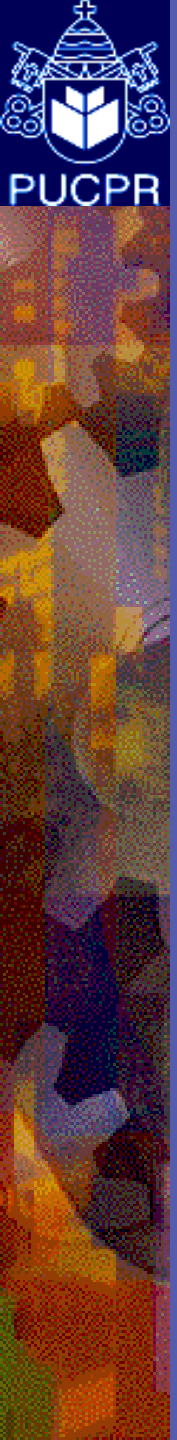


Aprendizagem de Máquina

Algoritmo *k*–Means

Alessandro L. Koerich

Mestrado/Doutorado em Informática (PPGIa)
Pontifícia Universidade Católica do Paraná (PUCPR)

Problema do Agrupamento

- Seja $x = (x_1, x_2, ..., x_d)$ um vetor d dimensional de características

- Seja D um conjunto de x vetores,

$$D = \{ x(1), x(2), ..., x(N) \}$$

- Problema do Agrupamento: Tendo D desejamos agrupar os N vetores em k grupos tal que o agrupamento seja ótimo.

Algoritmo *k*–Means

- O algoritmo *k*–Means ou *k*–Médias é uma técnica iterativa muito simples e poderosa para particionar um conjunto de dados em grupos separados, onde o valor de *k*, deve ser pré–determinado.

- *k* = número de grupos.

- A distância Euclidiana entre os vetores de atributos x_i e os representantes dos *clusters* Θ_i é utilizada como medida de dissimilaridade.

Algoritmo *k*–Means

- Entrada:
  - um conjunto de exemplos D contendo N vetores de atributos d–dimensionais
  - k = número de *clusters*
- Saída:
  - k vetores de média, isto é, os centros dos k clusters)
  - afiliação (*membership*) para cada um dos N vetores de atributos de D.

Algoritmo *k*–Means

The Isodata or k-Means or c-Means Algorithm

- Choose arbitrary initial estimates $\theta_j(0)$ for the θ_j's, $j = 1, \ldots, m$.
- Repeat
  - —For $i = 1$ to N
    - * Determine the closest representative, say θ_j, for $\boldsymbol{x}_i$.
    - * Set $b(i) = j$.
  - —End { For }
  - —For $i = 1$ to m
    - * *Parameter updating:* Determine θ_j as the mean of the vectors $\boldsymbol{x}_i \in X$ with $b(i) = j$.
  - —End { For }.
- Until no change in θ_j's occurs between two successive iterations.

An advantage of this algorithm is its computational simplicity. Also, as with all the algorithms that use point representatives, isodata is suitable for recovering compact clusters.

Algoritmo *k*–Means

1. Escolha estimativas iniciais arbitrárias $\Theta_j(0)$ para os Θ_j's, j=1, ..., m (i.e. para os centróides dos m clusters.
2. Repetir
   - Para i = 1 a N
     - Determine o representante mais próximo, isto é, Θ_j (i.e., o centróide mais próximo) de x_i.
     - Faça $b(i) = j$
   - Fim
   - Para i = 1 a m
     - Atualização de parâmetros: Determinar Θ_j como a média dos vetores $x_i \in X$ com $b(i) = j$
   - Fim
3. Até que não ocorra mudanças em Θ_j entre duas iterações sucessivas.

Algoritmo *k*–Means

- A vantagem deste algoritmo é sua simplicidade computacional.
- Na prática, existem diversas variações deste algoritmo que empregam operações de particionamento, união e descarte dos *clusters* resultantes.

