FUZZY PATTERN RECOGNITION

Fuzzy sets were introduced by Zadeh (1) to represent nonstatistical uncertainty. Suppose you must advise a driving student when to apply the brakes of a car. Would you say "begin braking *74.2 feet* from the crosswalk"? Or would you say "apply the brakes *pretty* soon"? You would choose the second instruction because the first one is *too precise* to be implemented. So, precision can be useless, while vague directions can be interpreted and acted upon. Fuzzy sets are used to endow computational models with the ability to recognize, represent, manipulate, interpret, and use (act on) nonstatistical imprecision.

Conventional (crisp) sets contain objects that satisfy *precise properties.* The set $H = \{r \in \Re | 6 \leq r \leq 8\}$ is crisp. H can be described by its membership function,

$$m_{\mathrm{H}}(r) = \begin{cases} 1 & 6 \leq r \leq 8 \\ 0 & \text{otherwise} \end{cases}$$

Since m_{H} maps all real numbers onto the two points $\{0, 1\}$, crisp sets correspond to 2-valued logic; every real number either is in H or is not.

Consider the set F of real numbers that are close to seven. Since "close to seven" is fuzzy, there is not a unique membership function for F. Rather, the modeler must decide, based on the potential application and imprecise properties of F, what m_F *should* be. Properties that seem plausible for this F include: (1) normality ($m_F(7) = 1$); (2) unimodality (only $m_F(7) = 1$); (3) the closer r is to 7, the closer $m_F(\mathrm{r})$ is to 1, and conversely; and (4) symmetry (numbers equally far left and right of 7 should have equal memberships). Infinitely many functions satisfy these intuitive constraints. For example, $m_{1F}(r) = e^{-(r-7)^2}$ and $m_{2\mathrm{F}}(r) = 1/(1 + (r - 7)^2)$. Notice that no physical entity corresponds to F. Fuzzy sets are realized only through membership functions, so it is correct to call m_F the fuzzy set F, even though it is a function.

Formally, *every* function $m: X \mapsto [0, 1]$ could be a fuzzy subset of any set X, but functions like this become fuzzy sets when and only when they match some intuitively plausible semantic description of imprecise properties of the objects in X.

A question that continues to spark much debate is whether or not fuzziness is just a clever disguise for probability. The answer is no. Fuzzy memberships represent similarities of objects to imprecisely defined properties; probabilities convey information about relative frequencies. Another common misunderstanding is that fuzzy models are offered as replacements for crisp or probabilistic models. But most schemes that use fuzziness use it in the sense of embedding: Conventional structure is preserved as a special case of fuzzy structure, just as the real numbers are a special case of the complex numbers. Zadeh (2) first discussed models that had both fuzziness and probability. A recent publication about this is special issue 2(1) of the *IEEE Transactions on Fuzzy Systems,* 1994.

PATTERN RECOGNITION: DATA, LABEL VECTORS, AND MEASURES OF SIMILARITY

There are two major approaches to pattern recognition: numerical (3) and syntactic (4). Discussed here are three areas of numerical pattern recognition for object data: clustering, classifier design, and feature analysis. The earliest reference to fuzzy pattern recognition was Bellman et al. (5). Fuzzy techniques for numerical pattern recognition are now fairly mature. Reference 6 is an edited collection of 51 papers on this subject that span the development of the field from 1965 to 1991.

Object data are represented as $X = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\}$, a set of n feature vectors in feature space $\Re^p$. The jth object is a physical entity such as a fish, medical patient, and so on. Column vector $\boldsymbol{x}_j$ is the object's numerical representation; x_{kj} is its kth *feature.* There are four types of class labels—crisp, fuzzy, probabilistic and possibilistic. Let integer c denote the number of classes, $1 < c < n$. Define three sets of label vectors in $\Re^c$ as follows:

$$\text{let } [0,1]^c = \underbrace{[0,1] \times \cdots \times [0,1]}_{c \text{ times}},$$

$$N_{pc} = \{\boldsymbol{y} \in \Re^c : y_i \in [0,1] \forall i, y_i > 0 \exists i\} = [0,1]^c - \{\boldsymbol{0}\} \qquad (1)$$

$$N_{fc} = \{\boldsymbol{y} \in N_{pc} : \sum_{i=1}^{c} y_i = 1\} \qquad (2)$$

$$N_{hc} = \{\boldsymbol{y} \in N_{fc} : y_i \in \{0,1\} \forall i\} = \{\boldsymbol{e}_1, \boldsymbol{e}_2, \ldots, \boldsymbol{e}_c\} \qquad (3)$$

In Eq. (1) $\boldsymbol{0}$ is the zero vector in $\Re^c$. Note that $N_{hc} \subset N_{fc} \subset N_{pc}$. N_{hc} is the canonical (unit vector) basis of Euclidean c-space.

$$\boldsymbol{e}_i = (0, 0, \ldots, \underbrace{1}_{i}, \ldots, 0)^T,$$

the ith vertex of N_{hc}, is the crisp label for class i, $1 \le i \le c$. N_{fc}, a piece of a hyperplane, is the convex hull of N_{hc}. The vector $\boldsymbol{y} = (0.1, 0.6, 0.3)^T$ is a label vector in N_{f3}; its entries lie between 0 and 1 and are constrained to sum to 1. If $\boldsymbol{y}$ is a label vector for some $\boldsymbol{z} \in \Re^p$ generated by, say, the fuzzy c-means clustering method, $\boldsymbol{y}$ is a *fuzzy label* for $\boldsymbol{z}$. If $\boldsymbol{y}$ came from a method such as maximum likelihood estimation in mixture decomposition, it would be a *probabilistic label* for $\boldsymbol{z}$.

$N_{pc} = [0, 1]^c - \{\boldsymbol{0}\}$ is the unit hypercube in $\Re^c$, excluding the origin. Vectors such as $\boldsymbol{y} = (0.4, 0.2, 0.7)^T$ are *possibilistic* label vectors in N_{p3}. Labels in N_{pc} are produced, for example, by possibilistic clustering algorithms (7) and neural networks (8).

Most pattern recognition models are based on statistical or geometrical properties of substructure in X. Two key concepts for describing geometry are angle and distance. Let A be any positive-definite $p \times p$ matrix. For vectors $\boldsymbol{x}, \boldsymbol{v} \in \Re^p$,

$$\langle \boldsymbol{x}, \boldsymbol{v} \rangle_A = \boldsymbol{x}^T A \boldsymbol{v} \qquad (4)$$

$$\|\boldsymbol{x}\|_A = \sqrt{\boldsymbol{x}^T A \boldsymbol{x}} \qquad (5)$$

and

$$\delta_A(\boldsymbol{x}, \boldsymbol{v}) = \|\boldsymbol{x} - \boldsymbol{v}\|_A = \sqrt{(\boldsymbol{x} - \boldsymbol{v})^T A (\boldsymbol{x} - \boldsymbol{v})} \qquad (6)$$

are the inner product, norm (length), and norm metric (distance) induced on $\Re^p$ by A. The most important instances of Eq. (6), together with their common names and inducing matrices, are

$$\|\boldsymbol{x} - \boldsymbol{v}\|_I = \sqrt{(\boldsymbol{x} - \boldsymbol{v})^T (\boldsymbol{x} - \boldsymbol{v})} \qquad \text{Euclidean, } A = I \qquad (7)$$

$$\|\boldsymbol{x} - \boldsymbol{v}\|_{D^{-1}} = \sqrt{(\boldsymbol{x} - \boldsymbol{v})^T D^{-1} (\boldsymbol{x} - \boldsymbol{v})} \qquad \text{Diagonal, } A = D^{-1} \qquad (8)$$

$$\|\boldsymbol{x} - \boldsymbol{v}\|_{M^{-1}} = \sqrt{(\boldsymbol{x} - \boldsymbol{v})^T M^{-1} (\boldsymbol{x} - \boldsymbol{v})} \qquad \text{Mahalanobis, } A = M^{-1} \qquad (9)$$

