduction</td>
<td>66</td>
</tr>
<tr>
<td>4.2</td>
<td>Problem Statement: Context Fuzzy Sets and Objective Function</td>
<td>68</td>
</tr>
<tr>
<td>4.3</td>
<td>The Optimization Problem</td>
<td>70</td>
</tr>
<tr>
<td>4.4</td>
<td>Computational Considerations of Conditional Clustering</td>
<td>80</td>
</tr>
<tr>
<td>4.5</td>
<td>Generalizations of the Algorithm Through the Aggregation Operator</td>
<td>81</td>
</tr>
<tr>
<td>4.6</td>
<td>Fuzzy Clustering with Spatial Constraints</td>
<td>82</td>
</tr>
<tr>
<td>4.7</td>
<td>Conclusions</td>
<td>86</td>
</tr>
<tr>
<td></td>
<td>References</td>
<td>86</td>
</tr>
<tr>
<td>5</td>
<td>Clustering with Partial Supervision</td>
<td>87</td>
</tr>
<tr>
<td>5.1</td>
<td>Introduction</td>
<td>87</td>
</tr>
<tr>
<td>5.2</td>
<td>Problem Formulation</td>
<td>88</td>
</tr>
<tr>
<td>5.3</td>
<td>Design of the Clusters</td>
<td>90</td>
</tr>
<tr>
<td>5.4</td>
<td>Experimental Examples</td>
<td>91</td>
</tr>
<tr>
<td>5.5</td>
<td>Cluster-Based Tracking Problem</td>
<td>93</td>
</tr>
<tr>
<td>5.6</td>
<td>Conclusions</td>
<td>96</td>
</tr>
<tr>
<td></td>
<td>References</td>
<td>96</td>
</tr>
<tr>
<td>6</td>
<td>Principles of Knowledge-Based Guidance in Fuzzy Clustering</td>
<td>97</td>
</tr>
<tr>
<td>6.1</td>
<td>Introduction</td>
<td>97</td>
</tr>
<tr>
<td>6.2</td>
<td>Examples of Knowledge-Oriented Hints and Their General Taxonomy</td>
<td>99</td>
</tr>
<tr>
<td>6.3</td>
<td>The Optimization Environment of Knowledge-Enhanced Clustering</td>
<td>102</td>
</tr>
</tbody>
</table>
6.4 Quantification of Knowledge-Based Guidance Hints and Their Optimization 105
6.5 Organization of the Interaction Process 107
6.6 Proximity-Based Clustering (P-FCM) 112
6.7 Web Exploration and P-FCM 117
6.8 Linguistic Augmentation of Knowledge-Based Hints 126
6.9 Conclusions 127
References 127

7 Collaborative Clustering 129
7.1 Introduction and Rationale 129
7.2 Horizontal and Vertical Clustering 131
7.3 Horizontal Collaborative Clustering 132
  7.3.1 Optimization Details 135
  7.3.2 The Flow of Computing of Collaborative Clustering 137
  7.3.3 Quantification of the Collaborative Phenomenon of Clustering 138
7.4 Experimental Studies 140
7.5 Further Enhancements of Horizontal Clustering 150
7.6 The Algorithm of Vertical Clustering 151
7.7 A Grid Model of Horizontal and Vertical Clustering 153
7.8 Consensus Clustering 155
7.9 Conclusions 157
References 157

8 Directional Clustering 158
8.1 Introduction 158
8.2 Problem Formulation 159
  8.2.1 The Objective Function 160
  8.2.2 The Logic Transformation Between Information Granules 161
8.3 The Algorithm 163
8.4 The Development Framework of Directional Clustering 166
8.5 Numerical Studies 167
8.6 Conclusions 174
References 176

9 Fuzzy Relational Clustering 178
9.1 Introduction and Problem Statement 178
9.2 FCM for Relational Data 179
9.3 Decomposition of Fuzzy Relational Patterns 181
13 Granular Prototyping 246

