COURSE

"APLICAÇÕES DE INTELIGÊNCIA ARTIFICIAL"

University of Aveiro

Mestrado em Eng. Computational

EXTRA INFO - this label will appear in slides with complementary information

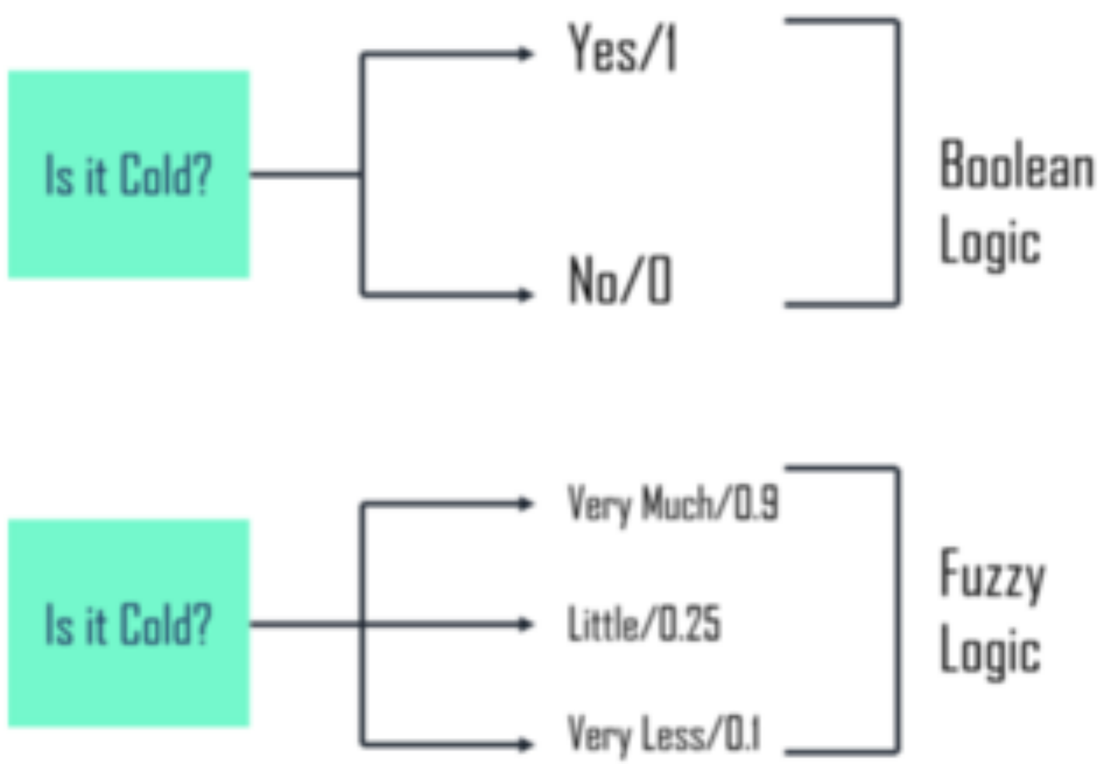
SUMMARY: About Neuro-Fuzzy:

[T] Fuzzy set theory; Algorithms for learning fuzzy model inference; Fuzzy clustering and Neuro-Fuzzy.

[P] Fuzzy min-max classifiers; Fuzzy tree classification; Exploring scikit-fuzzy and fylearn.

EXTRA INFO - this label will appear in slides with complementary information

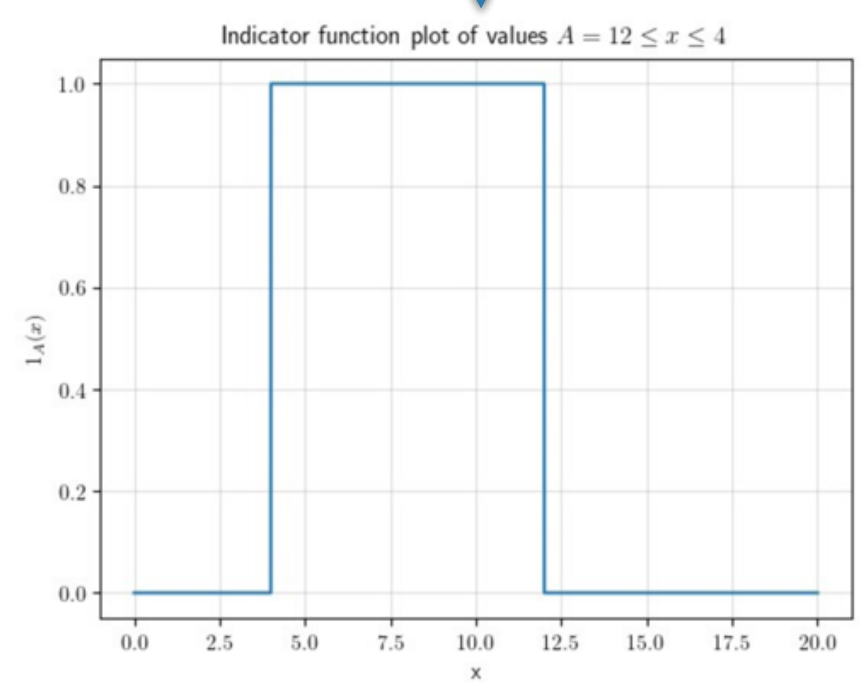
Fuzzy Logic (FL) is a method of reasoning that resembles **human reasoning**. This approach is similar to how humans perform decision making. And it involves all intermediate possibilities between **YES** and **NO**.



Fuzzy Logic (FL) is a method of reasoning that resembles **human reasoning**. This approach is similar to how humans perform decision making. And it involves all intermediate possibilities between **YES** and **NO**.

A ={i | i is an integer and 4<= i <= 12}

$$1_A(x) = \begin{cases} 1 & \text{if } 4 \leq x \leq 12 \\ 0, & \text{otherwise} \end{cases}$$



Fuzziness vs. randomness

"Hold an apple in your hand. Is it an apple? Yes. The object in your hand belongs to the clumps of space-time we call the set of apples — all apples anywhere, ever. Now take a bite, chew it, swallow it. Let your digestive tract take apart the apple's molecules. Is the object in your hand still an apple? Yes or no? Take another bite. Is the new object still an apple? Take another bite, and so on down to void." (Kosko, 1992, p. 4)

Fuzzy sets

that belongs to a finite universe of discourse:

$$A \subseteq \{x_1, x_2, \ldots, x_n\} = X$$

We have the following definitions for two fuzzy sets (A, μ_A) and (B, μ_B), where $A, B \subseteq X$:

- EQUALITY: $A = B$ iff $\mu_A(x) = \mu_B(x)$ for all $x \in X$
- INCLUSION: $A \subseteq B$ iff $\mu_A(x) \leq \mu_B(x)$ for all $x \in X$
- CARDINALITY: $|A| = \displaystyle\sum_{i=1}^{n} \mu_A(x_i)$
- EMPTY SET: A is empty iff $\mu_A(x) = 0$ for all $x \in X$.
- α-CUT: Given $\alpha \in [0, 1]$, the α-cut of A is defined by $A_\alpha = \{x \in X \mid \mu_A(x) \geq \alpha\}$

Operations on Fuzzy Sets

Let $(A, \mu_A), (B, \mu_B)$ be a fuzzy sets.

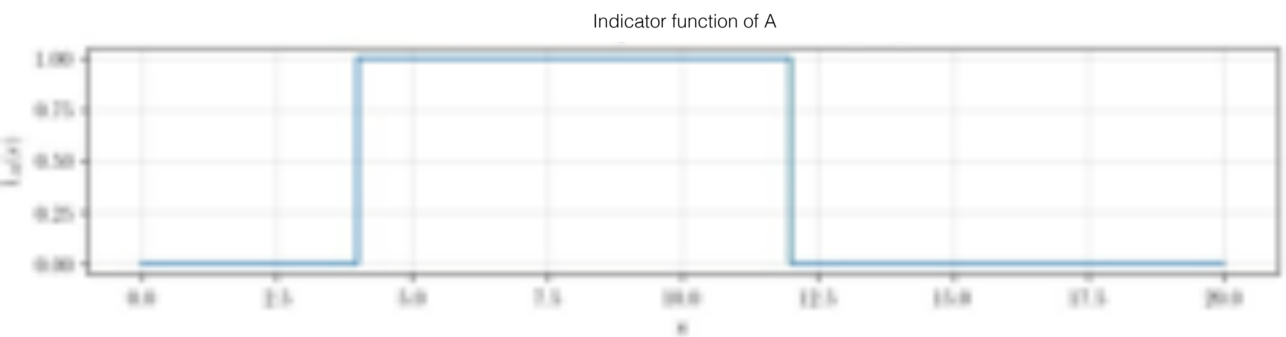
- <u>Complementation</u>: $(\neg A, \mu_{\neg A})$, where $\mu_{\neg A} = 1 - \mu_A$
- <u>Height</u>: $h(A) = \max_{x \in X} \mu_A(x)$
- <u>Support</u>: $\mathrm{supp}(A) = \{x \in X \mid \mu_A(x) > 0\}$
- <u>Core</u>: $C(A) = \{x \in X \mid \mu_A(x) = 1\}$
- <u>Intersection</u>: $C = A \cap B$, where $\mu_C = \min_{x \in X}\{\mu_A, \mu_B\}$
- <u>Union</u>: $C = A \cup B$, where $\mu_C = \max_{x \in X}\{\mu_A, \mu_B\}$
- <u>Bounded Sum</u>: $C = A + B$, where $\mu_C(x) = \min\{1, \mu_A(x) + \mu_B(x)\}$
- <u>Bounded Difference</u>: $C = A - B$, where $\mu_C(x) = \max\{0, \mu_A(x) - \mu_B(x)\}$

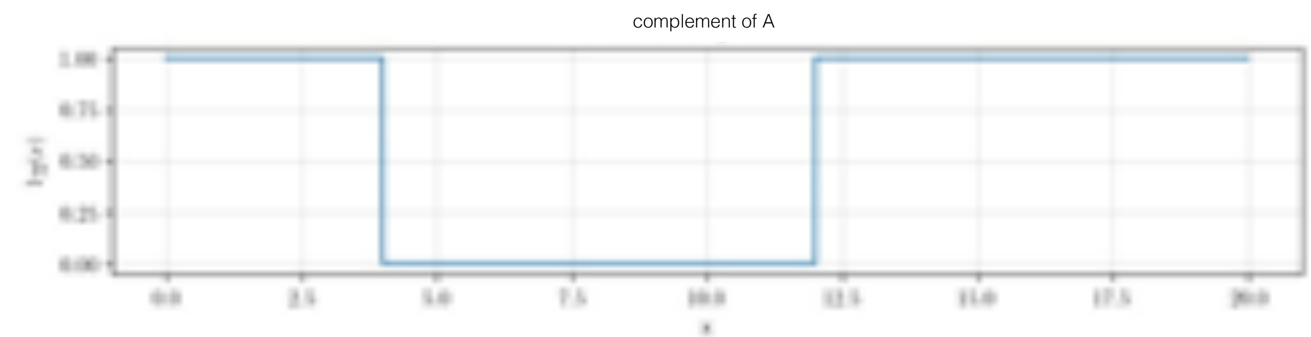
- <u>Exponentiation</u>: $C = A^\alpha$ where $\mu_C = (\mu_A)^\alpha$ for $\alpha > 0$
- <u>Level Set</u>: $C = \alpha A$ where $\mu_C = \alpha\mu_A$ for $\alpha \in [0, 1]$
- <u>Concentration</u>: $C = A^\alpha$ where $\alpha > 1$
- <u>Dilation</u>: $C = A^\alpha$ where $\alpha < 1$

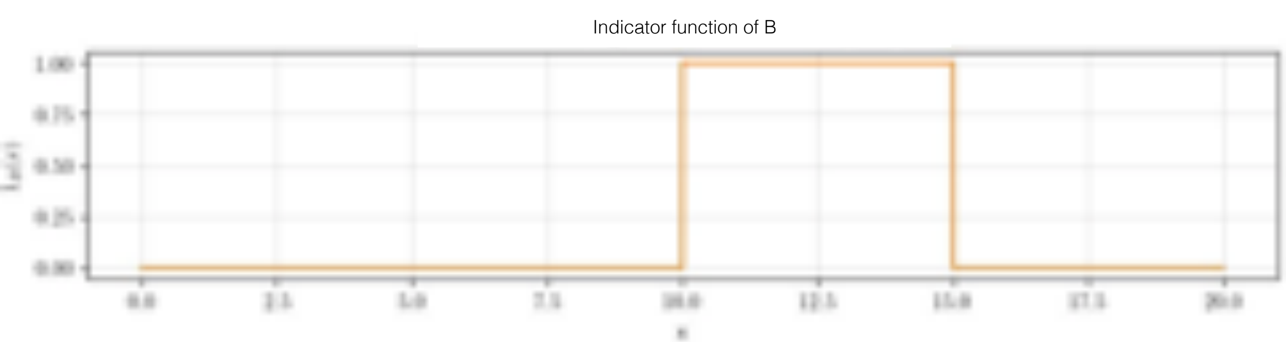
Note that $A \cap \neg A$ is not necessarily the empty set, as would be the case with classical set theory. Also, if A is crisp, then $A^\alpha = A$ for all α. We will define the Cartesian product $A \times B$ to be the same as $A \cap B$.

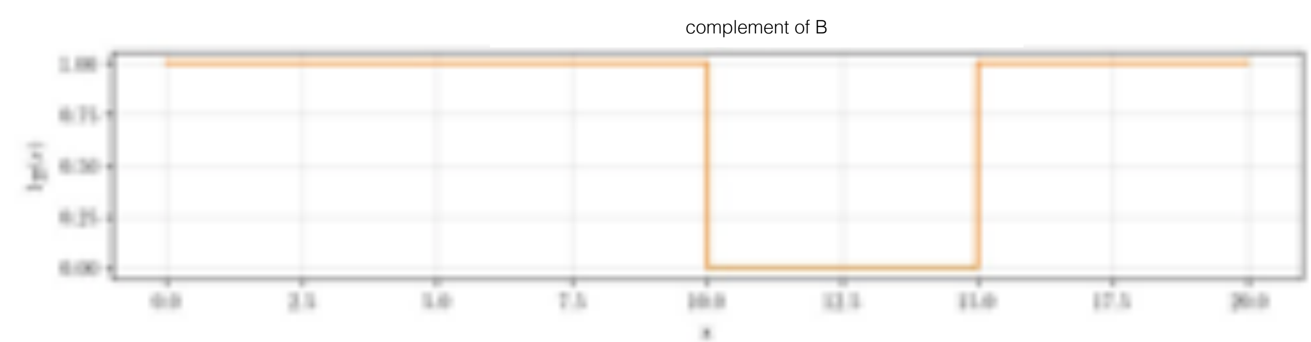
Some operations

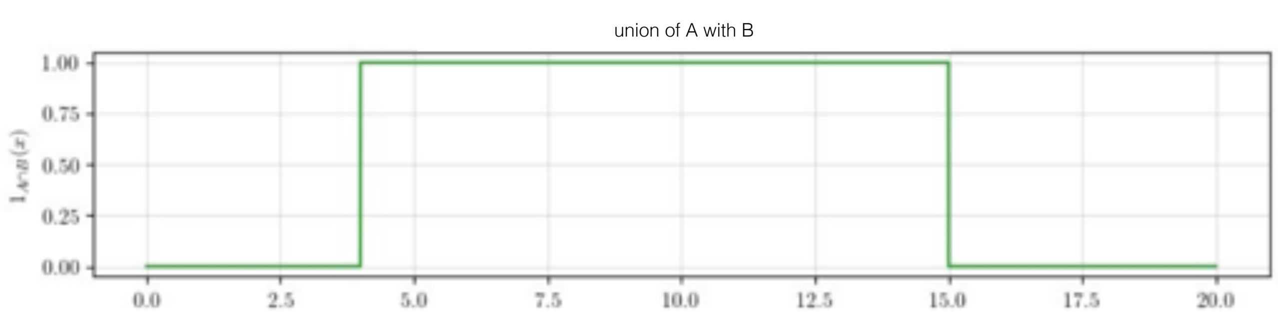
A ={i | i is an integer and 4<= i <= 12}

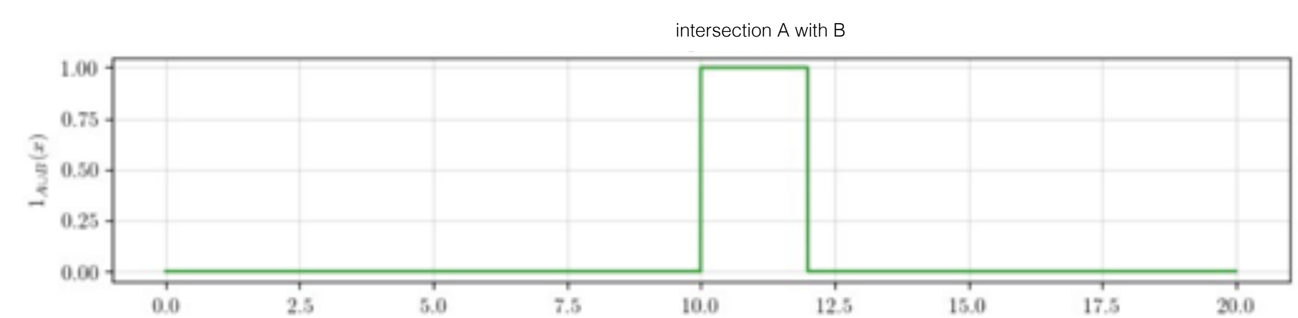












Usual membership functions

We will usually consider one of the following membership functions:

- TRIANGULAR: $\mathrm{tri}(x; a, b, c) = \max\left\{\min\left\{\frac{x-a}{b-a}, \frac{c-x}{c-b}\right\}, 0\right\}$
- TRAPEZOIDAL: $\mathrm{trap}(x; a, b, c, d) = \max\left\{\min\left\{\frac{x-a}{b-a}, \frac{d-x}{d-c}, 1\right\}, 0\right\}$
- GAUSSIAN: $\mathrm{gauss}(x; c, \sigma) = \exp\left[-\frac{1}{2}\left(\frac{x-c}{\sigma}\right)^2\right]$
- GENERALISED BELL: $\mathrm{gbell}(x; a, b) = \dfrac{1}{1 + \left|\frac{x-b}{a}\right|^{2a}}$

Applications

The Fuzzy logic works on the levels of possibilities of input to achieve a definite output. Now, talking about the implementation of this logic:

- It can be implemented in systems with different sizes and capabilities such as **micro-controllers, large networked**, or **workstation-based systems.**
- Also, it can be implemented in **hardware, software**, or a combination of **both**.

Applications

The Fuzzy logic works on the levels of possibilities of input to achieve a definite output. Now, talking about the implementation of this logic:

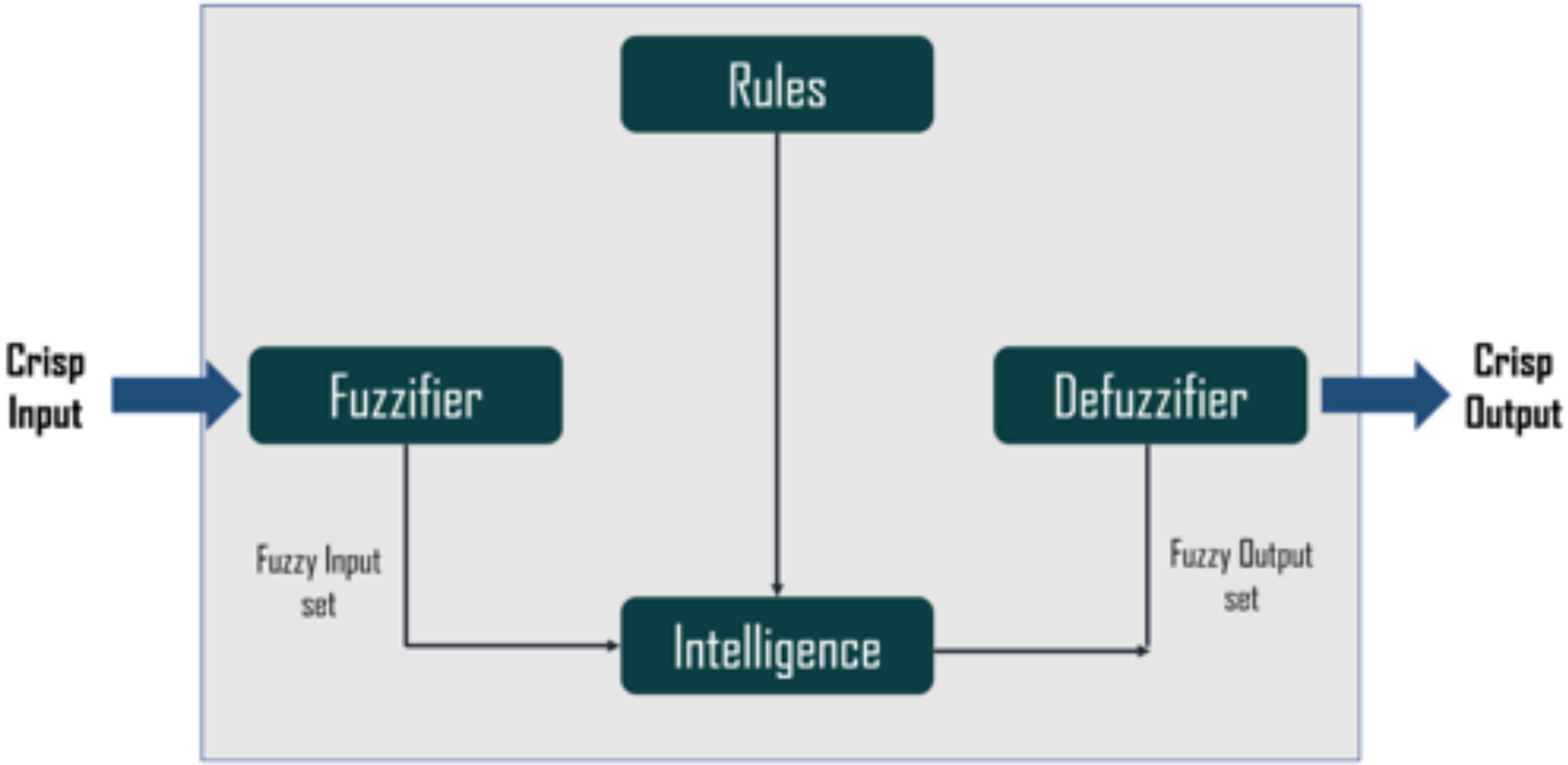
- It can be implemented in systems with different sizes and capabilities such as **micro-controllers, large networked**, or **workstation-based systems.**
- Also, it can be implemented in **hardware, software,** or a combination of **both**.

Generally, we use the fuzzy logic system for both commercial and practical purposes such as:

- It **controls machines** and **consumer products**
- If not accurate reasoning, it at least provides **acceptable reasoning**
- This helps in dealing with the **uncertainty in engineering**

So, now that you know about Fuzzy logic in AI and why do we use it, let's move on and understand the architecture of this logic.