Algoritmo *k*–Means

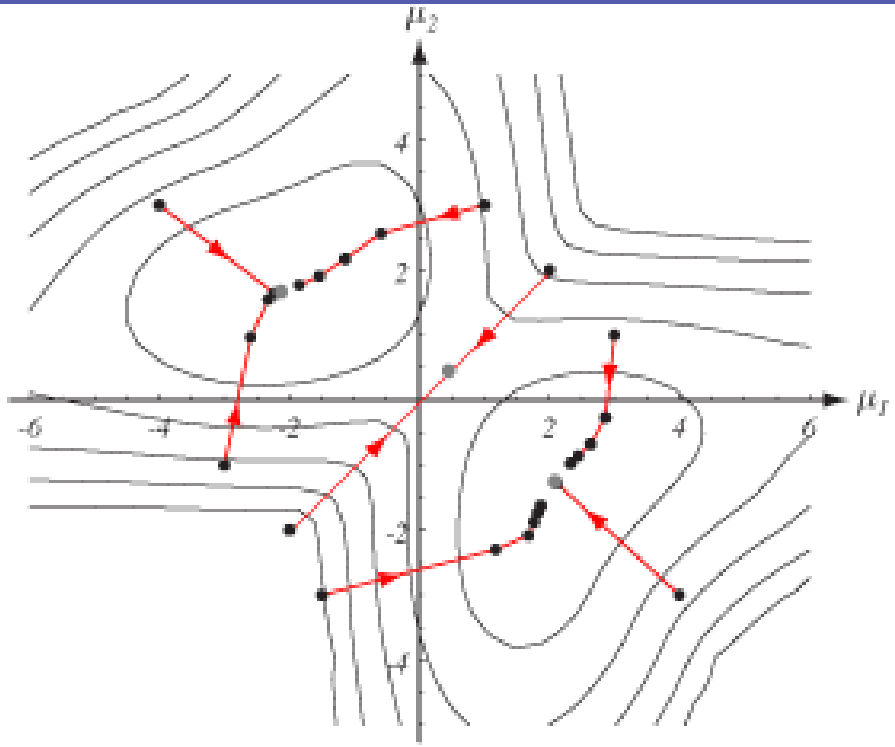


FIGURE 10.2. The *k*-means clustering procedure is a form of stochastic hill climbing in the log-likelihood function. The contours represent equal log-likelihood values for the one-dimensional data in Fig. 10.1. The dots indicate parameter values after different iterations of the *k*-means algorithm. Six of the starting points shown lead to local maxima, whereas two (i.e., $\mu_1(0) = \mu_2(0)$) lead to a saddle point near $\boldsymbol{\mu} = \mathbf{0}$. From: Richard O. Duda, Peter E. Hart, and David G. Stork, *Pattern Classification*. Copyright © 2001 by John Wiley & Sons, Inc.

Algoritmo *k*–Means

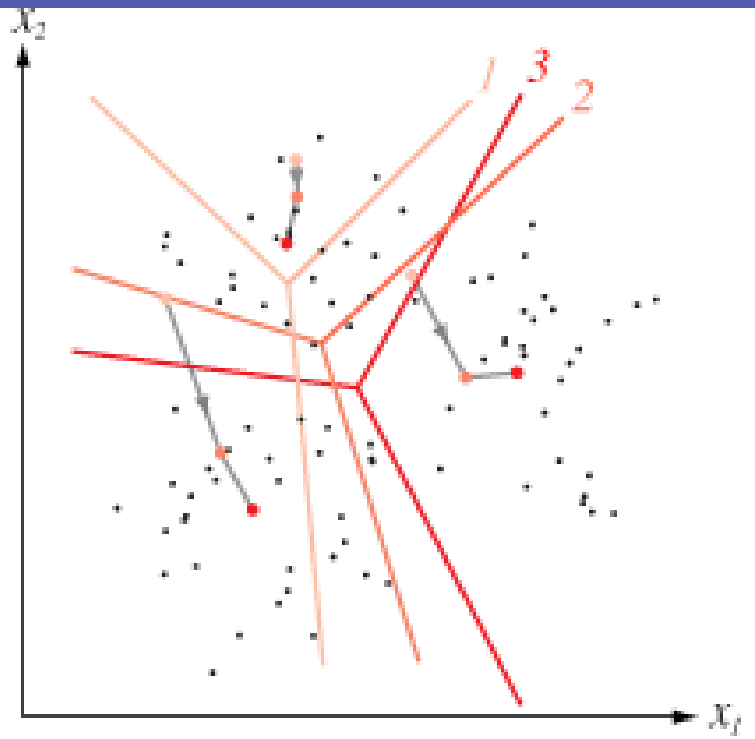


FIGURE 10.3. Trajectories for the means of the *k*-means clustering procedure applied to two-dimensional data. The final Voronoi tesselation (for classification) is also shown—the means correspond to the "centers" of the Voronoi cells. In this case, convergence is obtained in three iterations. From: Richard O. Duda, Peter E. Hart, and David G. Stork, *Pattern Classification*. Copyright © 2001 by John Wiley & Sons, Inc.