13.1 Introduction 246
13.2 Problem Formulation 247
  13.2.1 Expressing Similarity Between Two Fuzzy Sets 247
  13.2.2 Performance Index (Objective Function) 248
13.3 Prototype Optimization 251
13.4 Development of Granular Prototypes 263
  13.4.1 Optimization of the Similarity Levels 263
  13.4.2 An Inverse Similarity Problem 264
13.5 Conclusions 268
  References 268

14 Granular Mappings 270

14.1 Introduction and Problem Statement 270
14.2 Possibility and Necessity Measures as the Computational Vehicles of Granular Representation 271
14.3 Building the Granular Mapping 272
14.4 Designing Multivariable Granular Mappings Through Fuzzy Clustering 275
14.5 Quantification of Granular Mappings 278
14.6 Experimental Studies 278
14.7 Conclusions 280
  References 282

15 Linguistic Modeling 283

15.1 Introduction 283
15.2 Cluster-Based Representation of Input-Output Mapping 285
15.3 Conditional Clustering in the Development of a Blueprint of Granular Models 287
15.4 The Granular Neuron as a Generic Processing Element in Granular Networks 290
15.5 The Architecture of Linguistic Models Based on Conditional Fuzzy Clustering 293
15.6 Refinements of Linguistic Models 294
15.7 Conclusions 295
  References 296

Bibliography 297

Index 315
It is always a challenging task to write a foreword to a work authored by Professor Pedrycz. The reason is that, as a rule, what he writes about goes far beyond what can be found in the existing literature. This is particularly true in the instance of Knowledge-Based Clustering: From Data to Information Granules or Knowledge-Based Clustering, for short. Knowledge-Based Clustering is a magnum opus which touches upon some of the most basic facets of human cognition. It does so with authority, originality, erudition, insight, and high expository skill. Profusion of examples, figures, and references make Professor Pedrycz’s work a pleasure to read.

In Knowledge-Based Clustering, Professor Pedrycz addresses a vast array of linked subjects. Starting with an exposition of clustering and fuzzy clusters, he moves to computing with granular information, with a granule being a clump of attribute-values drawn together by indistinguishability, equivalence, similarity, proximity, or functionality. Professor Pedrycz’s co-authorship of a recent text on granular computing provides him with an effective framework for linking clustering with granular computing. In granular computing, the objects of computation are granules rather than singletons. In its general form, granular computing subsumes computing with intervals, computing with rough sets, and computing with probability distributions. The linkage between granular computing and cluster analysis plays a pivotal role throughout Professor Pedrycz’s work, and is an important novel feature of his approach to cluster analysis.

The chapters that are focused on granular computing serve as a foundation for the core of the book—knowledge-based clustering. In this mode of clustering, clustering is guided by the knowledge that underlies data. There is much that is new in this part of the book, especially in chapters dealing with conditioned fuzzy clustering, collaborative clustering, directional clustering, fuzzy relational clustering, and clustering of nonhomogeneous patterns. The last part of Professor Pedrycz’s work is an informative exposition of applications of knowledge-based clustering to generic models. In this part, we find a range of unconventional concepts and techniques, among them hyperbox modeling, linguistic modeling, and granular mapping.

To see the importance of Pedrycz’s work in a proper perspective, an observation is in order. As we move further into the age of machine intelligence and automated reasoning, a daunting problem becomes harder and harder to master. How can we cope with the explosive growth in data, information, and knowledge? How can we locate and infer from decision-relevant information that is embedded in a large database that is unstructured, imprecise, and not totally reliable?
What these issues point to is an imperative need for new ideas and new techniques in the realm of organization of data, information, and knowledge. In effect, information is organized data and knowledge is organized information.

A key concept that underlies the concept of organization is that of relatedness and, more specifically, the concepts of clustering and granulation. In this perspective, the concepts, ideas, and methods that are the objects of discussion in Professor Pedrycz’s work are of direct relevance to the goal of devising organizational structures that can cope with the explosive growth in data, information, and knowledge.

There is another observation that I should like to make. There is an enormous literature dealing with cluster analysis and related subjects but, strangely enough, what cannot be found in this literature is an operational definition of the concept of a cluster. There is a brief discussion of the concept of cluster validity in Professor Pedrycz’s work, but it stops short of defining the concept of a cluster.