Fuzzy Logic Architecture



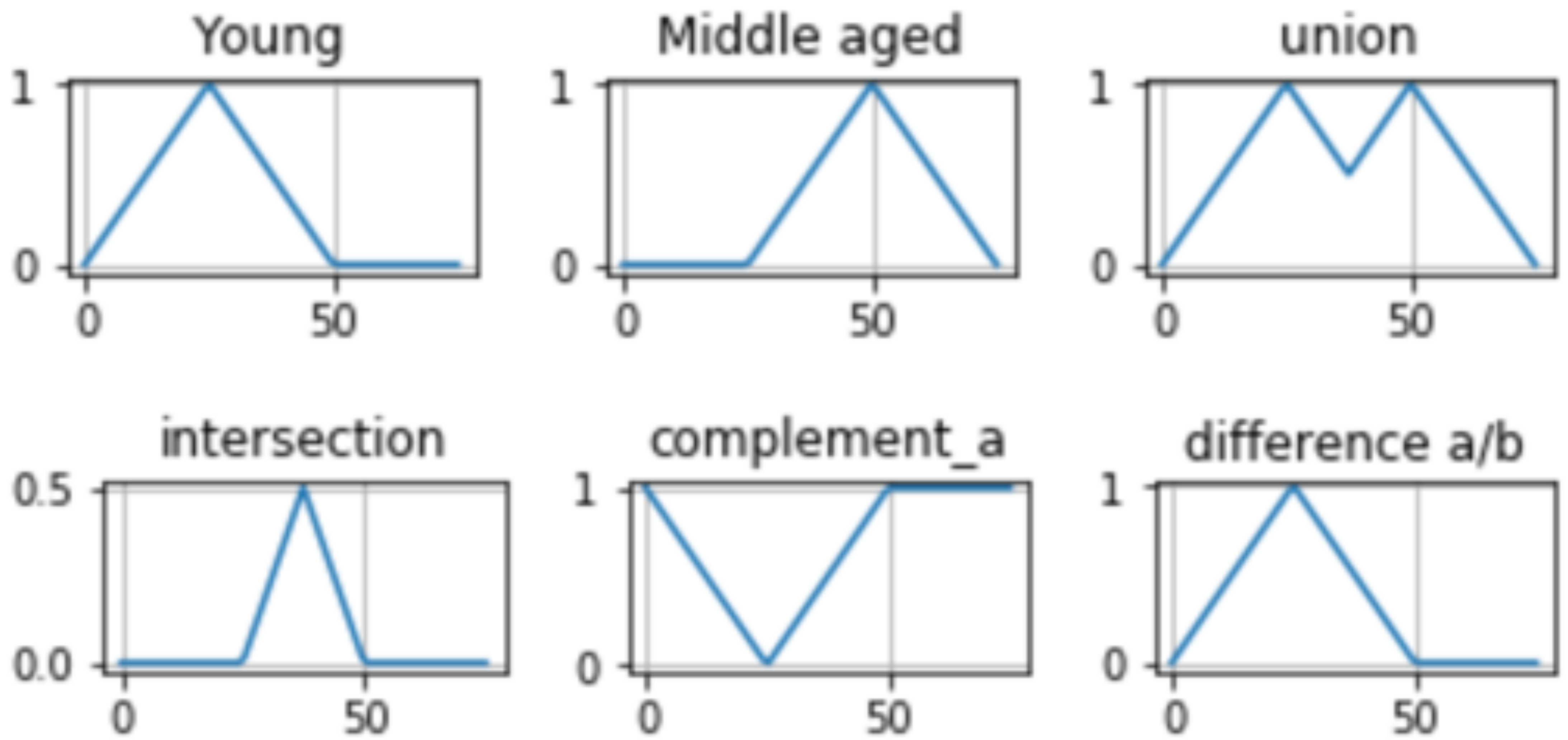
Crisp data: values that are not fuzzy;

Fuzzification: turn crisp data into fuzzy sets

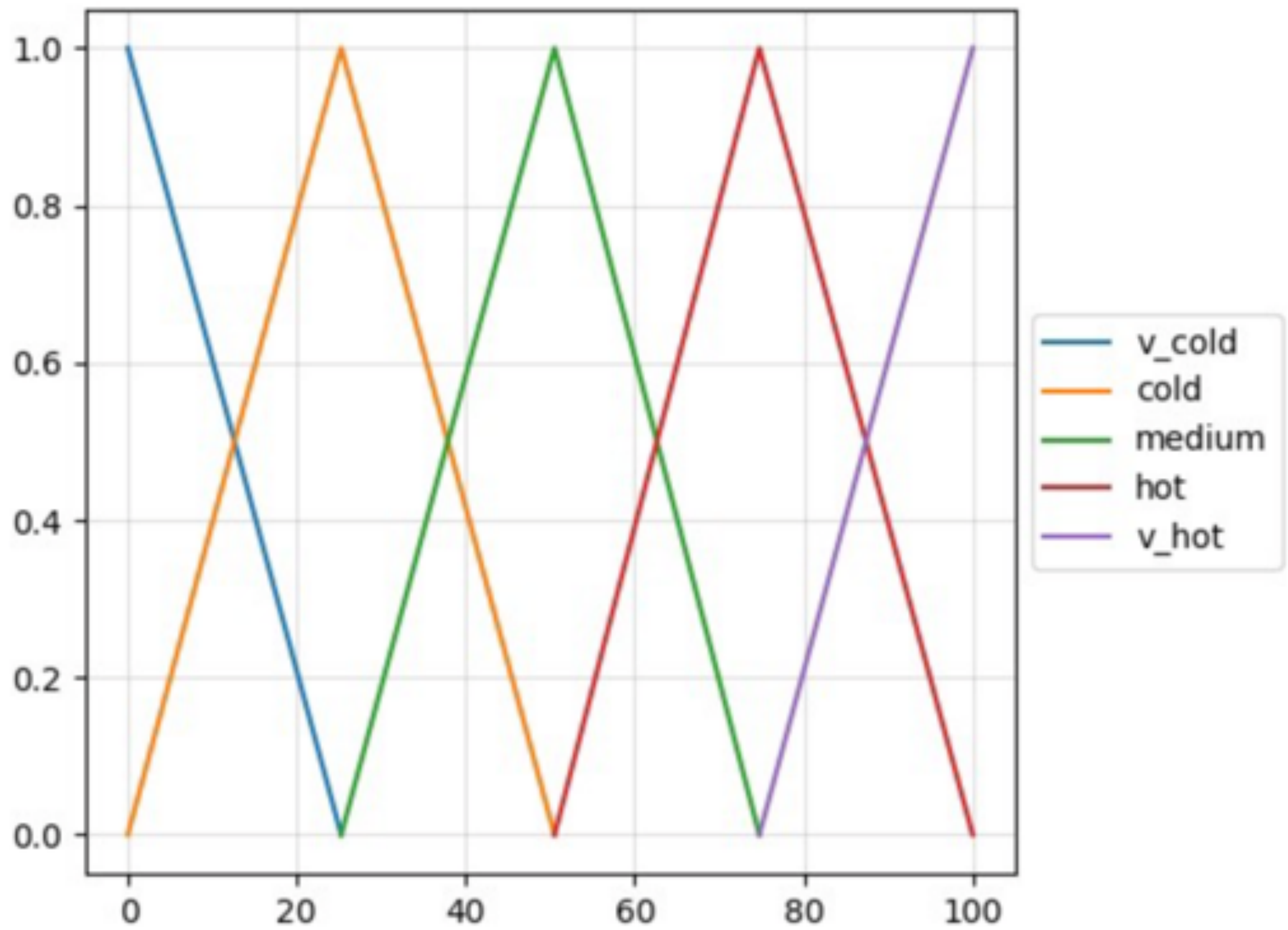
Fuzzy rules: IF THEN rules of inference

Defuzzification: transform fuzzy sets into crisp values (centroid, maxima)

Fuzzification (example with AGE)



Fuzzification (example with COLD)



1. Play with FuzzyLogic01.ipynb (Basic operations)

1. Play with FuzzyLogic01.ipynb (Basic operations)

2. Play with FuzzyLogic02.ipynb (Defussification)

1. Play with FuzzyLogic01.ipynb (Basic operations)

2. Play with FuzzyLogic02.ipynb (Defussification)

3. Learn FuzzyLogic03.ipynb (Better code)

Use FuzzyLogic03.ipynb and implement
your own problem!

1. Play with FuzzyLogic01.ipynb (Basic operations)

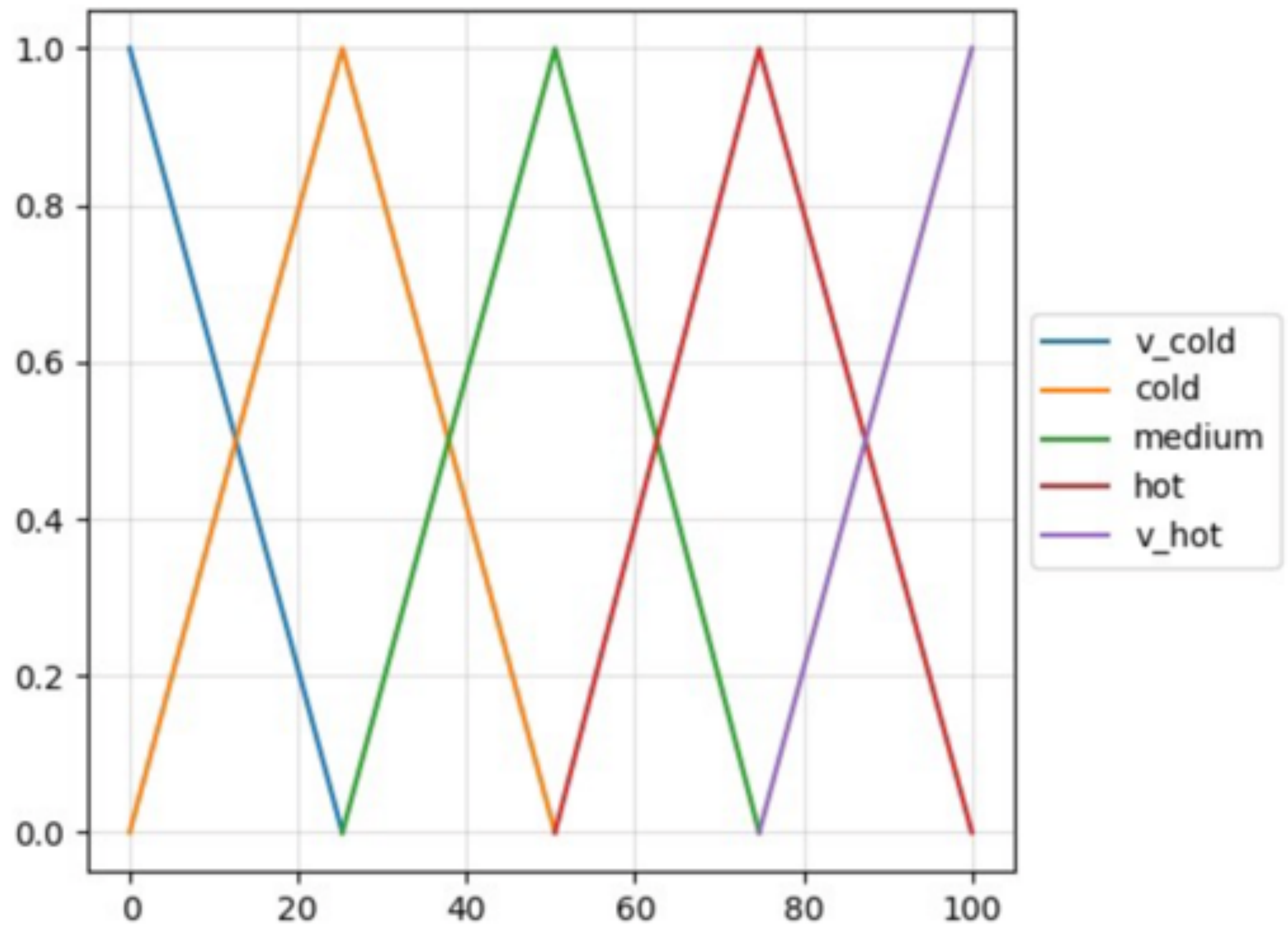
2. Play with FuzzyLogic02.ipynb (Defussification)

3. Learn FuzzyLogic03.ipynb (Better code)

Use FuzzyLogic03.ipynb and implement your own problem!

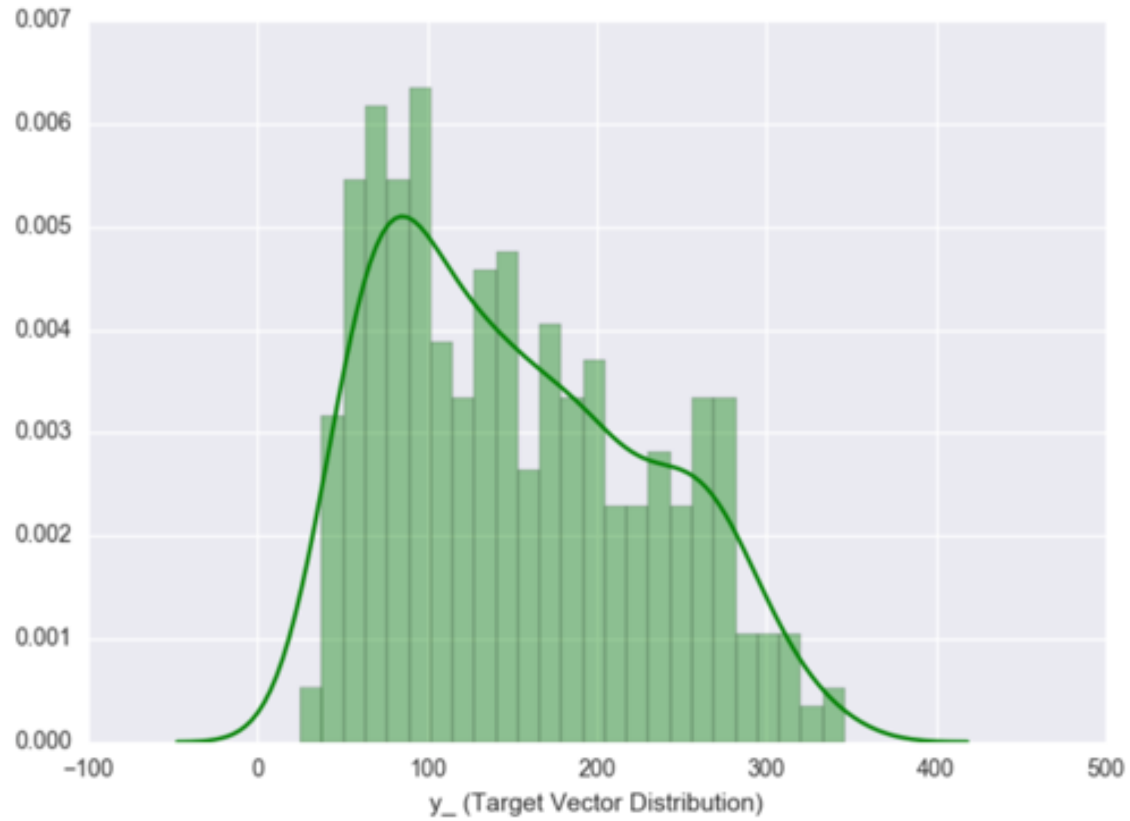
Fuzzification (recall the example with COLD)

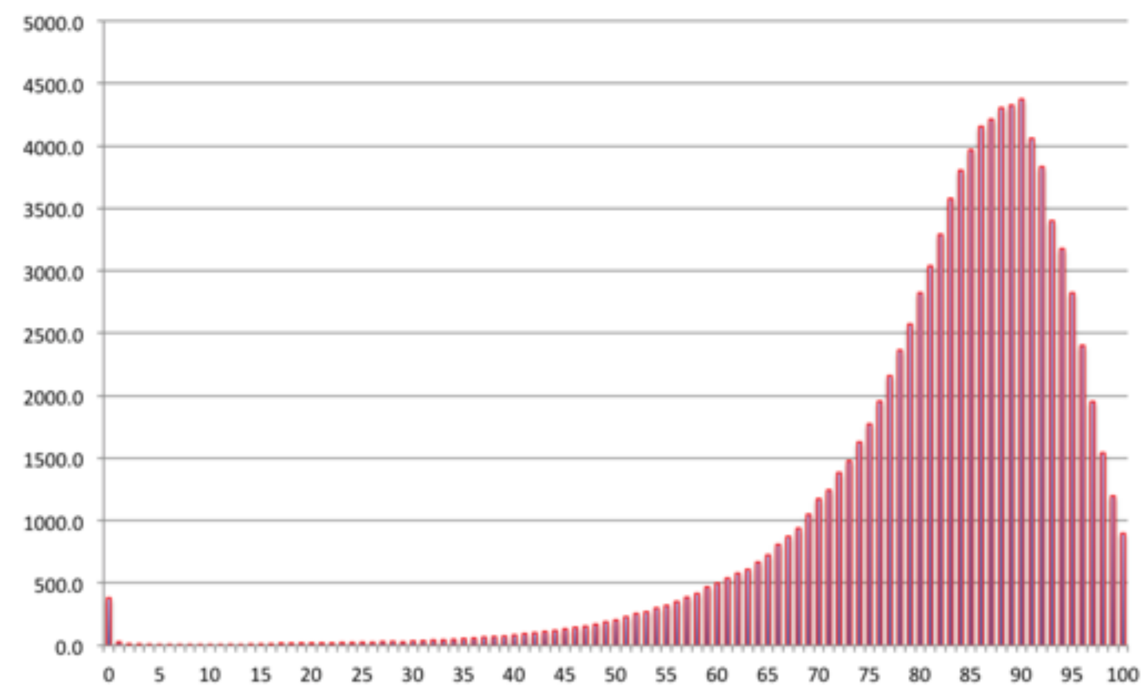
What is the best way to do Fuzzification with real data?

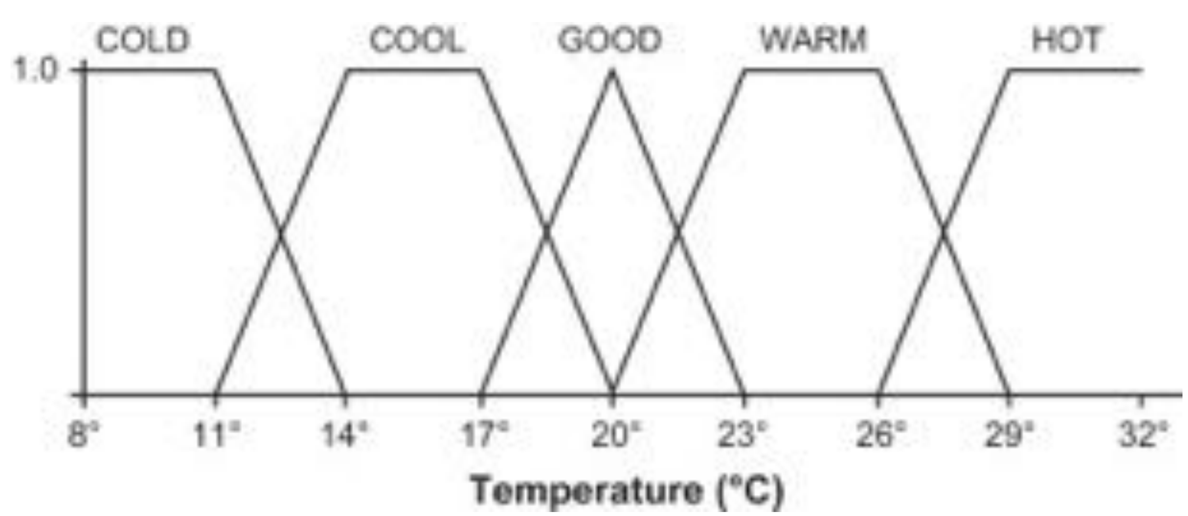


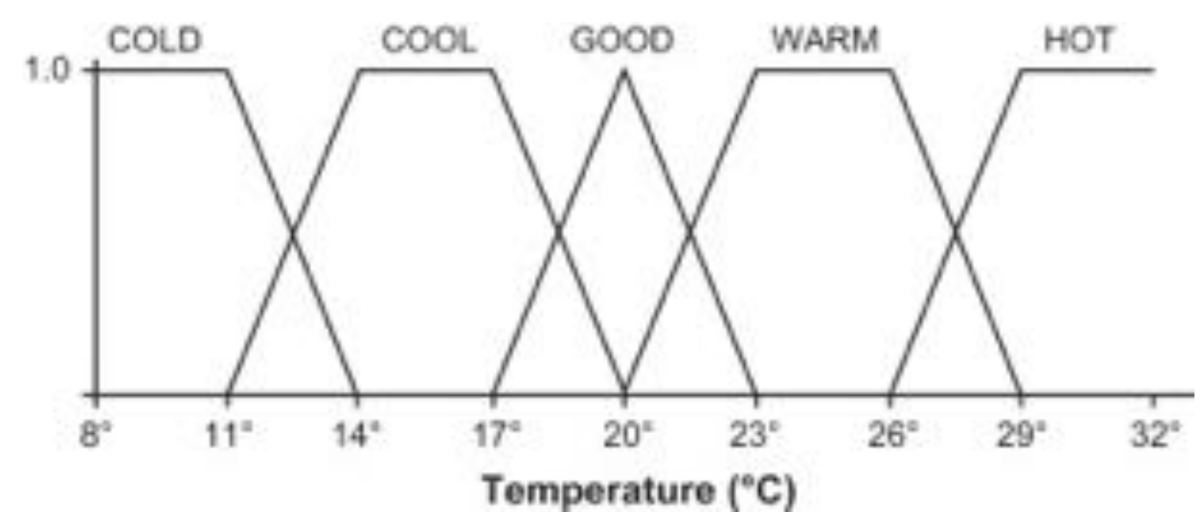
What is the best way to do Fuzzification?

We are still just scratching the surface



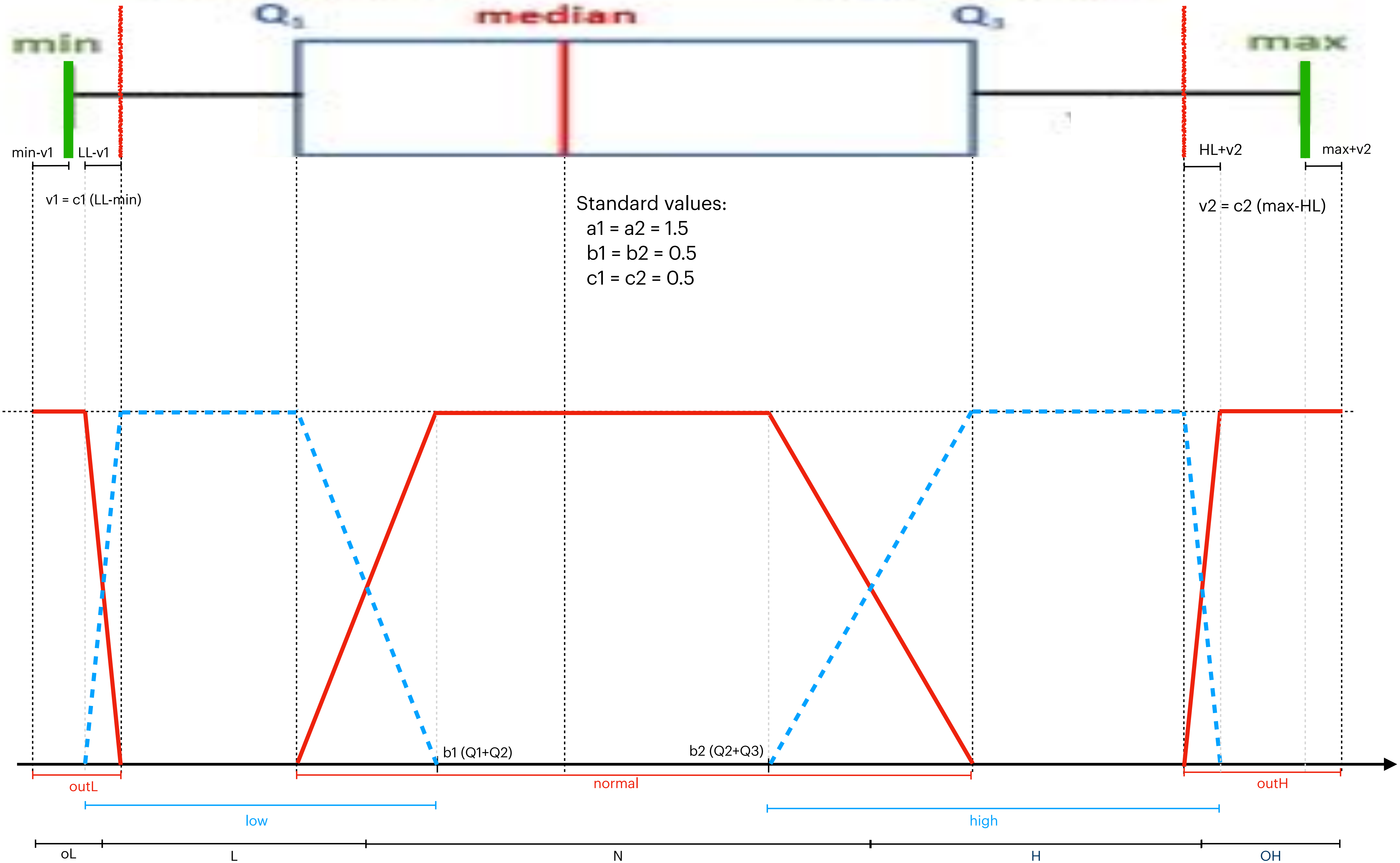






LL = MAX{min, Q1 - a1 (Q3 - Q1)}

HL = MIN{max, Q3 + a2 (Q3 - Q1)}



Standard values:
a1 = a2 = 1.5
b1 = b2 = 0.5
c1 = c2 = 0.5

Operations on Fuzzy Sets

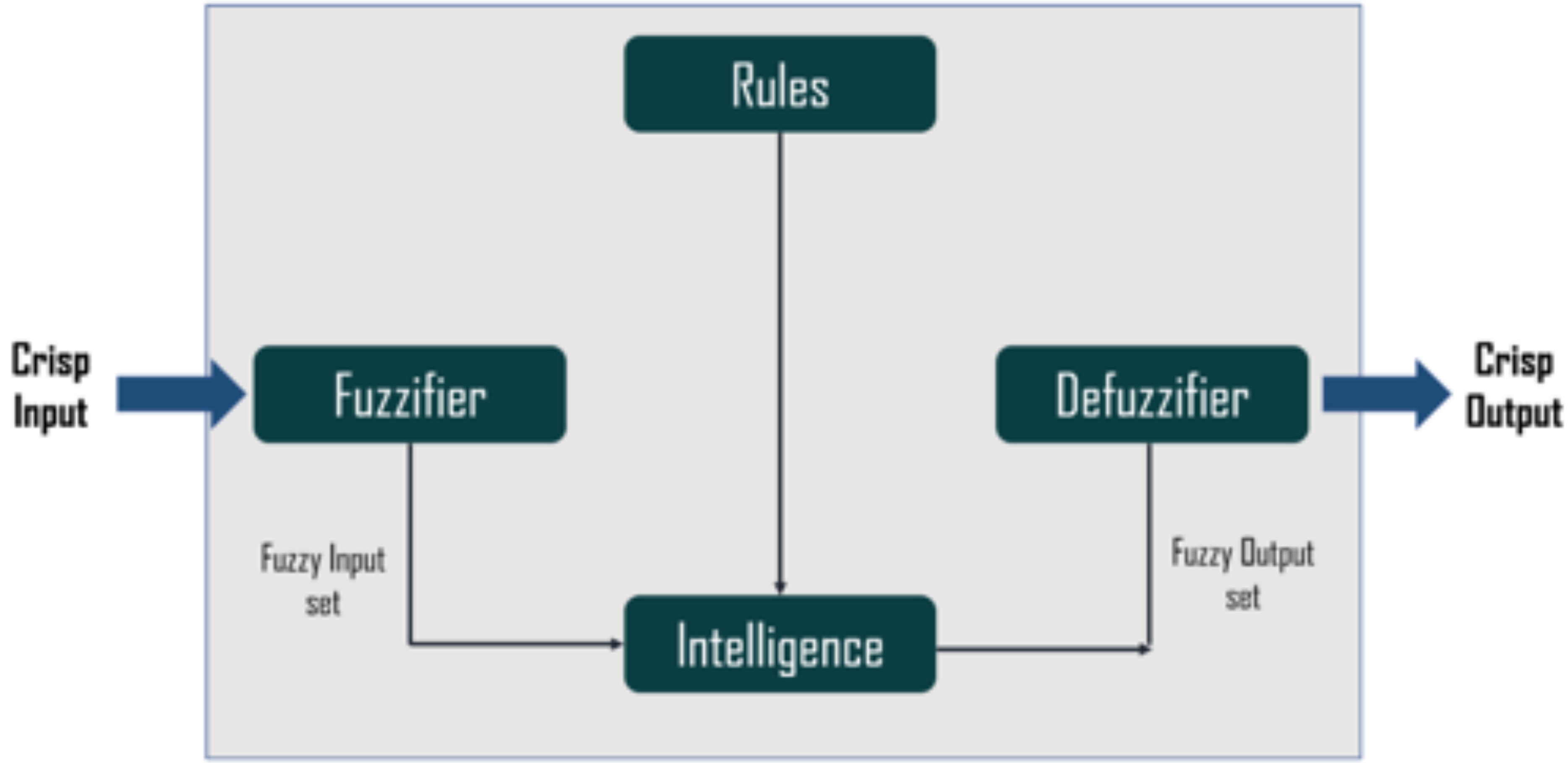
We are still just scratching the surface

Let $(A, \mu_A), (B, \mu_B)$ be a fuzzy sets.