In Eq. (7) I is the $p \times p$ identity matrix. Equations (8) and (9) use the covariance matrix of X, $M = \text{cov}(X) = \Sigma_{k=1}^{n} (\boldsymbol{x}_k - \overline{\boldsymbol{v}})(\boldsymbol{x}_k - \overline{\boldsymbol{v}})^T/n$, where $\overline{\boldsymbol{v}} = \Sigma_{k=1}^{n} \boldsymbol{x}_k/n$. D is the diagonal matrix extracted from M by deletion of its off-diagonal entries. A second family of commonly used lengths and distances are the *Minkowski norm* and *norm metrics*:

$$\|\boldsymbol{x}\|_q = \left(\sum_{j=1}^{p} \left| x_j \right|^q \right)^{1/q}, \qquad q \ge 1 \qquad (10)$$

$$\delta_q(\boldsymbol{x}, \boldsymbol{v}) = \|\boldsymbol{x} - \boldsymbol{v}\|_q = \left(\sum_{j=1}^{p} \left| x_j - v_j \right|^q \right)^{1/q}, \qquad q \ge 1 \qquad (11)$$

Three are commonly used:

$$\|\boldsymbol{x} - \boldsymbol{v}\|_1 = \left(\sum_{j=1}^{p} \left| x_j - v_j \right| \right) \qquad \text{City block (1-norm); } q = 1 \qquad (12)$$

$$\|\boldsymbol{x} - \boldsymbol{v}\|_2 = \left(\sum_{j=1}^{p} \left| x_j - v_j \right|^2 \right)^{1/2} \qquad \text{Euclidean (2-norm); } q = 2 \qquad (13)$$

$$\|\boldsymbol{x} - \boldsymbol{v}\|_\infty = \max_{1 \le j \le p} \left\{ \left| x_j - v_j \right| \right\} \qquad \text{Sup or Max norm; } q \to \infty \qquad (14)$$

Equations (7) and (13) both give the Euclidean norm metric, the only one in both of the inner product and Minkowski norm metric families.

FUZZY CLUSTER ANALYSIS

This field comprises three problems: tendency assessment, clustering and validation. Given an unlabeled data set X, is there substructure in X? This is clustering tendency—should you look for clusters at all? Very few methods—fuzzy or otherwise—address this problem. Jain and Dubes (9) discuss some formal methods for assessment of cluster tendency, but most users begin clustering without checking the data for possible tendencies. Why? Because it is impossible to guess what structure your data may have in p dimensions, so hypothesis tests cast against structure that cannot be verified are hard to interpret. The usefulness of tendency assessment lies with its ability to rule out certain types of cluster structure.

Different clustering algorithms produce different partitions of X, and it is never clear which one(s) may be most useful. Once clusters are obtained, how shall we pick the best clustering solution (or solutions)? This is *cluster validation* (4,5,9,10). Brevity precludes a discussion of this topic here.

Clustering (or unsupervised learning) in unlabeled X is the assignment of (hard *or* fuzzy *or* probabilistic *or* possibilistic) label vectors to the $\{\boldsymbol{x}_k\}$. Cluster substructure is represented by a $c \times n$ matrix $U = [\boldsymbol{U}_1 \ . \ . \ . \ \boldsymbol{U}_k \ . \ . \ . \ \boldsymbol{U}_n] = [u_{ik}]$, where $\boldsymbol{U}_k$ denotes the kth column of U. A c-partition of X belongs to one of three sets:

$$M_{pcn} = \{U \in \Re^{cn}: \boldsymbol{U}_k \in N_{pc} \forall k\} \tag{15}$$

$$M_{fcn} = \left\{U \in M_{pcn}: \boldsymbol{U}_k \in N_{fc} \forall k;\ 0 < \sum_{k=1}^{n} u_{ik} \forall i\right\} \tag{16}$$

$$M_{hcn} = \{U \in M_{fcn}: \boldsymbol{U}_k \in N_{hc} \forall k\} \tag{17}$$

Equations (15)–(17) define, respectively, the sets of possibilistic, fuzzy or probabilistic, and crisp c-partitions of X. Each column of U in $M_{pcn}(M_{fcn}, M_{hcn})$ is a label vector from $N_{pc}(N_{fc}, N_{hc})$. Note that $M_{hcn} \subset M_{fcn} \subset M_{pcn}$. The reason these matrices are called *partitions* follows from the interpretation of u_{ik}. If U is crisp or fuzzy, u_{ik} is the membership of $\boldsymbol{x}_k$ in the ith partitioning fuzzy subset (cluster) of X. If U is probabilistic, u_{ik} is usually the (posterior) probability that, given $\boldsymbol{x}_k$, it came from class i. When U is possibilistic, u_{ik} is the typicality of $\boldsymbol{x}_k$ to class i.

Since definite class assignments are often the ultimate goal, labels in N_{pc} or N_{fc} are usually transformed into crisp labels. Most noncrisp partitions are converted to crisp ones using the hardening function $\boldsymbol{H}$: $N_{pc} \mapsto N_{hc}$, that is,

$$\boldsymbol{H}(\boldsymbol{y}) = \boldsymbol{e}_i \Leftrightarrow \|\boldsymbol{y} - \boldsymbol{e}_i\|_2 \le \|\boldsymbol{y} - \boldsymbol{e}_j\|_2 \Leftrightarrow y_i \ge y_j; \quad j \ne i \tag{18}$$

In Eq. (18), ties are broken randomly. $\boldsymbol{H}$ finds the crisp label vector $\boldsymbol{e}_i$ in N_{hc} closest to $\boldsymbol{y}$. Alternatively, $\boldsymbol{H}$ finds the maximum coordinate of $\boldsymbol{y}$ and assigns the corresponding crisp label to the object $\boldsymbol{z}$ that $\boldsymbol{y}$ labels. For fuzzy partitions, hardening each column of U with Eq. (18) is called defuzzification by maximum membership (MM):

$$\boldsymbol{U}_{MM,k} = \boldsymbol{H}(\boldsymbol{U}_k) = \boldsymbol{e}_i \Leftrightarrow u_{ik} \ge u_{jk} \qquad \forall j \ne i;\ 1 \le k \le n \tag{19}$$

Crisp c-partitions of X obtained this way are denoted by $U^{\boldsymbol{H}} = [\boldsymbol{H}(U_1) \ . \ . \ . \ \boldsymbol{H}(U_n)]$.

Example 1. Let $O = \{o_1 = \text{peach}, o_2 = \text{plum}, o_3 = \text{nectarine}\}$, and let $c = 2$. Typical 2-partitions of O are as follows:

$$\begin{array}{lccc}
\textit{Object} & o_1\ o_2\ o_3 & o_1\ \ o_2\ \ o_3 & o_1\ \ o_2\ \ o_3 \\
\begin{array}{l}\textit{Peaches}\\ \textit{Plums}\end{array} & \begin{bmatrix}1 & 0 & 0\\ 0 & 1 & 1\end{bmatrix} & \begin{bmatrix}1 & 0.2 & 0.4\\ 0 & 0.8 & 0.6\end{bmatrix} & \begin{bmatrix}1 & 0.2 & 0.5\\ 0 & 0.8 & 0.8\end{bmatrix} \\
 & U_1 \in M_{h23} & U_2 \in M_{f23} & U_3 \in M_{p23}
\end{array}$$

The nectarine, o_3, is labeled by the last column of each partition, and in the crisp case it must be (erroneously) given full membership in one of the two crisp subsets partitioning X. In U_1, o_3, is labeled plum. Noncrisp partitions enable models to (sometimes!) avoid such mistakes. The last column of U_2 allocates most (0.6) of the membership of o_3 to the plums class but also assigns a lesser membership (0.4) to o_3 as a peach. U_3 illustrates a possibilistic partition, and its third column exhibits a possibilistic label for the nectarine. The values in the third column indicate that this nectarine is more typical of plums than of peaches.

Columns like the ones for the nectarine in U_2 and U_3 serve a useful purpose: Lack of strong membership in a single class is a signal to "take a second look." In this example the nectarine is a *hybrid* of peaches and plums, and the memberships shown for it in the last column of either U_2 or U_3 seem more plausible physically than crisp assignment of o_3 to an incorrect class. M_{pcn} and M_{fcn} can be more realistic than M_{hcn} because boundaries between many classes of real objects are badly delineated (i.e., really fuzzy). M_{fcn} reflects the degrees to which the classes share $\{o_k\}$ because of the constraint that $\sum_{i=1}^{c} u_{ik} = 1$. M_{pcn} reflects the degrees of typicality of $\{o_k\}$ with respect to the prototypical (ideal) members of the classes.