Is this an omission or are there some problems in defining a cluster? In my view, there are two basic problems. First, the concept of a cluster is a fuzzy concept in the sense that it is a matter of degree. And second, the concept of a cluster is a second-order concept in the sense that a cluster is a set of points rather than a singleton. Examples of second-order concepts are convex set, edge, and a mountain. In fact, the concepts of a cluster and mountain have the same deep structure.

The intrinsic problem is that, in general, second-order fuzzy concepts cannot be defined within the conceptual structure of bivalent logic. This is the principal reason why an operational definition of a cluster cannot be found in the literature of cluster analysis. A question that arises is: if the concept of a cluster cannot be defined within the conceptual structure of bivalent logic, then how can it be defined? In my view, what is needed for this purpose is PNL (Precisiated Natural Language)—a language that is based on fuzzy logic—a logic in which everything is or is allowed to be a matter of degree. To define a concept through the use of PNL, with PNL serving as a definition language, two steps are needed. First, the concept is defined in a natural language; second, the natural language definition is precisiated. Unlike bivalent-logic-based definitions, PNL-based definitions are context-dependent rather than context-free. This is the price that has to be paid to achieve a closer rapport with reality.

The fact that there is no definition of a cluster in Professor Pedrycz’s work or, for that matter, anywhere else in the literature, in no way detracts from its importance. Knowledge-Based Clustering is a major contribution that is must reading for anyone who is interested in cluster analysis and, more generally, in the conception, design, and utilization of advanced knowledge-based systems. Knowledge-Based Clustering is a superlative work and its author, Professor Pedrycz, and the publisher, John Wiley & Sons, Inc. deserve our loud applause and congratulations.

Lotfi A. Zadeh
Preface

Data and patterns are an integral part of the cultural fabric of our information society. The challenge we are confronted with every day is to cope with the flood of data generated by banking transactions, millions of sensors, World Wide Web log records, communication traffic of cellular calls, satellite image collection systems, and networks of intelligent home appliances, to name just a few evident examples.

Making sense of data has become a critical objective of intelligent data analysis (IDA), data mining (DM), sensor fusion, image understanding, and logic-driven system modeling. As never before, we are faced with the growing need to construct a powerful computer “eye”—a human-centric, human-interactive, and human-sensitive computer environment that helps us understand data and make sensible decisions.

Clustering is one of the well-established manifestations of such a computer eye. With its agenda of venturing into data spaces and discovering their structure—clusters of data—clustering is an ideal vehicle for exploration of vast territories of data spaces. From the early concepts of the 1930s, this field has recently undergone a rapid expansion fueled by new conceptual and computing challenges. The omnipresence of clustering today is astonishing. Even a quick and fairly unsophisticated search of the Web or a simple search of any library database returns thousands of hits revealing an impressive breadth of applications: from biomedicine to marketing, engineering, economics, biological sciences, chemistry, military, food engineering, finance, and education.

Clustering has become a synonym for a diversified suite of methodologies and algorithms that are almost exclusively data-driven and in which any optimization is predominantly, if not exclusively, data-oriented. Clustering gives rise to a variety of information granules whose use reveals the structure of data. The formalisms of granular computing help design clustering methods designed to meet user-defined objectives. In this diversified landscape of clustering, the algorithms operating within the framework of fuzzy sets have assumed an important and unique position. The reason is obvious: fuzzy sets regarded as basic information granules are human-centric. Dealing with concepts and groups (clusters) that allow for partial membership is highly appealing. Identifying data (patterns) that are of borderline character and may require special attention as potential outliers is a useful value-added feature of fuzzy clustering. Discovering the most typical patterns (with the highest membership values) in the cluster is another important feature of by fuzzy sets.
In light of the recent applications and new forms of agent-based technology, Web-based pursuits, and rapidly growing dimensionality and heterogeneity of data sets, the human-centricity of clustering has become even more essential. The paradigm of data-centric clustering has to be augmented. The paradigm of knowledge-based clustering I introduce in this book is concerned with reconciling two important driving forces of clustering activities: gaining data and domain knowledge and building a coherent platform of navigation in highly dimensional and often heterogeneous data spaces. The user plays a basic role in forming an essential feedback loop in any highly interactive data analysis. Needless to say, we require a carefully selected conceptual and algorithmic layer of human-machine communication.

