Máquinas de Aprendizaje Extremo

Motivación

- Las redes neuronales tipo back-propagation son lentas y la razón es:

  Entrenamiento utilizando algoritmos de aprendizaje basados en gradiente, requieren de muchas iteraciones para lograr un mejor rendimiento.

- La Máquina de Aprendizaje Extrema (ELM) fue propuesta para superar estos problemas y ofrecer un mejor desempeño de generalización.

Motivación

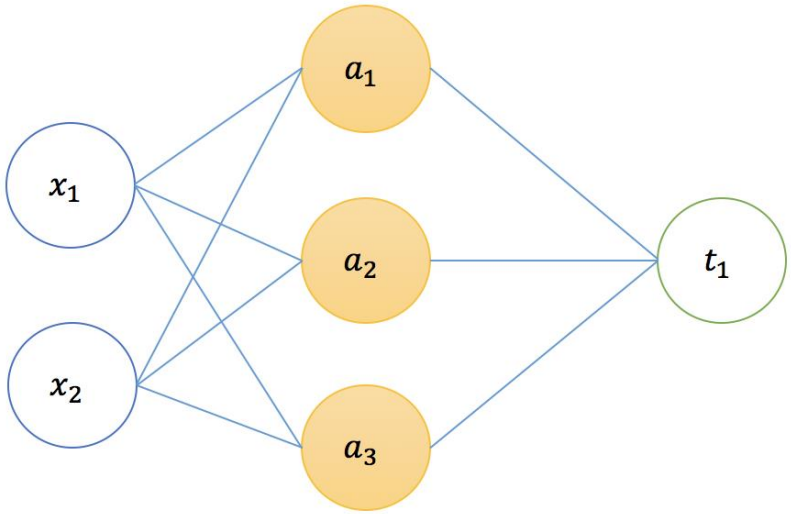
- Las ELM se usan tanto en problemas de clasificación como de regresión.

- La principal ventaja de las ELM es la ausencia del ajuste en la capa oculta del modelo neuronal por lo que su tiempo de aprendizaje es mucho menor que un BP clásico

Algoritmo ELM

- Una sola capa oculta
- Los pesos de entrada y los bias son escogidos aleatoriamente.
- Esos pesos y bias no se ajustan
- Entrenamiento consiste en encontrar mínimo cuadrado de β' del sistema lineal $H\,\beta=T$
- Analíticamente determina los pesos de salida

ELM

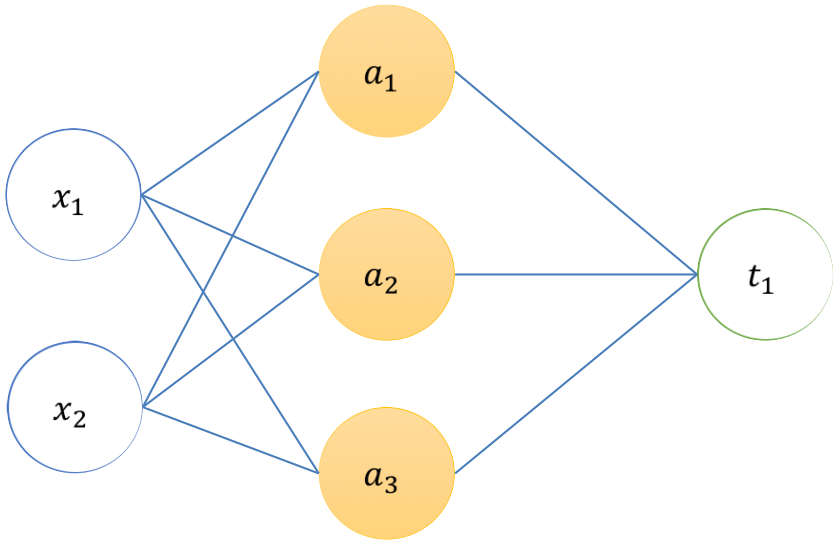


X W b β T

Algoritmo ELM

- T etiquetas, N observaciones, N' neuronas escondidas (N=N')
- H=(w_1, …, $w_{N'}$, b_1, …, $b_{N'}$), matriz cuadrada de salida capa oculta
- Funciones de activación no lineales diferenciables
- No se actualizan los pesos de entrada ni bias de la capa oculta
- sistema lineal H β=T tiene solo una solución
- Puede aprender N distintas observaciones con error=0

ELM



X W b β T

$$\mathbf{H}\beta = \mathbf{T},$$

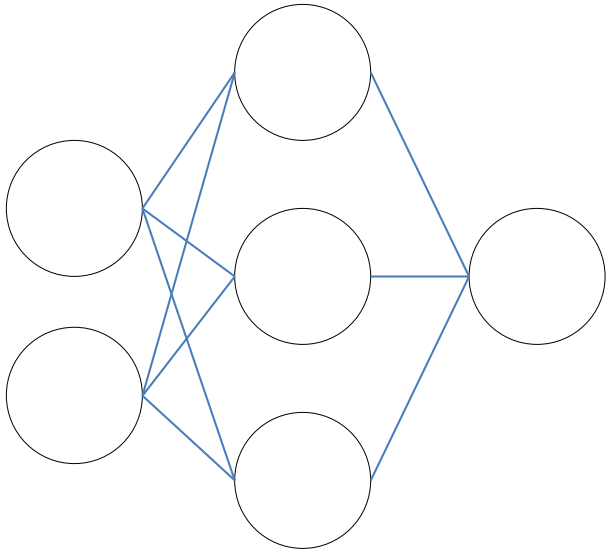
where

$$\mathbf{H}(\mathbf{w}_1, \ldots, \mathbf{w}_{\tilde{N}}, b_1, \ldots, b_{\tilde{N}}, \mathbf{x}_1, \ldots, \mathbf{x}_N) = \begin{bmatrix} g(\mathbf{w}_1 \cdot \mathbf{x}_1 + b_1) & \cdots & g(\mathbf{w}_{\tilde{N}} \cdot \mathbf{x}_1 + b_{\tilde{N}}) \\ \vdots & \cdots & \vdots \\ g(\mathbf{w}_1 \cdot \mathbf{x}_N + b_1) & \cdots & g(\mathbf{w}_{\tilde{N}} \cdot \mathbf{x}_N + b_{\tilde{N}}) \end{bmatrix}_{N \times \tilde{N}},$$

$$\beta = \begin{bmatrix} \beta_1^{\mathrm{T}} \\ \vdots \\ \beta_{\tilde{N}}^{\mathrm{T}} \end{bmatrix}_{\tilde{N} \times m} \quad \text{and} \quad \mathbf{T} = \begin{bmatrix} \mathbf{t}_1^{\mathrm{T}} \\ \vdots \\ \mathbf{t}_N^{\mathrm{T}} \end{bmatrix}_{N \times m}.$$

$\tilde{N} = \#\ Hidden\ units$
$N = \#\ of\ observation$

ELM



Capa de entrada | Capa oculta | Capa de salida

$$\mathbf{H}\boldsymbol{\beta} = \mathbf{T}$$

$$\mathbf{H} = \begin{bmatrix} g(\mathbf{w}_1\mathbf{x}_1 + b_1) & \cdots & g(\mathbf{w}_j\mathbf{x}_1 + b_j) \\ \vdots & \ddots & \vdots \\ g(\mathbf{w}_1\mathbf{x}_N + b_1) & \cdots & g(\mathbf{w}_j\mathbf{x}_N + b_j) \end{bmatrix}$$

$$\min_{\boldsymbol{\beta}} \left\| \mathbf{H}\boldsymbol{\beta} - \mathbf{T} \right\|$$

$$\boldsymbol{\beta} = (\mathbf{H}^T\mathbf{H})^{-1}\mathbf{H}^T\mathbf{T}$$

$\mathbf{H}: N \times j$, matrizidden layer output matrix

$\boldsymbol{\beta}: j \times m$, the output weight matrix

$\mathbf{T}: N \times m$, target

$$\text{weight} \in [-1,1], \text{bias} \in [0,1]$$

Los pesos entre la capa de entrada y la capa oculta y los sesgos de las neuronas en la capa oculta se eligen al azar.

Algoritmo ELM

Los pesos de salida se pueden obtener analíticamente mediante la matriz pseudoinversa

- Dado un conjunto de *N* vectores de entrada existen los parámetros β_i, $\mathbf{w}_i$ y b_i con:

$$\sum_{i=1}^{H} \beta_i \, \theta(\mathbf{w}_i x_j + b_i) = t_j$$

donde $j = 1, \ldots, N$ y t_i es el *i*−´esimo vector *target*.

- Esta ecuación se puede expresar de forma matricial como:

$$\mathbf{H}\,\beta = \mathbf{T},$$

donde **H** es la matriz $N \times H$ de salidas de la capa oculta, β es la matriz $H \times n$ de pesos de salida y **T** es una matriz $N \times n$ de *N targets*.

- El entrenamiento consiste en un problema de mínimos cuadrados con pesos ´óptimos $\beta' = \mathbf{H}^{+}\mathbf{T}$,

Propiedades de la solución H β=T

Minimo error de entrenamiento

$$\|\mathbf{H}\hat{\beta} - \mathbf{T}\| = \|\mathbf{H}\mathbf{H}^{\dagger}\mathbf{T} - \mathbf{T}\| = \min_{\beta} \|\mathbf{H}\beta - \mathbf{T}\|$$

Norma más pequeña de pesos

$$\|\hat{\beta}\| = \|\mathbf{H}^{\dagger}\mathbf{T}\| \leqslant \|\beta\|,$$

$$\forall \beta \in \left\{\beta : \|\mathbf{H}\beta - \mathbf{T}\| \leqslant \|\mathbf{H}\mathbf{z} - \mathbf{T}\|, \forall \mathbf{z} \in \mathbf{R}^{\tilde{N}\times N}\right\}$$

Solución unica del Sistema no lineal es minimo cuadrado $\beta' = H^{+}T$

Calculo de la pseudoinverso H^+

- Fat Matrix (Case $m < n$):
  - $H^+ = H^T(HH^T)^{-1}$
- Skinny Matrix (Case $m > n$):
  - $H^+ = (H^T H)^{-1}H^T$
- If H is not in full rank the pseudoinverse can be computed using Singular Value Decomposition (SVD) or other method.

Algoritmo de ELM

Algorithm ELM: Given a training set $\aleph = \{(\mathbf{x}_i, \mathbf{t}_i) | \mathbf{x}_i \in \mathbf{R}^n, \mathbf{t}_i \in \mathbf{R}^m, i = 1, \ldots, N\}$, activation function $g(x)$, and hidden node number $\tilde{N}$,

Step 1: Randomly assign input weight $\mathbf{w}_i$ and bias b_i, $i = 1, \ldots, \tilde{N}$.

Step 2: Calculate the hidden layer output matrix $\mathbf{H}$.

Step 3: Calculate the output weight β

$$\beta = \mathbf{H}^{\dagger}\mathbf{T}, \tag{16}$$

where $\mathbf{T} = [\mathbf{t}_1, \ldots, \mathbf{t}_N]^{\mathrm{T}}$.

ELM tienen mejor generalización en comparación con los algoritmos basados en gradiente como BP

ELM	Backpropagation
Solución mínima única	Propenso a converger en mínimos locales
Alcance el error de entrenamiento más pequeño y la norma de peso más pequeño.	Error de entrenamiento más pequeño
No necesita un método de parada	Sobreentrenamiento cuando la función objetivo no tiene métodos de parada adecuados.

Aprendizaje Incremental e Hibrído

Incremental Learning

- A diferencia de los seres humanos, que incrementalmente aprende la información a medida que se introduce, ANNs aprende todo de una vez.

- Una ANN existente no puede adaptarse a un sistema de cambio dinámico. Por lo tanto, cuando un sistema cambia, una nueva ANN debe ser creado desde cero.

- El objetivo es crear una ANN más adaptable, que aprenda incrementalmente, como los seres humanos.