Finally, observe that $U_1 = U_2^{\boldsymbol{H}} = U_3^{\boldsymbol{H}}$. Crisp partitions of data do not possess the information content to suggest fine details of infrastructure such as hybridization or mixing that are available in U_2 and U_3. Here, hardening U_2 and U_3 with $\boldsymbol{H}$ destroys useful information.

The *c*-Means Clustering Models

How can we find partitions of X such as those in Example 1? The c-means models are used more widely than any other clustering methods for this purpose. The optimization problem that defines the hard (H), fuzzy (F), and possibilistic (P) c-means (HCM, FCM, and PCM, respectively) models is

$$\min_{(U,\mathbf{V})} \left\{ J_m(U, \mathbf{V}; \boldsymbol{w}) = \sum_{i=1}^{c} \sum_{k=1}^{n} u_{ik}^m D_{ik}(\boldsymbol{x}_k, \boldsymbol{v}_i) + \sum_{i=1}^{c} w_i \sum_{k=1}^{n} (1 - u_{ik})^m \right\} \tag{20}$$

where

$U \in M_{hcn}, M_{fcn}$ or M_{pcn},	depending on the approach
$\mathbf{V} = (\boldsymbol{v}_1, \boldsymbol{v}_2, \ldots, \boldsymbol{v}_c) \in \Re^{cp}$,	$\boldsymbol{v}_i$ specifies the ith point prototype
$\boldsymbol{w} = (w_1, w_2, \ldots, w_c)^{\mathrm{T}}$,	$w_i \ge 0$ are *user-specified* penalty weights

$m \ge 1$ is a weighting exponent that controls the degree of fuzzification of U, and $D_{ik}(\boldsymbol{x}_k, \boldsymbol{v}_i) = D_{ik}$ is the deviation of $\boldsymbol{x}_k$ from the ith cluster prototype.

Optimizing $J_m(U, \boldsymbol{V}; \boldsymbol{w})$ when D_{ik} is an inner product norm metric, $D_{ik} = \|\boldsymbol{x}_k - \boldsymbol{v}_i\|_A^2$, is usually done by alternating optimization (AO) through the first-order necessary conditions on $(U, \boldsymbol{V})$:

HCM: Minimize over $M_{hcn} \times \Re^{cp}$: $m = 1$: $w_i = 0\ \forall\ i$. $(U, \boldsymbol{V})$ may minimize J_1 only if

$$u_{ik} = \begin{cases} 1; & \|\boldsymbol{x}_k - \boldsymbol{v}_i\|_A \le \|\boldsymbol{x}_k - \boldsymbol{v}_j\|_A, \quad j \ne i \\ 0, & \textit{otherwise} \end{cases}$$

$$\forall i, k; \text{ties are broken randomly} \tag{21}$$

$$\boldsymbol{v}_i = \left(\sum_{k=1}^{n} u_{ik}\boldsymbol{x}_k \Big/ \sum_{k=1}^{n} u_{ik} \right) \qquad \forall i \tag{22}$$

FCM: Minimize over $M_{fcn} \times \Re^{cp}$: assume $\|\boldsymbol{x}_k - \boldsymbol{v}_i\|_A^2 > 0\ \forall\ i, k$: $m > 1$: $w_i = 0\ \forall\ i$. $(U, \boldsymbol{V})$ may minimize J_m only if

$$u_{ik} = \left[\sum_{j=1}^{c} (\|\boldsymbol{x}_k - \boldsymbol{v}_i\|_A / \|\boldsymbol{x}_k - \boldsymbol{v}_j\|_A)^{2/(m-1)} \right]^{-1} \qquad \forall i, k \tag{23}$$

$$\boldsymbol{v}_i = \left(\sum_{k=1}^{n} u_{ik}^m \boldsymbol{x}_k \Big/ \sum_{k=1}^{n} u_{ik}^m \right) \qquad \forall i \tag{24}$$

PCM: Minimize over $M_{pcn} \times \Re^{cp}$: $m > 1$: $w_i > 0\ \forall\ i$. $(U, \boldsymbol{V})$ may minimize J_m only if

$$u_{ik} = \left[1 + (\|\boldsymbol{x}_k - \boldsymbol{v}_i\|_A^2 / w_i)^{1/(m-1)} \right]^{-1} \qquad \forall i, k \tag{25}$$

$$\boldsymbol{v}_i = \left(\sum_{k=1}^{n} u_{ik}^m \boldsymbol{x}_k \Big/ \sum_{k=1}^{n} u_{ik}^m \right) \qquad \forall i \tag{26}$$

The HCM/FCM/PCM-AO Algorithms: Inner Product Norms Case

Store: Unlabeled Object Data $X = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\} \subset \Re^p$

Pick: Numbers of clusters: $1 < c < n$. Rule of thumb: Limit c to $c \le \sqrt{n}$
Maximum number of iterations: T
Weighting exponent: $1 \le m < \infty$ ($m = 1$ for HCM)
Norm for similarity of data to prototypes in J_m: $\langle \boldsymbol{x}, \boldsymbol{x} \rangle_A = \|\boldsymbol{x}\|_A^2 = \boldsymbol{x}^T A \boldsymbol{x}$
Norm for termination criterion: $E_t = \|\boldsymbol{V}_t - V_{t-1}\|_{err}$
Termination threshold: $0 < \epsilon$
Weights for penalty terms: $w_i > 0\ \forall\ i$ ($\boldsymbol{w} = \boldsymbol{0}$ for FCM/HCM)

Guess: Initial prototypes: $\boldsymbol{V}_0 = (\boldsymbol{v}_{10}, \ldots, \boldsymbol{v}_{c0}) \in \Re^{cp}$ {or initial partition $U_0 \in M_{pcn}$}

Iterate: For $t = 1$ to T: {reverse U and $\boldsymbol{V}$ if initializing with $U_0 \in M_{pcn}$}
Calculate U_t with $\boldsymbol{V}_{t-1}$ and (21, 23, or 25)
Update $\boldsymbol{V}_{t-1}$ to $\boldsymbol{V}_t$ with U_t and (22, 24, or 26)
If $E_t \le \epsilon$, exit for loop; Else
Next t
$(U, \boldsymbol{V}) = (U_t, \boldsymbol{V}_t)$

In theory, iterate sequences of these algorithms possess subsequences that converge to either local minima or saddle points of their objective functions (6). In practice they almost always terminate at useful solutions within a reasonable number of iterations. Justifying a choice of m in FCM or PCM is a challenge. FCM-AO will produce equimembership partitions that approach $\overline{U} = [1/c]$ as $m \to \infty$; but in practice, terminal partitions usually have memberships very close to $(1/c)$ for values of m not much larger than 20. At the other extreme, as m approaches 1 from above, FCM reduces to HCM, and terminal partitions become more and more crisp. Thus, m controls the degree of fuzziness exhibited by the soft boundaries in U. Most users choose m in the range [1.1, 5], with $m = 2$ an overwhelming favorite.

Example 2. Table 1 lists the coordinates of 20 two-dimensional points $X = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_{20}\}$. Figure 1(a) plots the data. HCM, FCM, and PCM were applied to X with the following protocols: The similarity and termination norms were both Euclidean; $c = p = 2$; $n = 20$; $\epsilon = 0.01$, $T = 50$, $m = 2$ for FCM and PCM; initialization for HCM and FCM was at the $\boldsymbol{V}_0$ shown below the columns labeled $\boldsymbol{U}_{10}$ and $\boldsymbol{U}_{20}$; initialization for PCM was the terminal $\boldsymbol{V}_f$ from FCM shown below the FCM columns labeled $\boldsymbol{U}_{1f}$ and $\boldsymbol{U}_{2f}$; and the weights for PCM were fixed at $w_1 = 0.15$, $w_2 = 0.16$.