This book is divided into three parts. The first parts consisting of Chapters 1 to 3, provides a concise, carefully structured introduction to the subject. Three interrelated components are presented. First, I discuss the fundamentals of fuzzy clustering. Second, I review fuzzy computing, regarded as an important realization of granular computing, focused on the issues of fuzzy clustering. Third, I elaborate on the logic-based neurons and ensuing neural networks. The core of the book, Chapters 4 to 10, presents a highly diversified landscape of knowledge-based clustering. The third part of the book, consisting of Chapters 11 to 15, is devoted to generic models whose design is directly linked to the paradigm of knowledge-based clustering. First, I concentrate on hyperbox models of clusters, demonstrating how the essential structure can be captured in terms of hyperbox geometry. This is followed by studies of granular mappings and linguistic models.

Throughout the book, I adhere to the standard notation used in pattern recognition and system analysis, as well as the standard terminology used there. The terms “data” and “pattern” are used interchangeably to emphasize the unified way of treating various forms of pattern recognition, system modeling, and data analysis. The book is self-contained. While the reader can benefit from some initial familiarity with computational Intelligence (CI), this is not a must. CI helps place the material in perspective and allows the reader to fully appreciate the ideas of information granularity and information granules as building blocks of various CI architectures.

The purpose of this book is to present the main ideas in a fairly general format and not to skew the subject by limiting the discussion to selected application areas. The algorithmic aspects are also kept quite general, and no attempt is made to strive for the most efficient yet intricate implementations possible. This makes the book of interest to a broad audience. Those readers interested in clustering, fuzzy clustering, unsupervised learning, neural networks, fuzzy sets, and pattern recognition, as well as those involved in numerous tasks of data analysis, will find the book thought-provoking and intellectually stimulating. Readers involved in system modeling will view knowledge-driven clustering as an attractive vehicle of rapid prototyping of granular models.

Knowledge-based clustering has already emerged. This book outlines its fundamentals, presents the essential algorithmic developments, and discusses its
application-driven aspects. No attempt has been made to cover the subject completely. However, the material selected paints a coherent picture of the most recent developments central to this rapidly evolving area.

Witold Pedrycz
1 Clustering and Fuzzy Clustering

This chapter provides a comprehensive, focused introduction to clustering, viewed as a fundamental means of exploratory data analysis, unsupervised learning, data granulation, and information compression. We discuss the underlying principles, elaborate on the basic taxonomy of numerous clustering algorithms (including such essential classes as hierarchical, objective function-based algorithms), and review the main interpretation mechanisms associated with various clustering algorithms.

1.1. INTRODUCTION

Making sense of data is an ongoing task of researchers and professionals in almost every practical endeavor. The age of information technology, characterized by a vast array of data, has enormously amplified this quest and made it even more challenging. Data collection anytime and everywhere has become the reality of our lives. Understanding the data, revealing underlying phenomena, and visualizing major tendencies are major undertakings pursued in intelligent data analysis (IDA), data mining (DM), and system modeling.

Clustering is a general methodology and a remarkably rich conceptual and algorithmic framework for data analysis and interpretation (Anderberg, 1973; Bezdek, 1981; Bezdek et al., 1999; Devijver and Kittler, 1987; Dubes, 1987; Duda et al., 2001; Fukunaga, 1990; Hoppner et al., 1999; Jain et al., 1999, 2000; Kaufmann and Rousseeuw, 1990; Babu and Murthy, 1994; Dave, 1990; Dave and Bhaswan, 1992; Kersten, 1999; Klawonn and Keller, 1998; Mali and Mitra, 2002; Webb, 2002). In this chapter, we introduce basic notions, explain the functional components essential to the formulation of clustering problems, and discuss the main classes of clustering algorithms. These algorithms are accompanied by the formalisms of granular computing, including sets, fuzzy sets, shadowed sets, and rough sets.