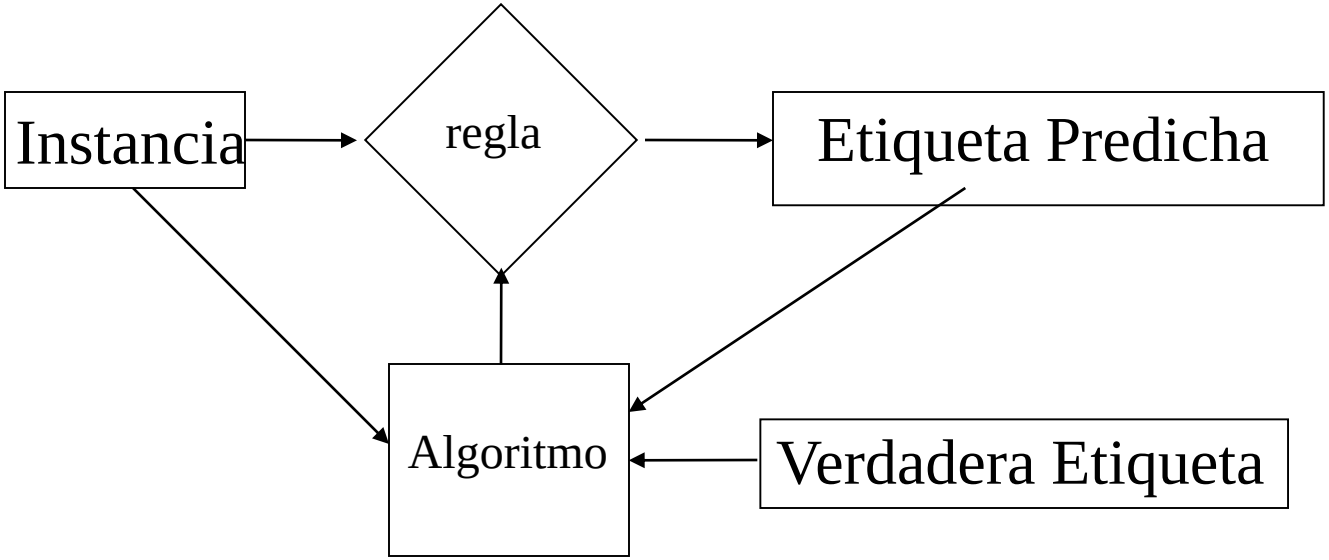
Incremental Learning

- Suponga que obtiene retroalimentación sobre sus predicciones
- Refina sus reglas con comentarios de etiqueta
- Permite que la regla se ajuste a las condiciones cambiantes

Online Learning

- Algoritmo no tiene instancias al inicio
- Algoritmo obtiene una sola instancia cada prueba
- Un ensayo se compone de tres pasos
  - Obtener Instancia
  - Predecir Etiqueta
  - Descubrir la etiqueta verdadera y usar esa información para refinar la regla de predicción

Online Learning



Ejemplos Online Learning

- Mercado financiero
- Predicción del tiempo
- Deportes
- Prediciendo el futuro
- Detección de spam
- Cocina

Otro Enfoque

- La búsqueda basada en poblaciones, como GA
  - Crea una matriz de probabilidad contando el número de 1s y 0s en cada posición del gen
  - Genera nueva población usando la matriz de probabilidad
  - ¡No se transmite información de generación en generación!
- Aprendizaje competitivo (CL) supervisado, p. LVQ
  - Ganador ganador-toma-todo y hace un Aprendizaje del reforzador en la ANN
  - Winner es una especie de prototipo de la muestra presentada
- PBIL = GA + CL
  - Captura la tendencia del mejor intérprete

PBIL

P ← initialize probability vector (each position = 0.5)
while (generations++ < limit)
    for each vector i do
        for each position j do
            generate $V_i(j)$ according to P(j)
        end-do
        evaluate $f(V_i)$
    end-do
    $V_{max} = \max(f(V_i))$
    update P according to V_{max}
    if random(0,1] < P_{mutate}
        mutate P
    end-if
end-while

Reglas

- Regla de Actualización

$$P_i(j) = P_i(j) * (1.0 - \alpha) + V_{\max}(j) * \alpha$$

- Regla de Mutación

$$P_i(j) = P_i(j) * (1.0 - \beta) + rand[0.0|1.0] * \beta$$

  - P_{mutate} = 0.02
  - □ = 0.05

Conclusión

- El aprendizaje en línea es un modelo teóricamente fuerte para el aprendizaje

- El aprendizaje en línea permite un estilo incremental y adaptativo de aprendizaje que podría estar más cerca de la forma en que los seres humanos y los animales aprenden

Aprendizaje supervisado






Carros





Motocicletas

Aprendizaje semi-supervisado








Imágenes sin etiquetas (solo carros/motocicletas)

Carro

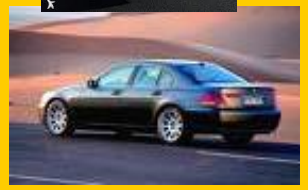
Motocicleta

Aprendizaje semi-supervisado










Imágenes sin etiquetas

Carro

Motocicleta

Aprendizaje Híbrido

Modelos que son capaces de combinar correctamente (bajo algunos criterios) el aprendizaje supervisado y no supervisado, para resolver sistemas con las siguientes características:

- Dado un conjunto de datos etiquetados, formar un modelo de clases.
- Dado un conjunto de datos no etiquetados, formar un modelo para generar clusters.
- Dado un conjunto de datos etiquetados y no etiquetados, formar un modelo para generar clases y clusters, simultáneamente.
- Dadas nuevas muestras entrantes y un modelo entrenado, asignar cada muestra a una clase o cluster existente, si es posible.
- Dadas nuevas muestras entrantes, el modelo puede evolucionar su estructura actual de clases o cluster, si es necesario, sin reentrenar.

Aprendizaje Híbrido

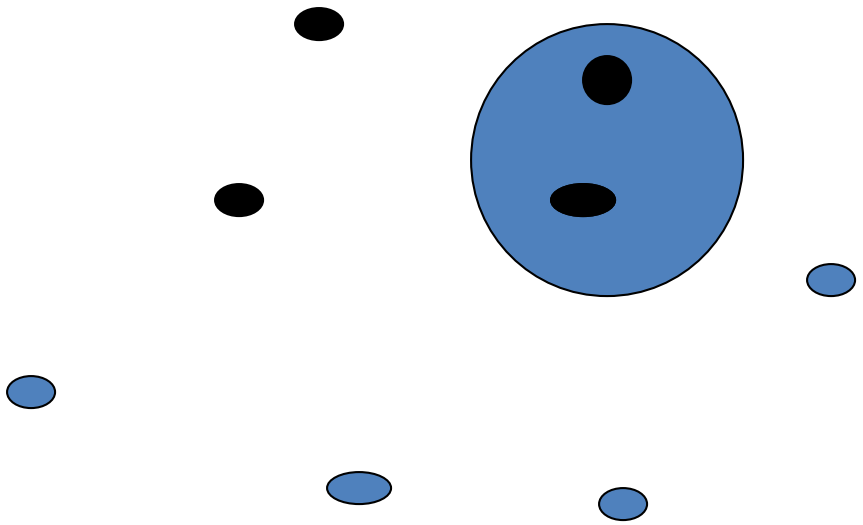
HHA-EMCC combina dos técnicas bien conocidas: KNN y K-means.

- **Clasificación basada en la distancia**
  - El algoritmo K Vecinos más cercanos (KNN) es uno de los algoritmos de aprendizaje más simples y efectivos basados en la distancia para la clasificación.
- **Clustering basado en prototipos**
  - K-means es un algoritmo de aprendizaje clásico basado en prototipos, que divide consecutivamente un conjunto de datos hasta K clusters que minimizan una función objetivo.

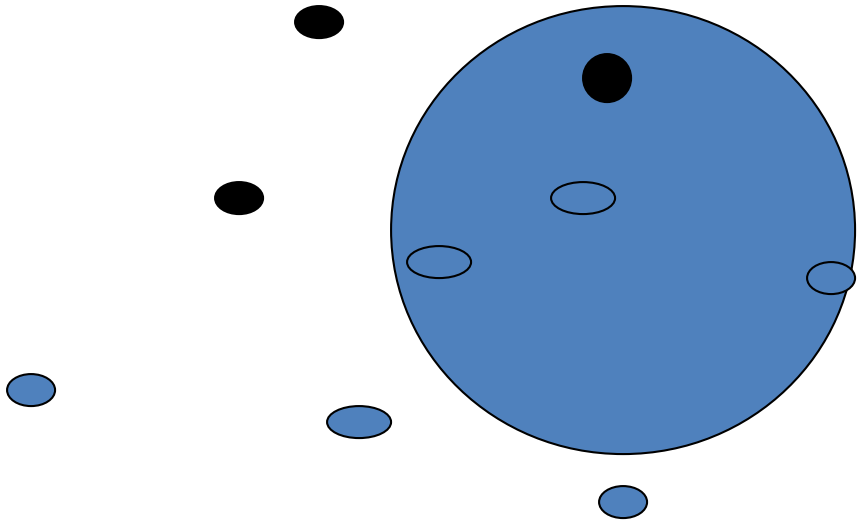
K Vecinos más cercanos (KNN)

- Características
  - Todas las instancias corresponden a puntos en un espacio euclidiano de dimensión n
  - La clasificación se retrasa hasta que llega una nueva instancia
  - Clasificación se realiza comparando los vectores de características de los diferentes puntos
  - La función objetivo puede ser discreta o real

1-NN



3- NN



KNN

- Una instancia arbitraria está representada por $(a_1(x), a_2(x), a_3(x),.., a_n(x))$
  - $a_i(x)$ denota características
- Distancia euclidiana entre dos instancias
  - $d(x_i, x_j)=\text{sqrt (sum for r=1 to n } (a_r(x_i) - a_r(x_j))^2)$
- Función objetivo evaluada continuamente
  - Valor medio de los k ejemplos de entrenamiento más cercanos

KNN

COMIENZO

Entrada: $D = \{(\mathbf{x}_1, c_1), \ldots, (\mathbf{x}_N, c_N)\}$

$\mathbf{x} = (x_1, \ldots, x_n)$ nuevo caso a clasificar

PARA todo objeto ya clasificado (x_i, c_i)