Algoritmo *k*–Means

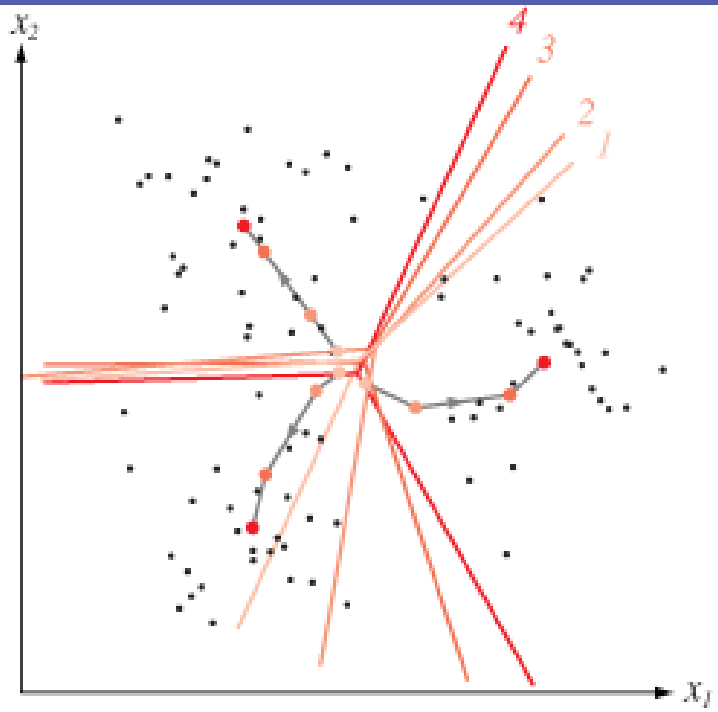
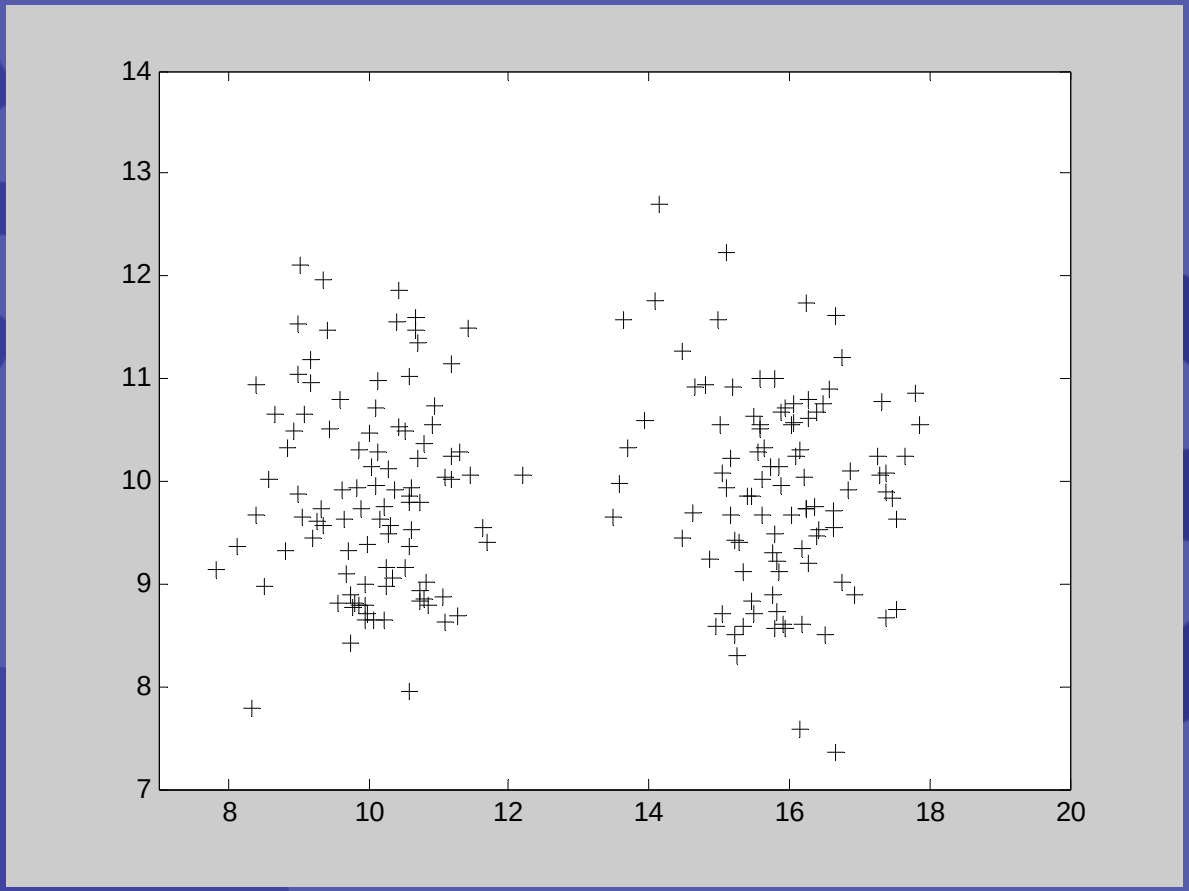