All three algorithms terminated in less than 10 iterations at the partition matrices U_f (rows are shown transposed) and point prototypes $\boldsymbol{V}_f$ shown in the table. HCM and FCM began with the first 16 points in crisp cluster 1. HCM terminated with 10 points in each cluster as indicated by the boundaries in Figure 1(b). FCM and PCM terminated at the fuzzy and possibilistic partitions of X shown in Table 1. The difference between these two partitions can be seen, for example, by looking at the memberships of point $\boldsymbol{x}_7$ in both clusters (the values are underlined in Table 1). The fuzzy memberships are (0.96, 0.04), which sum to 1 as they must. This indicates that $\boldsymbol{x}_7$ is a very strong member of fuzzy cluster 1 and is barely related to cluster 2. The PCM values are (0.58, 0.06). These numbers indicate that $\boldsymbol{x}_7$ is a fairly typical member of cluster 1 (on a scale from 0 to 1), while it cannot be regarded as typical of cluster 2. When hardened with Eq. (19), the FCM and PCM partitions coincide with the HCM result; that is, $U_{HCM} = U_{FCM}^{\boldsymbol{H}} = U_{PCM}^{\boldsymbol{H}}$. This is hardly ever the case for data sets that do not have compact, well-separated clusters.

Data point $\boldsymbol{x}_{13}$, partially underlined in Table 1, is more or less in between the two clusters. Its memberships, the fuzziest ones in the FCM partition (0.41, 0.59), point to this anomaly. The possibilities (0.22, 0.31) in the PCM partition are also low and roughly equal, indicating that $\boldsymbol{x}_{13}$ is not typical of either cluster.

Finally, note that HCM estimates of the subsample means of the two groups ($\overline{\boldsymbol{v}}_2$ for points 11–20 in Table 1 and Fig. 1) are exact. The FCM estimates differ from the means by at most 0.07, and the PCM estimates differ from the means by at most 0.10. In this simple data set then, all three algorithms produce roughly the same results. The apples and pears in the first column of Table 1 and the point $\boldsymbol{z}$ in Fig. 1 are discussed in the next section.

FUZZY CLASSIFIER DESIGN

A *classifier* is any function $\boldsymbol{D}$: $\Re^p \mapsto N_{pc}$. The value $\boldsymbol{y} = \boldsymbol{D}(\boldsymbol{z})$ is the label vector for $\boldsymbol{z}$ in $\Re^p$. $\boldsymbol{D}$ is a *crisp classifier* if $\boldsymbol{D}[\Re^p] = N_{hc}$. Designing a classifier means the following: Use

Table 1. Example 2 Data, Initialization, and Terminal Outputs of HCM, FCM, and PCM

	X			Initialization		HCM		FCM		PCM	
$\boldsymbol{e}_i$	$\boldsymbol{x}_i$	x_1	x_2	$\boldsymbol{U}_{10}$	$\boldsymbol{U}_{20}$	$\boldsymbol{U}_{1f}$	$\boldsymbol{U}_{2f}$	$\boldsymbol{U}_{1f}$	$\boldsymbol{U}_{2f}$	$\boldsymbol{U}_{1f}$	$\boldsymbol{U}_{2f}$
🍎	1	1.00	0.60	1	0	1	0	0.97	0.03	0.70	0.07
🍎	2	1.75	0.40	1	0	1	0	0.77	0.23	0.35	0.16
🍎	3	1.30	0.10	1	0	1	0	0.96	0.04	0.49	0.07
🍎	4	0.80	0.20	1	0	1	0	0.94	0.06	0.36	0.05
🍎	5	1.10	0.70	1	0	1	0	0.95	0.05	0.72	0.08
🍎	6	1.30	0.60	1	0	1	0	0.97	0.03	0.90	0.10
🍎	7	0.90	0.50	1	0	1	0	<u>0.96</u>	<u>0.04</u>	<u>0.58</u>	<u>0.06</u>
🍎	8	1.60	0.60	1	0	1	0	0.84	0.16	0.51	0.15
🍎	9	1.40	0.15	1	0	1	0	0.95	0.05	0.51	0.08
🍎	10	1.00	0.10	1	0	1	0	0.95	0.05	0.42	0.05
🍐	11	2.00	0.70	1	0	0	1	0.33	0.67	0.19	0.34
🍐	12	2.00	1.10	1	0	0	1	0.19	0.81	0.14	0.43
🍐	13	1.90	0.80	1	0	0	1	<u>0.41</u>	<u>0.59</u>	<u>0.22</u>	<u>0.31</u>
🍐	14	2.20	0.80	1	0	0	1	0.10	0.90	0.13	0.59
🍐	15	2.30	1.20	1	0	0	1	0.04	0.96	0.08	0.75
🍐	16	2.50	1.15	1	0	0	1	0.01	0.99	0.07	0.90
🍐	17	2.70	1.00	0	1	0	1	0.01	0.99	0.06	0.73
🍐	18	2.90	1.10	0	1	0	1	0.05	0.95	0.05	0.45
🍐	19	2.80	0.90	0	1	0	1	0.03	0.97	0.05	0.56
🍐	20	3.00	1.05	0	1	0	1	0.06	0.94	0.04	0.36
		$\overline{\boldsymbol{v}}_1$	$\overline{\boldsymbol{v}}_2$	$\boldsymbol{v}_{10}$	$\boldsymbol{v}_{20}$	$\boldsymbol{v}_{1f}$	$\boldsymbol{v}_{2f}$	$\boldsymbol{v}_{1f}$	$\boldsymbol{v}_{2f}$	$\boldsymbol{v}_{1f}$	$\boldsymbol{v}_{2f}$
		1.22	2.43	1.57	2.85	1.22	2.43	1.21	2.50	1.23	2.45
		0.40	0.98	0.61	1.01	0.40	0.98	0.41	1.00	0.50	1.02

X to find a specific $\boldsymbol{D}$ from a specified family of functions (or algorithms). If the data are labeled, finding $\boldsymbol{D}$ is called *supervised learning.* Classifier models based on statistical, heuristic and network structures are discussed elsewhere in this Encyclopedia. This section describes some of the basic (and often most useful) classifier designs that have fuzzy generalizations.

The Nearest Prototype Classifier

Synonyms for the word *prototype* include vector quantizer, signature, template, codevector, paradigm, centroid, and exemplar. The common denominator in all prototype generation schemes is a mathematical definition of how well prototype $\boldsymbol{v}_i$ represents a set of vectors X_i. Any measure of similarity or dissimilarity on $\Re^p$ can be used; the usual choice is one of the distances at Eqs. (7)–(9) or (12)–(14).

Definition (1-np classifier). Given $(\boldsymbol{V}, \boldsymbol{E}) = \{(\boldsymbol{v}_i, \boldsymbol{e}_i): i = 1, \ldots, c\} \in \Re^{cp} \times N_{hc}^{c}$, c crisply labeled prototypes (one per class) and *any* distance measure δ on $\Re^p$. The *crisp nearest prototype* (1-np) classifier $\boldsymbol{D}_{\boldsymbol{V},\boldsymbol{E},\delta}$ is defined, for $\boldsymbol{z} \in \Re^p$, as

$$\text{Decide } \boldsymbol{z} \in \text{class } i \Leftrightarrow \boldsymbol{D}_{\boldsymbol{V},\boldsymbol{E},\delta}(\boldsymbol{z}) = \boldsymbol{e}_i \Leftrightarrow \delta(\boldsymbol{z}, \boldsymbol{v}_i) \le \delta(\boldsymbol{z}, \boldsymbol{v}_j) \quad \forall\, j \ne i \qquad (27)$$

Equation (27) says: Find the closest prototype to $\boldsymbol{z}$, and assign its label to $\boldsymbol{z}$. Ties are broken randomly. For example, the Euclidean distances from the point ❖ = $\boldsymbol{z} = (2, 0.5)^{\mathrm{T}}$ to the subsample means (shown as dashed lines in Fig. 1) are $\|\boldsymbol{z} - \overline{\boldsymbol{v}}_2\| = 0.64 < \|\boldsymbol{z} - \overline{\boldsymbol{v}}_1\| = 0.79$, so $\boldsymbol{z}$ acquires the label of $\overline{\boldsymbol{v}}_2$; that is, $\boldsymbol{z}$ is in class 2. If the first 10 points are class 1 = apples and the second 10 points are class 2 = pears as shown in column 1 of Table 1, then the crisp labels for the 20 data points are $\boldsymbol{e}_i = \boldsymbol{e}_1 = (1, 0)^T$, $i = 1, \ldots, 10$; $\boldsymbol{e}_i = \boldsymbol{e}_2 = (0, 1)^T$, $i = 11, \ldots, 20$, and $\boldsymbol{z}$ is declared a pear by $\boldsymbol{D}_{\boldsymbol{V},\boldsymbol{E},\delta_2}$.