calcular $d_i = d(\mathbf{x}_i, \mathbf{x})$

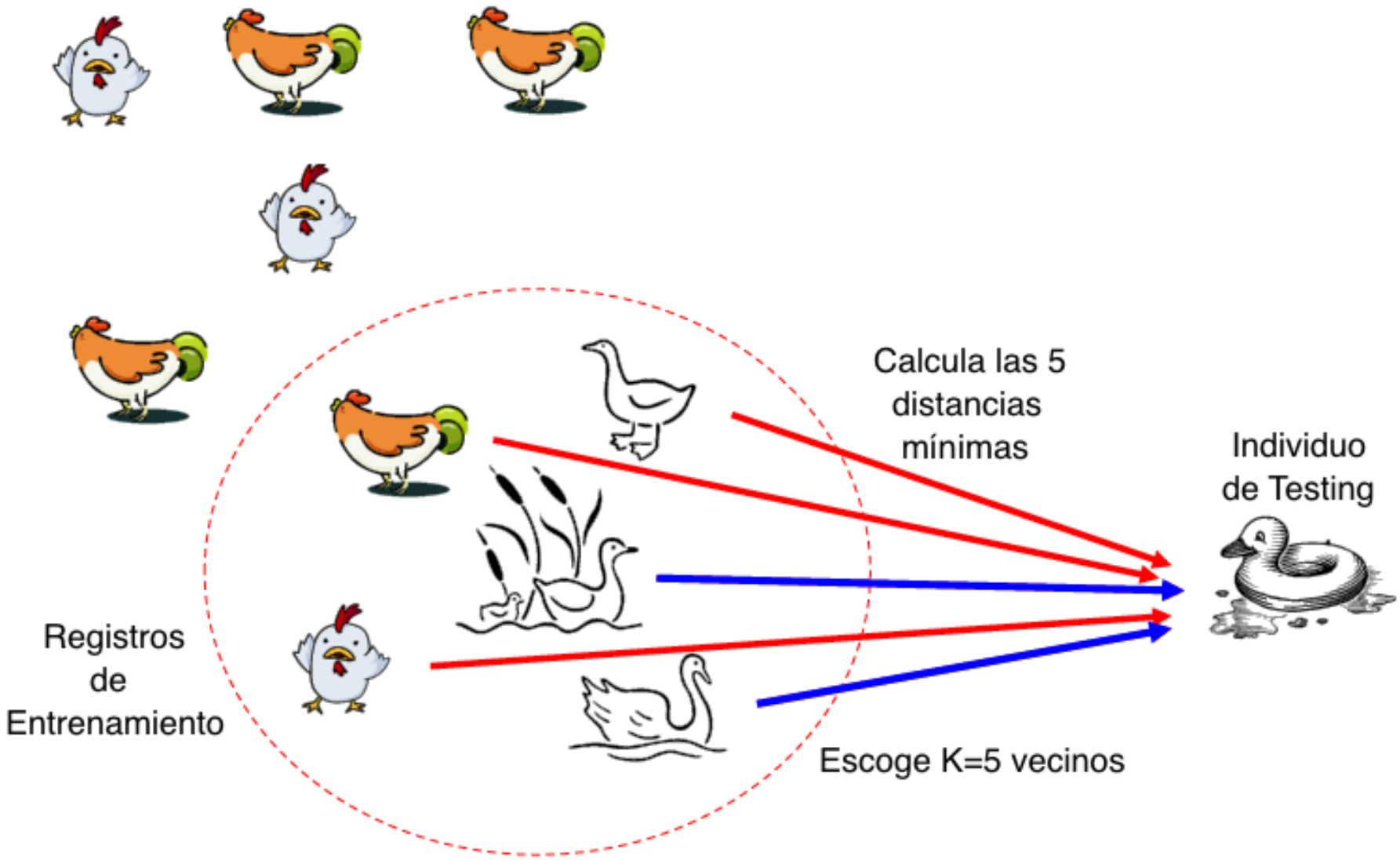
Ordenar $d_i (i = 1, \ldots, N)$ en orden ascendente

Quedarnos con los K casos $D_{\mathbf{x}}^K$ ya clasificados más cercanos a $\mathbf{x}$

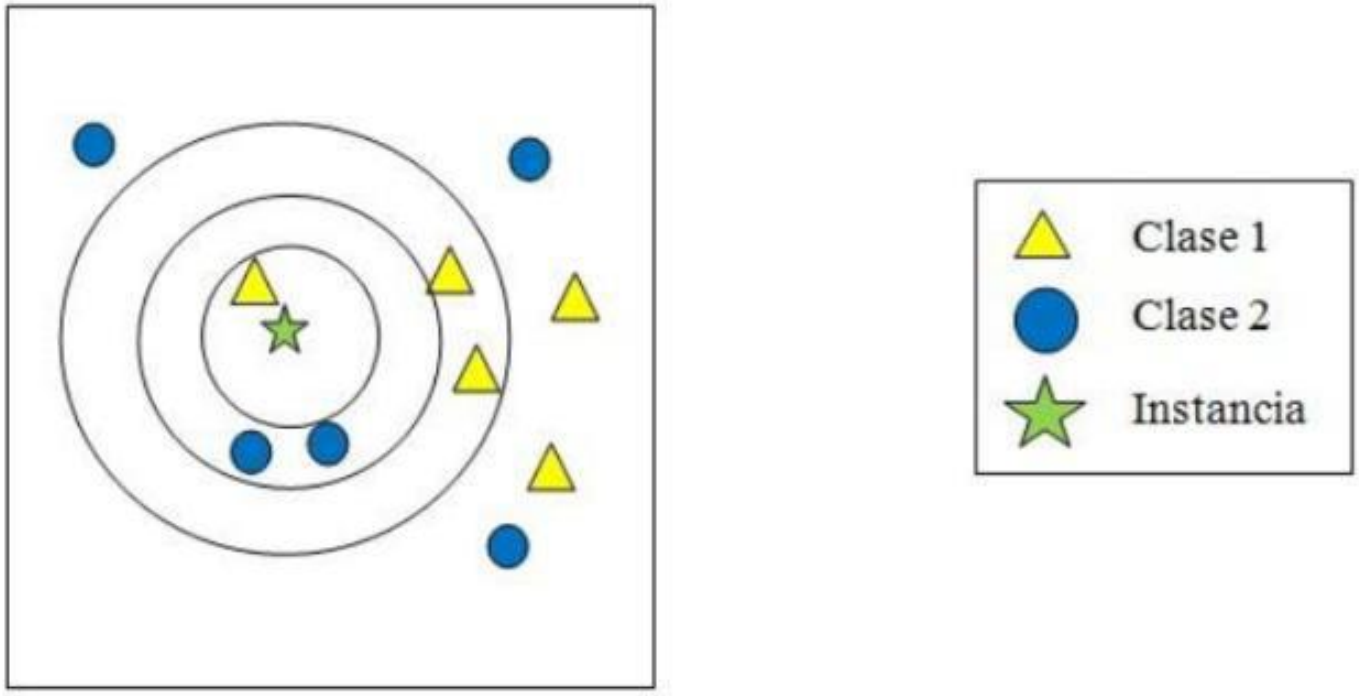
Asignar a $\mathbf{x}$ la clase más frecuente en $D_{\mathbf{x}}^K$

FIN

KNN


Calcula las 5
distancias
mínimas
Individuo
de Testing
Registros
de
Entrenamiento
Escoge K=5 vecinos

KNN



Para K=1 (círculo más pequeño), la clase de la nueva instancia sería la Clase 1, ya que es la clase de su vecino más cercano, mientras que para K=3 la clase de la nueva instancia sería la Clase 2 pues habrían dos vecinos de la Clase 2 y solo 1 de la Clase 1

¿Cómo escoger K?

- Si K es muy pequeño el modelo será muy sensitivo a puntos que son atípicos o que son ruido (datos corruptos)
- Si K es muy grande, el modelo tiende a asignar siempre a la clase más grande.

Mediante una Tabla de Aprendizaje el modelo escogerá el valor de K que mejor clasificación logre en esta tabla, es decir, prueba con K=1, K=2, ….

- Esto pude ser muy caro computacionalmente.

¿Cómo escoger K?

- Si K es muy pequeño el modelo será muy sensitivo a puntos que son atípicos o que son ruido (datos corruptos)
- Si K es muy grande, el modelo tiende a asignar siempre a la clase más grande.

Mediante una Tabla de Aprendizaje el modelo escogerá el valor de K que mejor clasificación logre en esta tabla, es decir, prueba con K=1, K=2, ….

- Esto pude ser muy caro computacionalmente.

¿Cómo escoger la distancia?

- Euclidiana.

$$d(A,B) \equiv \sqrt{\sum_{i=1}^{n}(A_i - B_i)^2} = \sqrt{(A-B)^T(A-B)}$$

- Manhattan

$$d(x,y) = \sum_{i=1}^{n} |x_i - y_i|$$

Conclusión

- Método de inferencia inductiva altamente efectivo para datos de entrenamiento ruidosos y funciones objetivo complejas
- La función objetivo de un espacio entero puede describirse como una combinación de aproximaciones locales menos complejas
- El aprendizaje es muy simple
- La clasificación lleva mucho tiempo

Tipos de clustering

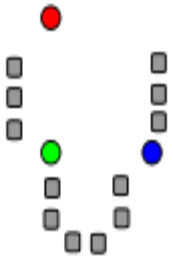
Basados en prototipos

- Un cluster se define por un conjunto de objetos, donde cada objeto esta más cerca (o es más similar) al **prototipo** que define al cluster donde fue asignado que a cualquier otro prototipo de otro cluster existente.

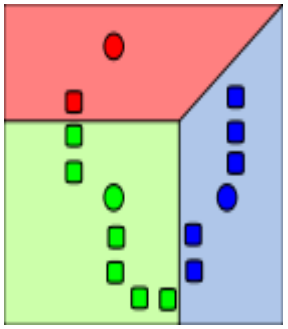
- El prototipo que define al cluster normalmente **denota sus características principales**, por ejemplo, para atributos solo numéricos el prototipo del cluster a menudo se representa como el centroide.

Algoritmo K-medias

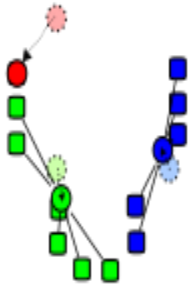
es un método de agrupamiento, que tiene como objetivo la partición de un conjunto (n) en k grupos en el que **cada ejemplo pertenece al grupo más cercano a la media**



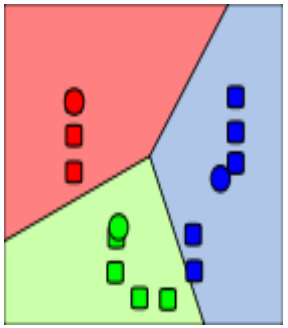
1) *k* centroides iniciales generados aleatoreamente (en este caso *k*=3)



2) *k* grupos son generados asociándole el punto



3) El centroide de cada uno de los *k* grupos se recalcula



4) Pasos 2 y 3 se repiten hasta que se logre la convergencia.

Algoritmo K-medias

1. **Método más utilizado** de clustering particional

2. La idea es situar **los prototipos o centros en el espacio**, de forma que los datos pertenecientes al mismo prototipo tengan características similares

3. Los datos se **asignan a cada centro según la menor distancia**, normalmente usando la distancia euclídea

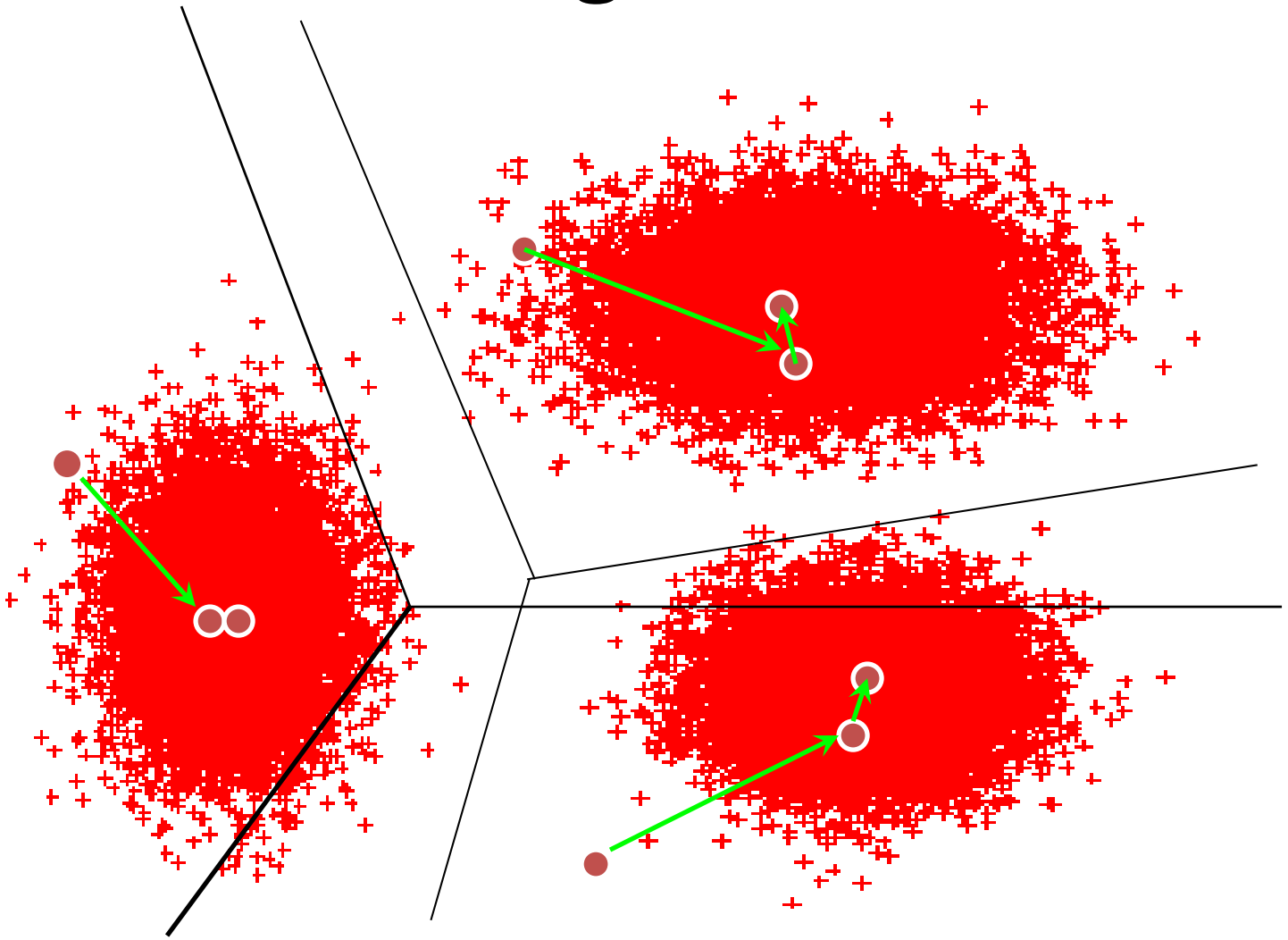
4. Una vez introducidos todos los datos, se **desplazan los prototipos hasta el centro de masas** de su nuevo conjunto, esto se repite hasta que no se desplazan más.