- COMPLEMENTATION: $(\neg A, \mu_{\neg A})$, where $\mu_{\neg A} = 1 - \mu_A$
- HEIGHT: $h(A) = \max_{x \in X} \mu_A(x)$
- SUPPORT: $\mathrm{supp}(A) = \{x \in X \mid \mu_A(x) > 0\}$
- CORE: $C(A) = \{x \in X \mid \mu_A(x) = 1\}$
- INTERSECTION: $C = A \cap B$, where $\mu_C = \min_{x \in X}\{\mu_A, \mu_B\}$
- UNION: $C = A \cup B$, where $\mu_C = \max_{x \in X}\{\mu_A, \mu_B\}$
- BOUNDED SUM: $C = A + B$, where $\mu_C(x) = \min\{1, \mu_A(x) + \mu_B(x)\}$
- BOUNDED DIFFERENCE: $C = A - B$, where $\mu_C(x) = \max\{0, \mu_A(x) - \mu_B(x)\}$

Why? Alternatives?

RELEVANT for Fuzzy Inference



Crisp data: values that are not fuzzy;

Fuzzification: turn crisp data into fuzzy sets

Fuzzy rules: IF THEN rules of inference   How can be computed?

Defuzzification: transform fuzzy sets into crisp values (centroid, maxima)

Operations on Fuzzy Sets

As mentioned before, there are in fact (infinitely) many ways for defining operations on fuzzy sets. That comes form the fact, that there is more than one:

1. Function that satisfies conditions for a T-norm, i.e., equivalent to intersection.
2. Function that satisfies conditions for a T- co-norm (S-norm), i.e., equivalent to union.
3. Function that satisfies conditions complement (negation).

We will see several examples of such functions.

Intersection, Union and (Negation)

The whole class of functions that are called T-norms can be used as fuzzy intersection.

Definition – T-norm

For any $a, b, c, d \in [0,1]$ *T-norm* is a function $T : [0,1]^2 \to [0,1]$ such that:

- Commutativity: $T(a,b) = T(b,a)$;
- Associativity: $T(a, T(b,c)) = T(T(a,b),c)$;
- Monotonicity: $T(a,b) \geq T(c,d)$ whenever $a \geq c$, $b \geq d$;
- Invariance for 1: $T(a,1) = a$

It is quite easy to see that the intersection of fuzzy sets defined by $T(a,b) = \min(a,b)$ is a proper T-norm. In fact, the function $\min(.,.)$ is a maximal element in the class of T-norms.

Analogously, the whole class of S-norms (T- co-norms) can be used as fuzzy union.

DEFINITION – S-NORM

For any $a, b, c, d \in [0, 1]$ *S-norm* (T- co-norm) is a function $S : [0, 1]^2 \to [0, 1]$ such that:

- Commutativity: $S(a, b) = S(b, a)$;
- Associativity: $S(a, S(b, c)) = S(S(a, b), c)$;
- Monotonicity: $S(a, b) \geq S(c, d)$ whenever $a \geq c$, $b \geq d$;
- Invariance for 0: $S(a, 0) = a$.

We have already seen examples of T-norms and S-norms such as:

- $T(a, b) = \max(0, a + b - 1)$, $S(a, b) = \min(1, a + b)$ - so called Łukasiewicz operators.
- T(a,b) = ab,   S(a,b) = a + b - ab   - so called product operators.

The complement of a fuzzy set (fuzzy negation) we can also define in great number of ways. All we need, is that negation conforms to a set of conditions.

Definition – fuzzy complement (negation)

For any $a, b \in [0,1]$ function $N : [0,1] \to [0,1]$ is called a complement (negation) operation if the following holds:

- Preservation of constants: $N(0) = 1; N(1) = 0$;
- Reversing of the order: $N(a) \leq N(b)$ iff $b \leq a$;
- Involution: $N(N(a)) = a$.

There are several functions that may be used as complement, but in 99.9% of applications (and further in this lecture) the only function used is $N(x) = 1 - x$ ($\mu_{\setminus A} = 1 - \mu_A$).
If in the definition above we cannot assure involution, the resulting operator is called an **intuitionistic negation**.

Duality

Once we have a negation operator, we can define an S-norm (T - co-norm) dual to a given T-norm.

Definition – dual S-norm

Given a T-norm $T : [0,1]^2 \to [0,1]$ we can define its dual co-norm (S-norm), and vice versa, by:

$$S(a,b) = N(T(N(a),N(b)))$$

As an exercise one may check if the examples of T-norms and S-norms presented previously are dual to each other.

As mentioned before, the pair of operations $\min(.,.)$ and $\min(.,.)$ play a special role. They are only idempotent operators in the class of T-norms and S-norms, respectively. They are also ones that conform to distributive laws: $A \cup (B \cap C) = (A \cup B) \cap (A \cup C)$ and $A \cap (B \cup C) = (A \cap B) \cup (A \cap C)$.

Lemma: For any real numbers x, y and z, we have:

1. $x - \min(y,z) = \max(x-y,x-z)$
2. $x - \max(y,z) = \min(x-y,x-z)$
3. $\min(x,y) - z = \min(x-z,y-z)$
4. $\max(x,y) - z = \max(x-z,y-z)$

Exerc 1: Proof that Lukasiewicz operators are (T,S)-norms.

(see corresponding notebook)

Lemma: For any real numbers x, y and z, we have:

1. x - min(y,z) = max(x-y,x-z)
2. x - max(y,z) = min(x-y,x-z)
3. min(x,y) - z = min(x-z,y-z)
4. max(x,y) - z = max(x-z,y-z)

Exerc 1: Proof that Lukasiewicz operators are (T,S)-norms.

(see corresponding notebook)

Exerc 2: Propose the operator "implication" for classical operators (Zodeh) and for Lukasiewicz operators.
Is it correct?

Lemma: For any real numbers x, y and z, we have:

1. x - min(y,z) = max(x-y,x-z)
2. x - max(y,z) = min(x-y,x-z)
3. min(x,y) - z = min(x-z,y-z)
4. max(x,y) - z = max(x-z,y-z)

Exerc 1: Proof that Lukasiewicz operators are (T,S)-norms.

(see corresponding notebook)

Exerc 2: Propose the operator "implication" for classical operators (Zodeh) and for Lukasiewicz operators.
Is it correct?

Exerc 3: Propose the operator "implication" for any (T,S)-norms.

Lemma: For any real numbers x, y and z, we have:

1. x - min(y,z) = max(x-y,x-z)
2. x - max(y,z) = min(x-y,x-z)
3. min(x,y) - z = min(x-z,y-z)
4. max(x,y) - z = max(x-z,y-z)

Exerc 1: Proof that Lukasiewicz operators are (T,S)-norms.
(see corresponding notebook)

Exerc 2: Propose the operator “implication” for classical operators (Zodeh) and for Lukasiewicz operators.
Is it correct?

Exerc 3: Propose the operator “implication” for any (T,S)-norms.

Exerc 4: What is the minimum set of operators needed to define a Fuzzy Logic System?

Fuzzy Inference

Linguistic rules are the statements of the form:

$$\textbf{IF } A_1 \textbf{ AND } A_2 \textbf{ AND ... AND } A_k \textbf{ THEN } D$$

where conditions $A_1, \ldots, A_k$ and decision D correspond to fuzzy sets. For example:

IF waether is **good** **AND** traffic is **light** **AND** we have **enough** fuel **THEN** we will reach the airport in **about 30 minutes**

such rules we may obtain form a human expert or discover (mine) from data. In order to use them in the context of fuzzy sets, we will employ fuzzy set operators.

Mainly two methods:

- Ebrahim **Mamdani**, London, 1975 (steam engine and boiler combined);
- Michio **Sugeno**, 1985

Fuzzy Inference

Fuzzy Inference System	Advantages
Mamdani	• Intuitive • Well-suited to human input • More interpretable rule base • Have widespread acceptance
Sugeno	• Computationally efficient • Work well with linear techniques, such as PID control • Work well with optimization and adaptive techniques • Guarantee output surface continuity • Well-suited to mathematical analysis

[1] Mamdani, E.H., and S. Assilian. 'An Experiment in Linguistic Synthesis with a Fuzzy Logic Controller'. *International Journal of Man-Machine Studies* 7, no. 1 (January 1975): 1–13. https://doi.org/10.1016/S0020-7373(75)80002-2.

[2] Sugeno, Michio, ed. *Industrial Applications of Fuzzy Control*. Amsterdam ; New York : New York, N.Y., U.S.A: North-Holland ; Sole distributors for the U.S.A. and Canada, Elsevier Science Pub. Co, 1985.

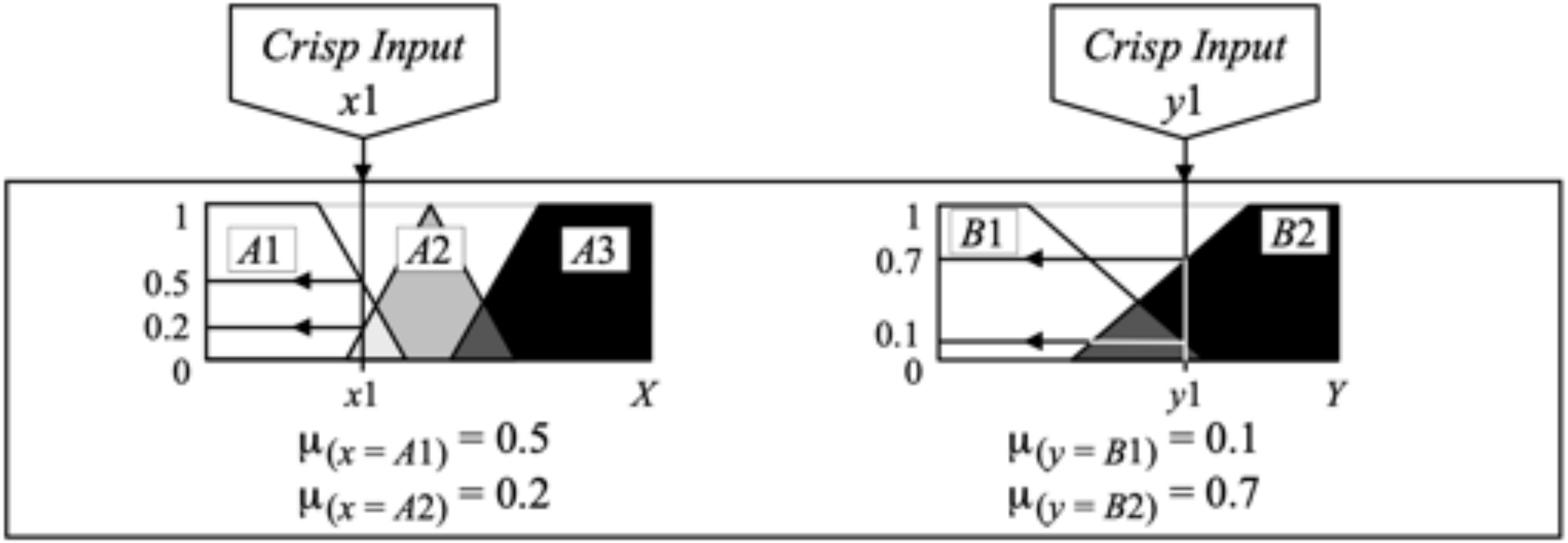
Classical fuzzy inference system steps

1. Fuzzification of input variables
2. Evaluation of each rule
   (output may be an area or a set of values)
3. Aggregation of rule results
   (output may be an area or a set of values)
4. Defuzzification
   (e.g. centroid for sets or median for values)

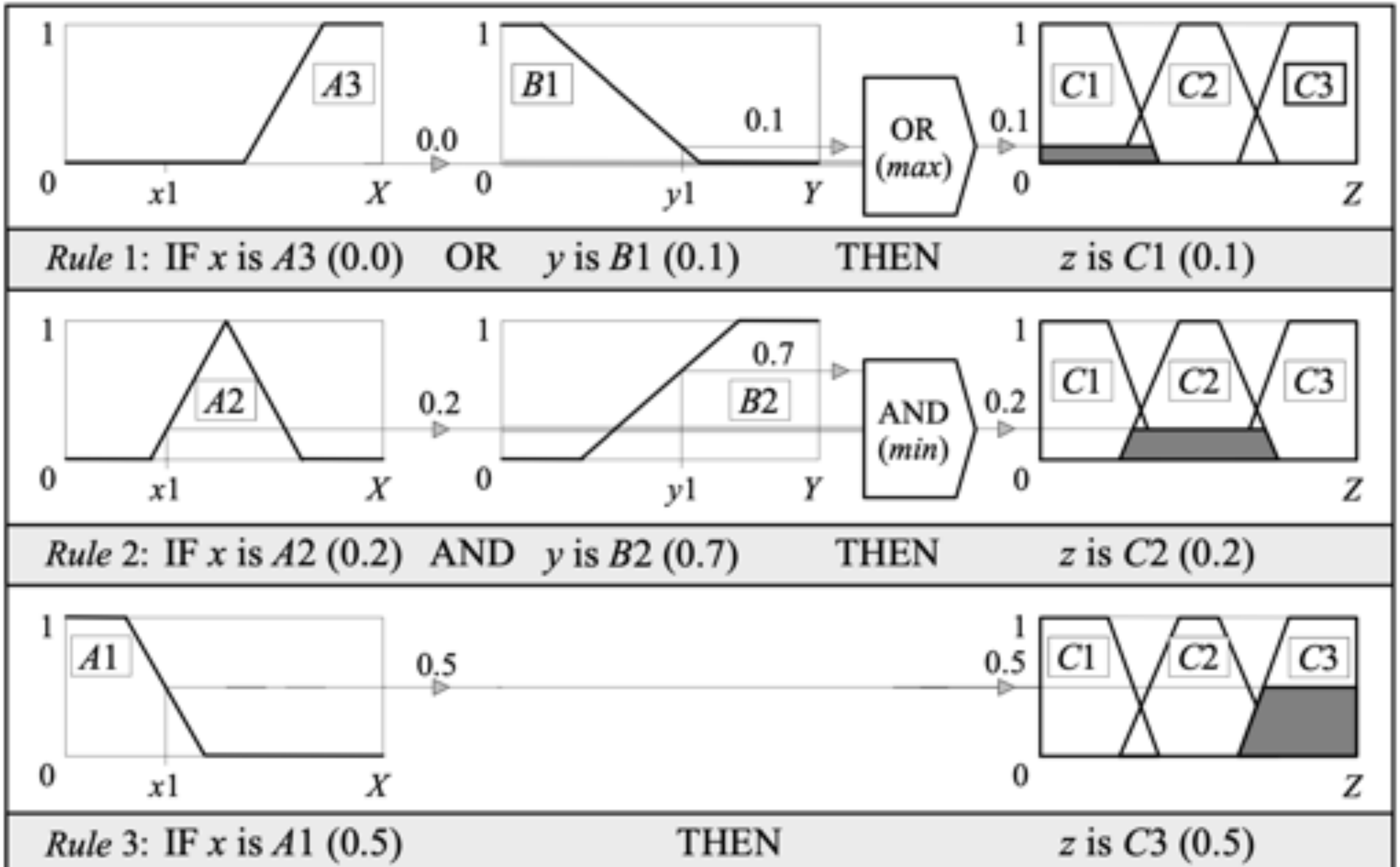
Classical fuzzy inference system steps

1. Fuzzification of input variables
2. Evaluation of each rule
   (output may be an area or a set of values)
3. Aggregation of rule results
   (output may be an area or a set of values)
4. Defuzzification
   (e.g. centroid for sets or median for values)

S1. Fuzzification

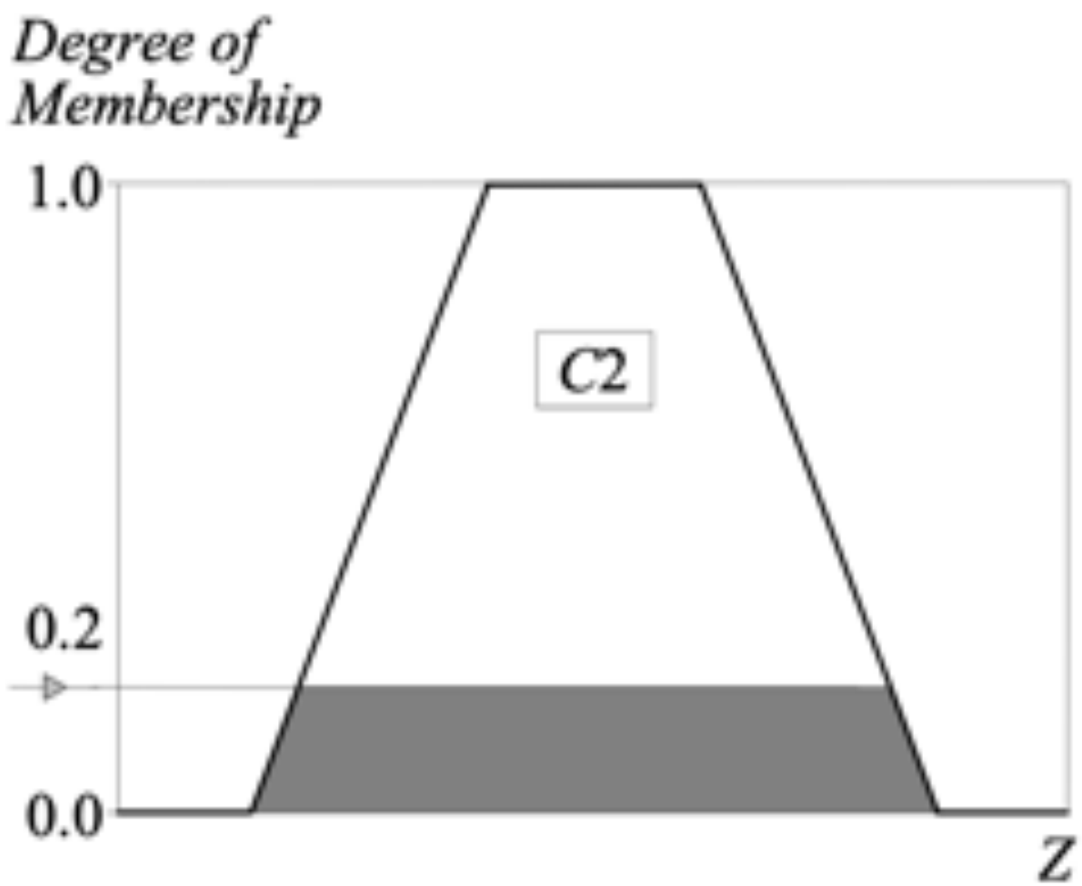


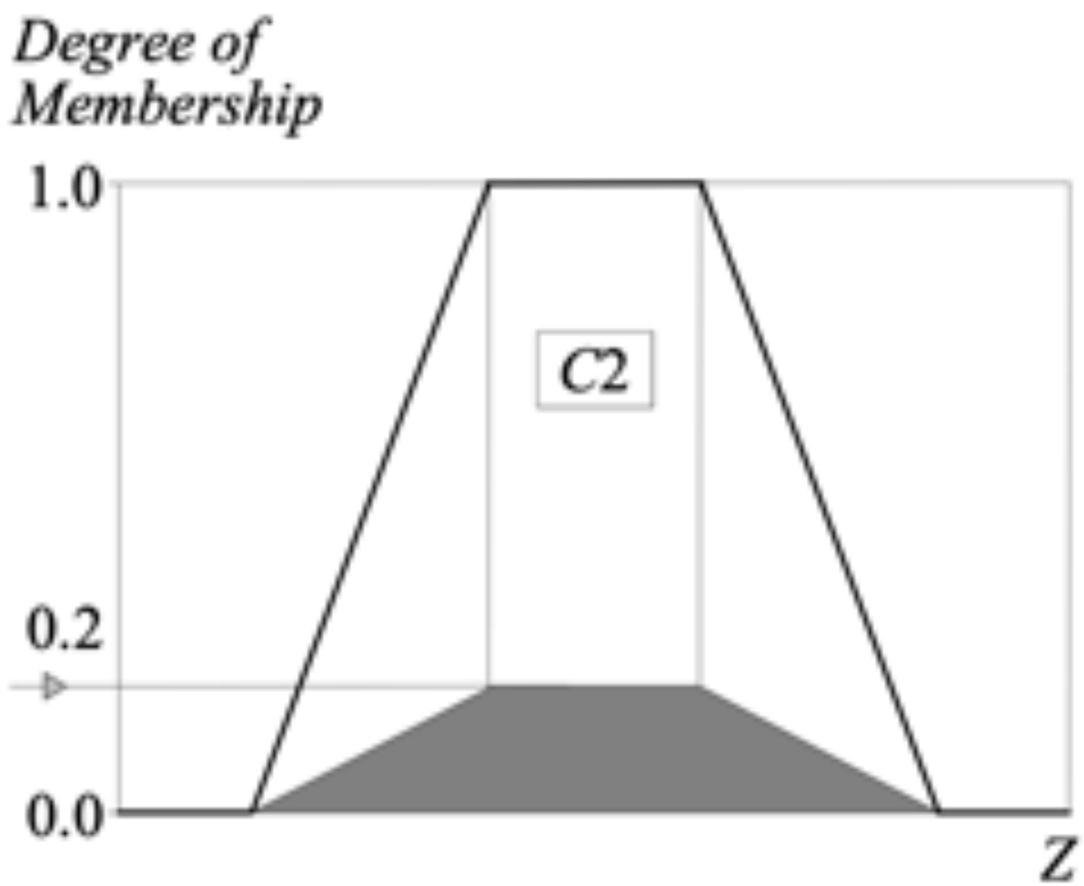
S2. Rule evaluation (e.g. via T-norm) [Mandani]



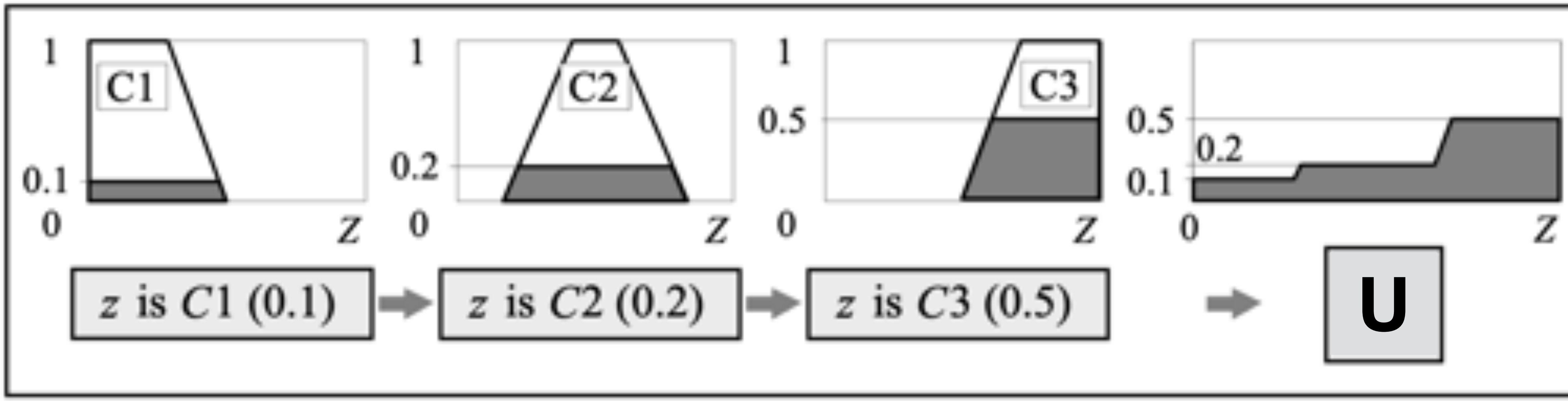
S2. Rule evaluation (e.g. via T-norm)

Clipping vs Scaling

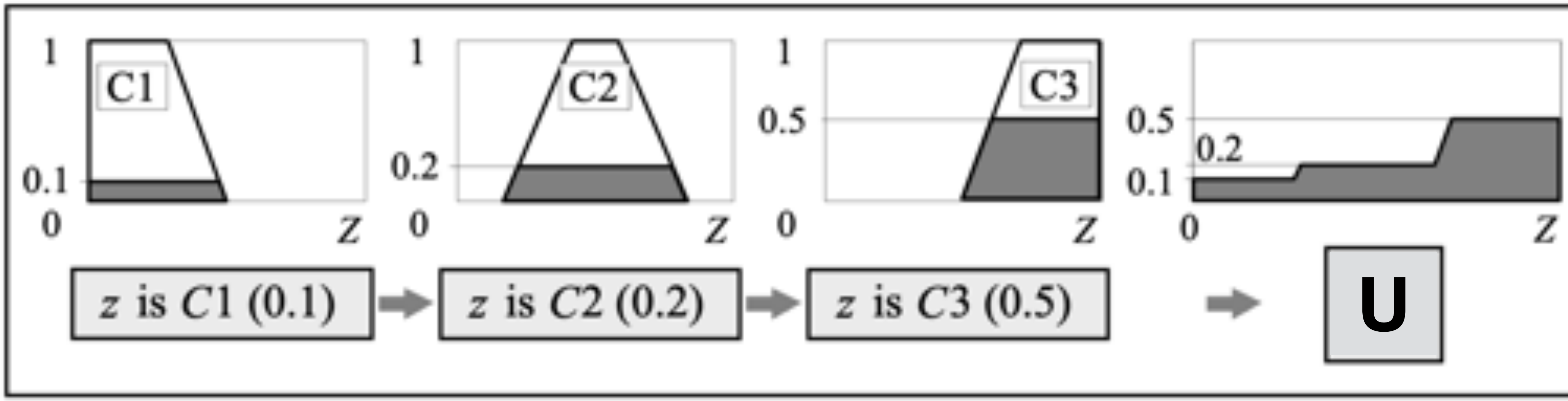




S3. Aggregation of the rule results [Mandani]



S3. Aggregation of the rule results [Mandani]