1.2. BASIC NOTIONS AND NOTATION

To establish a formal setting in which clustering can be carried out, we start with basic notions such as data types, distance, and similarity/resemblance.
1.2.1. Types of Data

The world surrounding us generates various types of data in abundance. The richness of data formats is impressive. The formal representation and organization of patterns reflect the way in which we intend to process the data. The most general taxonomy being in common use distinguishes among numeric (continuous), ordinal, and nominal variables. A numeric variable can assume any value in \( \mathbb{R} \). An ordinal variable assumes a small number of discrete states, and these states can be compared. For instance, there are four states, denoted \( a_1, a_2, a_3, \) and \( a_4 \), and we can say that \( a_1 \) and \( a_2 \) are closer (in some sense of similarity that we define in the next section) than \( a_1 \) and \( a_3 \). A nominal variable assumes a small number of states, but nothing can be said about their closeness. Regardless of this distinction, nominal and ordinal variables are represented as discrete variables. For computing purposes, we usually have several coding schemes, such as binary coding or binary coding with various options.

The variables can be organized into internal structures that reflect the specificity of the problem. If each pattern is described by a number of features, intuitively we arrange them into vectors—say, \( x, y, \) and \( z \). Depending upon the character of the variables involved, the entries can be real or binary. Obviously, this can give rise to a variety of vectors, including both types of entries. Vectors and matrices are “flat” structures in the sense that all variables are at the same level as individual entries of the feature vector, and they have no structure. Hierarchical structures like trees are used to visualize the relationship between objects (patterns) we are interested in when dealing with clustering or classification.

1.2.2. Distance and Similarity

The concept of dissimilarity (or distance) or dual similarity is the essential component of any form of clustering that helps us navigate through the data space and form clusters. By computing dissimilarity, we can sense and articulate how close together two patterns are and, based on this closeness, allocate them to the same cluster. Formally, the dissimilarity \( d(x, y) \) between \( x \) and \( y \) is considered to be a two-argument function satisfying the following conditions:

\[
\begin{align*}
  d(x, y) &\geq 0 \quad \text{for every } x \text{ and } y \\
  d(x, x) & = 0 \quad \text{for every } x \\
  d(x, y) & = d(y, x)
\end{align*}
\]

This list of requirements is intuitively appealing. We require a nonnegative character of the dissimilarity. The symmetry is also an obvious requirement. The dissimilarity attains a global minimum when dealing with two identical patterns, that is \( d(x, x) = 0 \).

Distance, (metric) is a more restrictive concept, as we require the triangular inequality to be satisfied; that is, for any pattern \( x, y, \) and \( z \) we have

\[
d(x, y) + d(y, z) \geq d(x, z)
\]
TABLE 1.1. Selected Distance Functions Between Patterns \( x \) and \( y \)

<table>
<thead>
<tr>
<th>Distance Function</th>
<th>Formula and Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>Euclidean distance</td>
<td>( d(x, y) = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2} )</td>
</tr>
<tr>
<td>Hamming (city block) distance</td>
<td>( d(x, y) = \sum_{i=1}^{n}</td>
</tr>
<tr>
<td>Tchebyschev distance</td>
<td>( d(x, y) = \max_{i=1,2,...,n}</td>
</tr>
<tr>
<td>Minkowski distance</td>
<td>( d(x, y) = \sqrt[p]{\sum_{i=1}^{n} (x_i - y_i)^p}, \ p &gt; 0 )</td>
</tr>
<tr>
<td>Canberra distance</td>
<td>( d(x, y) = \sum_{i=1}^{n} \frac{</td>
</tr>
</tbody>
</table>
| Angular separation                 | \( d(x, y) = \left( \frac{1}{2} \sum_{i=1}^{n} x_i^2 y_i^2 \right)^{1/2} \)