Clusterización: Algoritmo K-Medias

- Seleccionar centroides aleatorios
- Asignar cada objeto al grupo cuyo centroide sea el más cercano al objeto.
- Cuando todos los objetos hayan sido asignados, recalcular la posición de los k centroides.
- Repetir los pasos 2 y 3 hasta que los centroides no varíen

Distancia Euclídea $\delta^2 E(Xi, Xj) = ||Xi - Xj||^2$

Clusterización: Algoritmo K-Medias



Corrida en frio K-means

Input:

- K (number of clusters)
- Training set $\{x^{(1)}, x^{(2)}, \ldots, x^{(m)}\}$

Randomly initialize K cluster centroids $\mu_1, \mu_2, \ldots, \mu_K \in \mathbb{R}^n$

Repeat {

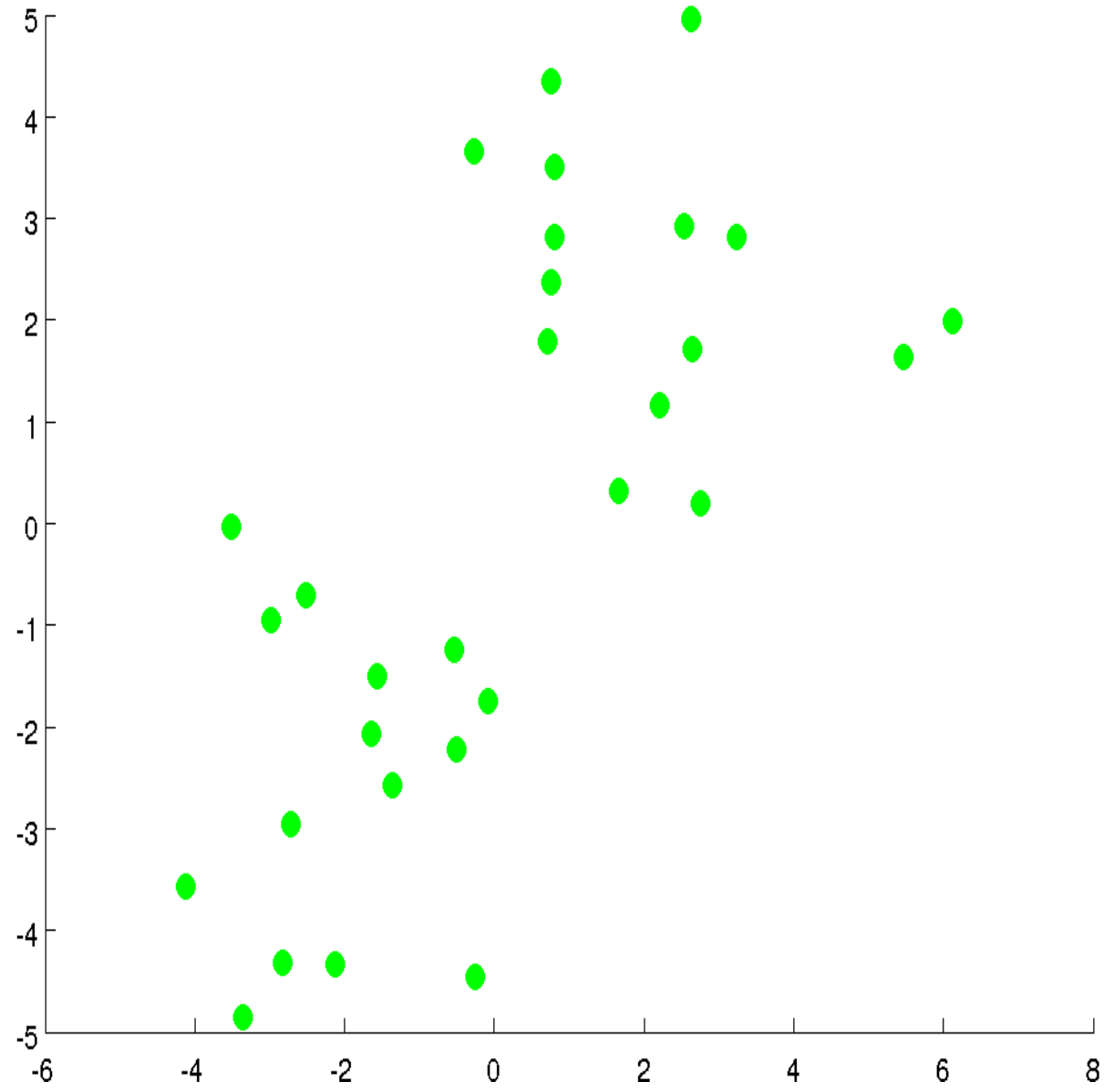
for i =1 to m

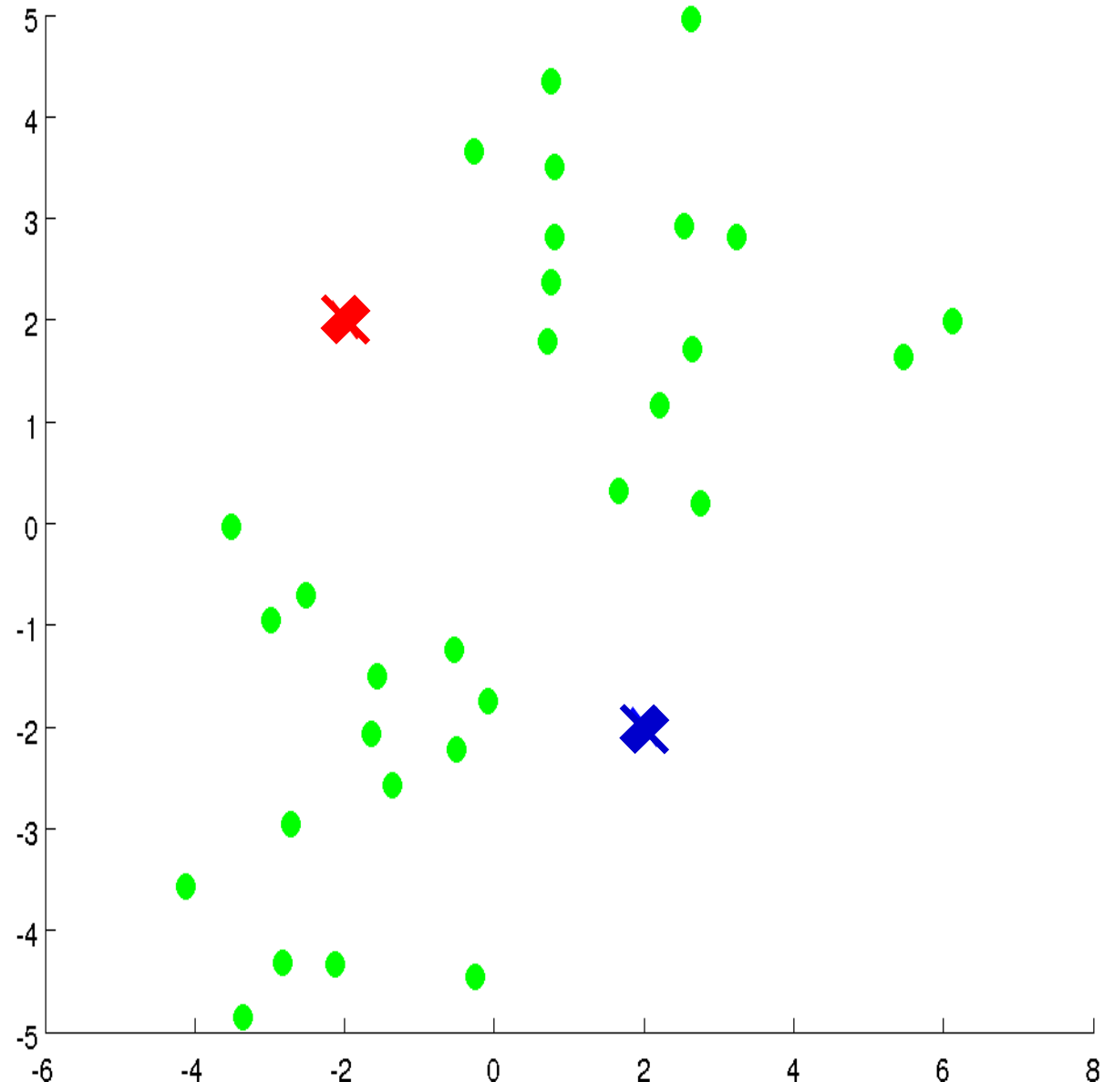
$c^{(i)}$:= index (from 1 to K) of cluster centroid closest to $x^{(i)}$