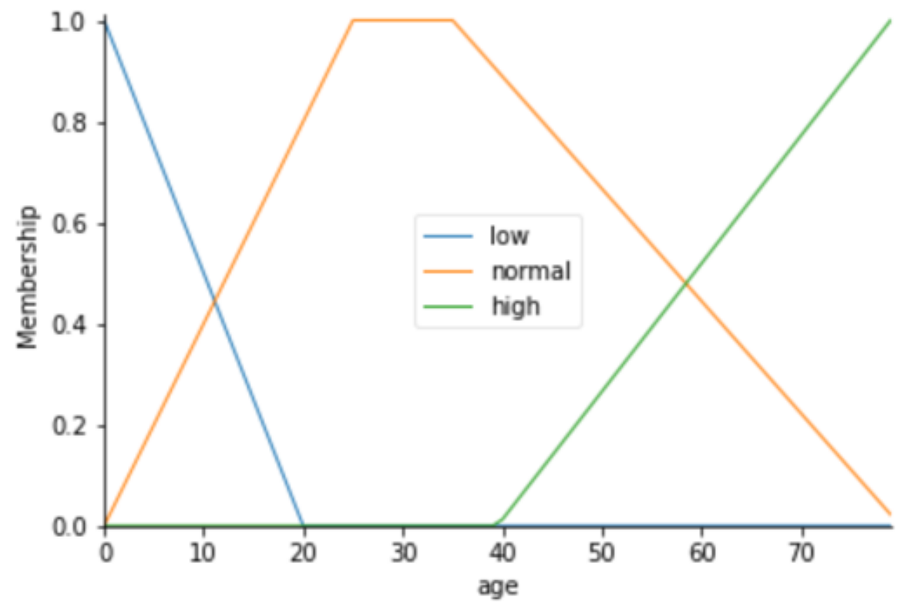


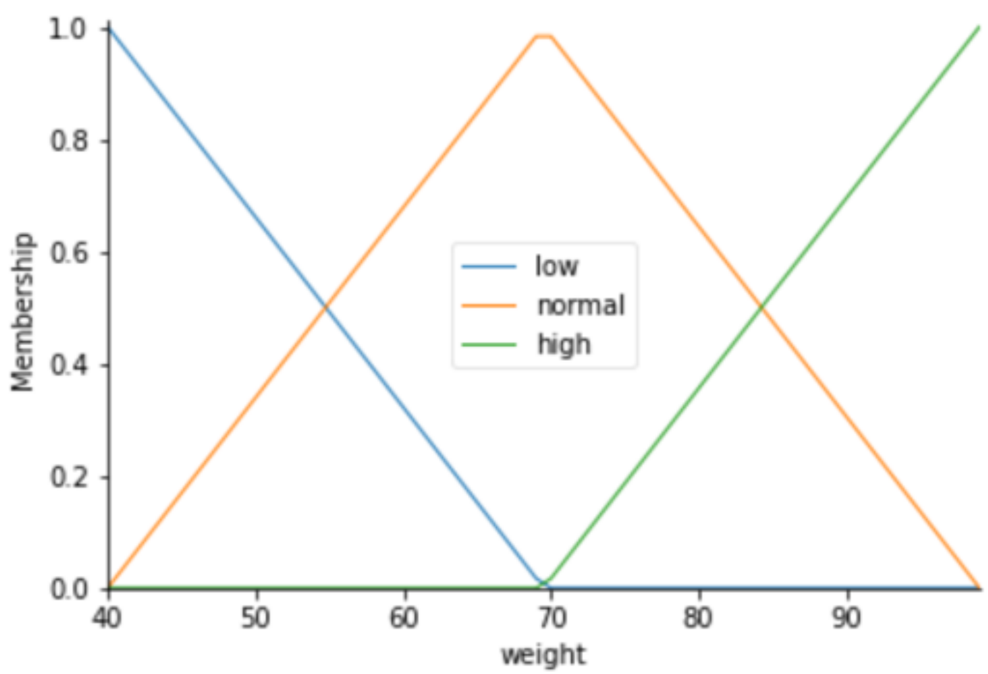
S4. Defuzzification [Mandani]

Centroid (center of gravity): $$COG = \frac{\int_a^b \mu_A(x)\, x\, dx}{\int_a^b \mu_A(x)\, dx}$$

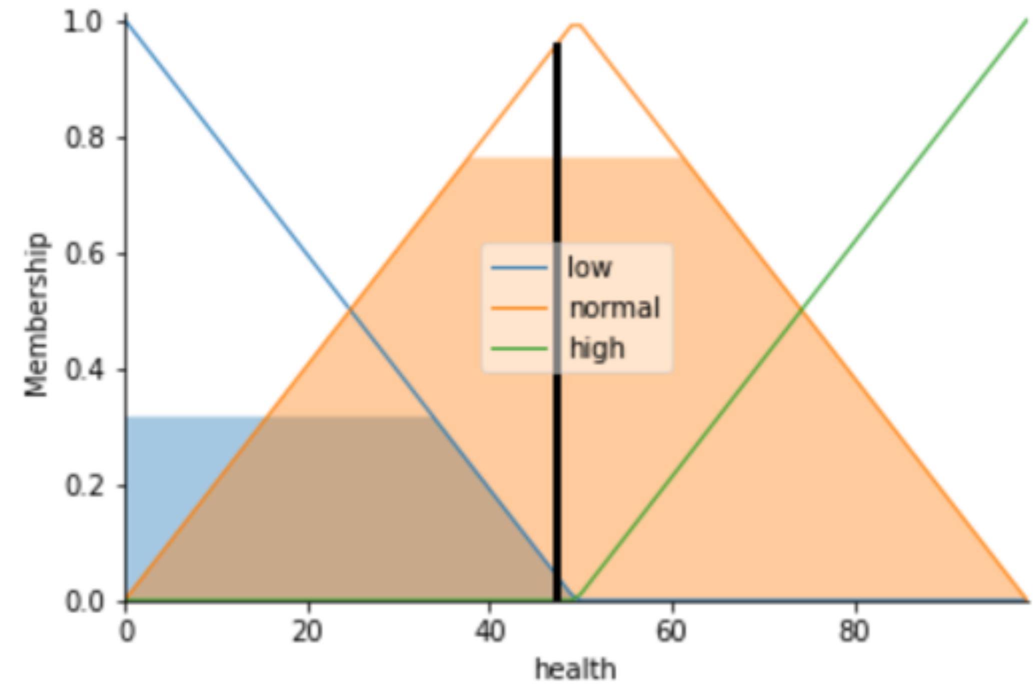
(not computacional efficient)

Exerc: Explique como se obtém o gráfico final.



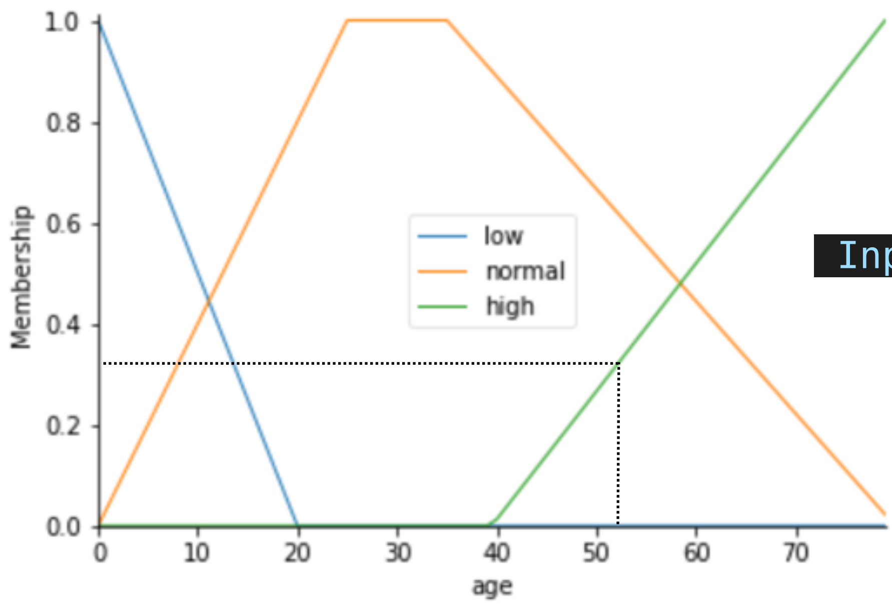



```
Input['age']    = 52
Input['weight'] = 47
```



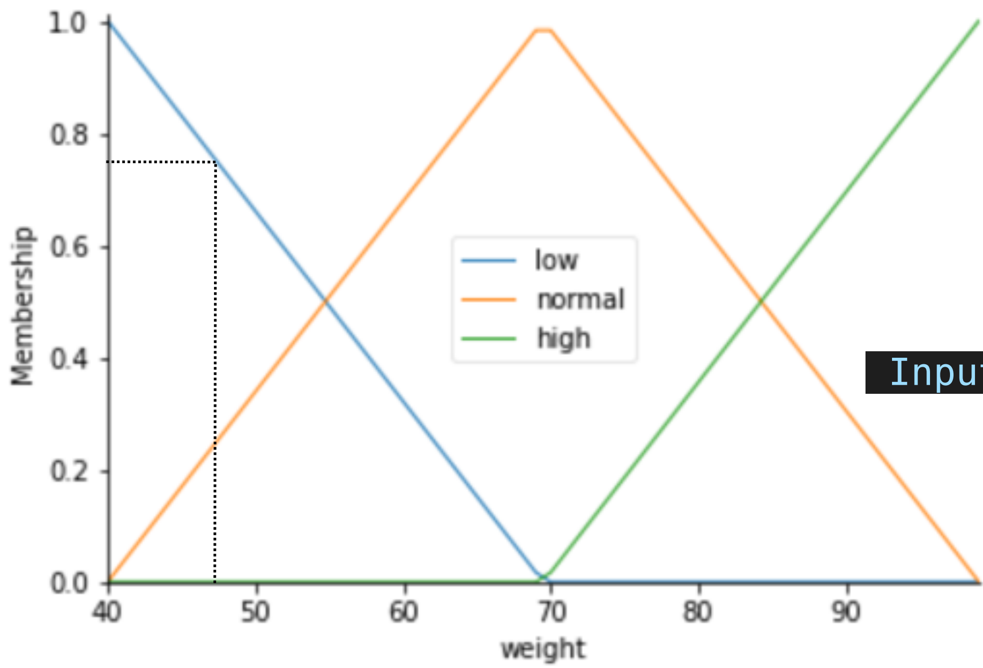

```
Rule1: IF weight['low'] AND age['high'] THEN health['low']
Rule2: IF weight['low'] OR age['low'] THEN health['normal']
```

Resposta: Para `Rule1: IF weight['low'] AND age['high'] THEN health['low']`



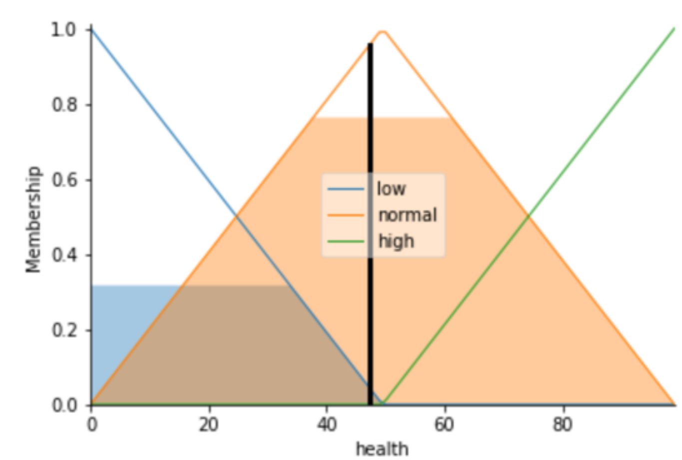
`Input['age'] = 52` → `age['high'] = 0.32`

`weight['low'] AND age['high'] = min(0.32,0.77) = 0.32`

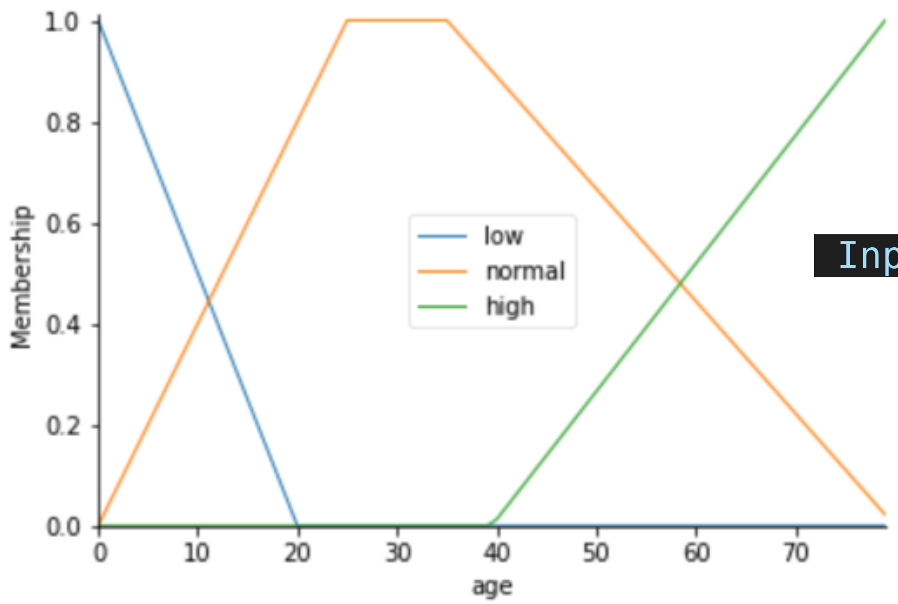


`Input['weight'] = 47` → `weight['low'] = 0.77`

The blue area is from Rule1



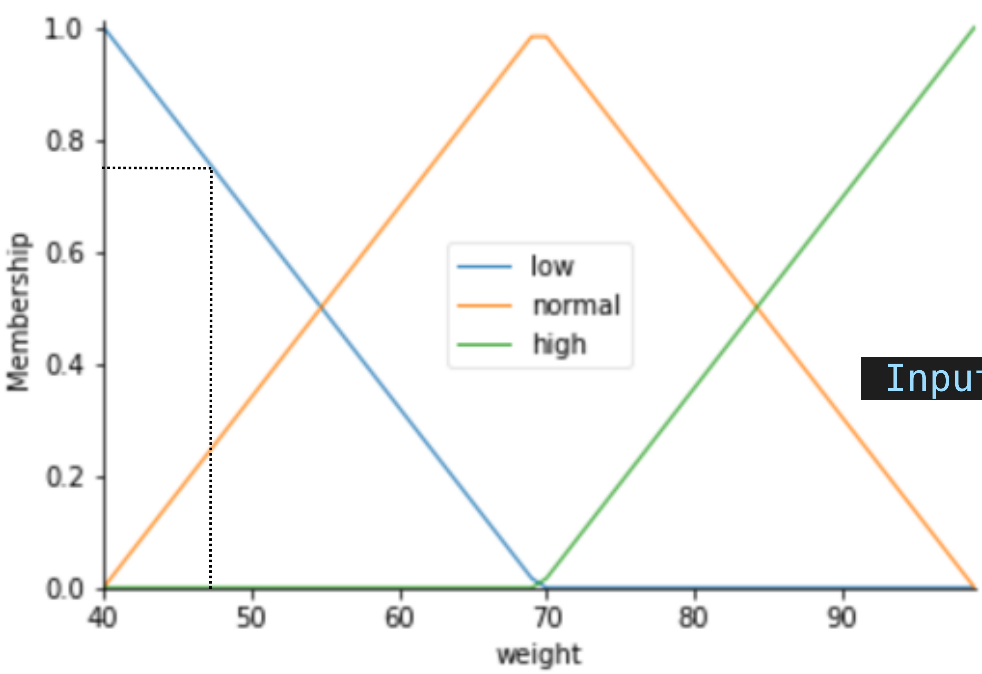
Resposta: Para `Rule2: IF weight['low'] OR age['low'] THEN health['normal']`



`Input['age'] = 52`

`age['low'] = 0.00`

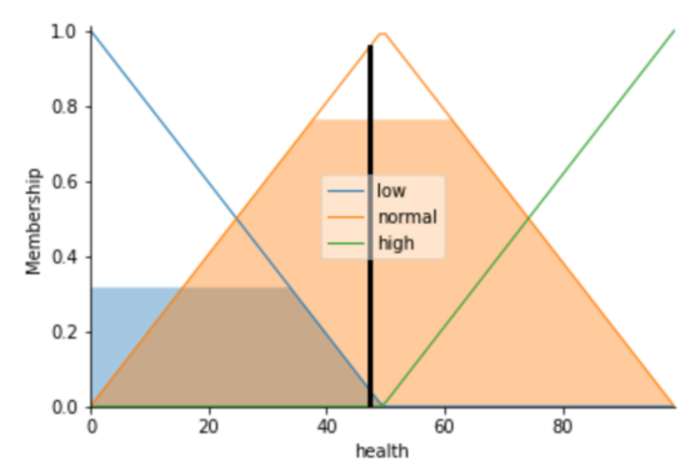
`weight['low'] OR age['low'] = max(0.00, 0.77) = 0.77`



`Input['weight'] = 47`

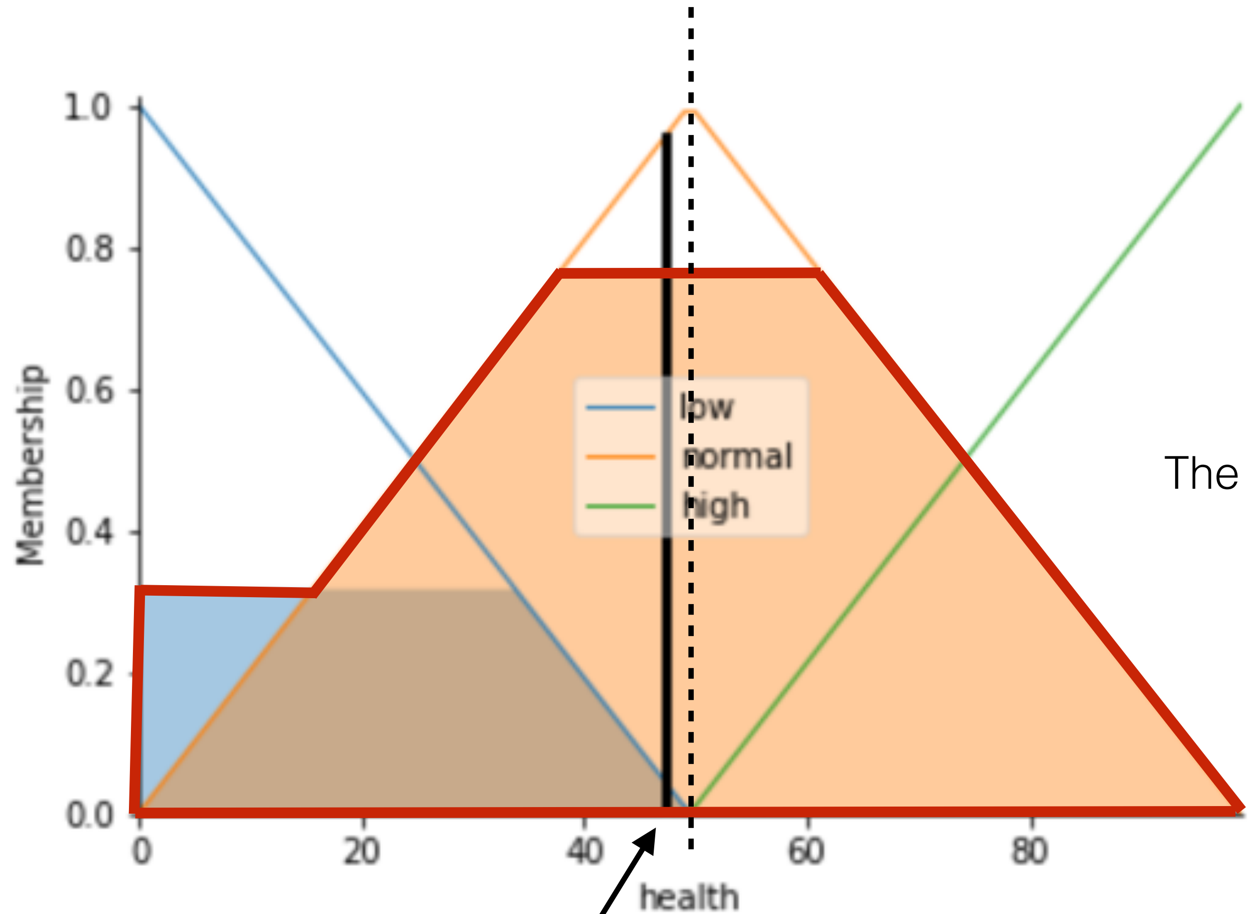
`weight['low'] = 0.77`

The orange area is from Rule2



Resposta: O passo final (defuzzification) é dado por

The blue area is from Rule1

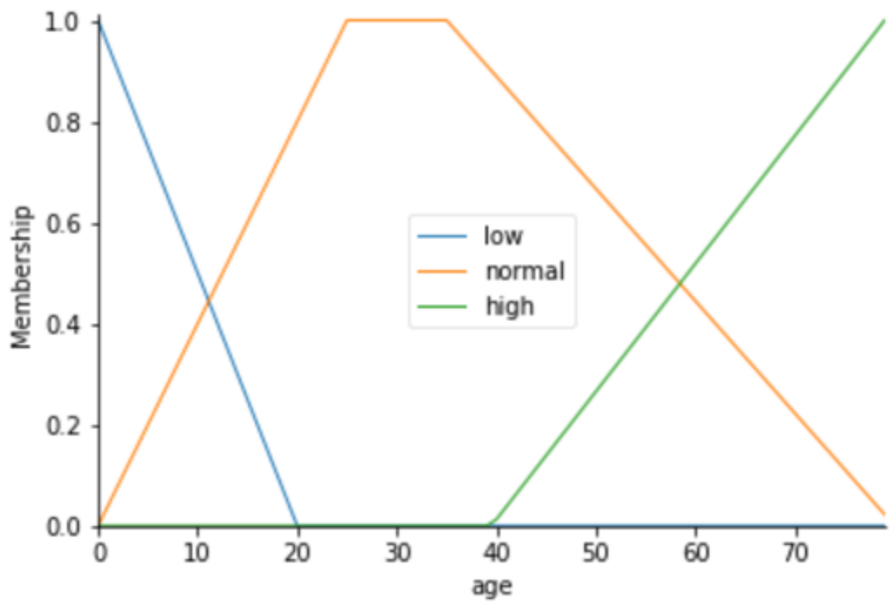


The orange area is from Rule2

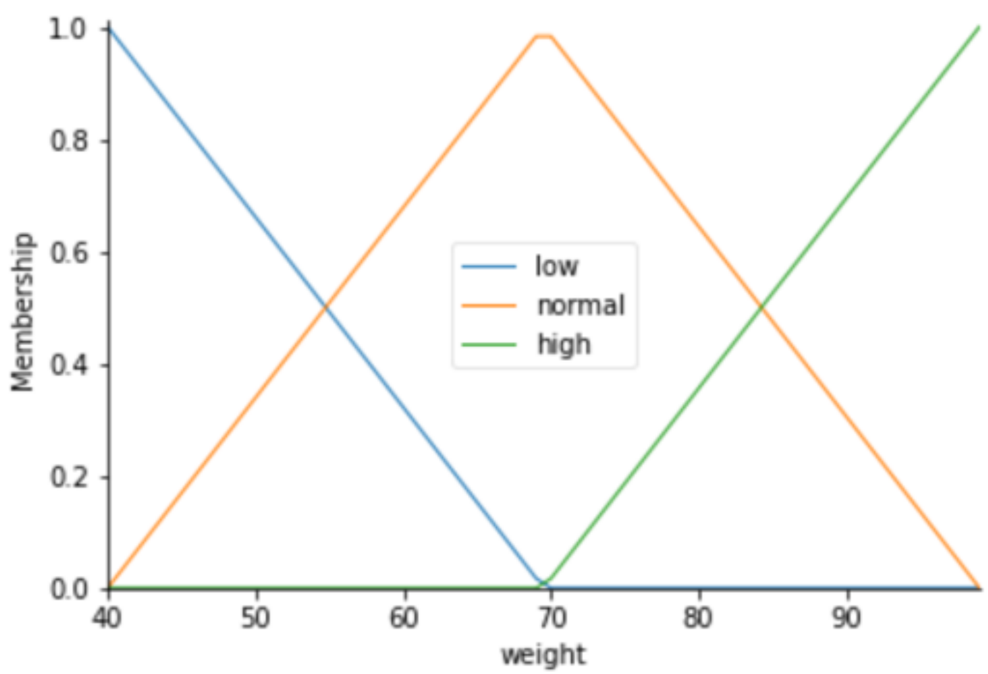
The results is just the Centroid (center of gravity) = 48

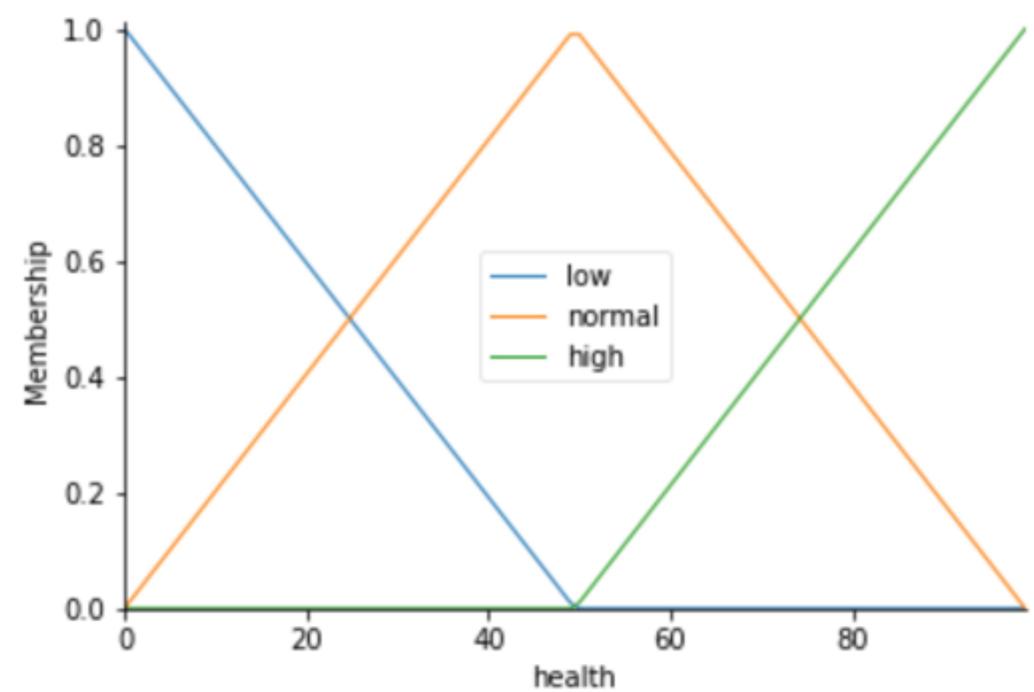
Exerc: Find the results, for the product operators.

