
**Universidad Veracruzana**

Maestría en Inteligencia Artificial


# Lógica difusa

## Tarea 9. Implementación del algoritmo de Fuzzy C-means en MATLAB.


*Ángel García Báez*

Dr. Sergio Hernández Méndez


28 de abril de 2025

# Índice

# 1. Introducción

En el presente reporte se plantea la idea de aplicar el algoritmo de agrupación con lógica difusa "Fuzzy C-means"para probarlo sobre un conjunto de datos de jugadores de futbol, del cual se tienen 2 variables medidas por cada uno de ellos: rapidez y resistencia.

# 2. Definiciones

## 2.1. Distancia

Acorde con Wang and Sun (2015), para un conjunto de datos tal que si $x, y, z \in M$ son vectores de datos de la misma dimensionalidad, se puede definir $D : M \times M \to R$ como una distancia métrica si se satisfacen las siguientes propiedades:

- No negatividad: $D(x, y)$.
- Coincidencia: $D(x, y) = 0$ si y solo si $x = y$.
- Simetría: $D(x, y) = D(y, x)$
- Subaditividad: $D(x, y) + D(y, z) \geq D(x, z)$

Las siguientes definiciones de distancias se obtuvieron del articulo de revisión de Wang and Sun (2015):

## 2.2. Distancia Euclidiana

$$d(x_1, x_2) = \sqrt{\sum (x_1 - x_2)^2}$$

Donde:

- $x_1$ es el vector fila de tamaño $1 \times P$.
- $x_2$ es el vector fila de tamaño $1 \times P$.
- $\sum (x_1 - x_2)^2$ Es la suma de las diferencias al cuadrado de cada componente de los vectores, el resultado es un escalar.

## 2.3. Distancia Mahalanobis

$$d(x_1, x_2) = \sqrt{(x_1 - x_2)\Sigma^{-1}(x_1 - x_2)^T}$$

Donde:

- $x_1$ es el vector fila de tamaño $1 \times P$.
- $x_2$ es el vector fila de tamaño $1 \times P$.
- $\Sigma^{-1}$ es la inversa de la matriz de varianzas y covarianzas de todo el conjunto de datos al que pertenecen los vectores.

## 2.4. Distancia Manhattan

$$d(x_1, x_2) = \sum |(x_1 - x_2)|$$

- $x_1$ es el vector fila de tamaño $1 \times P$.
- $x_2$ es el vector fila de tamaño $1 \times P$.
- $\sum (x_1 - x_2)^2$ Es la suma de las diferencias en valor absoluto de las componentes de los vectores, el resultado es un escalar.

## 2.5. Algoritmo para fuzzy c-means

El algoritmo fuzzy c-means fue propuesto en un inicio por Nascimento et al. (2000) como una alternativa para el agrupamiento de datos en el contexto del aprendizaje no supervisado de tal forma que se le pudiera agregar esta capa de computo suave al proceso con el fin de llegar a mejores resultados.

A continuación se muestra una versión del algoritmo que es explicada de forma más clara y concisa que fue obtenida de Edla et al. (2020):

### 2.5.1. Paso 1

Inicializa la matriz $U(t)$ (matriz de pertenencias) de tamaño $n \times k$ con valores aleatorios entre 0 y 1, de tal forma que cada fila sume 1.

### 2.5.2. Paso 2

Calcula el valor de los $k$ centroides haciendo uso de la matriz de pertenencias $U(t)$ acorde con la siguiente formula:

$$vk = \frac{\sum_{i=1}^{n} u_{ki}^{m} xi}{\sum_{i=1}^{n} u_{ki}^{m}}$$

- $v_k$: Centroide del cluster $k$ (vector de características promedio ponderado).
- $n$: Número total de datos o muestras.
- $u_{ki}$: Grado de pertenencia del dato $i$ al cluster $k$.
- $m$: Parámetro de difusidad o fuzzificación ($m > 1$).
- $x_i$: Vector de características del dato $i$.
- $\sum_{i=1}^{n} u_{ki}^{m} x_i$: Suma ponderada de los datos $x_i$ por el grado de pertenencia elevado a $m$.
- $\sum_{i=1}^{n} u_{ki}^{m}$: Suma de los grados de pertenencia elevados a $m$ (factor de normalización).