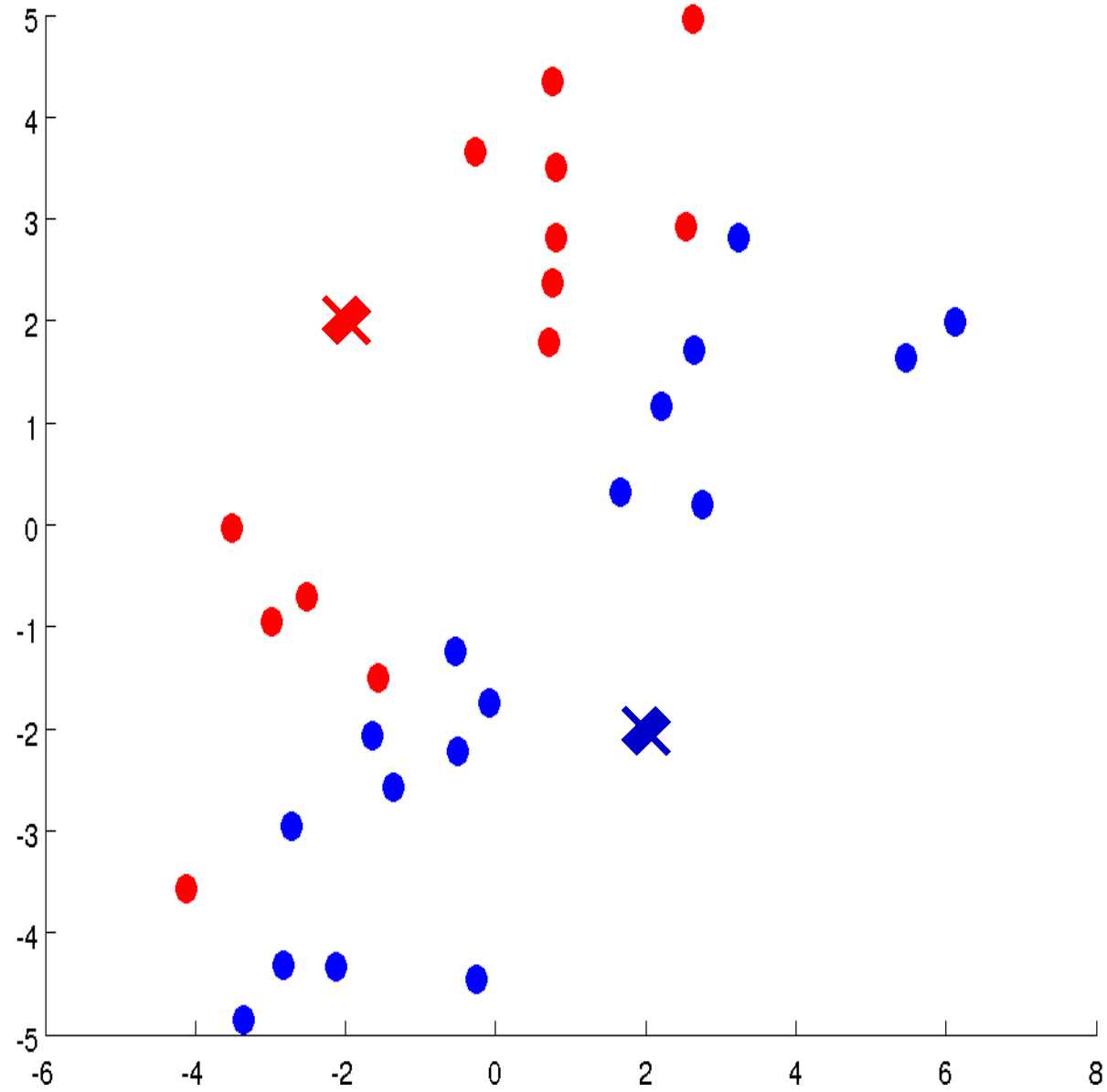
for k = 1 to K

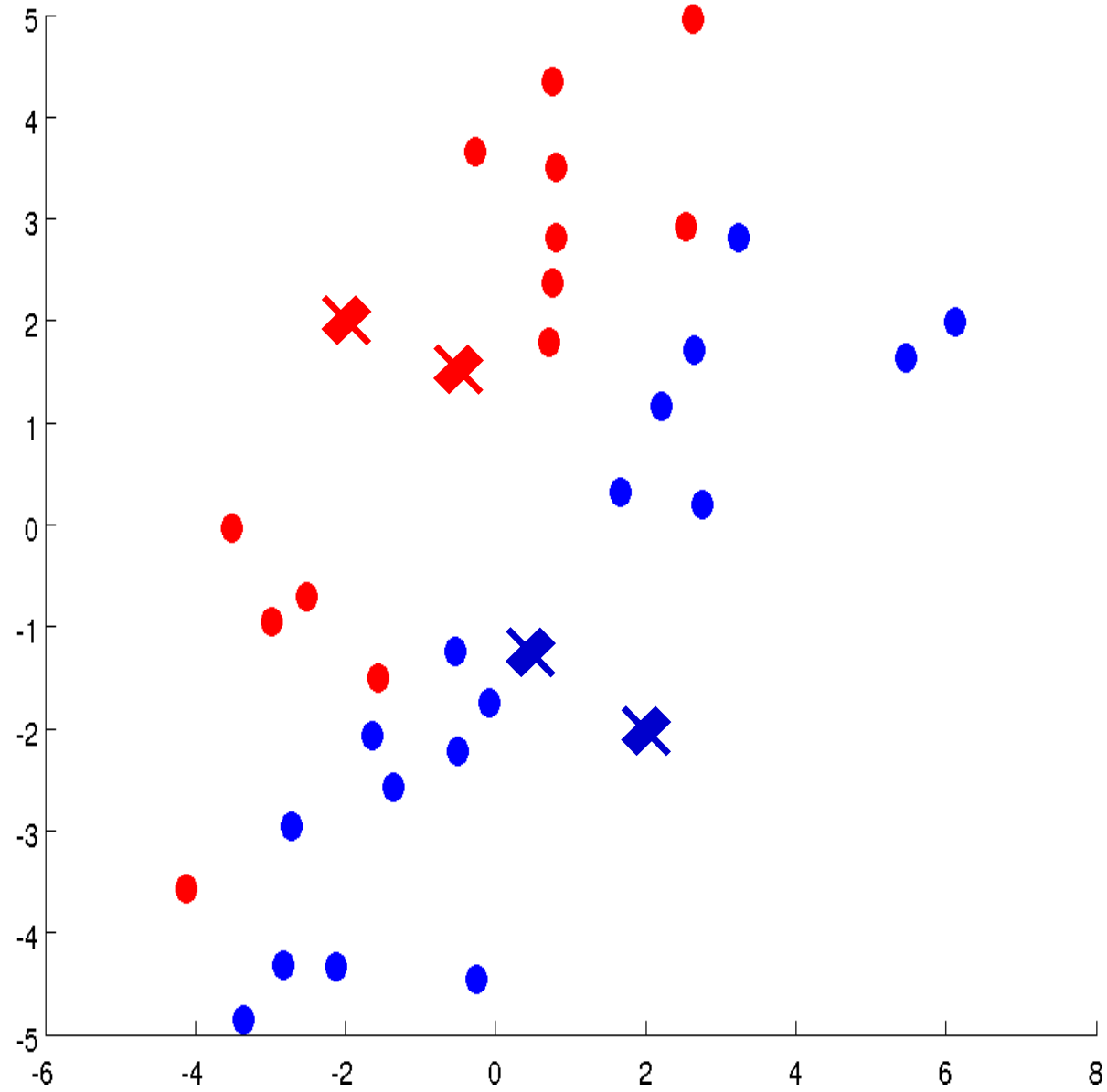
μ_k := average (mean) of points assigned to cluster