$T(x,y) = xy$
$S(x,y) = x+y-xy$




```
Input['age']    = 52
Input['weight'] = 47
```



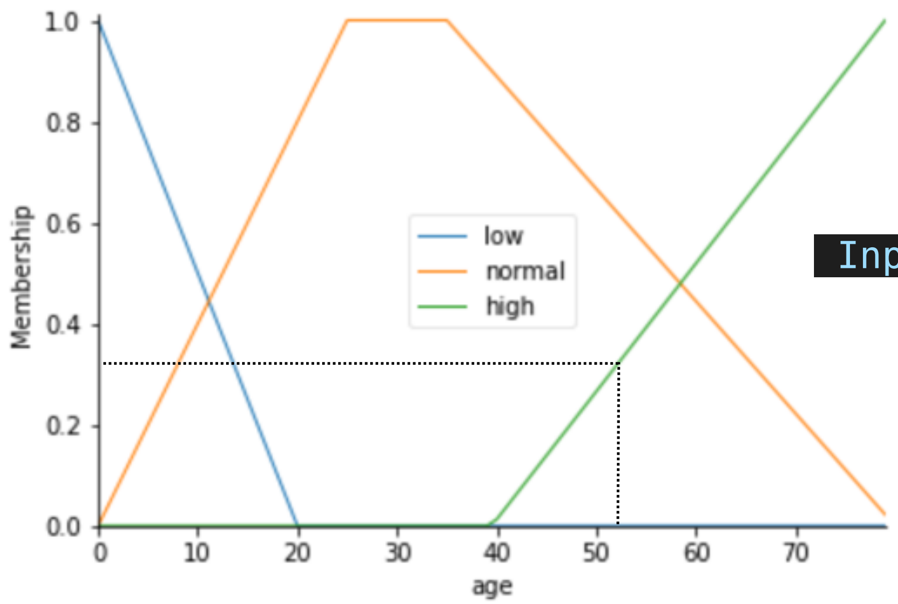


?

```
Rule1: IF weight['low'] AND age['high'] THEN health['low']
Rule2: IF weight['low'] OR age['low'] THEN health['normal']
```

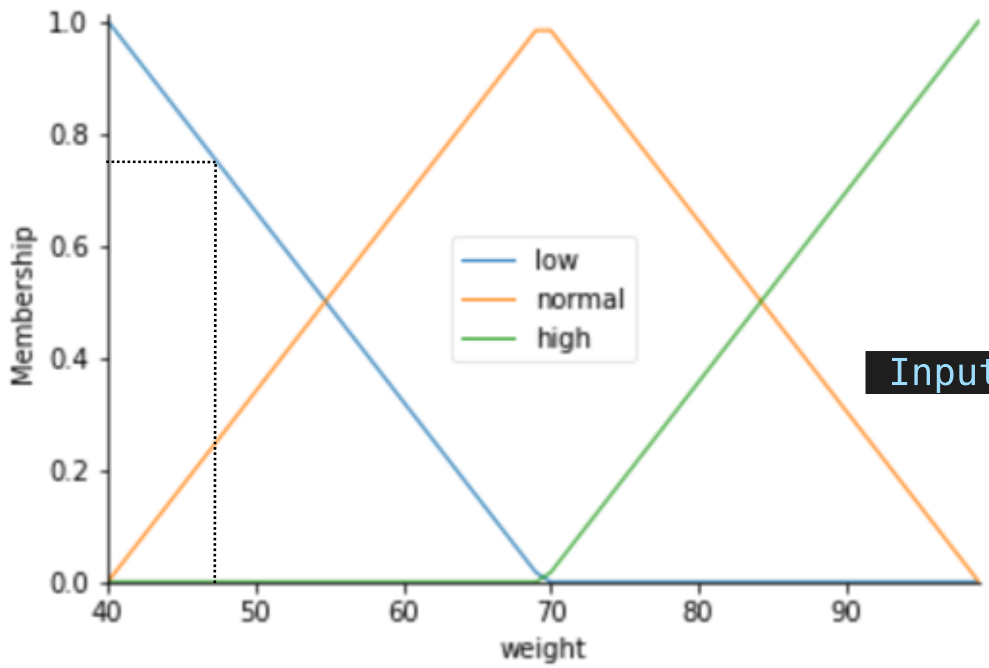
Resposta: Para `Rule1: IF weight['low'] AND age['high'] THEN health['low']`

$T(x,y) = xy$
$S(x,y) = x+y-xy$



`Input['age'] = 52` → `age['high'] = 0.32`

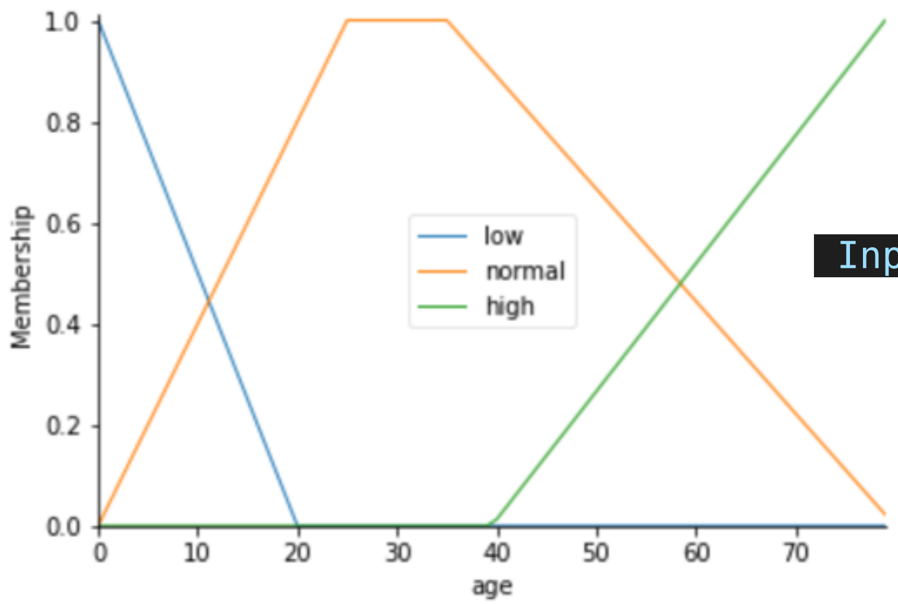
`weight['low'] AND age['high'] = T(0.32,0.77) = 0.32 x 0.77 = 0.25`



`Input['weight'] = 47` → `weight['low'] = 0.77`

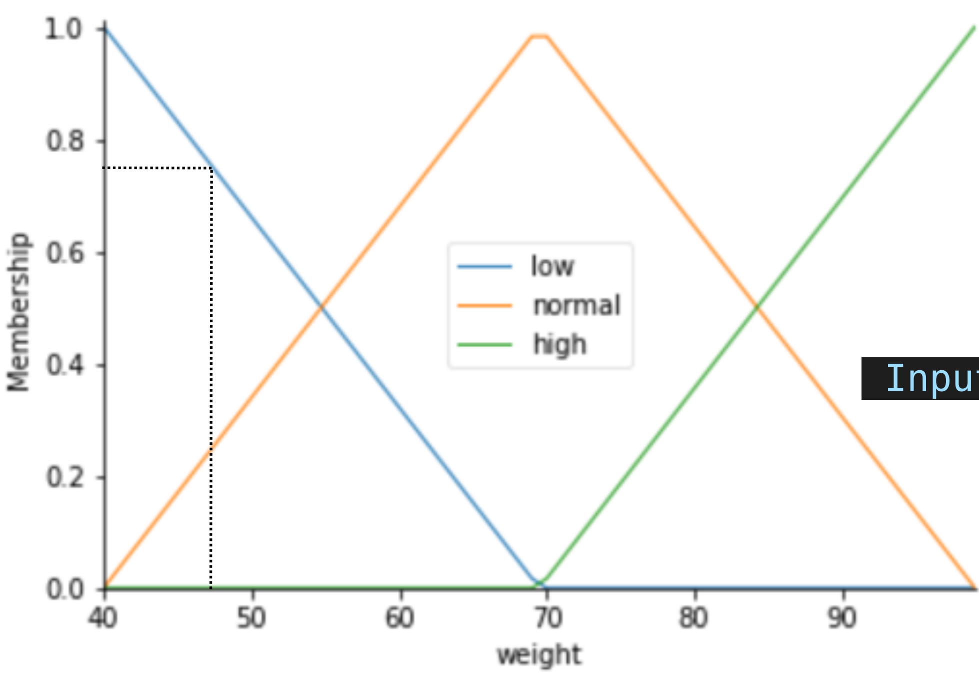
Resposta: Para `Rule2: IF weight['low'] OR age['low'] THEN health['normal']`

$T(x,y) = xy$
$S(x,y) = x+y-xy$



`Input['age'] = 52` → `age['low'] = 0.00`

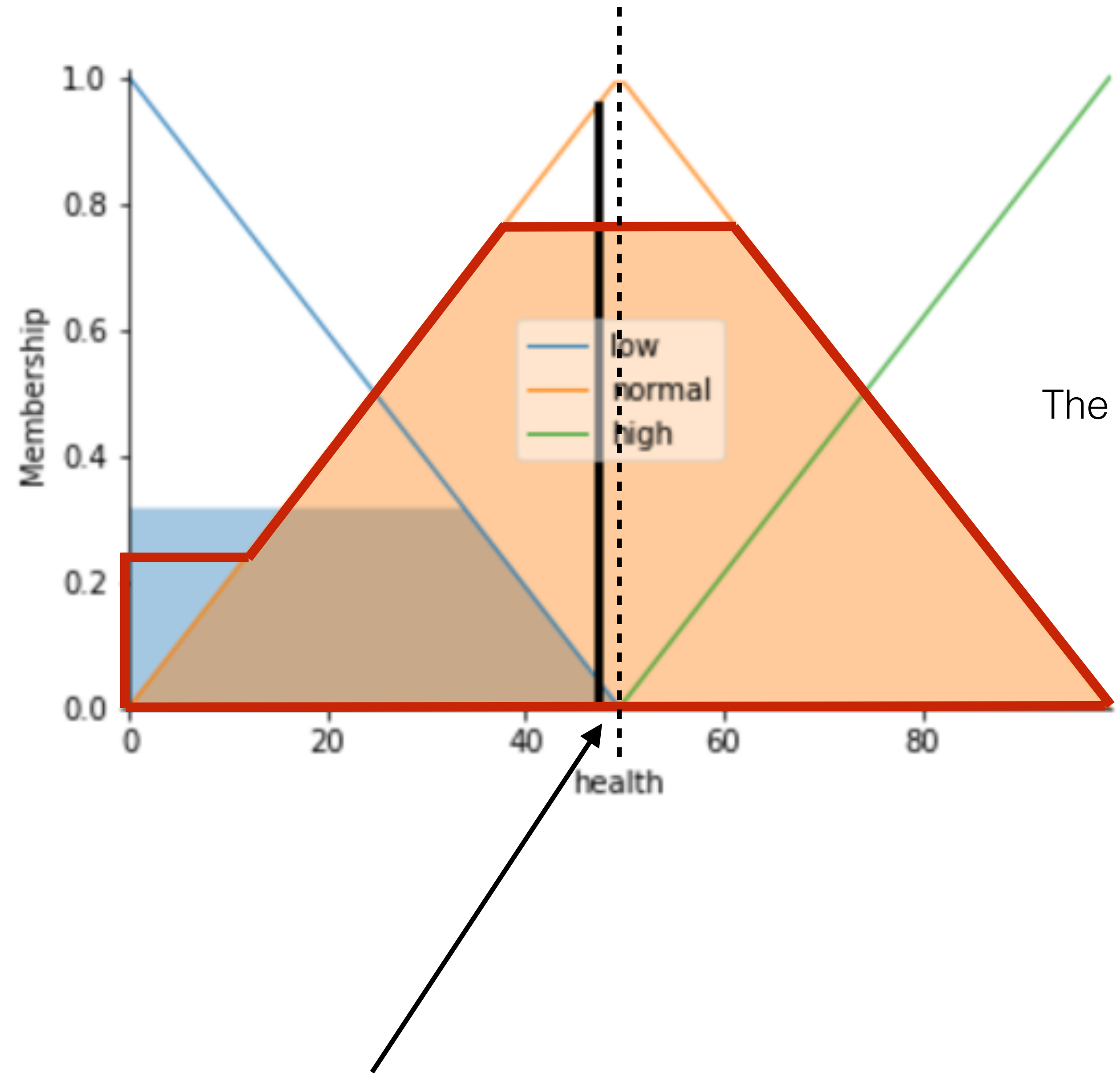
`weight['low'] OR age['low'] = S(0.00,0.77) = 0.00+0.77-0.00 = 0.77`



`Input['weight'] = 47` → `weight['low'] = 0.77`

Resposta: O passo final (defuzzification) é dado por

The blue area is from Rule1



The orange area is from Rule2

The results is between the previous value and the dotted line = 49

Sugeno’s method

- Try to overcome the computational issues
- Rule results are singletons (a value)
- Rule outputs are calculated as the image of a function

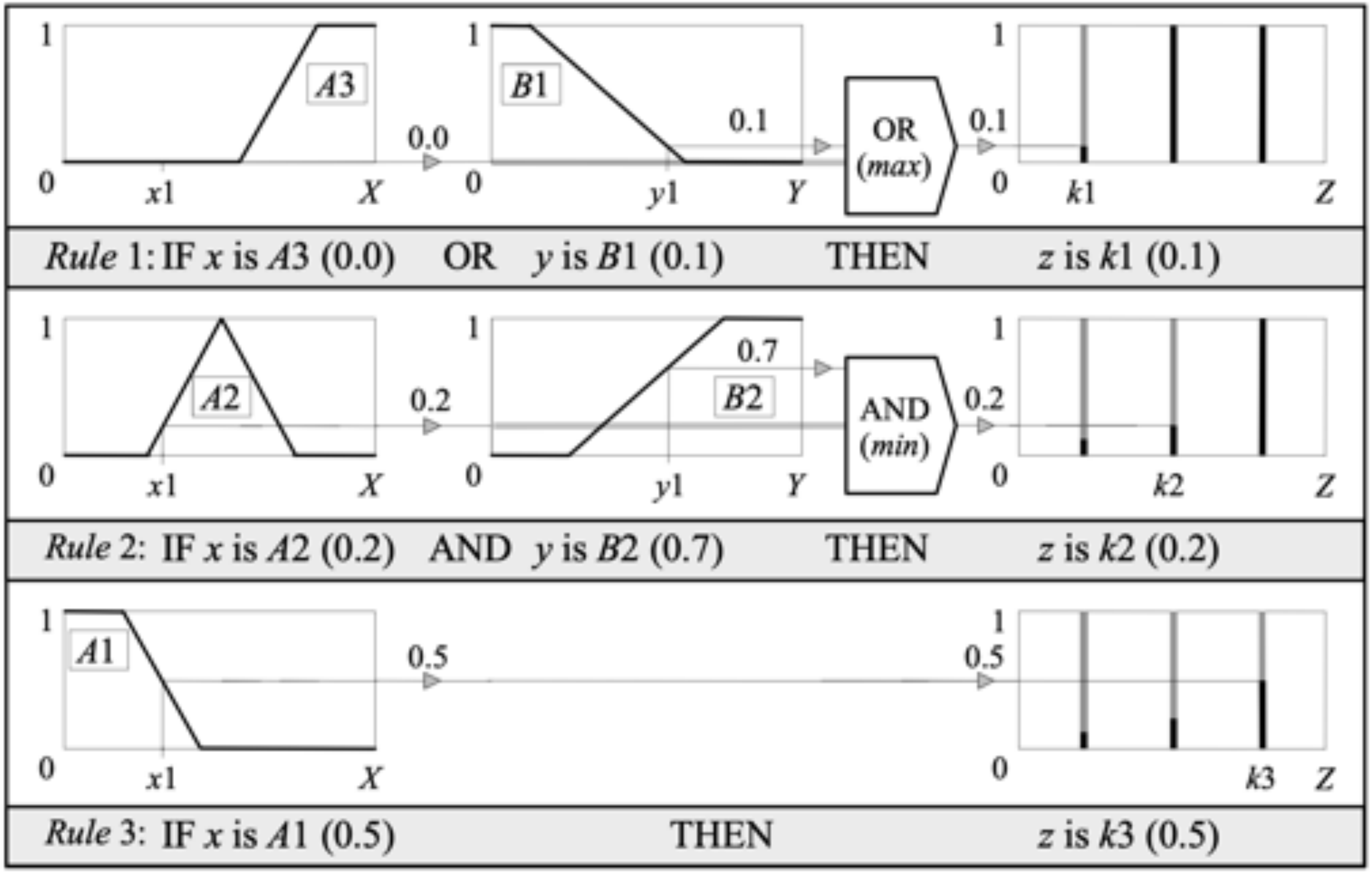
IF x is A **AND** y is B **THEN** z is f(x,y)

- Usually rules are zero order polynomials, i.e. $f(x,y) = k$

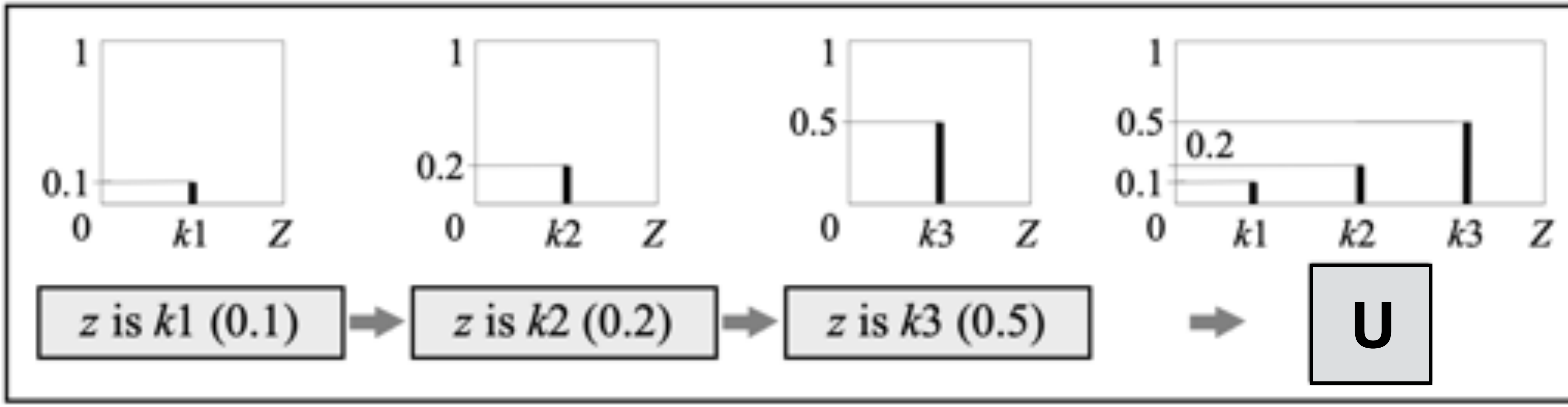
  or are first order polynomials, i.e. $f(x,y) = ax+by+c$

Sugeno's method (zero order)

IF x is A **AND** y is B **THEN** z is K



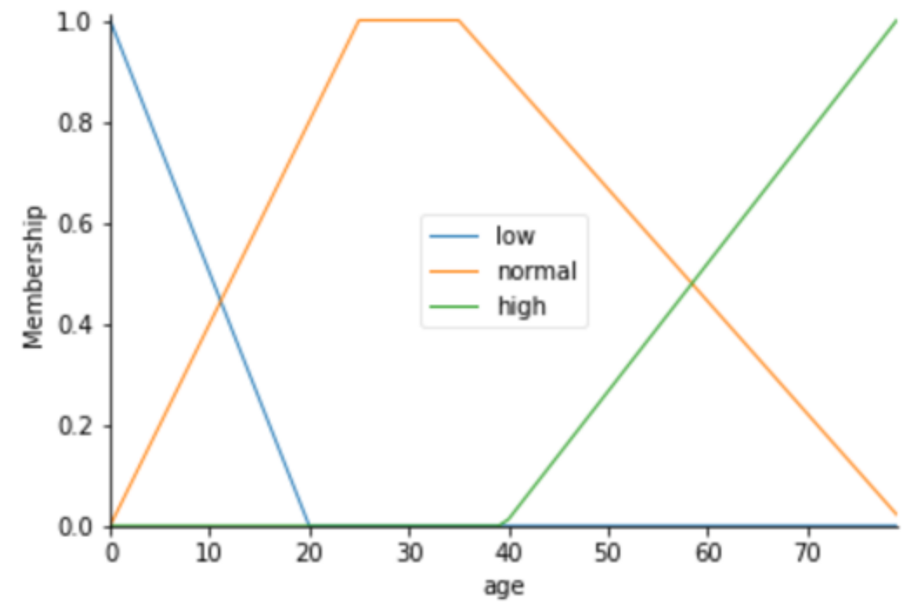
S3. Aggregation of the rule results [Sugeno]

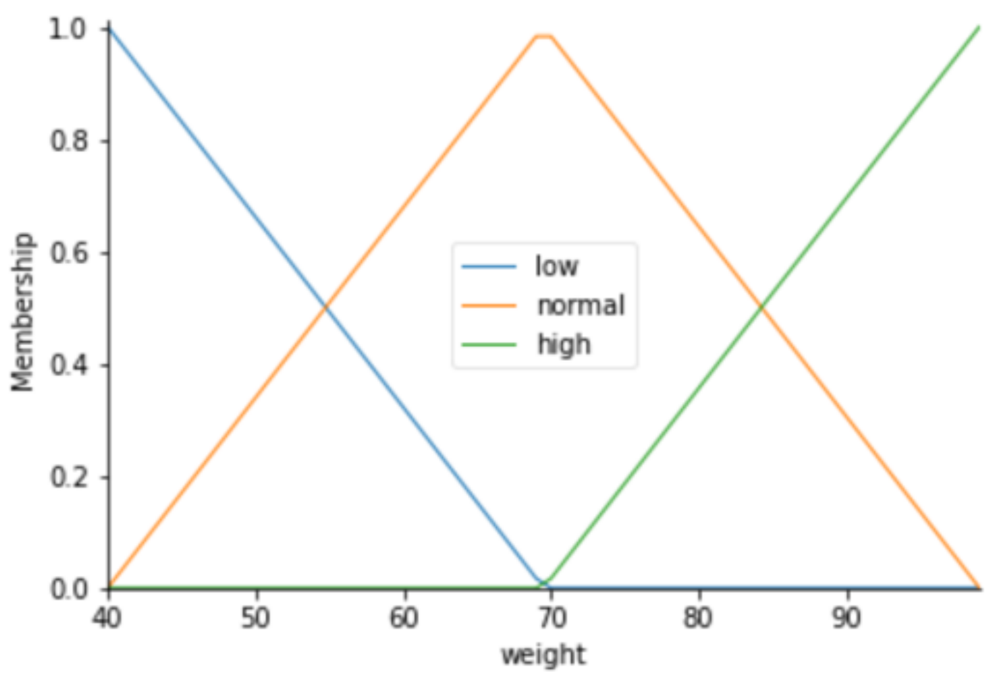


S4. Defuzzification [Sugeno]

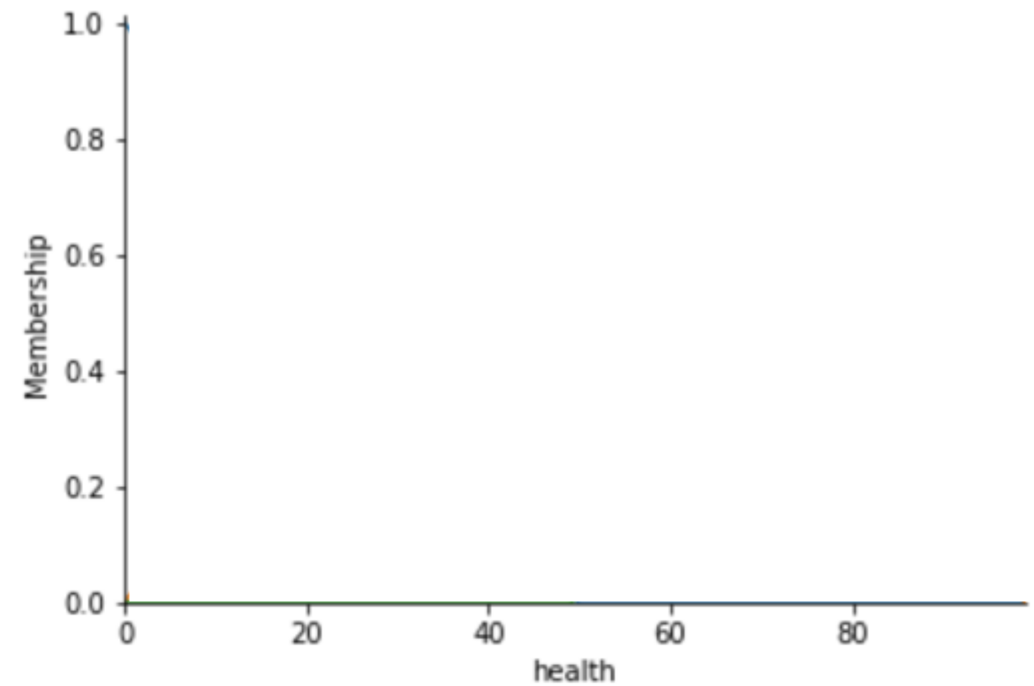
Aggregation of set data: $\text{Mean\_w} = \dfrac{\sum_{i=1}^{N} w_i z_i}{\sum_{i=1}^{N} w_i}$

Exerc: Find the results, for Sugeno’s fuzzy inference.






```
Input['age']    = 52
Input['weight'] = 47
```

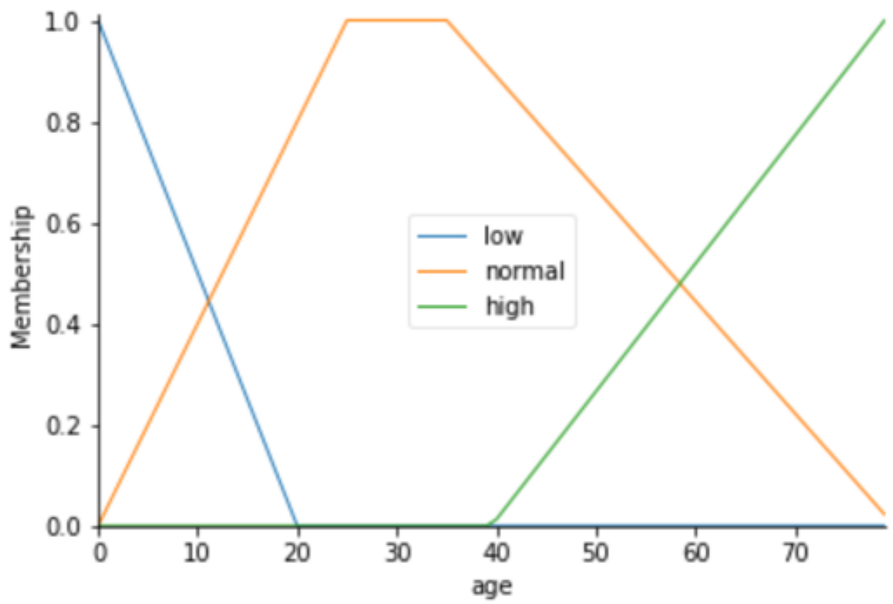


?