Note: this is a similarity measure that expresses the angle between the unit vectors in the direction of \( x \) and \( y \)

In the case of continuous features (variables), we have a long list of distance functions (see Table 1.1). Each of these functions implies a different view of the data because of their geometry. The geometry is easily illustrated when we consider only two features \( x = [x_1, x_2]^T \) and compute the distance of \( x \) from the origin. The contours of the constant distance (Figure 1.1) show what type of geometric construct becomes a focus of the search for structure. Here we become aware that the Euclidean distance favors circular shapes of data clusters. With the distance functions come some taxonomy; the Minkowski distance comprises an infinite family of distances, including well-known and commonly used ones such as the Hamming, Tchebyschev, and Euclidean distances.

The same effect shown in Figure 1.1 can be achieved when the value of the power in the Minkowski distance is changed; see Figure 1.2.

One commonly used generalization is the Mahalanobis distance

\[
d(x, y) = x^T A^{-1} y
\]

(1.3)

where \( A \) is a positive definite matrix. By choosing this matrix, we can control the geometry of potential clusters by rotating the ellipsoid (off diagonal entries of
A) and changing the length of its axes (the elements lying on the main diagonal of the matrix).

With binary variables, we traditionally focus on the notion of similarity rather than distance (or dissimilarity). Consider two binary vectors \( \mathbf{x} \) and \( \mathbf{y} \) that consist of two strings \([x_k], [y_k]\) of binary data; compare them coordinatewise and do the simple counting of occurrences:

- number of occurrences when \( x_k \) and \( y_k \) are both equal to 1
- number of occurrences when \( x_k = 0 \) and \( y_k = 1 \)
- number of occurrences when \( x_k = 1 \) and \( y_k = 0 \)
- number of occurrences when \( x_k \) and \( y_k \) are both equal to 0

**Figure 1.1.** Examples of distance functions—three-dimensional and contour plots: (a) Euclidean, (b) Hamming (city block), (c) Tchebyschev, (d) “combined” type of distance max (2/3 Hamming, Tchebyschev).
Figure 1.2. Examples of the Minkowski distance function for selected values of the power: (a) 1.5 (b) 2.5, and (c) 7.0.

These four numbers can be organized in a 2 by 2 co-occurrence matrix (contingency table) that visualizes how “close” these two strings are to each other.

\[
\begin{array}{cc}
1 & 0 \\
1 & a & b \\
0 & c & d \\
\end{array}
\]

Evidently the zero nondiagonal entries of this matrix point at the ideal matching (the highest similarity). Based on these four entries, there are several commonly encountered measure of similarity of binary vectors $\mathbf{x}$ and $\mathbf{y}$. The simplest
matching coefficient computes as the following ratio:

\[ \frac{a + d}{a + b + c + d} \]  

(1.4)

The Russell and Rao measure of similarity consists of the quotient

\[ \frac{a}{a + b + c + d} \]  

(1.5)

The Jacard index involves the case when both inputs assume values equal to 1:

\[ \frac{a}{a + b + c} \]  

(1.6)

The Czekanowski index is practically the same as the Jacard index, but by adding the weight factor of 2, it emphasizes the coincidence of situations where entries of \( x \) and \( y \) both assume values equal to 1:

\[ \frac{2a}{2a + b + c} \]  

(1.7)

### 1.3. MAIN CATEGORIES OF CLUSTERING ALGORITHMS

Clustering techniques are rich and diversified. They have been continuously developing for over a half century following a number of trends, depending upon the emerging optimization techniques, main methodology (system modeling, DM, signal processing), and application areas. At the very high end of the overall taxonomy we envision two main categories of clustering, known as hierarchical and objective function-based clustering.