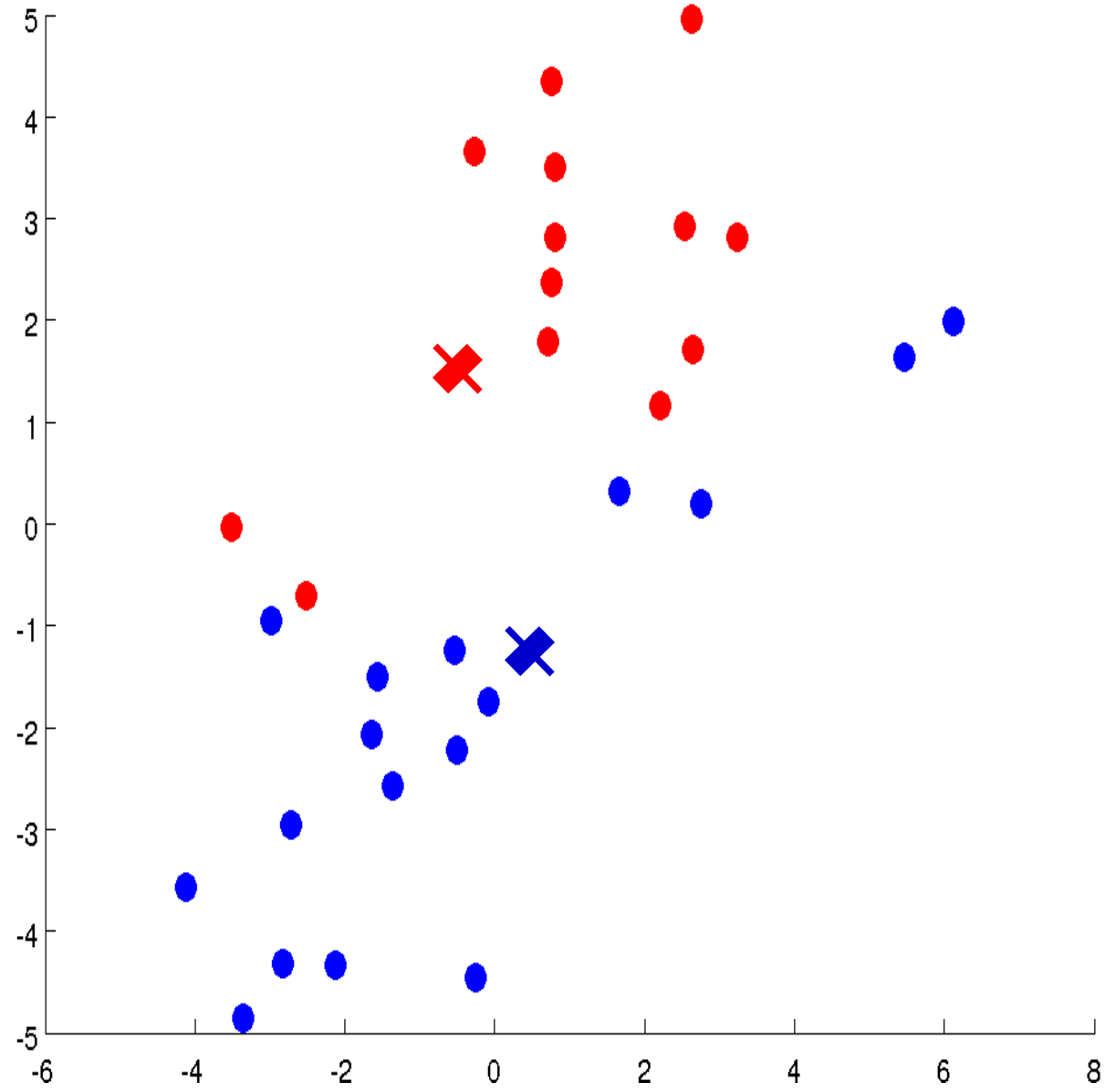
} until convergence criteria is met

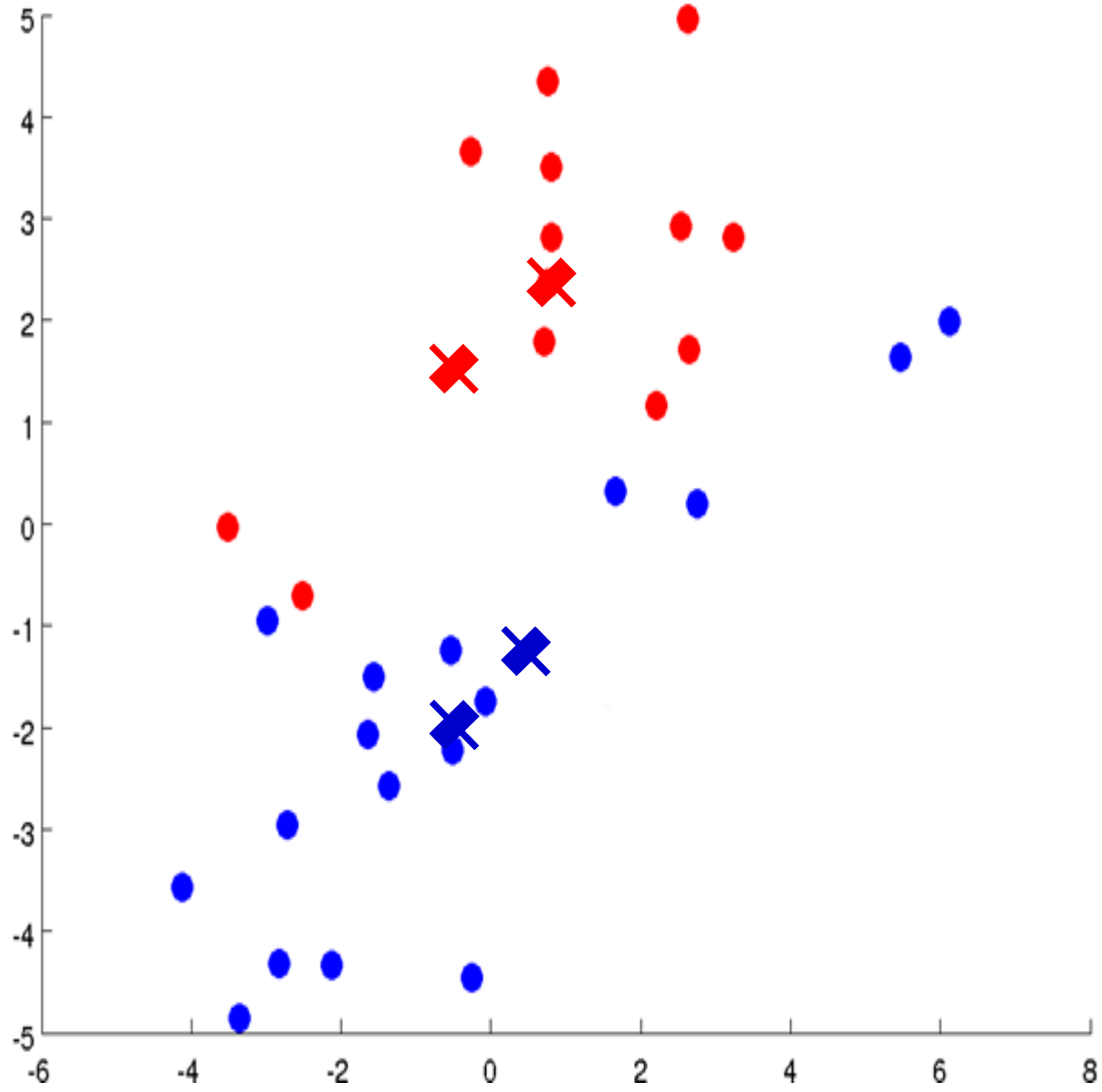

5
4
3
2
1
0
-1
-2
-3
-4
-5
-6
-4
-2
0
2
4
6
8

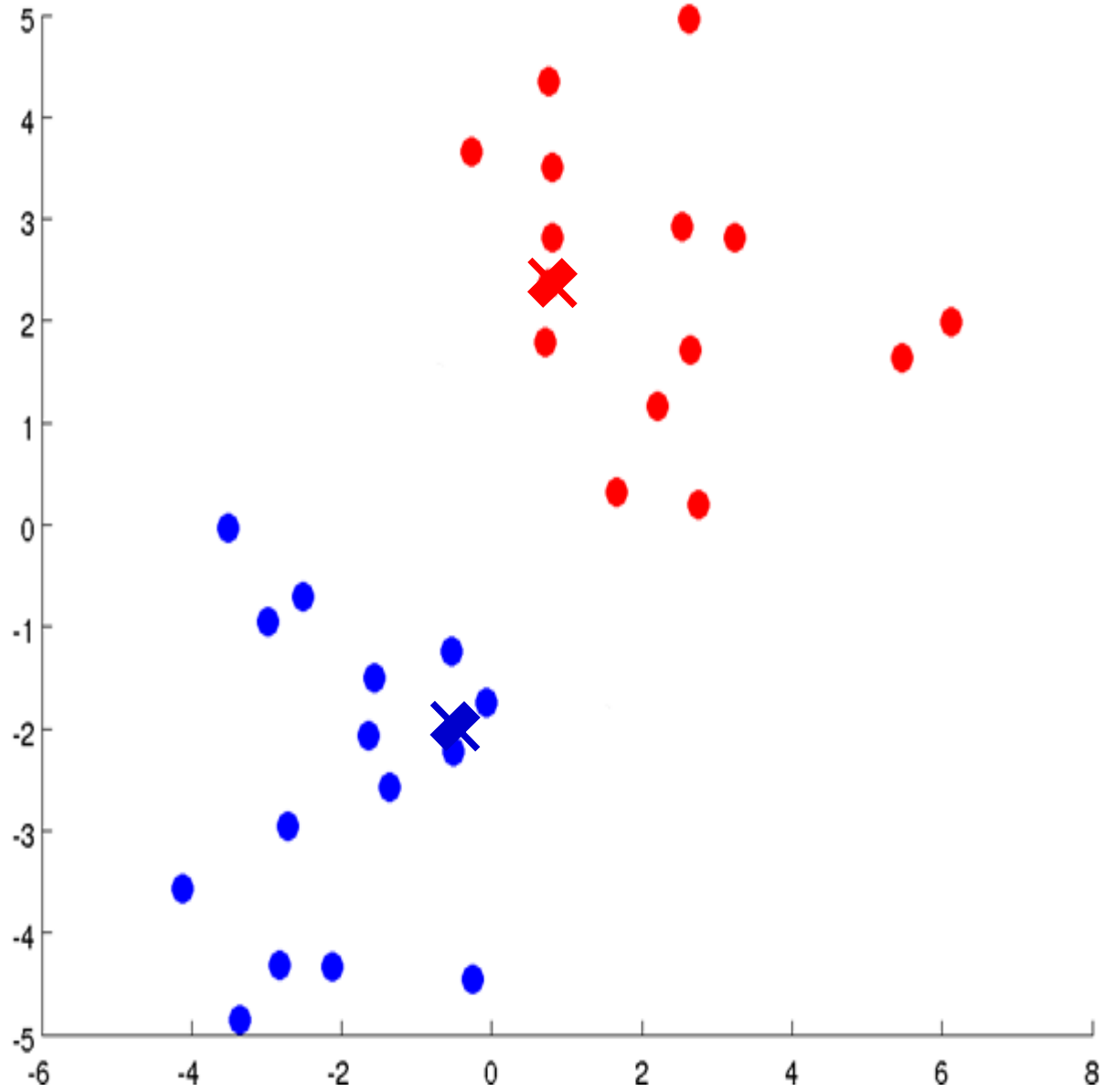

5
4
3
2
1
0
-1
-2
-3
-4
-5
-6
-4
-2
0
2
4
6
8

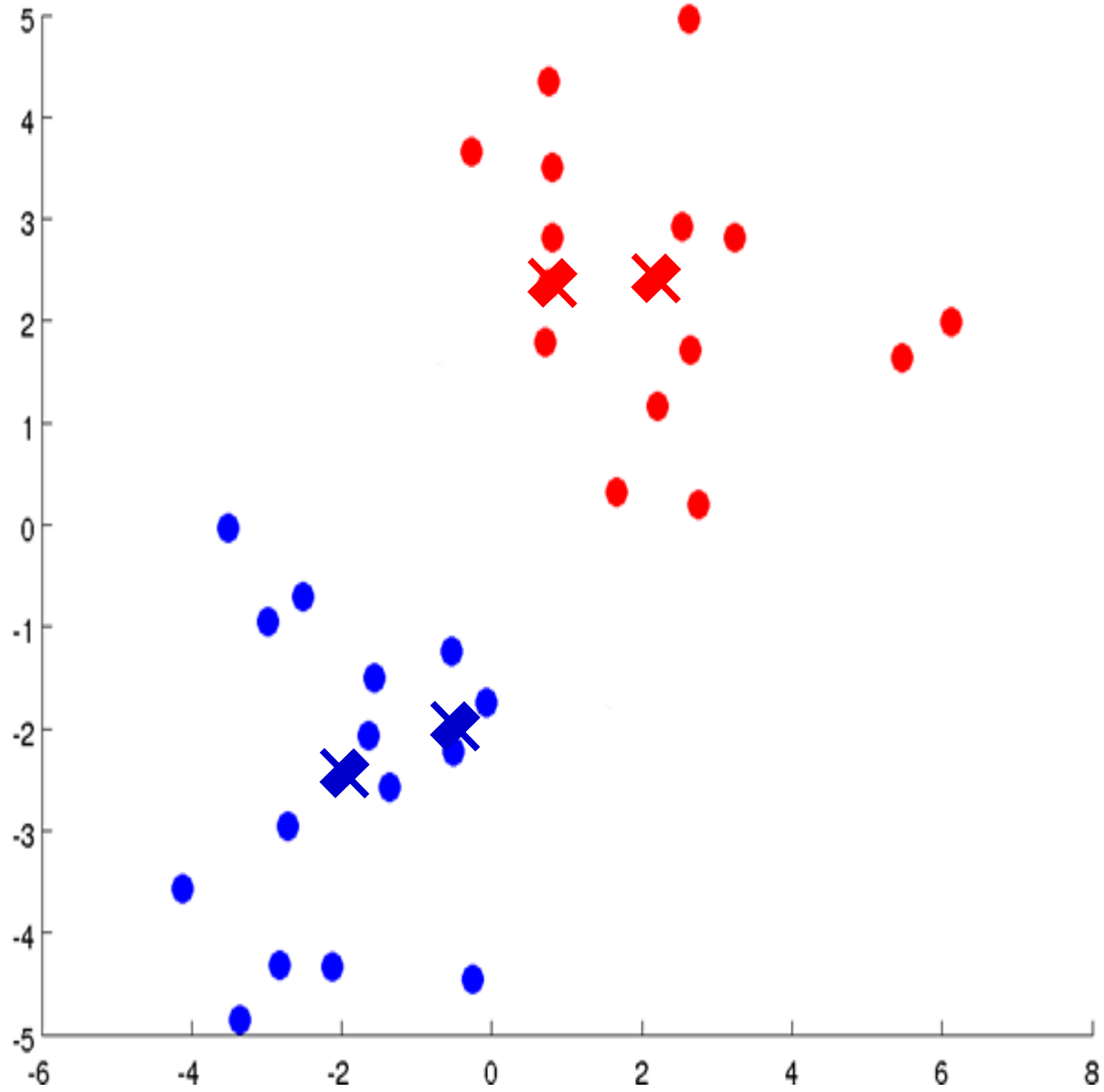

5
4
3
2
1
0
-1
-2
-3
-4
-5
-6
-4
-2
0
2
4
6
8

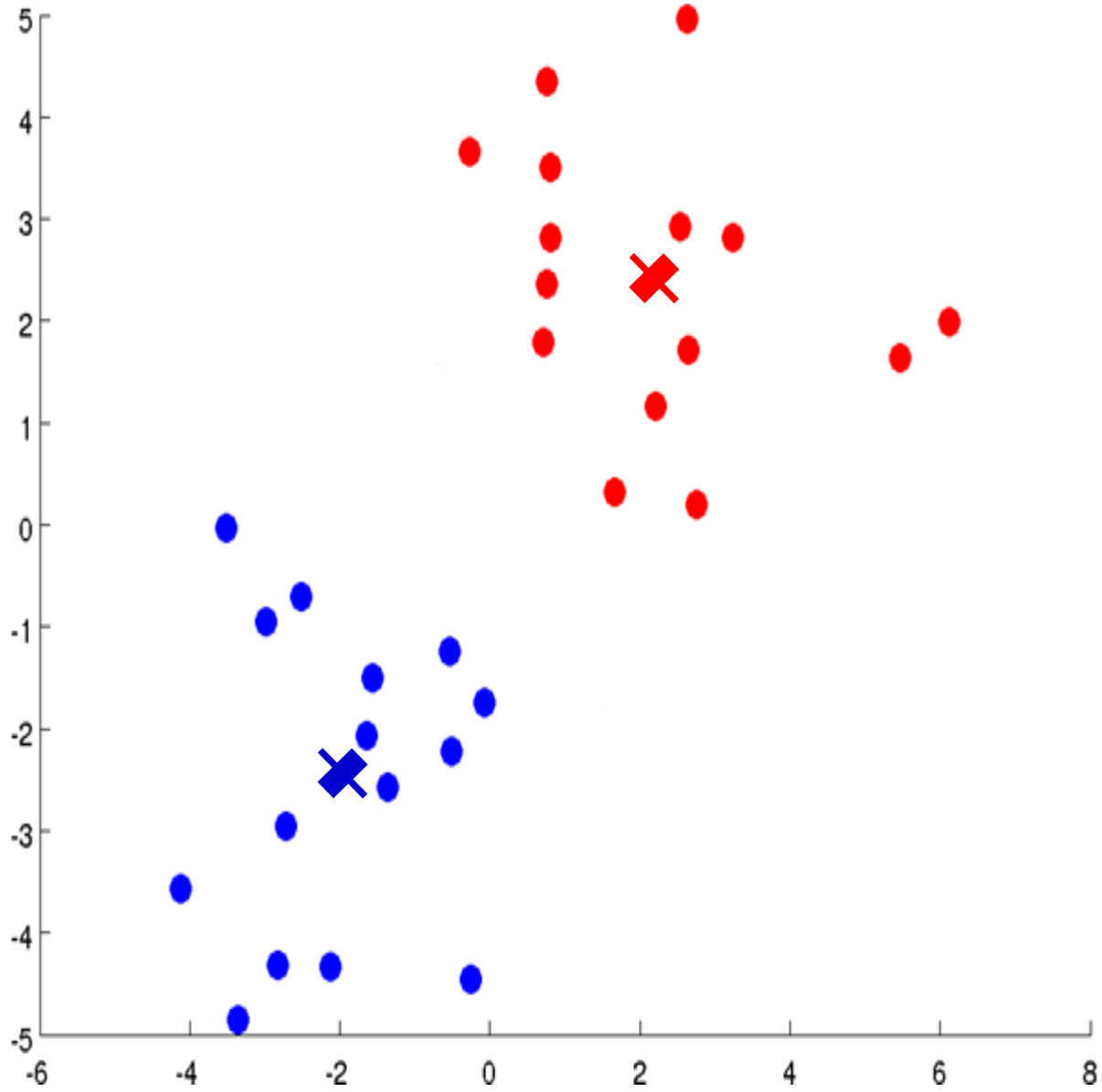

5
4
3
2
1
0
-1
-2
-3
-4
-5
-6
-4
-2
0
2
4
6
8


5
4
3
2
1
0
-1
-2
-3
-4