The notation for $\boldsymbol{D}_{\boldsymbol{V},\boldsymbol{E},\delta}$ emphasizes that there are three ways to alter Eq. (27): We can change $\boldsymbol{V}$, $\boldsymbol{E}$, or δ. As the measure of distance δ changes with $\boldsymbol{V}$ and $\boldsymbol{E}$ fixed, it is possible that the label assigned by Eq. (27) will too. If we use the 1-norm distance at Eq. (12) instead of the 2-norm distance at Eq. (13), then $\|\boldsymbol{z} - \overline{\boldsymbol{v}}_1\|_1 = 0.89 < \|\boldsymbol{z} - \overline{\boldsymbol{v}}_2\|_1 = 0.91$, so the decision is reversed: $\boldsymbol{z}$ is in class 1 = apples. Finally, if we use Eq. (14), then $\|\boldsymbol{z} - \overline{\boldsymbol{v}}_1\|_\infty = 0.78 > \|\boldsymbol{z} - \overline{\boldsymbol{v}}_2\|_\infty = 0.48$, so the label for $\boldsymbol{z}$ with this distance reverts to class 2 = pears. This shows why it is important to choose the distance carefully and understand the effect of changing it when using $\boldsymbol{D}_{\boldsymbol{V},\boldsymbol{E},\delta}$.

Second, we can change the prototype set $\boldsymbol{V}$ while holding $\boldsymbol{E}$ and δ fixed. The crisp 1-np design can be implemented using prototypes from *any* algorithm that produces them. $\boldsymbol{D}_{\boldsymbol{V},\boldsymbol{E},\delta}$ is crisp because of $\boldsymbol{E}$, even if $\boldsymbol{V}$ comes from a fuzzy, probabilistic, or possibilistic algorithm. Table 1 shows four different sets of prototypes for the data: the sample means $\overline{\boldsymbol{v}}_1$ and $\overline{\boldsymbol{v}}_2$, which coincide with the HCM estimates, and the FCM and PCM prototypes. Repeating the calculations of the last paragraph with the FCM or PCM prototypes leads here to the same labels for $\boldsymbol{z}$ using the three distances in Eqs. (12)–(14) because the sets of prototypes are nearly equal. But generally, this is not the case.

Third, the crisp labels $\boldsymbol{E}$ can be softened while holding $\boldsymbol{V}$ and δ fixed. In this case a more sophisticated approach based on aggregation of the soft label information possessed by several close prototypes is needed. This is a special case of the classifier we turn to next.

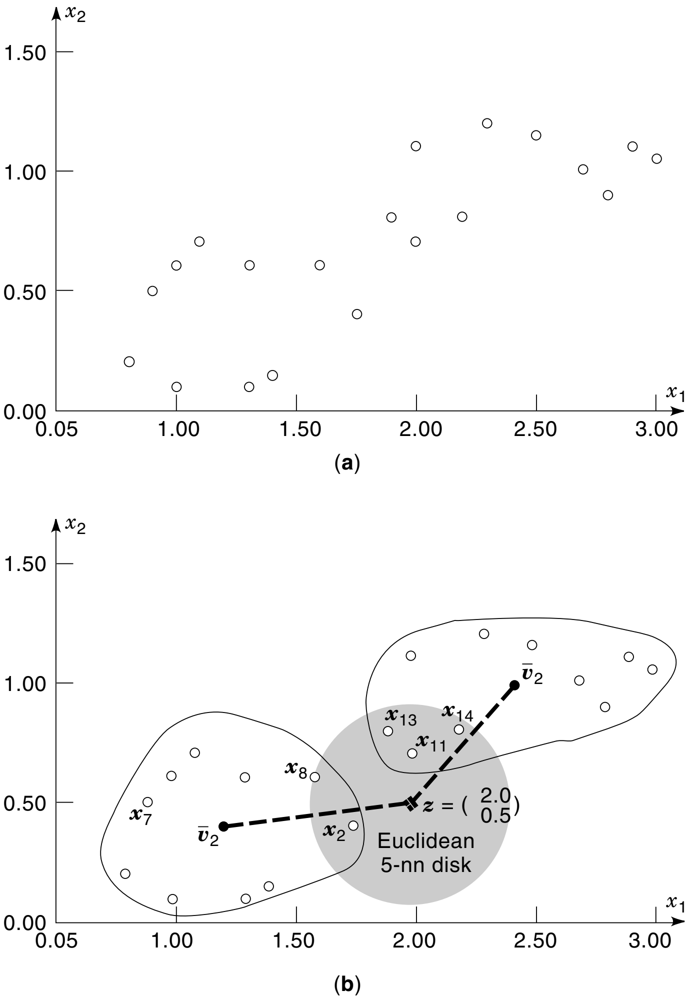

Figure 1. (a) The 20-point data set for Examples 2 and 3. (b) Clustering and classification results for Examples 2 and 3.

The Crisp *k*-Nearest Neighbor Classifier

Another widely used classifier with fuzzy and possibilistic generalizations is the k-Nearest Neighbor (k-nn) rule, which *requires* labeled samples from each class. As an example, the symbols in the first column of Table 1 enable each point in the data to serve as a labeled prototype. The crisp k-nn rule finds the k nearest neighbors (points in X) to $\boldsymbol{z}$, and then it aggregates the votes of the neighbors for each class. The majority vote determines the label for $\boldsymbol{z}$. Only two parameters must be selected to implement this rule: k, the *number* of nearest neighbors to $\boldsymbol{z}$; and δ, a measure of *nearness* (usually distance) between pairs of vectors in $\Re^p$.

Definition (k-nn Classifier). Given $(X, U) = \{(\boldsymbol{x}_k, \boldsymbol{U}_k)\colon k = 1, \ldots, n\} \in \Re^{np} \times N_{pc}^c$ and *any* distance measure δ on $\Re^p$. Let $\boldsymbol{z} \in \Re^p$ and let $\boldsymbol{U}_{(1)} \ldots \boldsymbol{U}_{(k)}$ denote the columns of U corresponding to the k nearest neighbors of $\boldsymbol{z}$. Aggregate votes (full or partial) for each class in the label vector $\boldsymbol{D}_{(X,U),k,\delta}(\boldsymbol{z}) = \sum_{j=1}^{k} \boldsymbol{U}_{(j)}/k$. The crisp k-nn classifier is defined as

$$\text{Decide } \boldsymbol{z} \in i \Leftrightarrow \boldsymbol{H}(\boldsymbol{D}_{(X,U),k,\delta}(\boldsymbol{z})) = \boldsymbol{e}_i \tag{28}$$

Example 3. Figure 1(b) shows a shaded disk with radius $\|\boldsymbol{x}_8 - \boldsymbol{z}\|_2 = 0.41$ centered at $\boldsymbol{z}$ which corresponds to the k = 5-nn rule with Euclidean distance for δ. The disk captures three neighbors—$\boldsymbol{x}_{11}$, $\boldsymbol{x}_{13}$, and $\boldsymbol{x}_{14}$—labeled pears in Table 1 and captures two neighbors—$\boldsymbol{x}_2$ and $\boldsymbol{x}_8$—labeled apples in Table 1. This 5-nn rule labels $\boldsymbol{z}$ a pear, realized by Eq. (28) as follows:

$$\begin{aligned}\boldsymbol{D}_{(X,U_{\text{crisp}}),5,\delta_2}(\boldsymbol{z}) &= \frac{\sum_{j=1}^{5} \boldsymbol{U}_{\text{crisp}(j)}}{5} \\ &= \frac{\begin{pmatrix}1\\0\end{pmatrix} + \begin{pmatrix}1\\0\end{pmatrix} + \begin{pmatrix}0\\1\end{pmatrix} + \begin{pmatrix}0\\1\end{pmatrix} + \begin{pmatrix}0\\1\end{pmatrix}}{5} \\ &= \begin{pmatrix}0.4\\0.6\end{pmatrix}\end{aligned} \tag{29}$$

$$\boldsymbol{H}(\boldsymbol{D}_{(X,U_{\text{crisp}}),5,\delta_2}(\boldsymbol{z})) = \boldsymbol{H}\begin{pmatrix}0.4\\0.6\end{pmatrix} = \begin{pmatrix}0\\1\end{pmatrix} = \boldsymbol{e}_2 \tag{30}$$

To see that k and δ affect the decision made by (28), Table 2 shows the labeling that (28) produces for $\boldsymbol{z}$ using k = 1 to 5 and the three distances shown in Eqs. (12)–(14) with U_{crisp}.