#### 1.3.1. Hierarchical Clustering

The clustering techniques in this category produce a graphic representation of data (Duda et al., 2001). The construction of graphs (as these methods reveal the structure by considering each individual pattern) is done in two ways: bottom-up and top-down. The other names used reflect the way a structure is revealed. In the bottom-up mode known as an agglomerative approach, we treat each pattern as a single-element cluster and then successively merge the closest clusters. At each pass of the algorithm, we merge the two closest clusters. The process is repeated until we get to a single data set or reach a certain predefined threshold value. The top-down approach, known as a divisive approach, works in the opposite direction: we start with the entire set treated as a single cluster and keep splitting it into smaller clusters. Considering the nature of the process, these methods are often computationally inefficient, with the possible exception of patterns with binary variables.
The results of hierarchical clustering are usually represented in the form of dendrograms (Figure 1.3). Dendrograms are visually appealing graphical constructs: they show how difficult it is to merge two clusters. The distance scale shown at the right-hand side of the graph helps us quantify the distance between the clusters. This implies a simple stopping criterion: given a certain threshold value of the distance, we stop merging the clusters once the distance between them exceeds this threshold, meaning that merging two distinct structures does not seem to be feasible.

An important issue is how to measure the distance between two clusters. Note that we have discussed how to express the distance between two patterns. Here, as each cluster may contain many patterns, computation of the distance is neither obvious nor unique. Consider two clusters, \( A \) and \( B \), illustrated in Figure 1.4. Let us describe the distance by \( d(A, B) \) and denote the number of patterns in \( A \) and \( B \) by \( n_1 \) and \( n_2 \), respectively. Intuitively, we can easily envision three typical ways of computing the distance between the two clusters.

![Figure 1.3. A dendrogram as a visualization of the structure of patterns; also shown are the distance values guiding the process of successive merging of the clusters.](image)

![Figure 1.4. Two clusters and three main ways of computing the distance between them: (a) single link, (b) complete link, and (c) group average link.](image)
Single-Link Method. The distance \( d(A, B) \) is based on the minimal distance between the patterns belonging to \( A \) and \( B \). It is computed in the form

\[
d(A, B) = \min_{x \in A, y \in B} d(x, y)
\]

In essence, the distance supports a sort of radically “optimistic” mode of expressing vicinity between clusters where we get involved the closest patterns located in different clusters. Clustering based on this distance is one of the most commonly used methods.

Complete-Link Method. This method is at the opposite end of the spectrum, as it is based on the distance between the two farthest patterns belonging to two clusters:

\[
d(A, B) = \max_{x \in A, y \in B} d(x, y)
\]

Group Average Link Method. In contrast to the two previous approaches, where the distance is determined on the basis of extreme values of the distance function, this method considers the average between the distances computed between all pairs of patterns, one from each cluster. We have

\[
d(A, B) = \frac{1}{\text{card}(A) \text{card}(B)} \sum_{x \in A, y \in B} d(x, y)
\]

Obviously, these computations are more intensive. However, they reflect a general tendency between the distances computed for individual pairs of patterns.

Obviously, we can develop other ways of expressing the distance between \( A \) and \( B \). For instance, the Hausdorff method of computing the distance between two sets of patterns could be an attractive alternative.

There is an interesting general expression for describing various agglomerative clustering approaches known as the Lance-Williams recurrence formula. It expresses the distance between clusters \( A \) and \( B \) and the cluster formed by merging them \((C)\)

\[
d_{A\cup B, C} = \alpha_A d_{A,C} + \alpha_B d_{B,C} + \beta d_{A,B} + \gamma |d_{A,C} - d_{B,C}|
\]

with the adjustable values of the parameters \( \alpha_A, \alpha_B, \beta, \gamma \). This is shown in Table 1.2, where the choice of values implies a certain clustering method.

1.3.2. Objective Function-Based Clustering

The second general category of clustering is concerned with building partitions (clusters) of data sets on the basis of some performance index known also as an objective function. In essence, partitioning \( N \) patterns into \( c \) clusters (groups)
TABLE 1.2. Values of the Parameters in the Lance-Williams Recurrence Formula
and the Resulting Agglomerative Clustering; \( n_A \), \( n_B \), and \( n_C \) Denote the Number of Patterns in the Corresponding Clusters