### 2.5.3. Paso 3

Actualizar la matriz $U(t)$, con la matriz $U(t+1)$ reemplazando los elementos de la matriz con los siguientes:

$$u_{ki} = \frac{1}{\sum_{j=1}^{c} \frac{||x_i - v_k||}{||x_i - v_j||}^{2/(m-1)}}$$

- $u_{ki}$: Grado de pertenencia del dato $i$ al cluster $k$.
- $c$: Número total de clusters.
- $j$: Índice que recorre todos los clusters de 1 a $c$.
- $x_i$: Vector de características del dato $i$.
- $v_k$: Centroide del cluster $k$.
- $v_j$: Centroide del cluster $j$.
- $||x_i - v_k||$: Distancia entre el dato $i$ y el centroide $k$.
- $||x_i - v_j||$: Distancia entre el dato $i$ y el centroide $j$.
- $m$: Parámetro de difusidad o fuzzificación ($m > 1$).
- $\sum_{j=1}^{c} \left(\frac{||x_i - v_k||}{||x_i - v_j||}\right)^{2/(m-1)}$: Suma que normaliza las pertenencias relativas del punto $x_i$ respecto a todos los centroides.

### 2.5.4. Paso 4

Verificar que $||U(t) - U(t+1)|| < \epsilon$, si la diferencia de la anterior con la nueva matriz de pertenencias es menor al limite de tolerancia permitido, detiene el proceso, si no, vuelve al paso 2 y sigue iterando hasta alcanzar un máximo de iteraciones (fijado a 100 para efectos prácticos).

## 3. Metodología

La metodología de experimentación propuesta para observar el comportamiento del algoritmo bajo distintas combinaciones de parámetros, es realizar 3 corridas por cada una de las distancias (Euclidiana, Mahalanobis y Manhattan), con $k = 3$ grupos, una tolerancia $\epsilon = 0{,}001$, un máximo de iteraciones de 100 y variando al parámetro $M$ por valores de 1.3, 1.6 y 1.9.

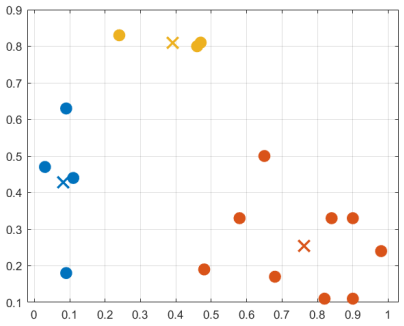
La base de datos para trabajar es la siguiente:

| **Rapidez** | **Resistencia** |
|---|---|
| 0.58 | 0.33 |
| 0.90 | 0.11 |
| 0.68 | 0.17 |
| 0.11 | 0.44 |
| 0.47 | 0.81 |
| 0.24 | 0.83 |
| 0.09 | 0.18 |
| 0.82 | 0.11 |
| 0.65 | 0.50 |
| 0.09 | 0.63 |
| 0.98 | 0.24 |
| 0.90 | 0.33 |
| 0.46 | 0.80 |
| 0.48 | 0.19 |
| 0.03 | 0.47 |
| 0.84 | 0.33 |

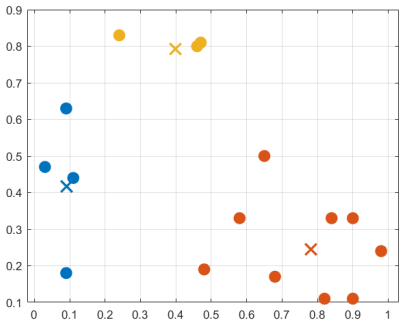
Cuadro 1: Valores de Rapidez y Resistencia

# 4. Resultados

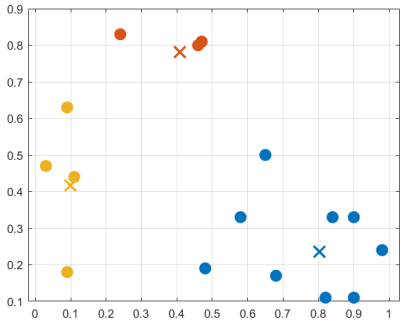
## 4.1. Resultados para la distancia Euclidiana