Distances from $\boldsymbol{z}$ to each of its five nearest neighbors are shown in the upper third of Table 2. The five nearest neighbors are ranked in the same order by all three distances, $\boldsymbol{x}_{(1)} = \boldsymbol{x}_{11}$ being closest to $\boldsymbol{z}$, and $\boldsymbol{x}_{(5)} = \boldsymbol{x}_8$ being furthest from $\boldsymbol{z}$, where $\boldsymbol{x}_{(k)}$ is the kth ranked nearest neighbor to $\boldsymbol{z}$. $L(\boldsymbol{x}_{(k)})$ is the crisp label for $\boldsymbol{x}_{(k)}$ from Table 1. The label sets—in order, left to right—that are used for each of the 15 decisions (3 distances by 5 rules) are shown in the middle third of Table 2. $L_q(\boldsymbol{z})$ in the lower third of Table 2 is the crisp label assigned to $\boldsymbol{z}$ by each k-nn rule for the q = 1, 2, and ∞ distances.

Whenever there is a tie, the label assigned to $\boldsymbol{z}$ is arbitrary. There are two kinds of ties: label ties and distance ties. The 1-nn rule labels $\boldsymbol{z}$ a pear with all three distances. All three rules yield a label tie using k = 2, so either label may be assigned to $\boldsymbol{z}$ by these three classifiers. For k = 3 the 1 and 2 norm distances label $\boldsymbol{z}$ a pear. The sup norm experiences a distance tie between $\boldsymbol{x}_{(3)}$ and $\boldsymbol{x}_{(4)}$ at k = 3, but both points are labeled *pear* so the decision is still pear regardless of how the tie is resolved. At k = 4 the 1 norm has a distance tie between $\boldsymbol{x}_{(4)}$ and $\boldsymbol{x}_{(5)}$. Since these two points have different labels, the output of this classifier will depend on which point is selected to break the distance tie. If the apple is selected, resolution of the distance tie results in a label tie, and a second tie must be broken. If the distance tie breaker results in the pear, there are three pears and one apple as in the other two cases at k = 4. And finally, for k = 5 all three classifiers agree that $\boldsymbol{z}$ is a pear. Table 2 illustrates that the label assigned by (28) is dependent on both k and δ.

Equation (28) is well-defined for fuzzy and possibilistic labels. If, for example, we use the FCM labels from Table 1 for the five nearest neighbors to $\boldsymbol{z}$ instead of the crisp labels used in Example 3, we have

$$\begin{aligned}&\boldsymbol{D}_{(X,U_{FCM}),5,\delta_2}(\boldsymbol{z}) \\ &\quad = \frac{\sum_{j=1}^{5} \boldsymbol{U}_{FCM(j)}}{5} \\ &\quad = \frac{\begin{pmatrix}0.33\\0.67\end{pmatrix} + \begin{pmatrix}0.77\\0.23\end{pmatrix} + \begin{pmatrix}0.41\\0.59\end{pmatrix} + \begin{pmatrix}0.10\\0.90\end{pmatrix} + \begin{pmatrix}0.84\\0.16\end{pmatrix}}{5} \\ &\quad = \begin{pmatrix}0.49\\0.51\end{pmatrix}\end{aligned} \tag{31}$$

Table 2. The k-nn Rule Labels $\boldsymbol{z}$ for Three Distances and Five Sets of Neighbors

Distances from $\boldsymbol{z}$ to the Ranked Neighbors

k	$\boldsymbol{x}_{(k)}$	$L(\boldsymbol{x}_{(k)})$	$\delta_1(\boldsymbol{z}, \boldsymbol{x}_{(k)})$	$\delta_2(\boldsymbol{z}, \boldsymbol{x}_{(k)})$	$\delta_\infty(\boldsymbol{z}, \boldsymbol{x}_{(k)})$
1	$\boldsymbol{x}_{11}$	🍐	0.20	0.20	0.20
2	$\boldsymbol{x}_{2}$	🍎	0.35	0.27	0.25
3	$\boldsymbol{x}_{13}$	🍐	0.40	0.32	0.30
4	$\boldsymbol{x}_{14}$	🍐	0.50	0.36	0.30
5	$\boldsymbol{x}_{8}$	🍎	0.50	0.41	0.40

Labels of the Ranked Neighbors

k	*Ranked Neighbors*	δ_1	δ_2	δ_∞
1	$\boldsymbol{x}_{11}$	🍐	🍐	🍐
2	$\boldsymbol{x}_{11}, \boldsymbol{x}_{2}$	🍐🍎 = Label tie	🍐🍎 = Label tie	🍐🍎 = Label tie
3	$\boldsymbol{x}_{11}, \boldsymbol{x}_{2}, \boldsymbol{x}_{13}$	🍐🍎🍐	🍐🍎🍐	🍐🍎🍐🍐 = δ tie
4	$\boldsymbol{x}_{11}, \boldsymbol{x}_{2}, \boldsymbol{x}_{13}, \boldsymbol{x}_{14}$	🍐🍎🍐🍐🍎 = δ tie	🍐🍎🍐🍐	🍐🍎🍐🍐
5	$\boldsymbol{x}_{11}, \boldsymbol{x}_{2}, \boldsymbol{x}_{13}, \boldsymbol{x}_{14}, \boldsymbol{x}_{8}$	🍐🍎🍐🍐🍎	🍐🍎🍐🍐🍎	🍐🍎🍐🍐🍎

Output Label for $\boldsymbol{z}$

k	*Ranked Neighbors*	$L_1(\boldsymbol{z})$	$L_2(\boldsymbol{z})$	$L_\infty(\boldsymbol{z})$
1	$\boldsymbol{x}_{11}$	🍐	🍐	🍐
2	$\boldsymbol{x}_{11}, \boldsymbol{x}_{2}$	Label tie	Label tie	Label tie
3	$\boldsymbol{x}_{11}, \boldsymbol{x}_{2}, \boldsymbol{x}_{13}$	🍐	🍐	🍐
4	$\boldsymbol{x}_{11}, \boldsymbol{x}_{2}, \boldsymbol{x}_{13}, \boldsymbol{x}_{14}$	Label − δ tie	🍐	🍐
5	$\boldsymbol{x}_{11}, \boldsymbol{x}_{2}, \boldsymbol{x}_{13}, \boldsymbol{x}_{14}, \boldsymbol{x}_{8}$	🍐	🍐	🍐

$$\boldsymbol{H}(\boldsymbol{D}_{(X,U),5,\delta_2}(\boldsymbol{z})) = \boldsymbol{H}\begin{pmatrix}0.49\\0.51\end{pmatrix} = \begin{pmatrix}0\\1\end{pmatrix} = \boldsymbol{e}_2 \Rightarrow \boldsymbol{z} = \text{pear} \tag{32}$$