-5
-6
-4
-2
0
2
4
6
8

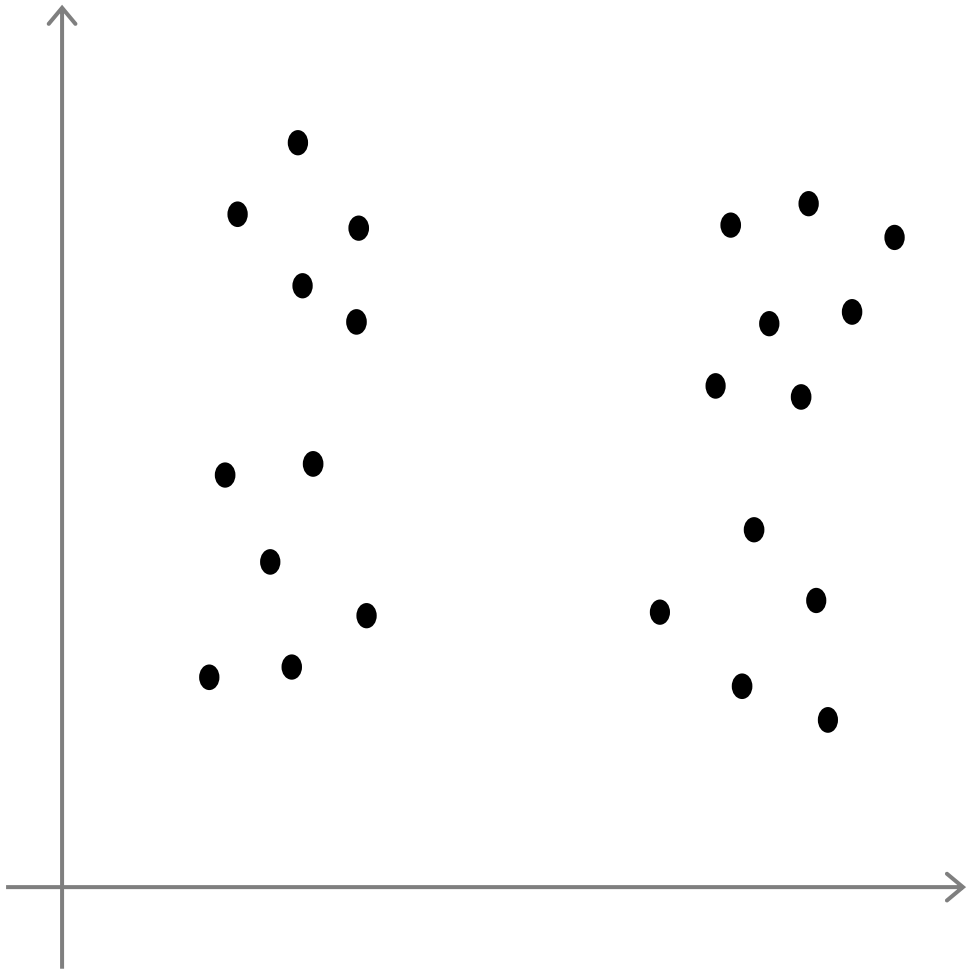

5
4
3
2
1
0
-1
-2
-3
-4
-5
-6
-4
-2
0
2
4
6
8


5
4
3
2
1
0
-1
-2
-3
-4
-5
-6
-4
-2
0
2
4
6
8


5
4
3
2
1
0
-1
-2
-3
-4
-5
-6
-4
-2
0
2
4
6
8

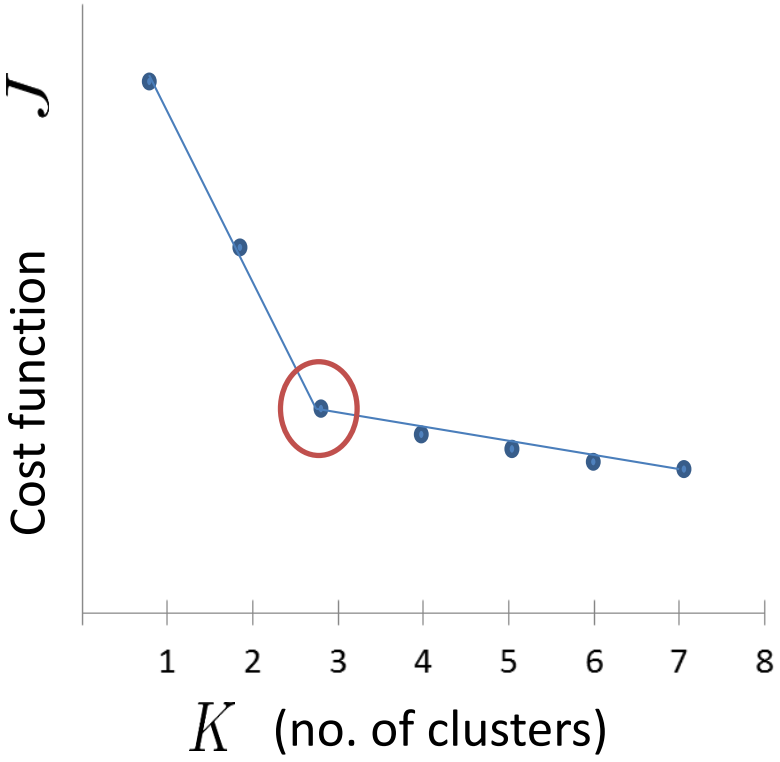

5
4
3
2
1
0
-1
-2
-3
-4
-5
-6
-4
-2
0
2
4
6
8

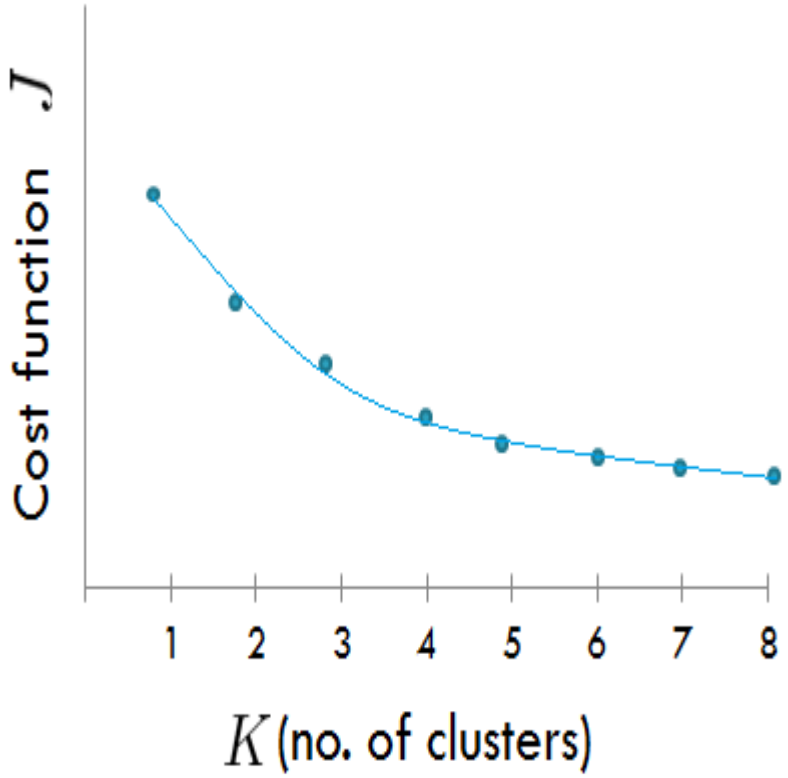
Cual es el correcto valor de K?



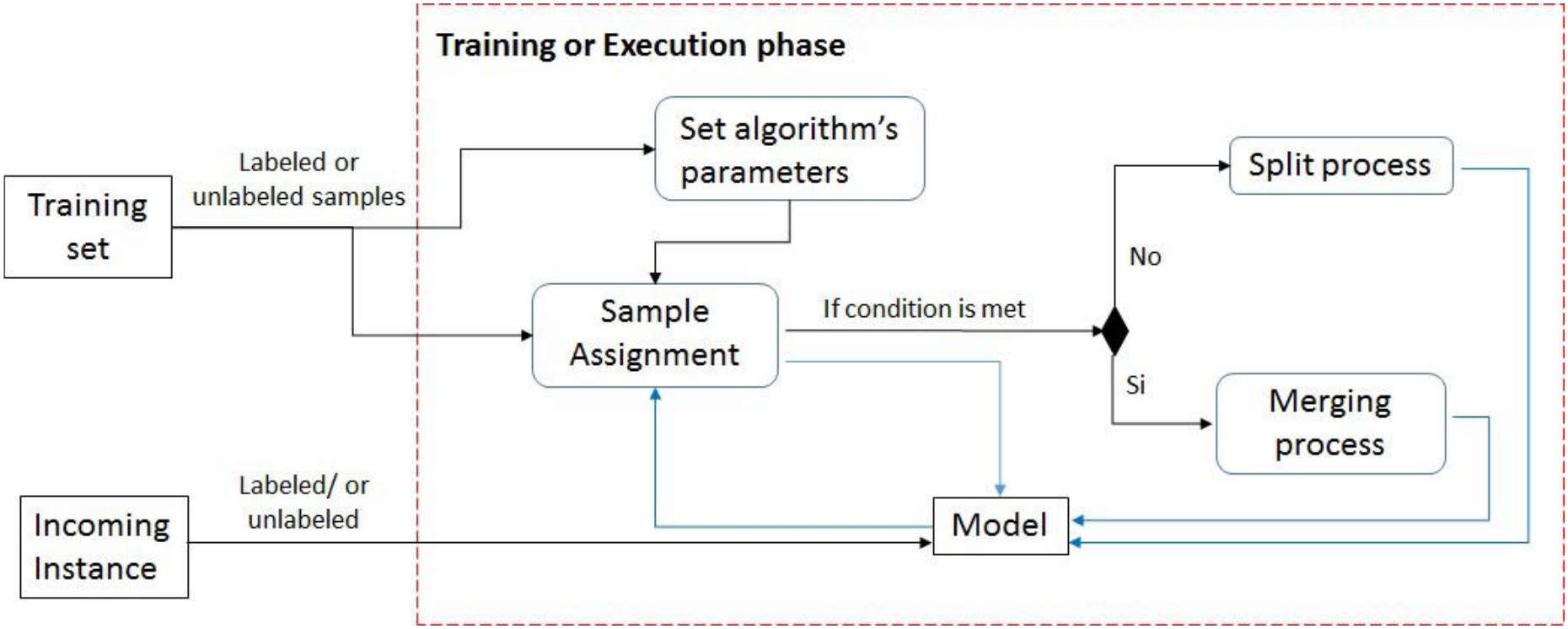
Escogiendo el valor de K

Método del codo:

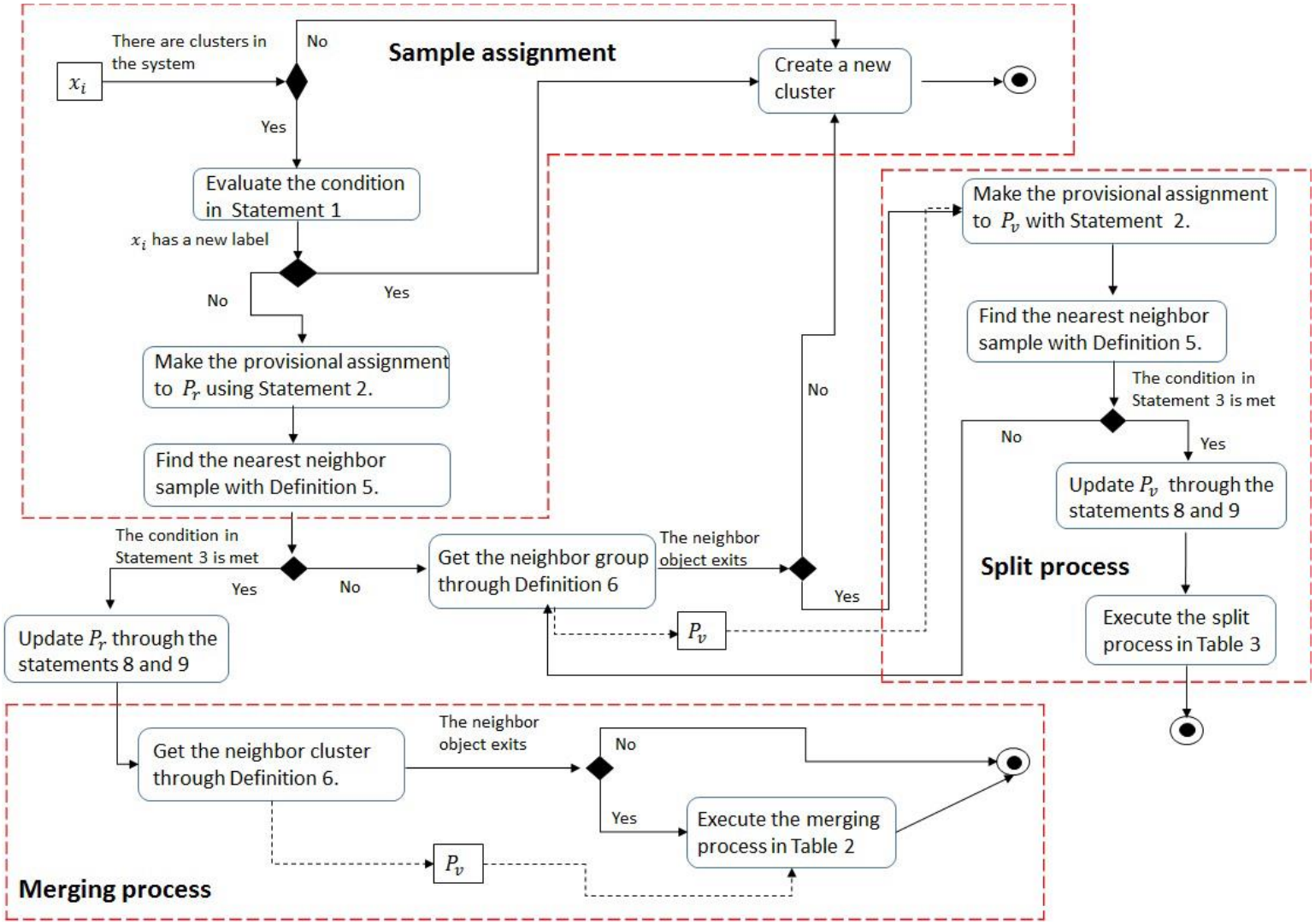




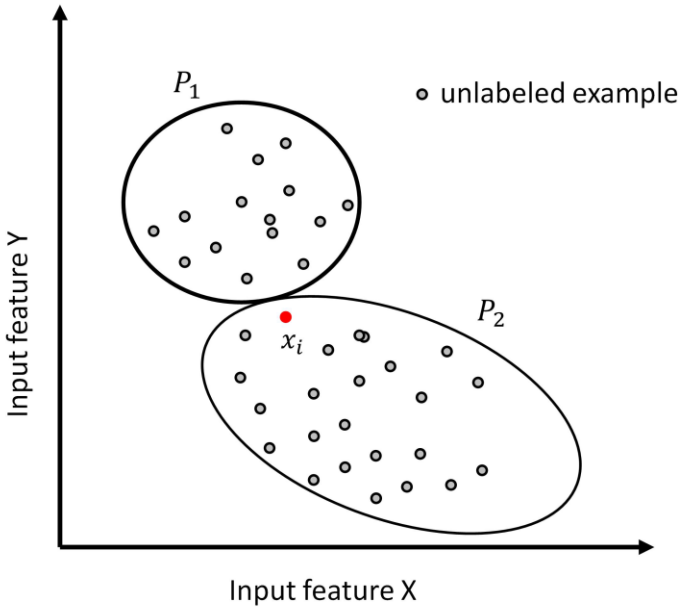
Aprendizaje Híbrido

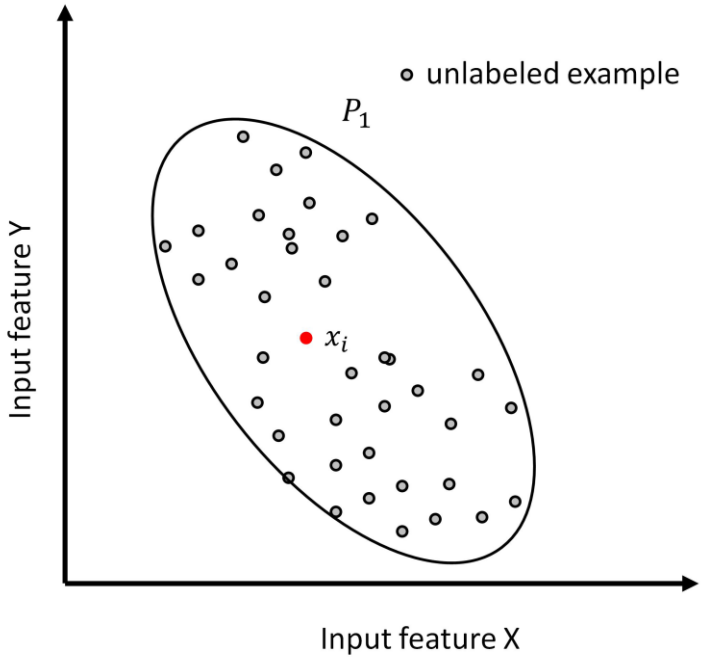


Aprendizaje Híbrido



Merging Procedure

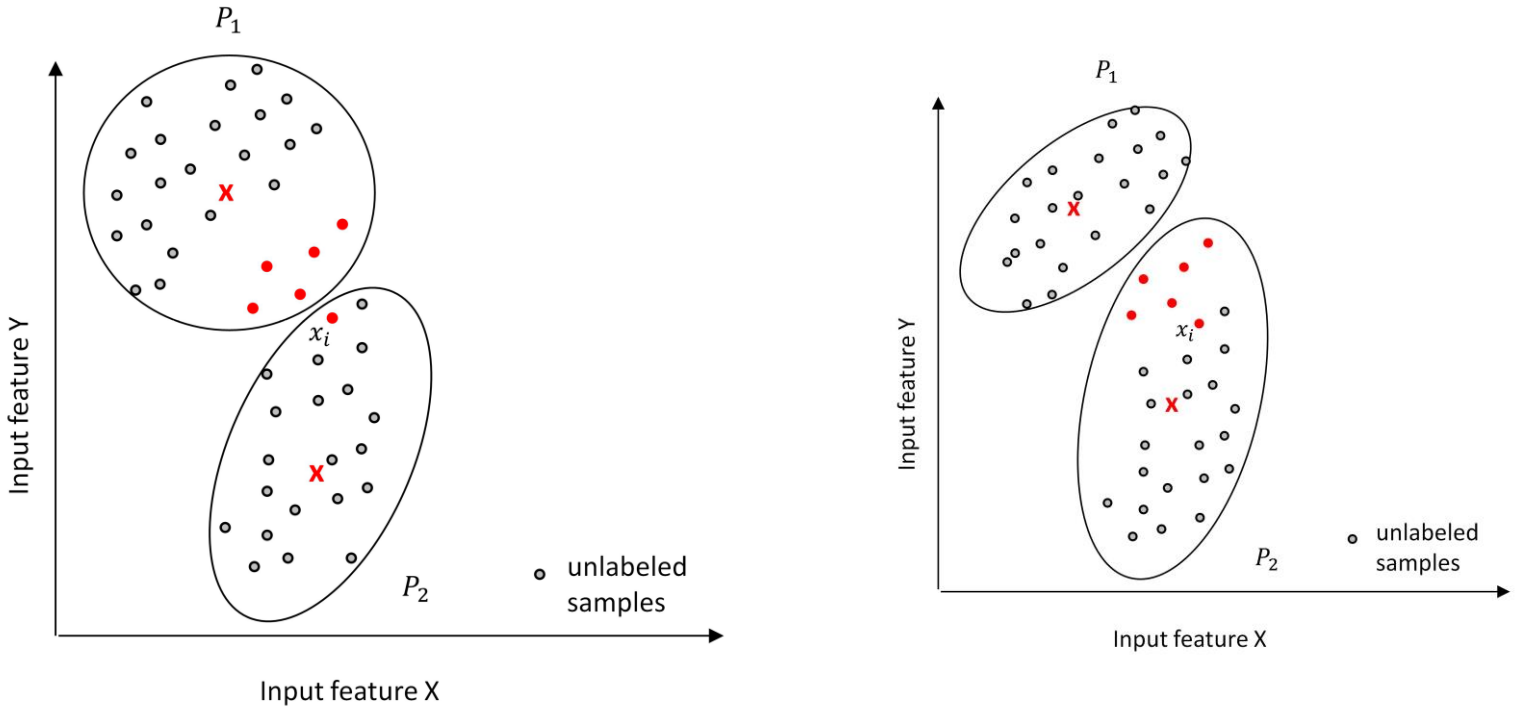




1. New sample xi assigned to P2
2. P1 and P2 are merged after verifying.

Two groups Pi and Pj are merged, if their corresponding Oij and Dij meet defined thresholds th1 and th2 ; i.e., Oij < th1 and Dij < th2.
The density Dij in the overlapped area between two groups Pi and Pj ,
Oij is the degree of overlapping between two groups Pi and Pj .

Split Procedure



1. New incoming sample x_i inducing high similarities to samples in other cluster
2. Separation of neighbors in P_1, as a result of the split process