<table>
<thead>
<tr>
<th>Clustering Method</th>
<th>( \alpha_A (\alpha_B) )</th>
<th>( \beta )</th>
<th>( \gamma )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Single link</td>
<td>1/2</td>
<td>0</td>
<td>(-1/2)</td>
</tr>
<tr>
<td>Complete link</td>
<td>1/2</td>
<td>0</td>
<td>1/2</td>
</tr>
<tr>
<td>Centroid</td>
<td>( \frac{n_A}{n_A + n_B} )</td>
<td>( \frac{n_A n_B}{(n_A + n_B)^2} )</td>
<td>0</td>
</tr>
<tr>
<td>Median</td>
<td>1/2</td>
<td>(-1/4)</td>
<td>0</td>
</tr>
</tbody>
</table>

is a nontrivial problem. First, the number of the partitions is expressed in the following form (Webb, 2002):

\[
\frac{1}{c!} \sum_{i=1}^{c} (-1)^{c-i} \binom{c}{i} i^N
\]  

(1.12)

This number increases very quickly, making any attempt to enumerate all of the partitions unfeasible. The minimization of a certain objective function can be treated as an optimization approach leading to some suboptimal configuration of the clusters (which, in practice, is an appealing solution). The main design challenge lies in formulating an objective function that is capable of reflecting the nature of the problem so that its minimization reveals a meaningful structure in the data set. The minimum variance criterion is one of the most common options. Having \( N \) patterns in \( \mathbb{R}^n \), and assuming that we are interested in forming \( c \) clusters, we compute a sum of dispersions between the patterns and a set of prototypes \( \mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_c \)

\[
Q = \sum_{i=1}^{c} \sum_{k=1}^{N} u_{ik} \| \mathbf{x}_k - \mathbf{v}_i \|^2
\]  

(1.13)

with \( \| \cdot \|^2 \) being a certain distance between \( \mathbf{x}_k \) and \( \mathbf{v}_i \). The important component in the above sum is a partition matrix \( U = [u_{ik}], i = 1, 2, \ldots, c, k = 1, 2, \ldots, N \) whose role is to allocate the patterns to the clusters. The entries of \( U \) are binary. Pattern \( k \) belongs to cluster \( i \) when \( u_{ik} = 1 \). The same pattern is excluded from the cluster when \( u_{ik} = 0 \). Partition matrices satisfy the following conditions:

Each cluster is nontrivial, that is, it does not include all patterns and is nonempty:

\[
0 < \sum_{k=1}^{N} u_{ik} < N, \quad i = 1, 2, \ldots, c
\]
Each pattern belongs to a single cluster:

$$\sum_{i=1}^{c} u_{ik} = 1, \quad k = 1, 2, \ldots, N$$

The family of partition matrices (viz., binary matrices satisfying these two conditions) will be denoted by $U$. As a result of minimization of $Q$, we construct the partition matrix and a set of prototypes. Formally we express this in the following way, which is just an optimization problem with constraints:

$$\text{Min } Q \text{ with respect to } v_1, v_2, \ldots, v_c \text{ and } U \in U$$  \hspace{1cm} (1.14)

Several methods are used to achieve this optimization. The most common one, named C-Means (Duda et al., 2001; Webb, 2002), is a well-established way of clustering data.

Partition matrices are an intuitively appealing form in which to illustrate the structure of the patterns. For instance, the matrix formed for $N = 8$ patterns split into $c = 3$ clusters is

$$U = \begin{bmatrix}
1 & 0 & 0 & 1 & 0 & 1 & 0 & 1 \\
0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 1 & 0
\end{bmatrix}$$

Each row describes a single cluster. Thus we have the following arrangement: the first cluster consists of patterns $\{1, 4, 6, 8\}$, the second involves patterns $\{2, 3\}$, and the third covers the remaining patterns, $\{5, 7\}$.

Graphically, the partition matrix (or, equivalently, the structure of the data set) can be shown in the form of a so-called star or radar diagram (Figure 1.5).

1.4. CLUSTERING AND CLASSIFICATION

The structure revealed through the clustering process allows us to set up a classifier. The “anchor” points of the classifier are the prototypes of the clusters. Each