Equation (32) is a crisp decision based on fuzzy labels, so it is still a crisp k-nn rule. Possibilistic labels for these five points from Table 1 would result in the same decision here, but this is not always the case. If all 20 sets of FCM and PCM memberships from Table 1 are used in Eq. (28), the 20-nn rules based on the HCM, FCM, and PCM columns in Table 1 yield

$$\boldsymbol{H}\left[\boldsymbol{D}_{(X,U_{\text{HCM}}),20,\delta_2}(\boldsymbol{z}) = \frac{\sum_{j=1}^{20}\boldsymbol{U}_{\text{HCM}(j)}}{20}\right] = \boldsymbol{H}\begin{pmatrix}0.50\\0.50\end{pmatrix} \Rightarrow \text{tie} \tag{33}$$

$$\boldsymbol{H}\left[\boldsymbol{D}_{(X,U_{\text{FCM}}),20,\delta_2}(\boldsymbol{z}) = \frac{\sum_{j=1}^{20}\boldsymbol{U}_{\text{FCM}(j)}}{20}\right] = \boldsymbol{H}\begin{pmatrix}0.52\\0.48\end{pmatrix} = \begin{pmatrix}1\\0\end{pmatrix} = \boldsymbol{e}_1 \Rightarrow \boldsymbol{z} = \text{apple} \tag{34}$$

$$\boldsymbol{H}\left[\boldsymbol{D}_{(X,U_{\text{PCM}}),20,\delta_2}(\boldsymbol{z}) = \frac{\sum_{j=1}^{20}\boldsymbol{U}_{\text{PCM}(j)}}{20}\right] = \boldsymbol{H}\begin{pmatrix}0.33\\0.31\end{pmatrix} = \begin{pmatrix}1\\0\end{pmatrix} = \boldsymbol{e}_1 \Rightarrow \boldsymbol{z} = \text{apple} \tag{35}$$

Terminology. The output of Eq. (31) and the argument of $\boldsymbol{H}$ in Eq. (34) are fuzzy labels based on fuzzy labels. Even though the final outputs are crisp in these two equations, some writers refer to the overall crisp decision as the fuzzy k-nn rule. More properly, however, the fuzzy k-nn rule is the algorithm that produces the fuzzy label which is subsequently hardened in Eq. (28). Similarly, the input or argument of $\boldsymbol{H}$ in (35) is properly regarded as the output of the possibilistic k-nn rule, but some authors prefer to call the output of Eq. (35) the possibilistic k-nn rule. The important point is that if all 20 labels are used, the rule based on crisp labels is ambiguous, while the fuzzy and possibilistic based rules both label $\boldsymbol{z}$ an apple. This shows that the type of label also impacts the decision made by Eq. (28).

The Crisp, (Fuzzy and Possibilistic) k-nn Algorithms

Problem:	To label $\boldsymbol{z}$ in $\Re^p$
Store:	Labeled object data $X = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\} \subset \Re^p$ and label matrix $U \in N_{pc}^n$
Pick:	k = number of nn's and $\delta: \Re^p \times \Re^p \mapsto \Re^+$ = any metric on $\Re^p$
Find:	The n distances $\{\delta_j \equiv \delta(\boldsymbol{z}, \boldsymbol{x}_j): j = 1, 2, \ldots, n\}$
Rank:	$\underbrace{\delta_{(1)} \le \delta_{(2)} \le \cdots \le \delta_{(k)}}_{k-\text{nn } indices} \le \delta_{(k+1)} \le \ldots \le \delta_{(n)}$
Compute:	$\boldsymbol{D}_{(X,U),k,\delta}(\boldsymbol{z}) = \sum_{j=1}^{k}\boldsymbol{U}_{(j)} \Big/ k$
Do:	Decide $\boldsymbol{z} \in i \Leftrightarrow \boldsymbol{H}(\boldsymbol{D}_{(X,U),k,\delta}(\boldsymbol{z})) = \boldsymbol{e}_i$

FEATURE ANALYSIS

Methods that explore and improve raw data are broadly characterized as *feature analysis.* This includes scaling, normalization, filtering, and smoothing. Any transformation

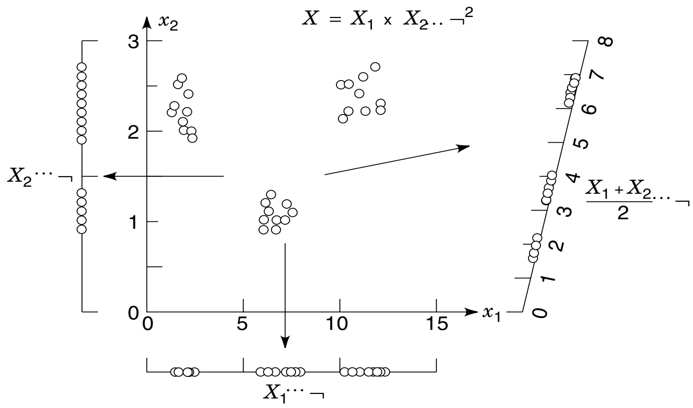

Figure 2. Feature selection and extraction on a 30 point data set.

$\Phi: \Re^p \mapsto \Re^q$ does feature extraction when applied to X. Usually $q \ll p$, but there are cases where $q \geq p$ too. Examples of feature extraction transformations include Fourier transforms, principal components, and features such as the digital gradient, mean, range, and standard deviation from intensities in image windows.

Feature *selection* consists of choosing subsets of the original measured features. Here Φ projects X onto a coordinate subspace of $\Re^p$. The goals of extraction and selection are as follows: to improve the data for solving a particular problem; to compress feature space to reduce time and space complexity; and to eliminate redundant (dependent) and unimportant (for the problem at hand) features.

Example 4. The center of Fig. 2 is a scatterplot of 30 two-dimensional points $X = \{(x_1, x_2)\}$ whose coordinates are listed in Table 3. The data are indexed so that points 1–10, 11–20, and 21–30 correspond to the three visually apparent clusters. Projection of X onto the first and second coordinate axes results in the one-dimensional data sets X_1 and X_2; this illustrates *feature selection.* The one-dimensional data set $(X_1 + X_2)/2$ in Fig. 2 (plotted to the right of X, not to scale) is made by averaging the coordinates of each vector in X. Geometrically, this amounts to orthogonal projection of X onto the line $x_1 = x_2$; this illustrates *feature extraction.*

Visual inspection should convince you that the three clusters seen in X, X_1 and $(X_1 + X_2)/2$ will be properly detected by most clustering algorithms. Projection of X onto its second axis, however, mixes the data and results in just two clusters in X_2. This suggests that projections of high-dimensional data into visual dimensions cannot be relied upon to show much about cluster structure in the original data.

The results of applying FCM to these four data sets with $c = 3$, $m = 2$, $\epsilon = 0.01$, and the Euclidean norm for both termination and J_m are shown in Table 3, which also shows