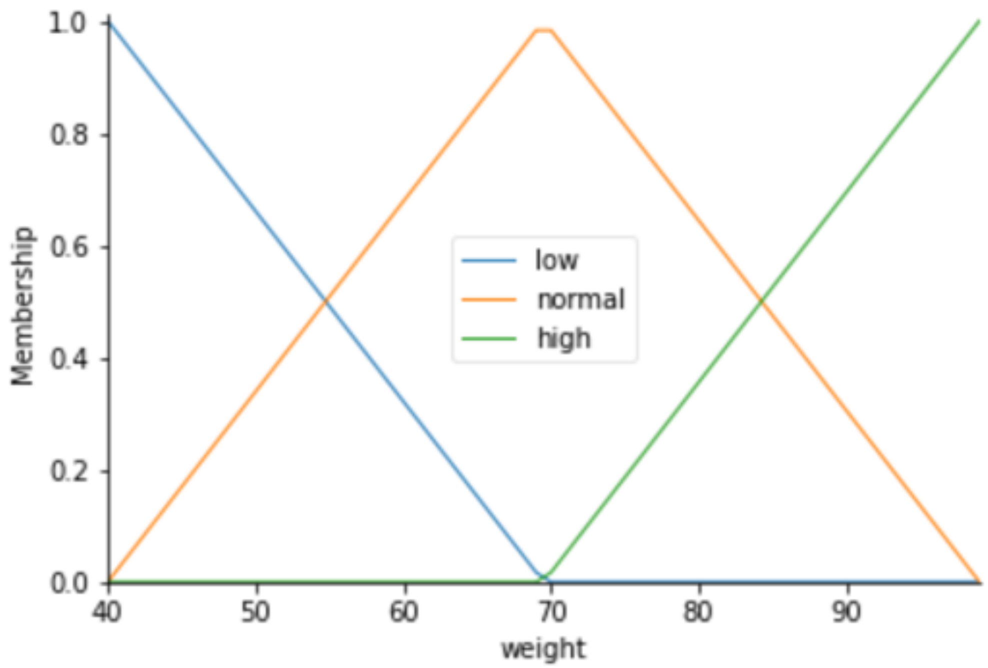
```
Rule1: IF weight['low'] AND age[‘high'] THEN health is k1=100

Rule2: IF weight['low'] OR age[‘low'] THEN health is k2=50
```

Answer: We have

```
Input['age']    = 52
Input['weight'] = 47
```




```
weight['low'] AND age['high'] = min(0.32,0.77) = 0.32
weight['low'] OR age['low'] = max(0.00, 0.77) = 0.77
```



$$\text{Mean\_w} = \frac{\sum_{i=1}^{N} w_i z_i}{\sum_{i=1}^{N} w_i} = \frac{0.32\text{x}80+0.77\text{x}50}{0.82} = 78.17$$

```
Rule1: IF weight['low'] AND age['high'] THEN health is k1=80
Rule2: IF weight['low'] OR age['low'] THEN health is k2=50
```

Q: What is the dependence of the result of the (k1,k2)? For example, (k1,k2)=(50,100). Then R=0.62.

FIS optimization

- Why to optimise?
  - trial and error tuning is laborious
  - it can be impossibly complicated if number of input parameters are large (100’s or more)
  - far too many parameters to tune: number of rules, membership functions, rule consequents
  - arbitrariness of FIS is eliminated
  - if not optimised, the FIS performance is not optimum

- Optimisation methods:
  - Adaptive Neuro-Fuzzy systems
    multilayer perceptron neural networks (ANFIS, FuNeI)
  - evolutionary techniques (genetic algorithms)
  - clustering methods (cluster analysis, Hard C-means, Fuzzy C-means)
  - ...

Extra Class Problem

Use the notebook *FuzzyLogic01.ipynb* and implement your own Sugeno fuzzy inference system.

Next Learning Step

- Fuzzy k-means clustering

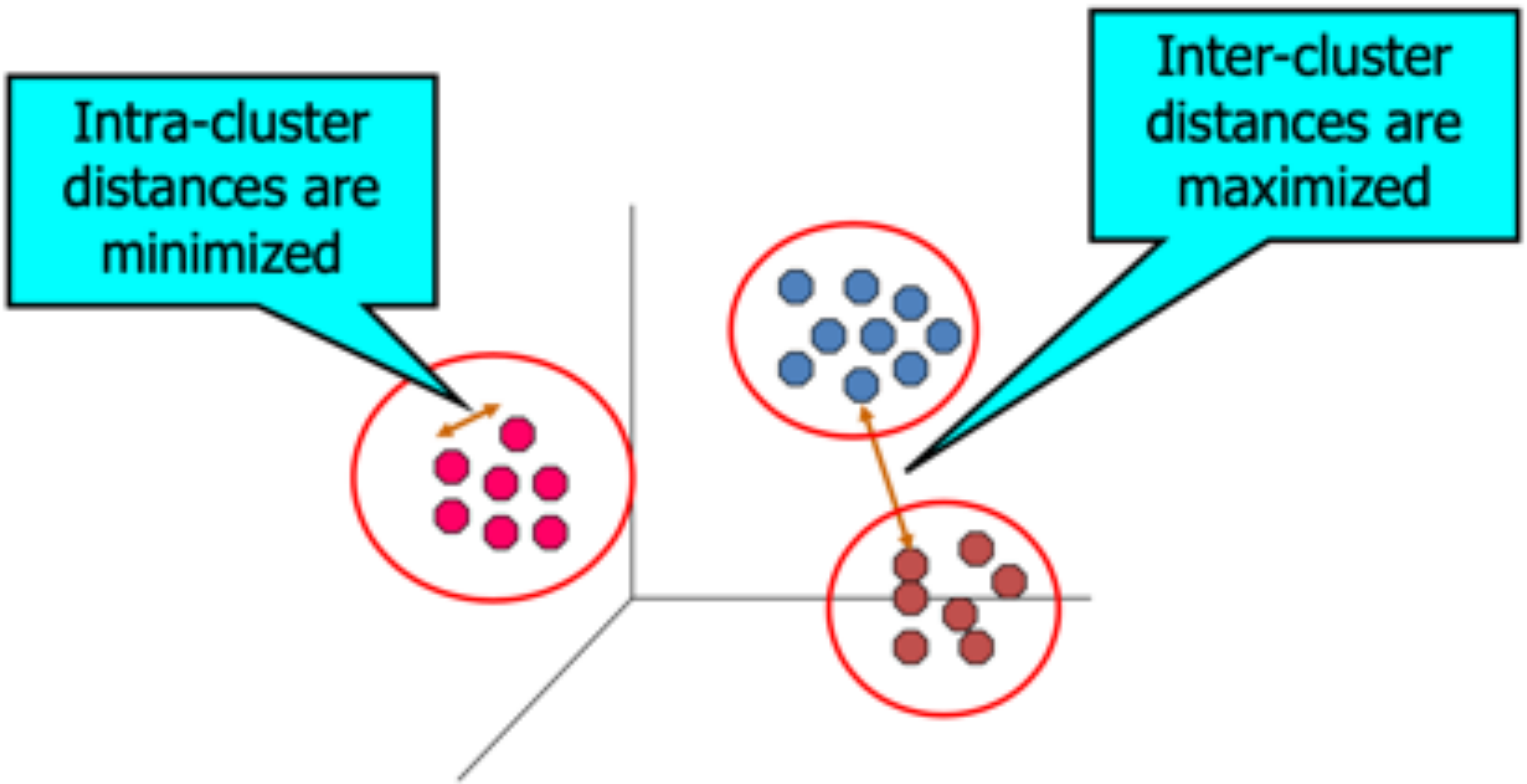
for that

- What is Clustering?

- Classic k-means clustering

What is a Clustering?

- In general a grouping of objects such that the objects in a group (cluster) are similar (or related) to one another and different from (or unrelated to) the objects in other groups



Early applications of cluster analysis

- John Snow, London 1854

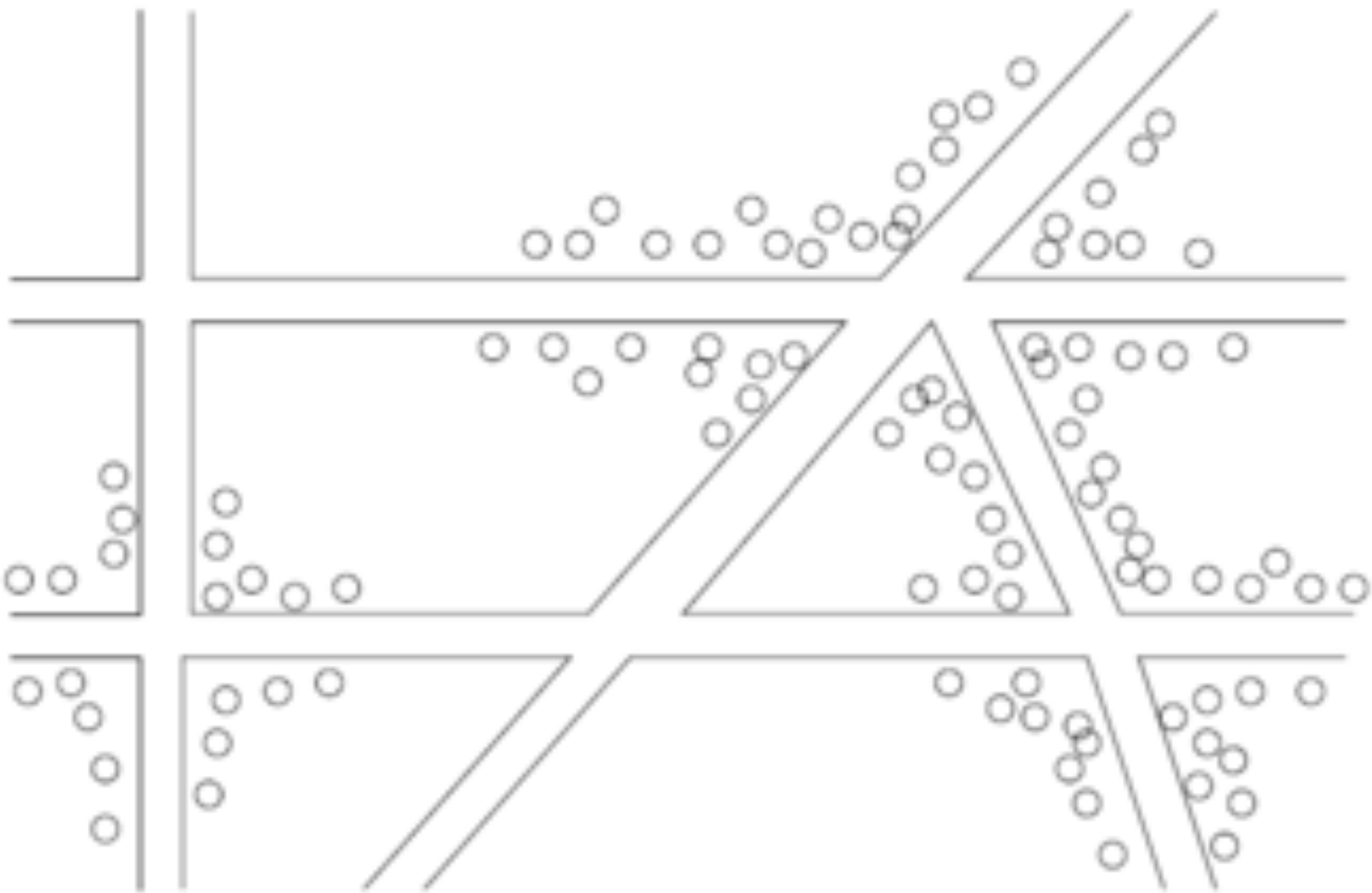
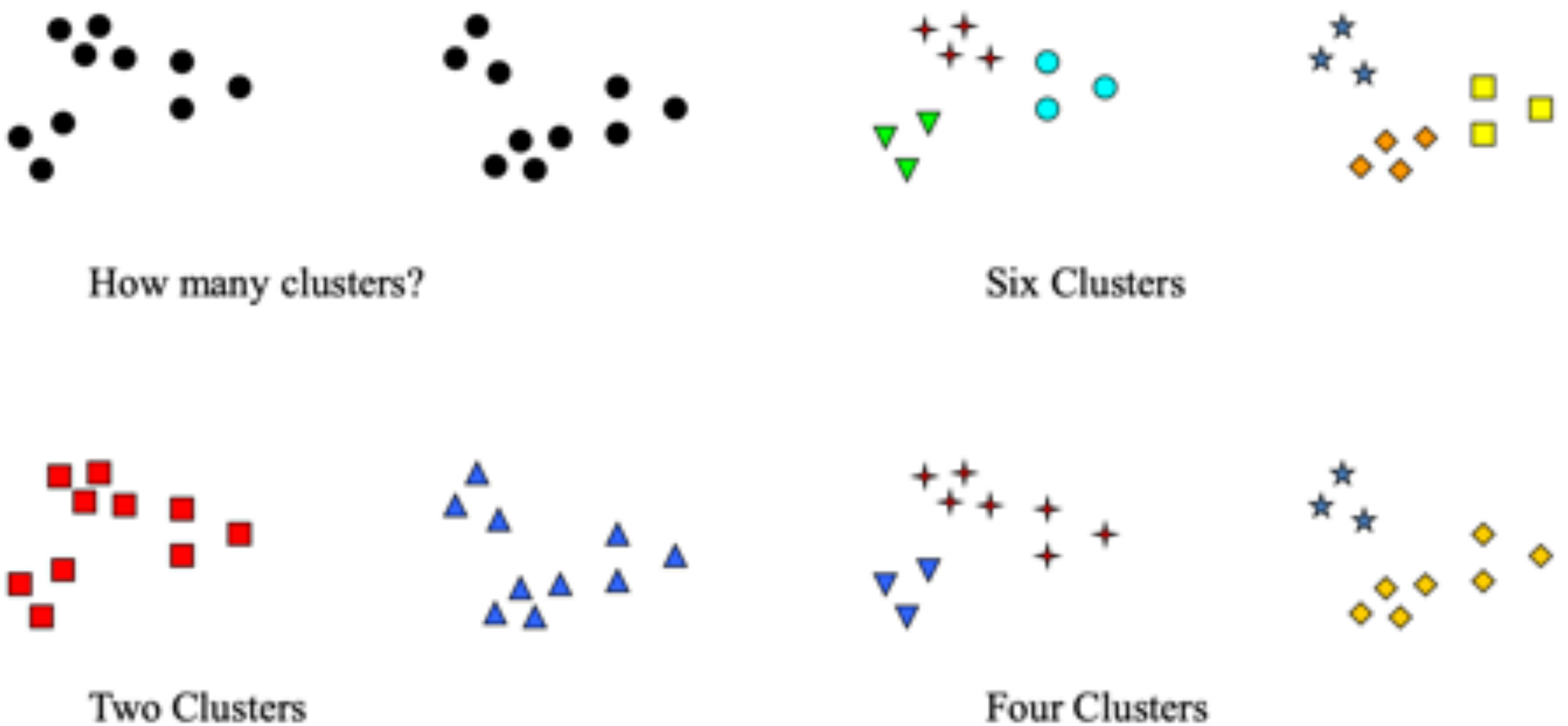


Figure 1.1: Plotting cholera cases on a map of London

Notion of a Cluster can be Ambiguous

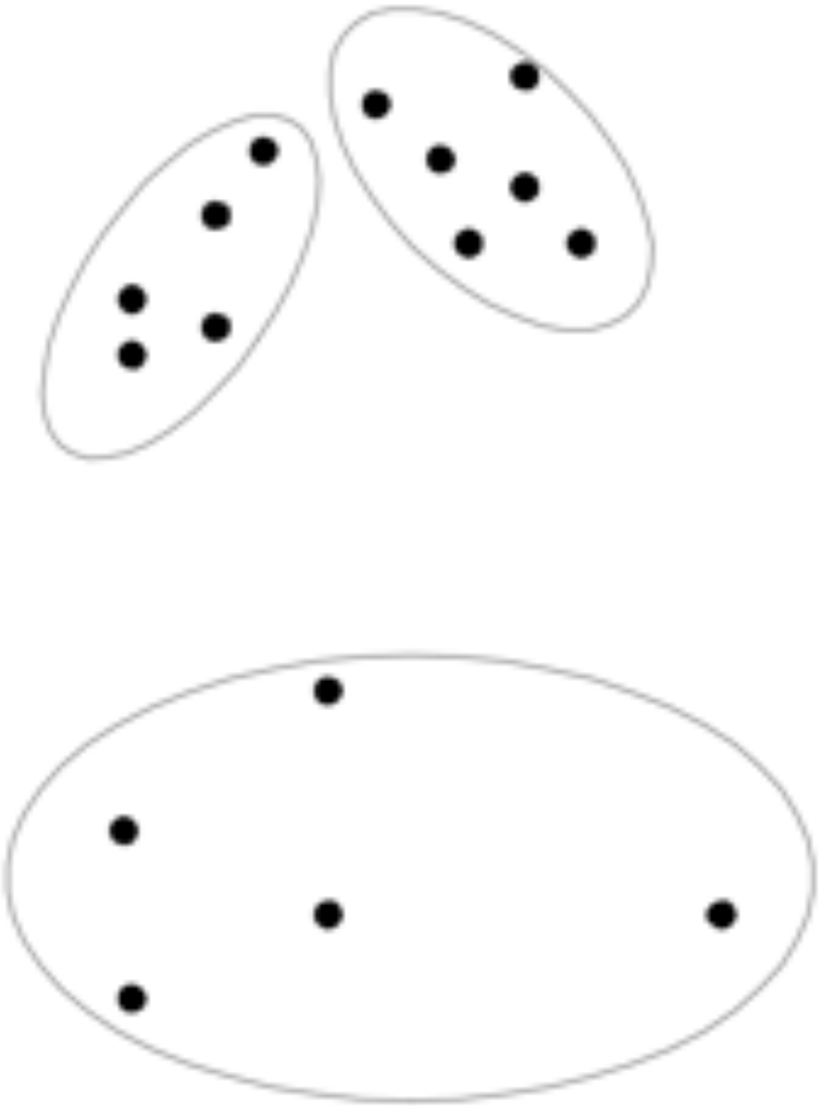


Types of Clusterings

- A clustering is a set of clusters
- Important distinction between hierarchical and partitional sets of clusters
- Partitional Clustering
  - A division data objects into subsets (clusters) such that each data object is in exactly one subset
- Hierarchical clustering
  - A set of nested clusters organized as a hierarchical tree

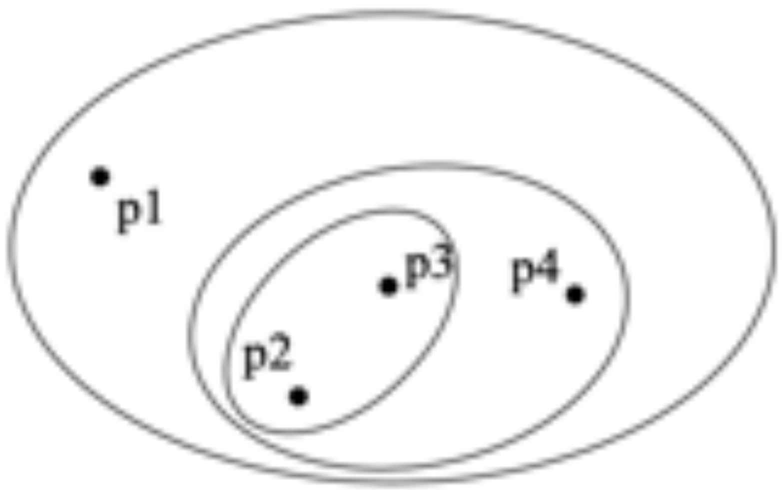
Partitional Clustering

Original Points



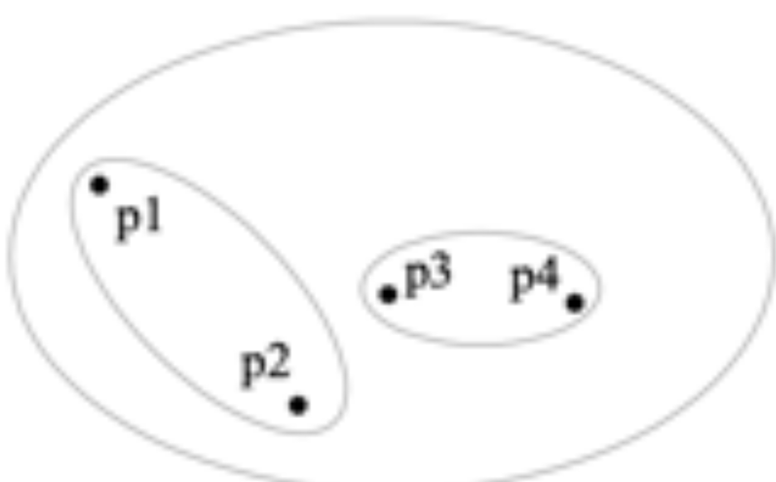
A Partitional Clustering

Hierarchical Clustering



Traditional Hierarchical Clustering

Traditional Dendrogram



Non-traditional Hierarchical Clustering

Non-traditional Dendrogram

Other types of clustering

- Exclusive (or non-overlapping) versus non-exclusive (or overlapping)
  - In non-exclusive clusterings, points may belong to multiple clusters.
    - Points that belong to multiple classes, or ‘border’ points
- Fuzzy (or soft) versus non-fuzzy (or hard)
  - In fuzzy clustering, a point belongs to every cluster with some weight between 0 and 1
- Partial versus complete
  - In some cases, we only want to cluster some of the data

Types of Clusters: Objective Function

- Clustering as an optimization problem
  - Finds clusters that minimize or maximize an objective function.
  - Enumerate all possible ways of dividing the points into clusters and evaluate the `goodness' of each potential set of clusters by using the given objective function. (NP Hard)
  - Can have global or local objectives.
    - Hierarchical clustering algorithms typically have local objectives
    - Partitional algorithms typically have global objectives
  - A variation of the global objective function approach is to fit the data to a parameterized model.
    - The parameters for the model are determined from the data, and they determine the clustering
    - E.g., Mixture models assume that the data is a 'mixture' of a number of statistical distributions.

K-means Clustering

- Partitional clustering approach
- Each cluster is associated with a centroid (center point)
- Each point is assigned to the cluster with the closest centroid
- Number of clusters, K, must be specified
- The objective is to minimize the sum of distances of the points to their respective centroid

K-means Clustering

- **Problem:** Given a set X of n points in a d-dimensional space and an integer K group the points into K clusters $C = \{C_1, C_2, \ldots, C_k\}$ such that

$$Cost(C) = \sum_{i=1}^{k} \sum_{x \in C_i} dist(x, c)$$

is minimized, where c_i is the centroid of the points in cluster C_i

K-means Clustering

- Most common definition is with euclidean distance, minimizing the Sum of Squares Error (SSE) function
  - Sometimes K-means is defined like that

- **Problem:** Given a set X of n points in a d-dimensional space and an integer K group the points into K clusters $C = \{C_1, C_2, \ldots, C_k\}$ such that

$$Cost(C) = \sum_{i=1}^{k} \sum_{x \in C_i} (x - c_i)^2$$

is minimized, where c_i is the mean of the points in cluster C_i

Sum of Squares Error (SSE)

Complexity of the k-means problem

- NP-hard if the dimensionality of the data is at least 2 ($d>=2$)
  - Finding the best solution in polynomial time is infeasible
- For $d=1$ the problem is solvable in polynomial time (how?)
- A simple iterative algorithm works quite well in practice

K-means Algorithm

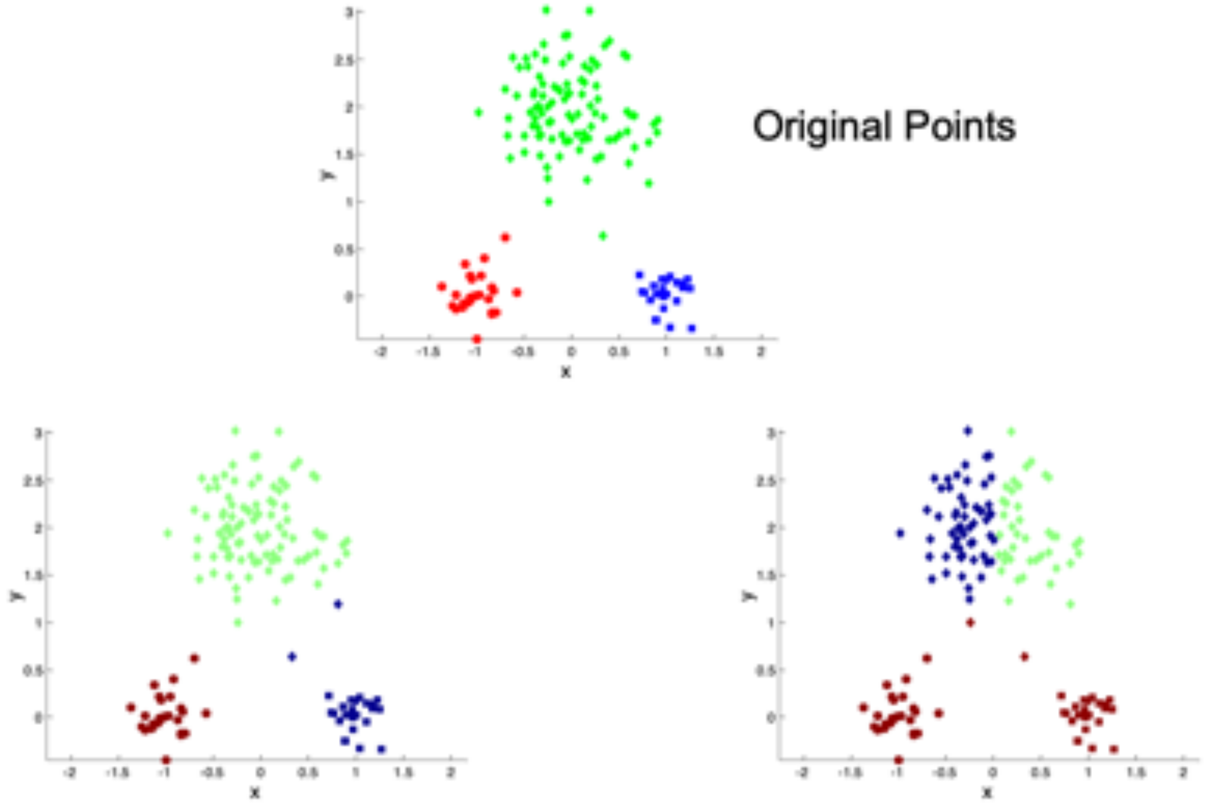
- Also known as Lloyd's algorithm.
- K-means is sometimes synonymous with this algorithm

Select K points as the initial centroids.
repeat
    Form K clusters by assigning all points to the closest centroid.
    Recompute the centroid of each cluster.
until The centroids don't change

K-means Algorithm – Initialization

- Initial centroids are often chosen randomly.
  - Clusters produced vary from one run to another.