(a) Euclidiana con M = 1.3



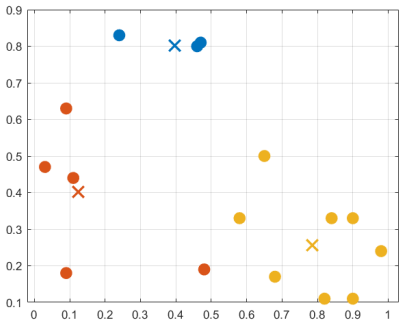
(b) Euclidiana con M = 1.6



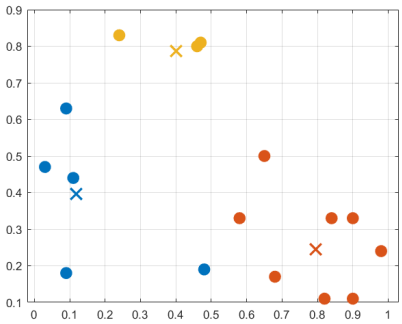
(c) Euclidiana con M = 1.9

Figura 1: Resumen de los resultados con distancia Euclidiana

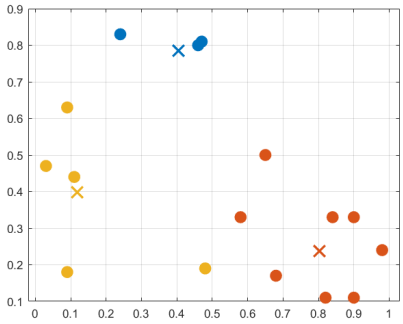
## 4.2. Resultados para la distancia de Mahalanobis




(a) Mahalanobis con M = 1.3



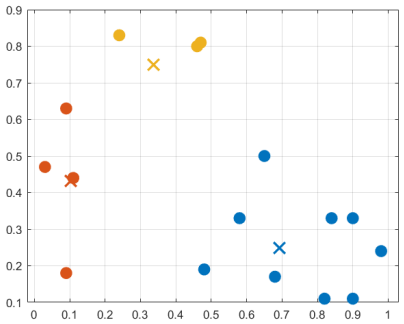

(b) Mahalanobis con M = 1.6



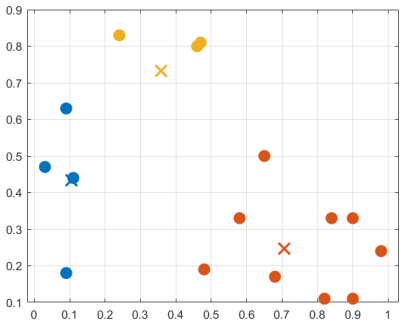

(c) Mahalanobis con M = 1.9

Figura 2: Resumen de los resultados con distancia Mahalanobis

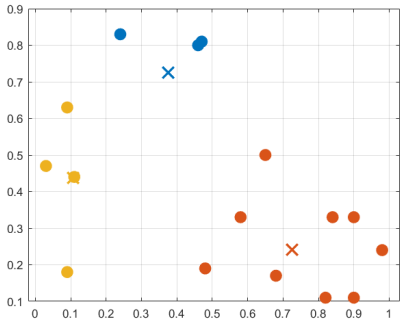
### 4.3. Resultados para la distancia de Manhattan



(a) Manhattan con M = 1.3



(b) Manhattan con M = 1.6



(c) Manhattan con M = 1.9

Figura 3: Resumen de los resultados con distancia Manhattan

## 5. Conclusiones

Se observa que el algoritmo logra detectar agrupaciones que se encuentran relativamente "visibiles", siendo que en para las 3 distancias, no hay diferencias internas al variar el parametro de fuzzy $M$. Tambien se encontro que la distancia Euclidiana y la distancia de Manhattan dan los mismos resultados, siendo que la distancia de Mahalanobis detecta que un punto que las demas distancias identifican cerca del centroide que se encuentra abajo a la derecha, realmente pertenece a el centroide de la izquierda, esto se puede explicar porque la distancia de Mahalanobis pondera en función de las varianzas y covarianzas.

Queda como trabajo futuro realizar la comparativa del desempeño de este algoritmo contra el algoritmo de K-medias clásico.

# 6. Referencias

## Referencias


Edla, D. R., Lone, T., Tapas, N., and Kuppili, V. (2020). Analysis of high dimensional brain data using prototype based fuzzy clustering. *Clinical Epidemiology and Global Health*, 8(4):1110–1118.

Nascimento, S., Mirkin, B., and Moura-Pires, F. (2000). A fuzzy clustering model of data and fuzzy c-means. In *Ninth IEEE International Conference on Fuzzy Systems. FUZZ- IEEE 2000 (Cat. No.00CH37063)*, volume 1, pages 302–307, San Antonio, TX, USA. IEEE.

Wang, F. and Sun, J. (2015). Survey on distance metric learning and dimensionality reduction in data mining. *Data Mining and Knowledge Discovery*, 29(2):534–564.

# 7. Anexos

Este reporte se envía con los códigos anexos que corresponden a:

1. El codigo en MATLAB del fuzzy cmeans con las pruebas
2. El archivo de excel donde se almacenaron los datos.