Table 3. Terminal FCM Partitions (Cluster 1 Only) for the Data Sets in Example 4

				Initialization			X	X_1	$(X_1 + X_2)/2$	X_2
	x_1	x_2	$(x_1 + x_2)/2$	$\boldsymbol{U}_{10}$	$\boldsymbol{U}_{20}$	$\boldsymbol{U}_{30}$	$\boldsymbol{U}_1$	$\boldsymbol{U}_1$	$\boldsymbol{U}_1$	$\boldsymbol{U}_1$
$\boldsymbol{x}_1$	1.5	2.5	2	1	0	0	0.99	1.00	1.00	0.00
$\boldsymbol{x}_2$	1.7	2.6	2.15	0	1	0	0.99	1.00	0.99	0.03
$\boldsymbol{x}_3$	1.2	2.2	1.7	0	0	1	0.99	0.99	0.98	0.96
$\boldsymbol{x}_4$	1.8	2	1.9	1	0	0	1.00	1.00	1.00	0.92
$\boldsymbol{x}_5$	1.7	2.1	1.9	0	1	0	1.00	1.00	1.00	0.99
$\boldsymbol{x}_6$	1.3	2.3	1.8	0	0	1	0.99	0.99	0.99	0.63
$\boldsymbol{x}_7$	2.1	2	2.05	1	0	0	0.99	0.99	1.00	0.92
$\boldsymbol{x}_8$	2.3	1.9	2.1	0	1	0	0.97	0.98	1.00	0.82
$\boldsymbol{x}_9$	2	2.4	2.2	0	0	1	0.99	1.00	0.98	0.17
$\boldsymbol{x}_{10}$	1.9	2.2	2.05	1	0	0	1.00	1.00	1.00	0.96
$\boldsymbol{x}_{11}$	6	1.2	3.6	0	1	0	0.01	0.01	0.01	0.02
$\boldsymbol{x}_{12}$	6.6	1	3.8	0	0	1	0.00	0.00	0.00	0.00
$\boldsymbol{x}_{13}$	5.9	0.9	3.4	1	0	0	0.02	0.02	0.07	0.02
$\boldsymbol{x}_{14}$	6.3	1.3	3.8	0	1	0	0.00	0.00	0.00	0.07
$\boldsymbol{x}_{15}$	5.9	1	3.45	0	0	1	0.02	0.02	0.05	0.00
$\boldsymbol{x}_{16}$	7.1	1	4.05	1	0	0	0.01	0.01	0.02	0.00
$\boldsymbol{x}_{17}$	6.5	0.9	3.7	0	1	0	0.00	0.00	0.00	0.02
$\boldsymbol{x}_{18}$	6.2	1.1	3.65	0	0	1	0.00	0.00	0.01	0.00
$\boldsymbol{x}_{19}$	7.2	1.2	4.2	1	0	0	0.02	0.02	0.03	0.02
$\boldsymbol{x}_{20}$	7.5	1.1	4.3	0	1	0	0.03	0.03	0.04	0.00
$\boldsymbol{x}_{21}$	10.1	2.5	6.3	0	0	1	0.01	0.01	0.01	0.00
$\boldsymbol{x}_{22}$	11.2	2.6	6.9	1	0	0	0.00	0.00	0.00	0.03
$\boldsymbol{x}_{23}$	10.5	2.5	6.5	0	1	0	0.01	0.01	0.00	0.00
$\boldsymbol{x}_{24}$	12.2	2.3	7.25	0	0	1	0.01	0.01	0.01	0.63
$\boldsymbol{x}_{25}$	10.5	2.2	6.35	1	0	0	0.01	0.01	0.01	0.96
$\boldsymbol{x}_{26}$	11	2.4	6.7	0	1	0	0.00	0.00	0.00	0.17
$\boldsymbol{x}_{27}$	12.2	2.2	7.2	0	0	1	0.01	0.01	0.01	0.96
$\boldsymbol{x}_{28}$	10.2	2.1	6.15	1	0	0	0.01	0.01	0.02	0.99
$\boldsymbol{x}_{29}$	11.9	2.7	7.3	0	1	0	0.01	0.01	0.01	0.09
$\boldsymbol{x}_{30}$	11.5	2.2	6.85	0	0	1	0.00	0.00	0.00	0.96

the initialization used. Only memberships in the first cluster are shown. As expected, FCM discovers three very distinct fuzzy clusters in X, X_1, and $(X_1 + X_2)/2$. Table 3 shows the three clusters blocked into their visually apparent subsets of 10 points each. For X, X_1, and $(X_1 + X_2)/2$, all memberships for the first 10 points are ≥ 0.97, and memberships of the remaining 20 points in this cluster are ≤ 0.07. For X_2, however, this cluster has eight anomalies with respect to the original data. When column $\boldsymbol{U}_1$ of X_2 is hardened, this cluster contains the 12 points (underlined in Table 3) numbered 3, 4, 5, 6, 7, 8, 10, 24, 25, 27, 28, and 30; the last five of these belong to cluster 3 in X, and the points numbered 1, 2, and 9 should belong to this cluster, but do not.

REMARKS ON APPLICATIONS OF FUZZY PATTERN RECOGNITION

Retrieval from the *Science Citation Index* for years 1994–1997 on titles and abstracts that contain the keyword combinations "fuzzy" and either "clustering" or "classification" yield 460 papers. Retrievals against "fuzzy" and either "feature selection" or "feature extraction" yield 21 papers. This illustrates that the literature contains some examples of fuzzy models for feature analysis, but they are widely scattered because this discipline is very data-dependent and, hence, almost always done on a case-by-case basis.

A more interesting metric for the importance of fuzzy models in pattern recognition lies in the diversity of applications areas represented by the titles retrieved. Here is a partial sketch:

Chemistry. Analytical, computational, industrial, chromatography, food engineering, brewing science.

Electrical Engineering. Image and signal processing, neural networks, control systems, informatics, automation, robotics, remote sensing and control, optical engineering, computer vision, parallel computing, networking, dielectrics, instrumentation and measurement, speech recognition, solid-state circuits.

Geology / Geography. Photogrammetry, geophysical research, geochemistry, biogeography, archeology.

Medicine. Magnetic resonance imaging, diagnosis, tomography, roentgenology, neurology, pharmacology, medical physics, nutrition, dietetic sciences, anesthesia, ultramicroscopy, biomedicine, protein science, neuroimaging, pharmocology, drug interaction.

Physics. Astronomy, applied optics, earth physics.

Environmental Sciences. Soils, forest and air pollution, meteorology, water resources.

Thus, it seems fair to assert that this branch of science and engineering has established a niche as a useful way to approach pattern recognition problems.

BIBLIOGRAPHY

1. L. A. Zadeh, Fuzzy sets, *Inf. Control,* **8**: 338–352, 1965.
2. L. A. Zadeh, Probability measures of fuzzy events, *J. Math. Anal. Appl.,* **23**: 421–427, 1968.
3. R. Duda and P. Hart, *Pattern Classification and Scene Analysis,* New York: Wiley-Interscience, 1973.
4. K. S. Fu, *Syntactic pattern recognition and applications,* Englewood Cliffs, NJ: Prentice Hall, 1982.
5. R. E. Bellman, R. Kalaba, and L. A. Zadeh, Abstraction and pattern classification, *J. Math. Anal. Appl.,* **13**: 1–7, 1966.
6. J. C. Bezdek and S. K. Pal, *Fuzzy Models for Pattern Recognition,* Piscataway, NJ: IEEE Press, 1992.
7. R. Krishnapuram and J. Keller, A possibilistic approach to clustering, *IEEE Trans. Fuzzy Syst.,* **1** (2), 98–110, 1993.
8. Y. H. Pao, *Adaptive Pattern Recognition and Neural Networks,* Reading, MA: Addison-Wesley, 1989.
9. A. Jain and R. Dubes, *Algorithms for Clustering Data,* Englewood Cliffs, NJ: Prentice Hall, 1988.
10. J. C. Bezdek, *Pattern Recognition with Fuzzy Objective Function Algorithms,* New York: Plenum, 1981.

Reading List

G. Klir and T. Folger, *Fuzzy Sets, Uncertainty and Information,* Englewood Cliffs, NJ: Prentice Hall, 1988.

D. Dubois and H. Prade, *Fuzzy Sets and Systems: Theory and Applications,* New York: Academic Press, 1980.

H. J. Zimmermann, *Fuzzy Set Theory—and Its Applications,* 2nd ed., Boston: Kluwer, 1990.

D. Schwartz, G. Klir, H. W. Lewis, and Y. Ezawa, Applications of fuzzy sets and approximate reasoning, *Proc. IEEE,* **82**: 482–498, 1994.

A. Kandel, *Fuzzy Techniques in Pattern Recognition,* New York: Wiley-Interscience, 1982.

S. K. Pal and D. K. Dutta Majumder, *Fuzzy Mathematical Approach to Pattern Recognition,* New York: Wiley, 1986.

B. Kosko, *Neural Networks and Fuzzy Systems: A Dynamical Approach to Machine Intelligence,* Englewood Cliffs, NJ: Prentice Hall, 1991.

Journals

IEEE Transactions on Fuzzy Systems, IEEE Transactions on Neural Networks, IEEE Transactions on Systems, Man Cybernetics, Fuzzy Sets and Systems, International Journal of Approximate Reasoning, International Journal of Intelligent Systems, Intelligent Automation and Soft Computing, Uncertainty, Fuzziness and Knowledge Based Systems, Journal of Intelligent and Fuzzy Systems

JAMES C. BEZDEK
University of West Florida
LUDMILA KUNCHEVA
University of Wales, Bangor

FUZZY QUERYING. See FUZZY INFORMATION RETRIEVAL AND DATABASES.