Two different K-means Clusterings



Optimal Clustering

Sub-optimal Clustering

Dealing with Initialization

- Do multiple runs and select the clustering with the smallest error

- Select original set of points by methods other than random . E.g., pick the most distant (from each other) points as cluster centers (K-means++ algorithm)

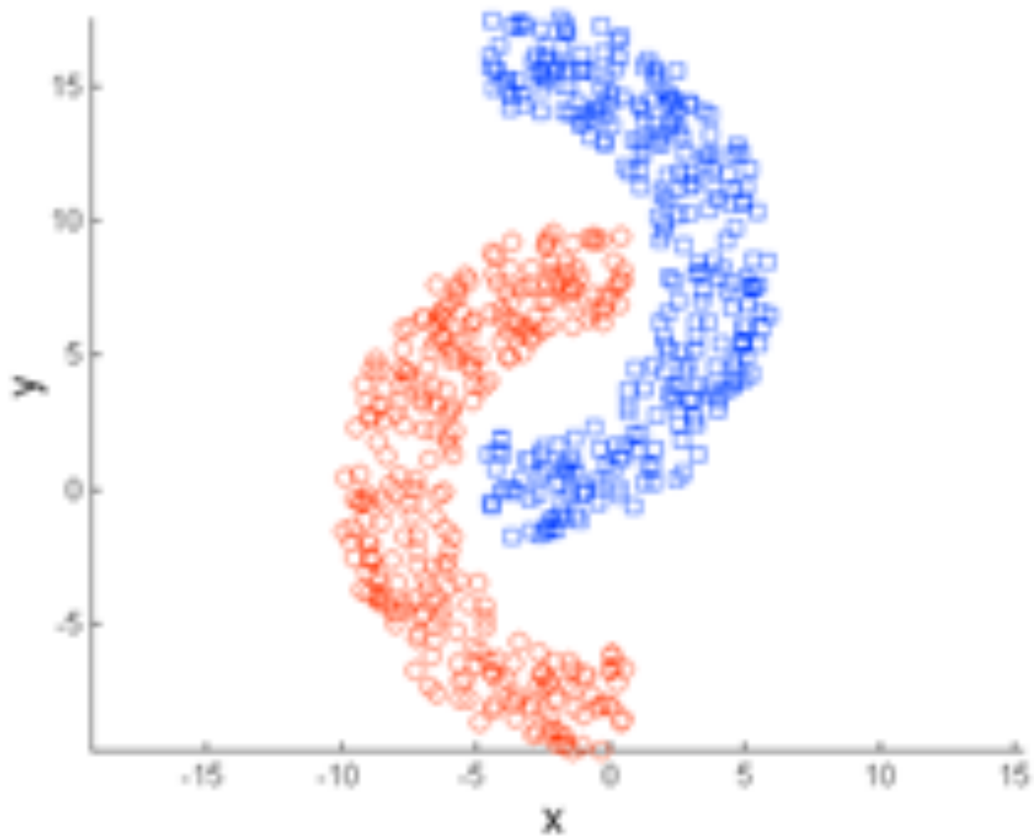
K-means Algorithm – Centroids

- The centroid depends on the distance function
  - The minimizer for the distance function
- ‘Closeness’ is measured by Euclidean distance (SSE), cosine similarity, correlation, etc.
- Centroid:
  - The mean of the points in the cluster for SSE, and cosine similarity
  - The median for Manhattan distance.
- Finding the centroid is not always easy
  - It can be an NP-hard problem for some distance functions
    - E.g., median form multiple dimensions

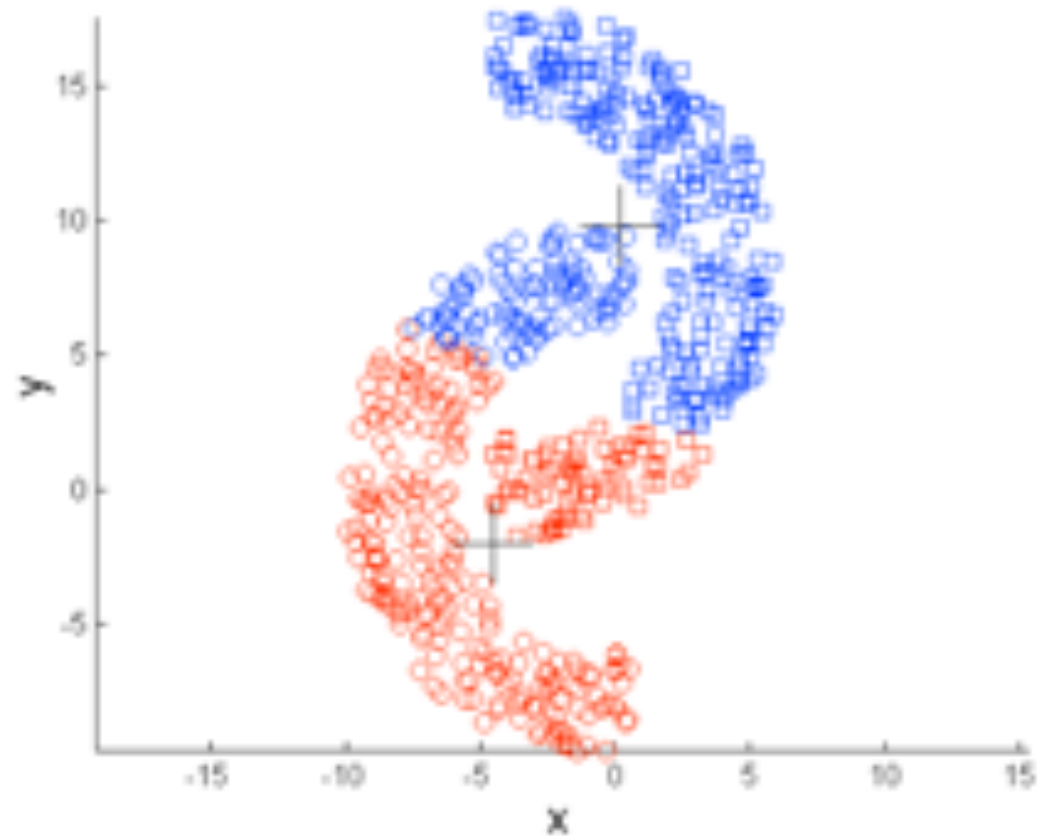
K-means Algorithm – Convergence

- K-means will converge for common similarity measures mentioned above.
  - Most of the convergence happens in the first few iterations.
  - Often the stopping condition is changed to 'Until relatively few points change clusters'
- Complexity is O(n * K * I * d)
  - n = number of points, K = number of clusters, I = number of iterations, d = dimensionality
- In general a fast and efficient algorithm

Limitations of K-means: Non-globular Shapes

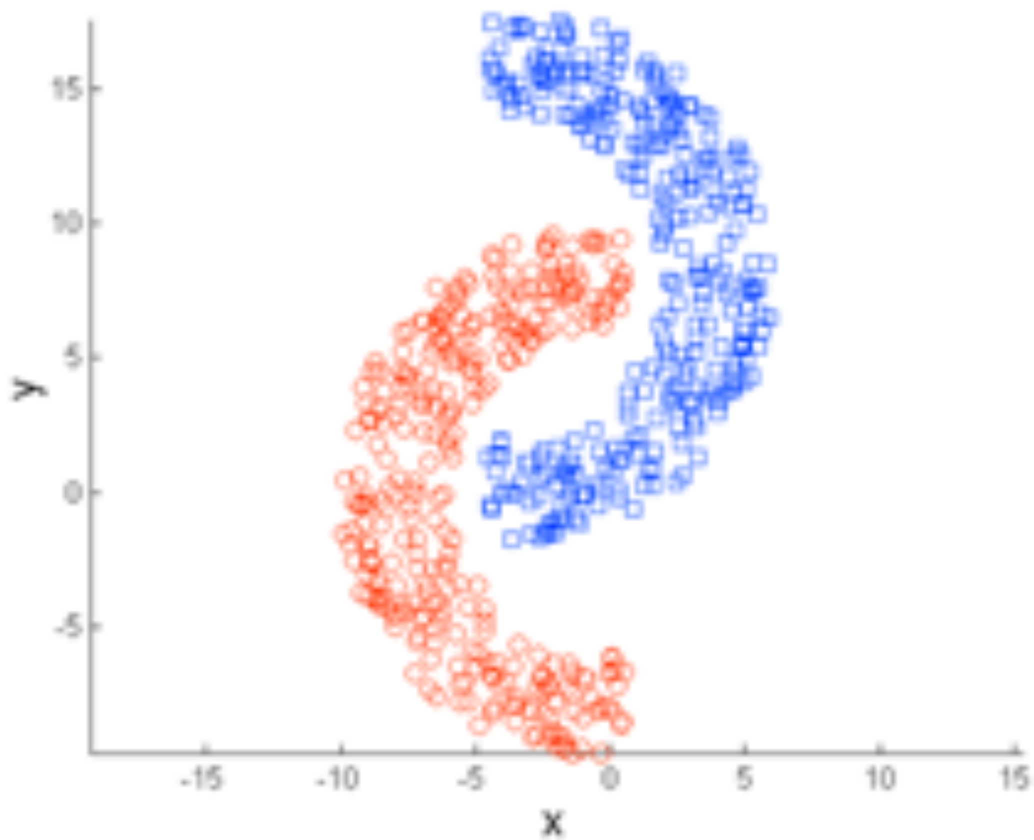


Original Points

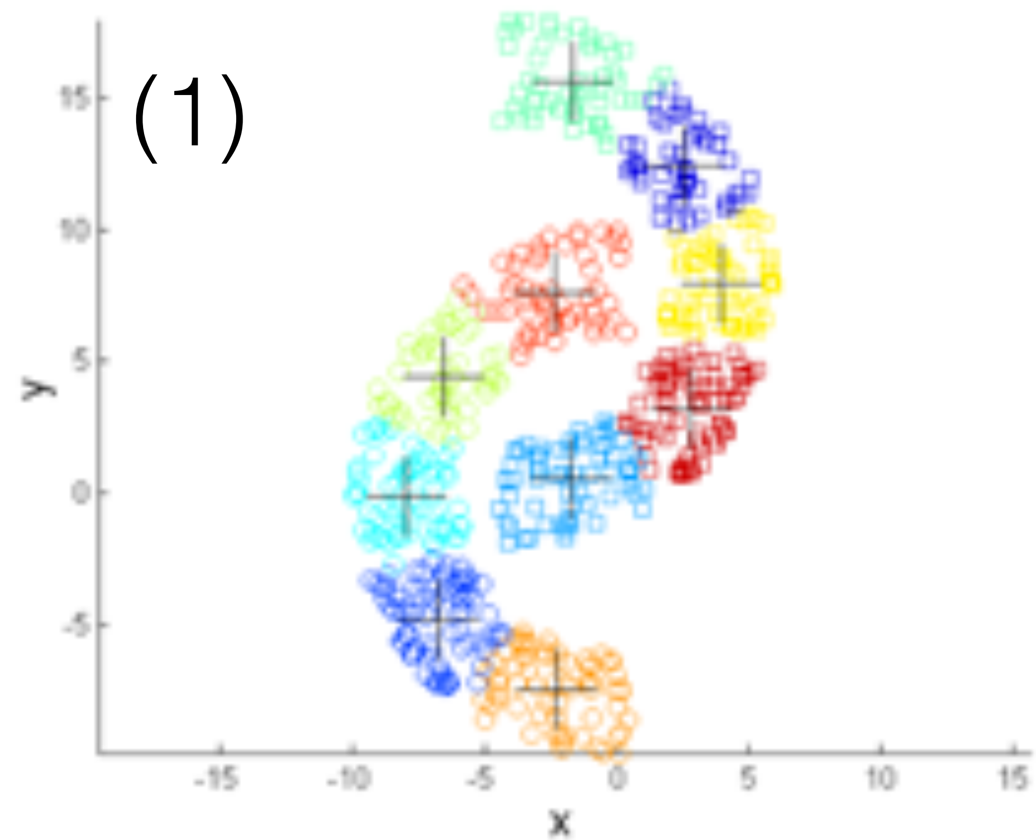


K-means (2 Clusters)

Overcoming K-means Limitations



Original Points



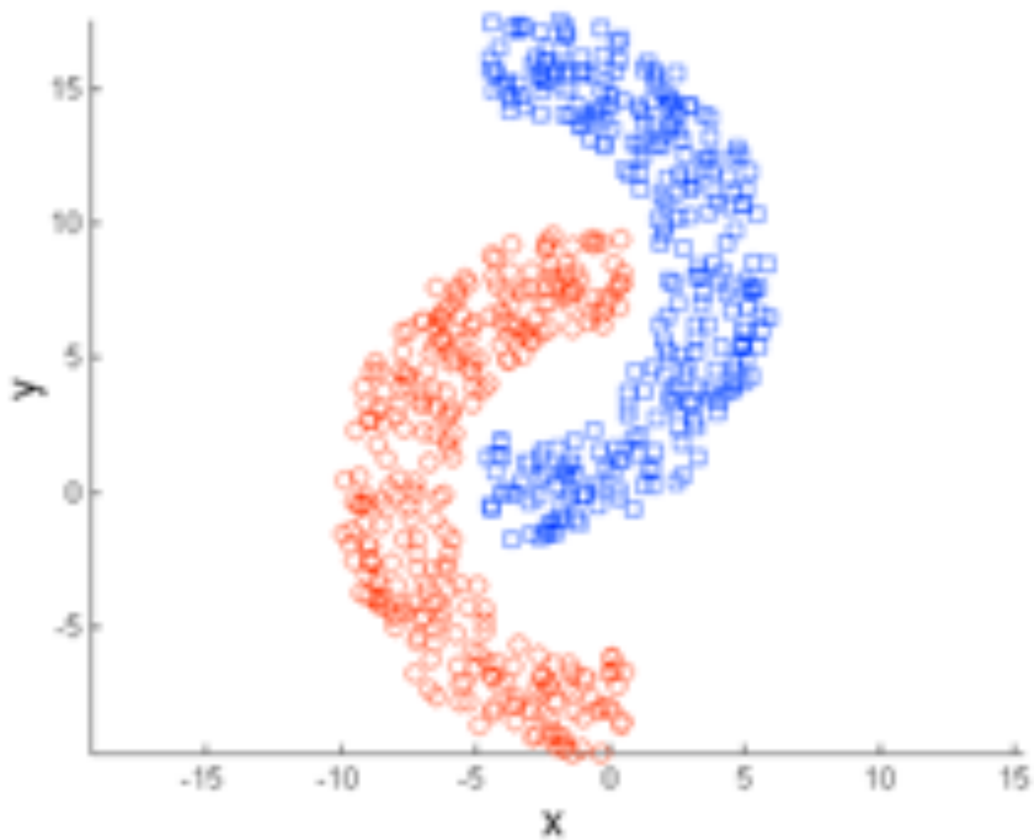
K-means Clusters

Overcoming K-means Limitations

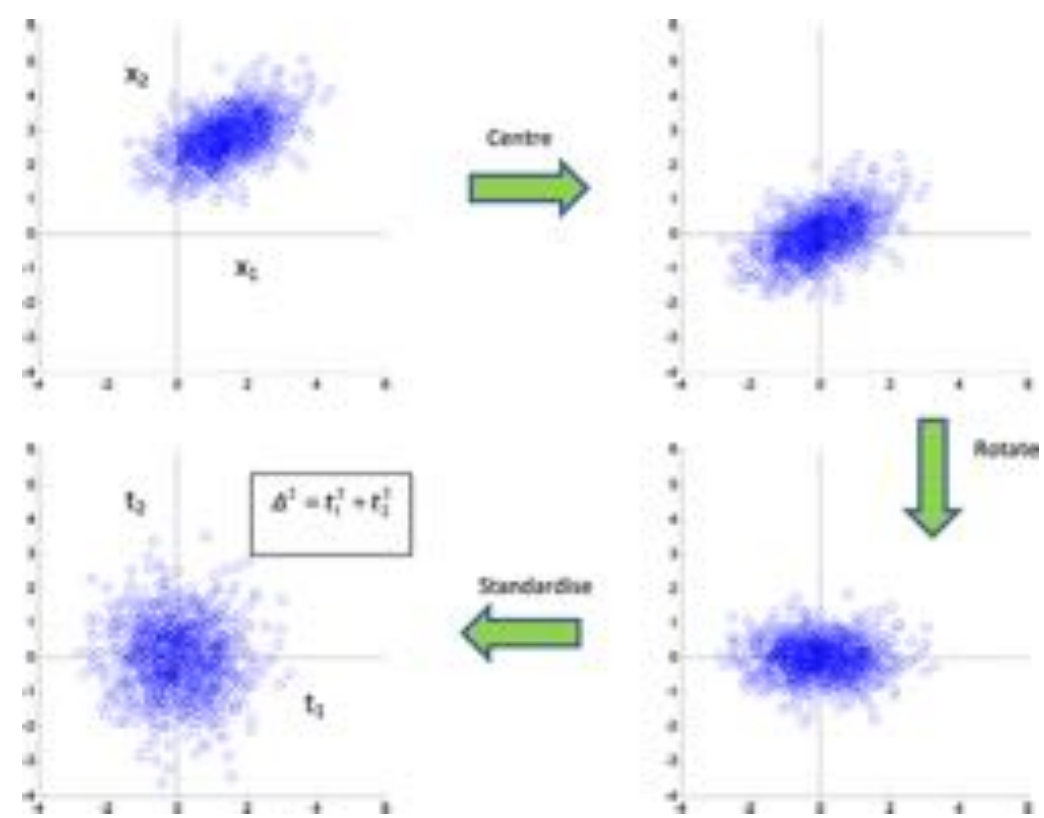
(2) Changing the distance/similarity
e.g. Mahalanobis distance

$$D_M(\vec{x}) = \sqrt{(\vec{x} - \vec{\mu})^\mathsf{T} \mathbf{S}^{-1} (\vec{x} - \vec{\mu})}.$$

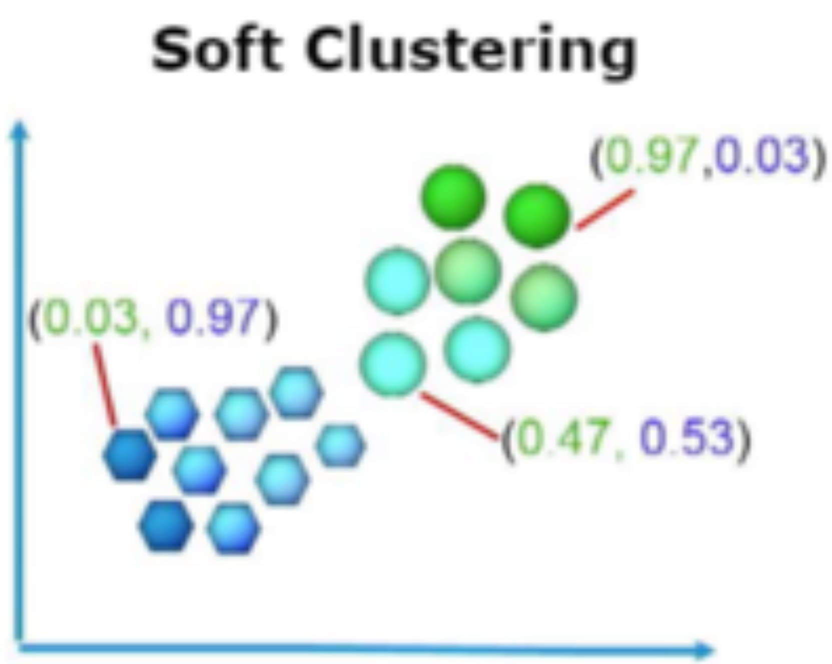
$$\operatorname{cov}[X_i, X_j] = \mathrm{E}[(X_i - \mathrm{E}[X_i])(X_j - \mathrm{E}[X_j])]$$



Original Points

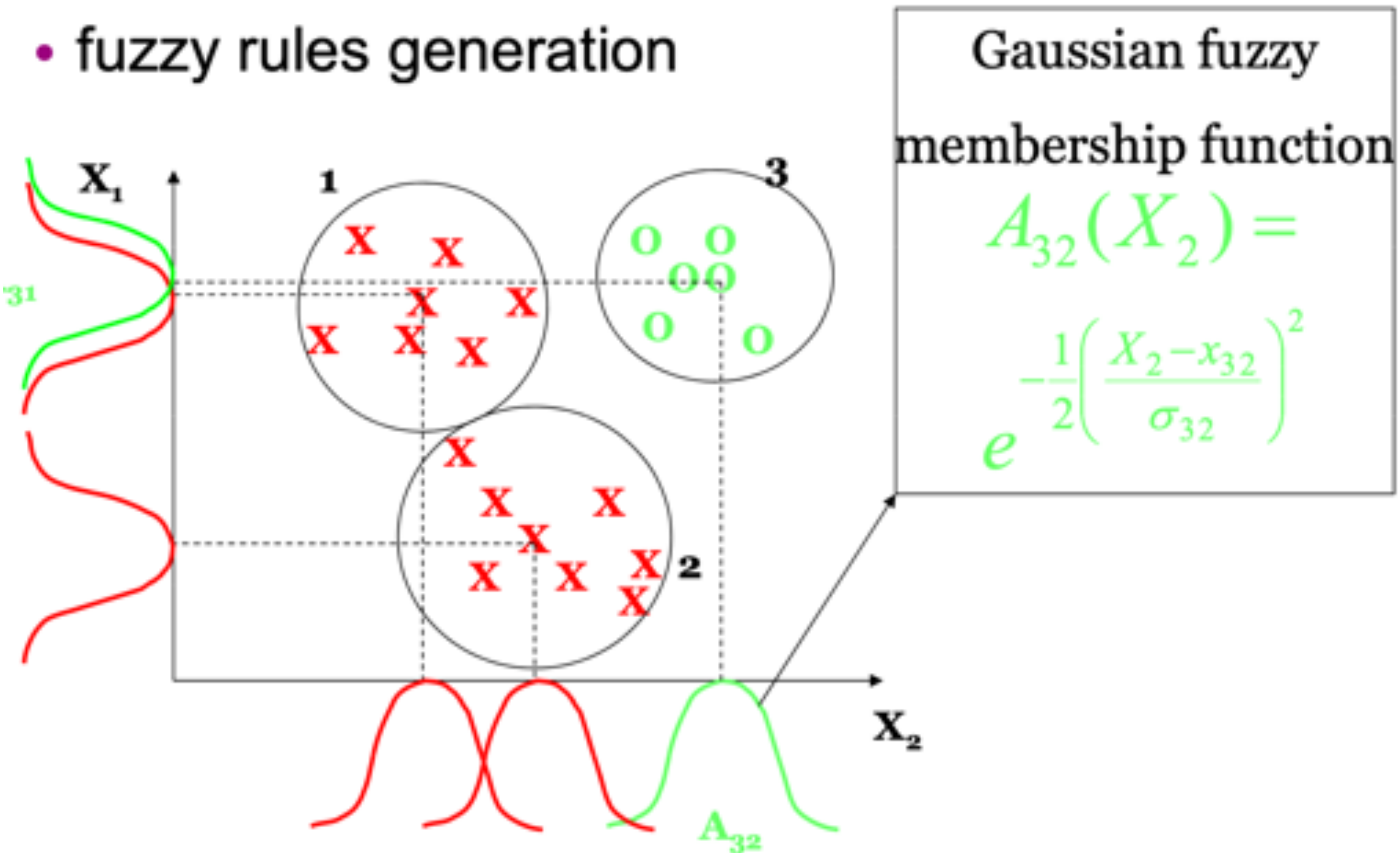


Fuzzy Clustering (idea)



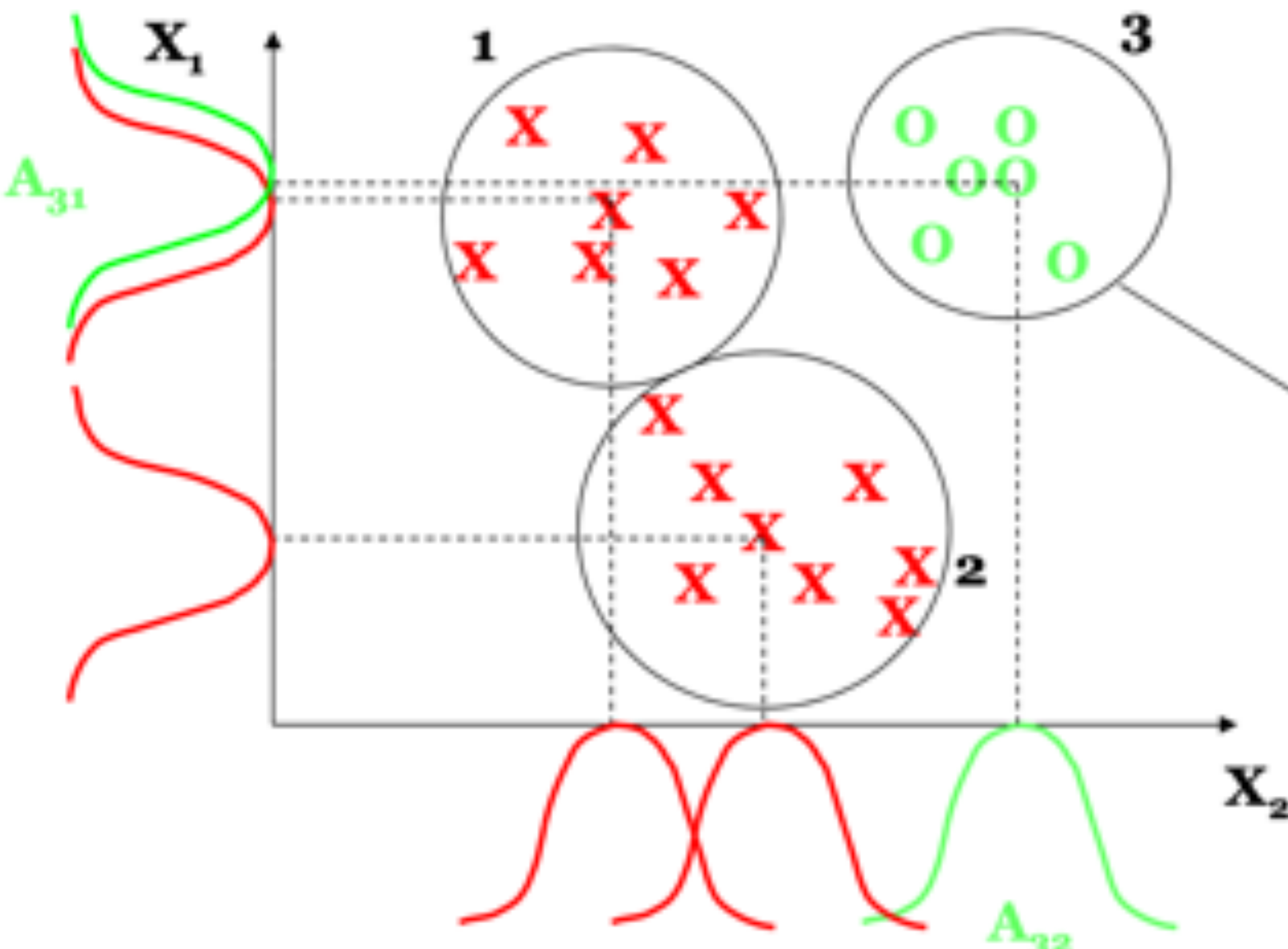
Fuzzy Clustering (idea)

- fuzzy rules generation



Fuzzy Clustering (idea)

- fuzzy rules generation



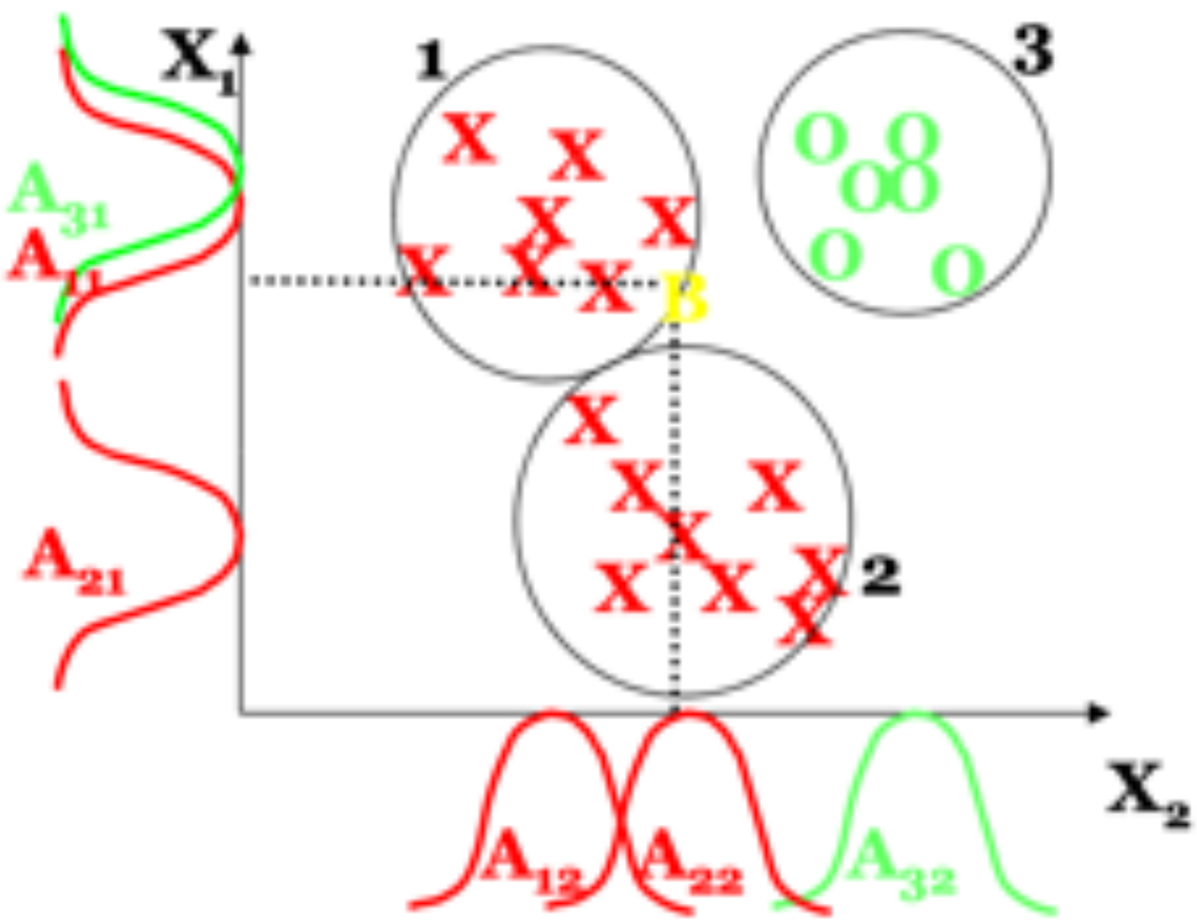
If X_1 is A_{31} and X_2 is A_{32}

Then

Class is O

Fuzzy Clustering (idea)

- Classification of an unseen vector B



If X_1 is A_{11} and X_2 is A_{12} Then Class is X

$A_{11}(X_1) * A_{12}(X_2) = 0.8 * 0.2 = \boxed{0 .16}$

If X_1 is A_{21} and X_2 is A_{22} Then Class is X

$A_{21}(X_1) * A_{22}(X_2) = 0.01 * 0.9 = 0.009$

If X_1 is A_{31} and X_2 is A_{32} Then Class is O

$A_{31}(X_1) * A_{32}(X_2) = 0.1 * 0.01 = 0.001$

B belongs to X

FCM

min $J_{\mathrm{FCM}}(U,V)=\sum_{j=1}^{N}\sum_{i=1}^{C}(u_{ij})^q(d_{ji})^2$ with

$(d_{ji})^2=\|x_j-v_i\|^2$
$u_{ij}=$ Membership level x_j in cluster to i;
$N=$ Total data;
$C=$ Total cluster;
$q=$ Fuzzifier parameter, $q>1$.

Step 1. Initialization Vector centroid, v_i (prototypes).

Step 2. Calculate the distance between feature vector (X) and the centroid vector $(V)[X\to V]$. Feature vectors with the closest distance to one of the centroid vectors then expressed as a cluster member.

Step 3. Calculate membership level of all feature vectors in all clusters by using the formula:

$$u_{ij}=\frac{1}{\sum_{k=1}^{K}\left[\frac{(d_{ji})^2}{(d_{jk})^2}\right]^{1/(q-1)}}=\frac{\left[\frac{1}{(d_{ji})^2}\right]^{1/(q-1)}}{\sum_{k=1}^{K}\left[\frac{1}{(d_{jk})^2}\right]^{1/(q-1)}}. \quad (3)$$

FCM

$$\min \; J_{\text{FCM}}(U,V) = \sum_{j=1}^{N}\sum_{i=1}^{C}(u_{ij})^q (d_{ji})^2 \quad \text{with}$$

$(d_{ji})^2 = \|x_j - v_i\|^2$

u_{ij} = Membership level x_j in cluster to i;
N = Total data;
C = Total cluster;
q = Fuzzifier parameter, $q > 1$.

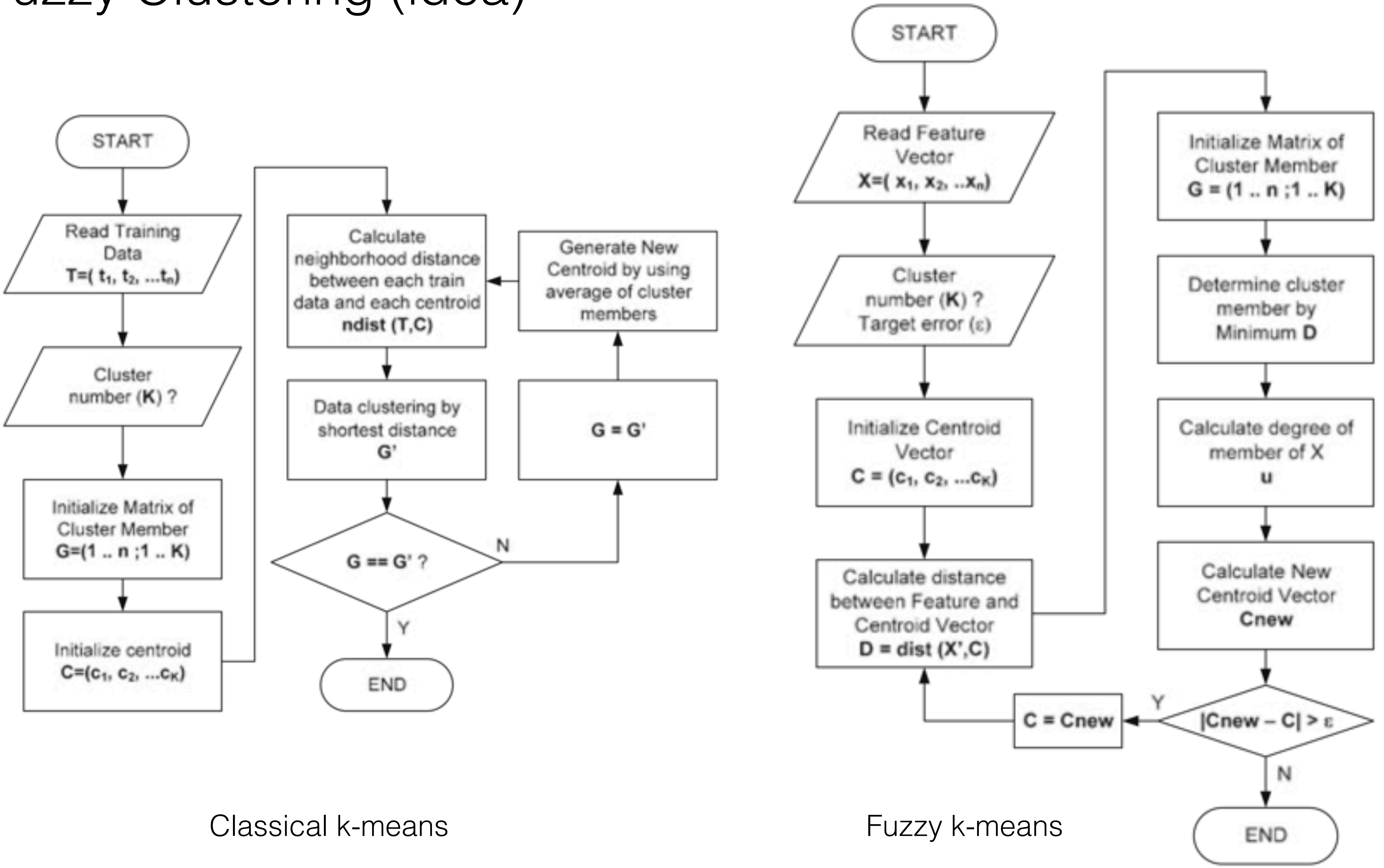
Step 4. Calculate new centroid using Eq. (4).

$$\hat{V}_i = \frac{\sum_{j=1}^{N}(u_{ij})^q X_j}{\sum_{j=1}^{N}(u_{ij})^q}. \tag{4}$$

Step 5. Recalculate the step 4, $u_{ij} \rightarrow \hat{u}_{ij}$ If, $\max_{ij} |u_{ij} - \hat{u}_{ij}| < \varepsilon$, where ε is termination criteria between 0 and 1. Then, the iteration process is stopped. If not go back to step 5.

U = Fuzzy datasets K-partition;
V = Set of prototype centroid;
$V = \{v_1, v_2, \cdots, v_C\} \subset R^P$;

Fuzzy Clustering (idea)



Classical k-means

Fuzzy k-means

Fuzzy Clustering (idea)

Evaluation of features — For the partitioning process, both versions use the same measures, entropy and information gain, in order to select the features to be used in the test nodes of the tree;

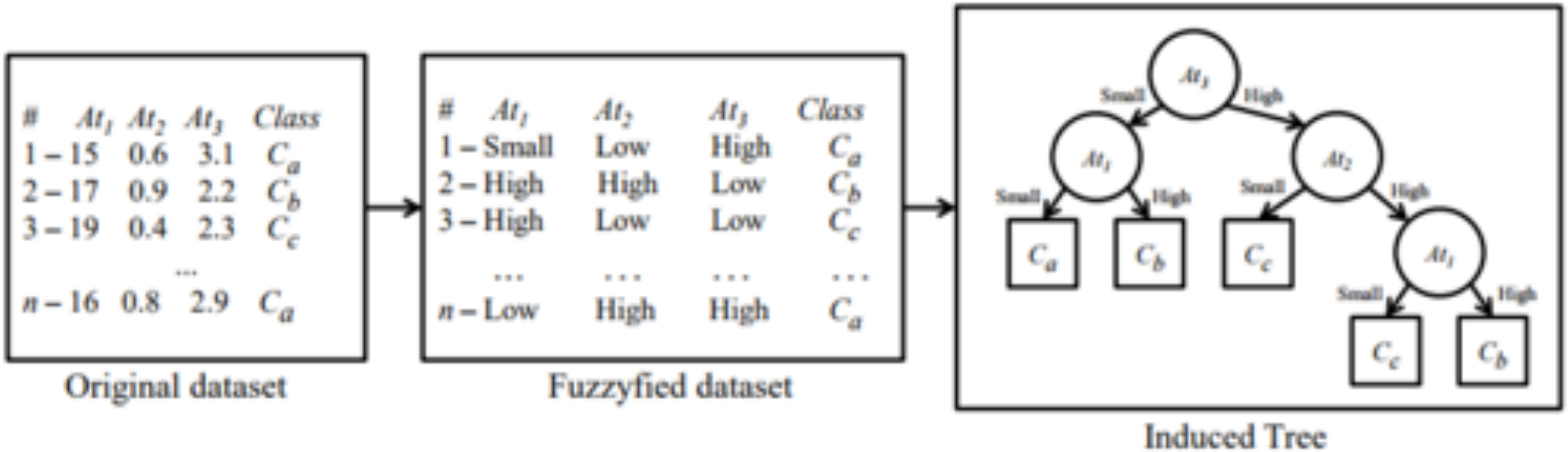
Induction process — Both versions use the same approach: repeated subdivision of the feature space using the most informative features until a leaf node is reached or no features or examples remain;

Some benefits

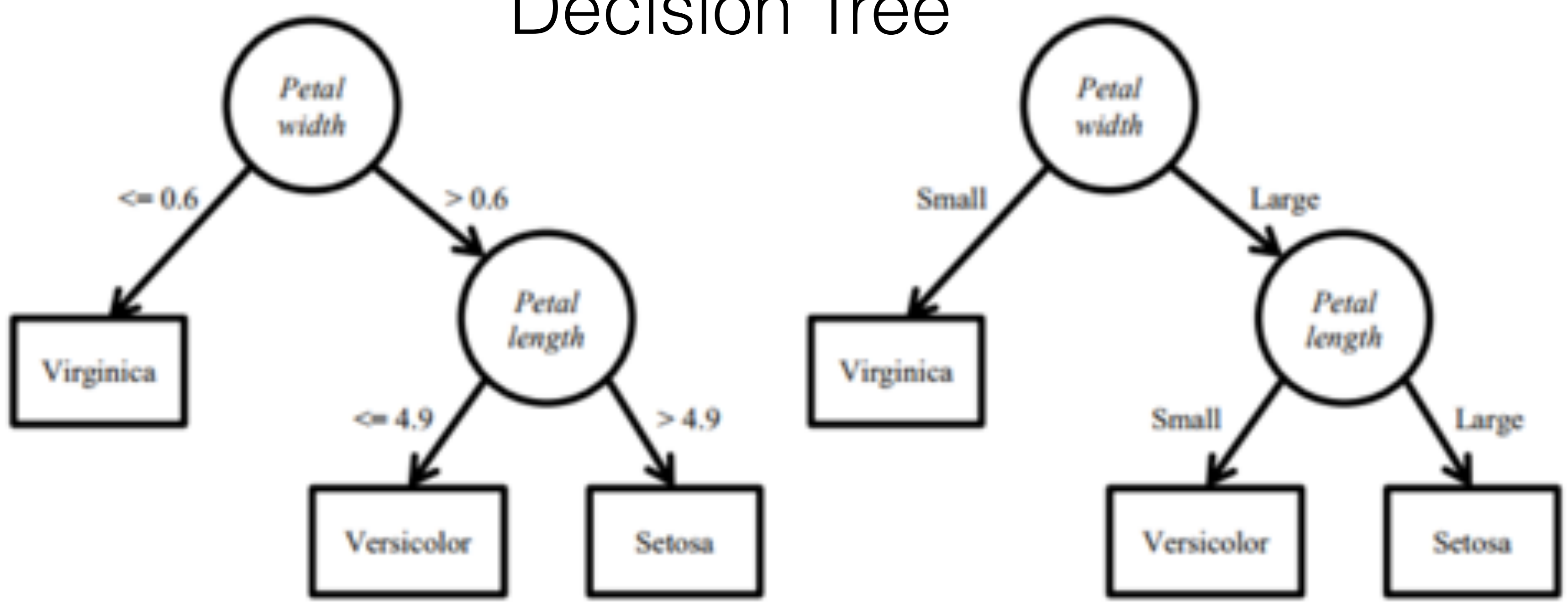
- Rule set reduction:

1. **IF** *Compactness* is $\leq$ 95 **AND** ... **AND** *Compactness* is $\leq$ 89
2. **IF** *Compactness* is $\leq$ 95 **AND** ... **AND** *Compactness* is $>$ 89
3. **IF** *Compactness* is $>$ 95
4. **IF** *Compactness* is $\leq$ 102
5. **IF** *Compactness* is $>$ 102
6. **IF** *Compactness* is $\leq$ 109 **AND** ... **AND** *Compactness* is $\leq$ 106
7. **IF** *Compactness* is $\leq$ 109 **AND** ... **AND** *Compactness* is $>$ 106
8. **IF** *Compactness* is $>$ 109
9. **IF** *Compactness* is $\leq$ 82 **AND** ... **AND** *Compactness* is $\leq$ 81
10. **IF** *Compactness* is $\leq$ 82 **AND** ... **AND** *Compactness* is $>$ 81
11. **IF** *Compactness* is $>$ 82 **AND** ... **AND** *Compactness* is $\leq$ 84
12. **IF** *Compactness* is $>$ 82 **AND** ... **AND** *Compactness* is $>$ 84

Fuzzy Machine Learning

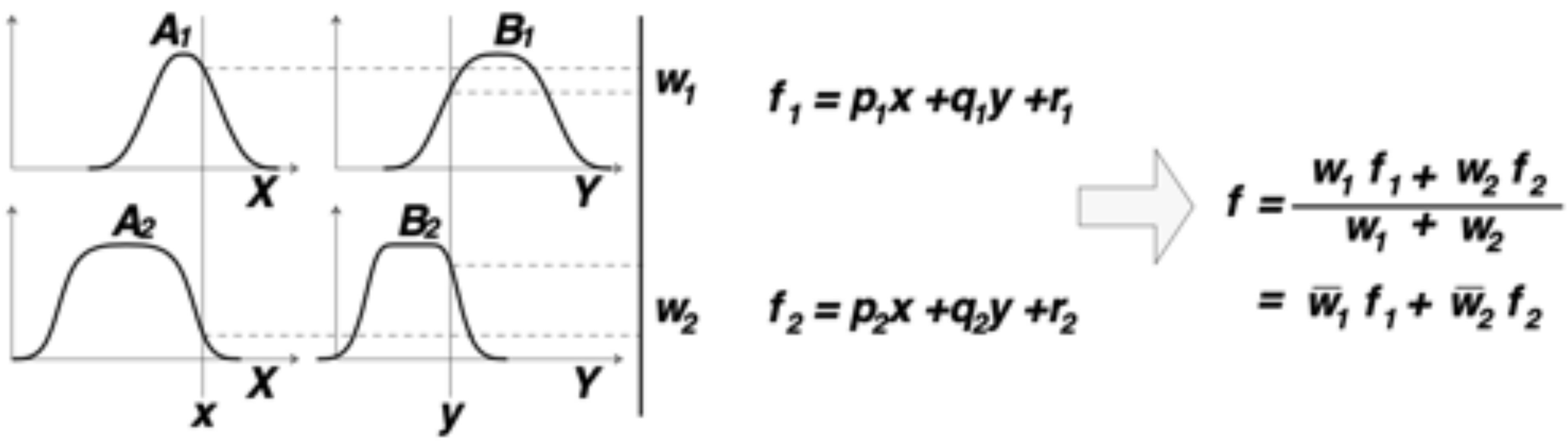


Decision Tree



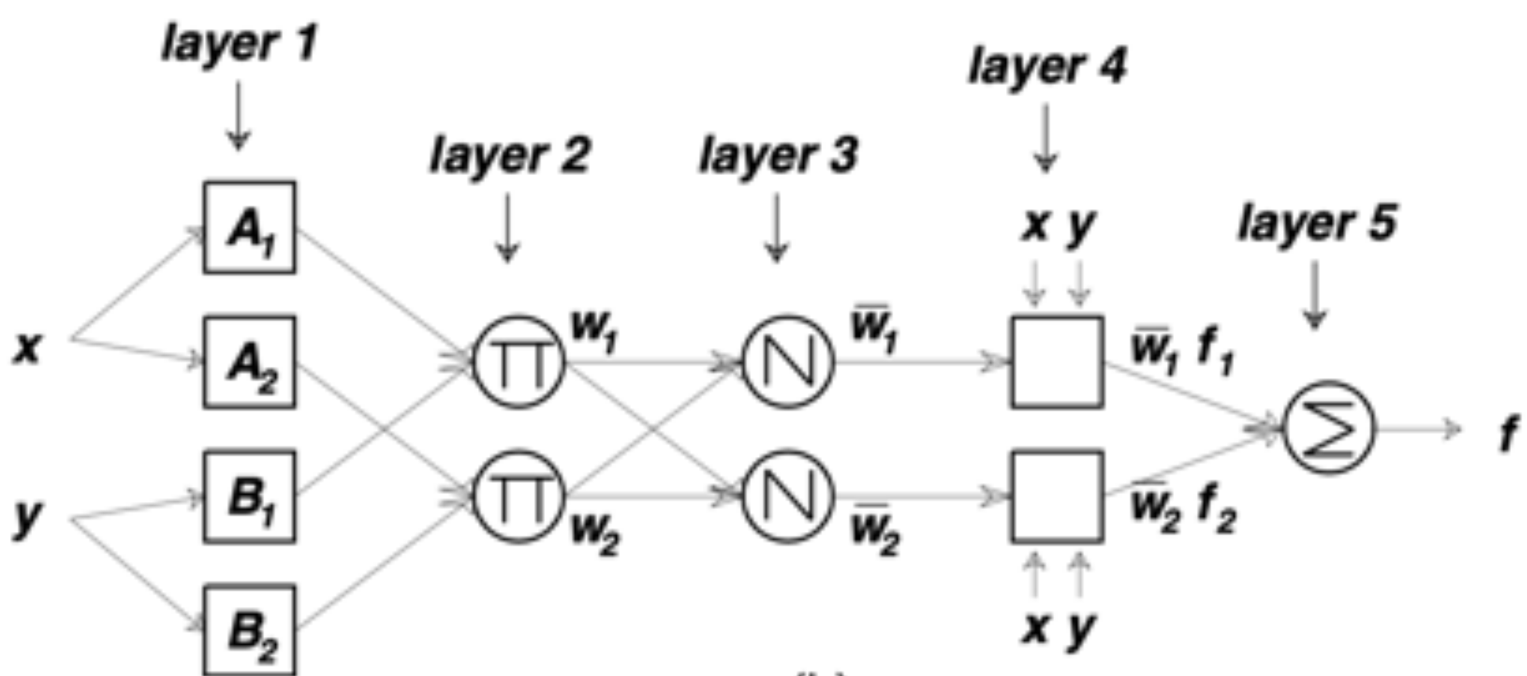
Neuro-Fuzzy

Fuzzy Model



(a)

Neuro Network



(b)

Pratical Tasks

1. Implement a “standard” sklearn kmeans
2. Explore FuzzyLogic04.ipynb
3. Create a new notebook FuzzyLogic05.ipynb and implement a Neuro-Fuzzy system.