FIGURE 10.4. At each iteration of the fuzzy *k*-means clustering algorithm, the probability of category memberships for each point are adjusted according to Eqs. 32 and 33 (here $b = 2$). While most points have nonnegligible memberships in two or three clusters, we nevertheless draw the boundary of a Voronoi tesselation to illustrate the progress of the algorithm. After four iterations, the algorithm has converged to the red cluster centers and associated Voronoi tesselation. From: Richard O. Duda, Peter E. Hart, and David G. Stork, *Pattern Classification*. Copyright © 2001 by John Wiley & Sons, Inc.

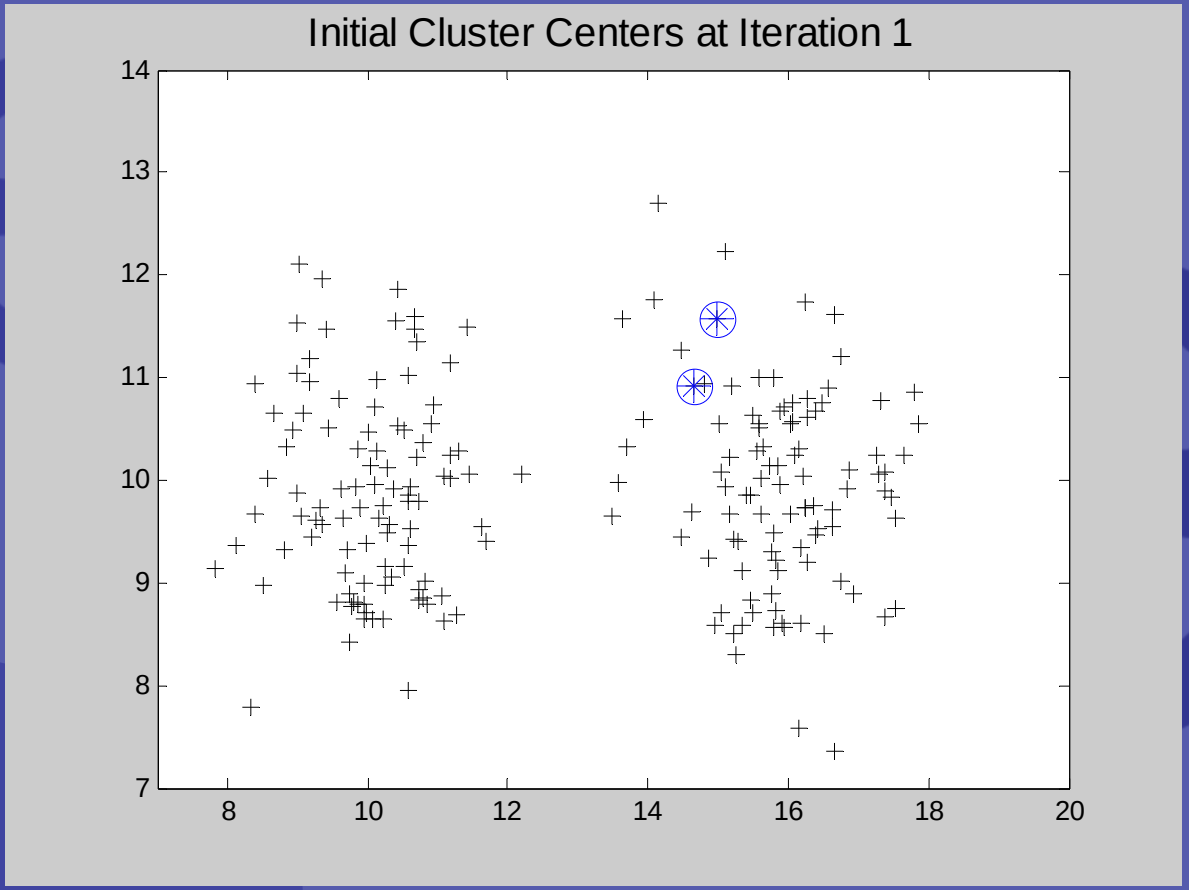
Algoritmo *k*–Means: Exemplo

- Conjunto de exemplos *D*: (2-dimensional)



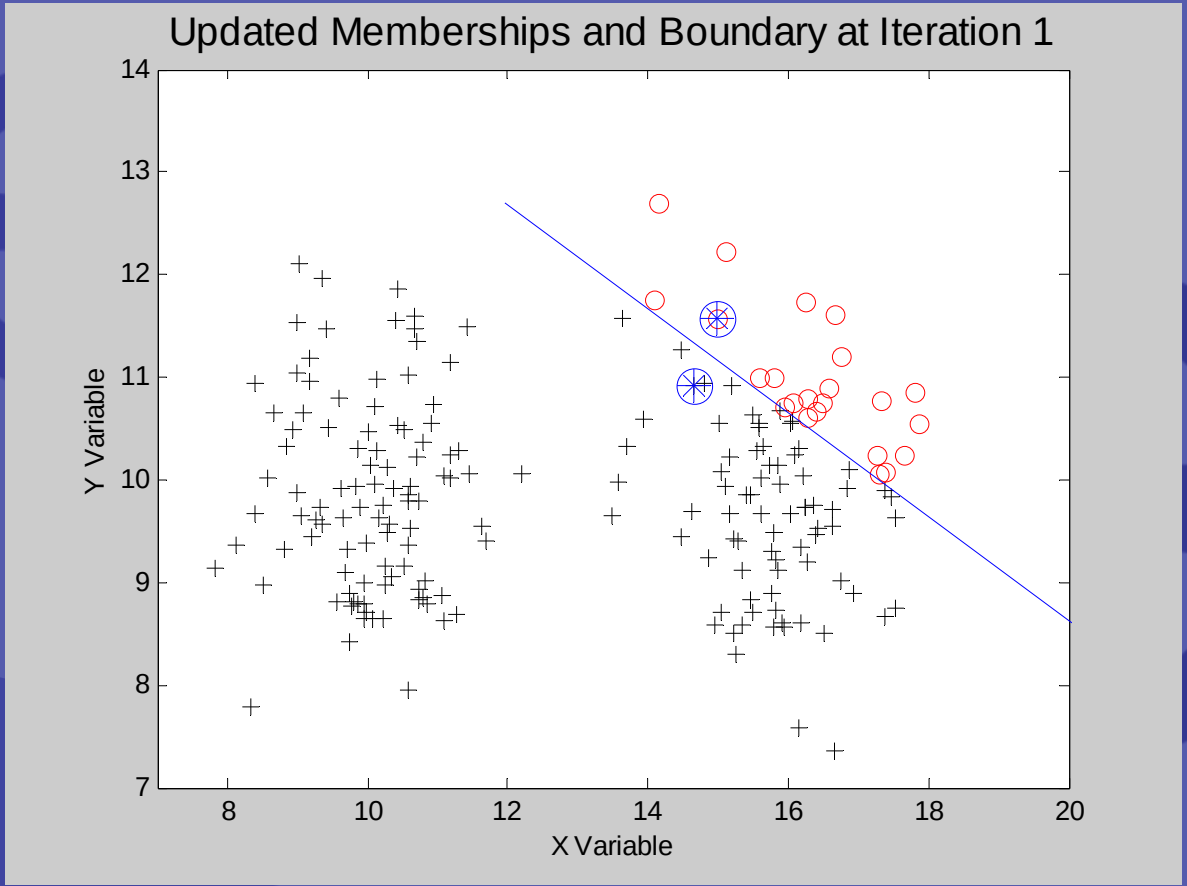
Algoritmo *k*–Means: Exemplo

- Centros iniciais dos clusters (k=2)



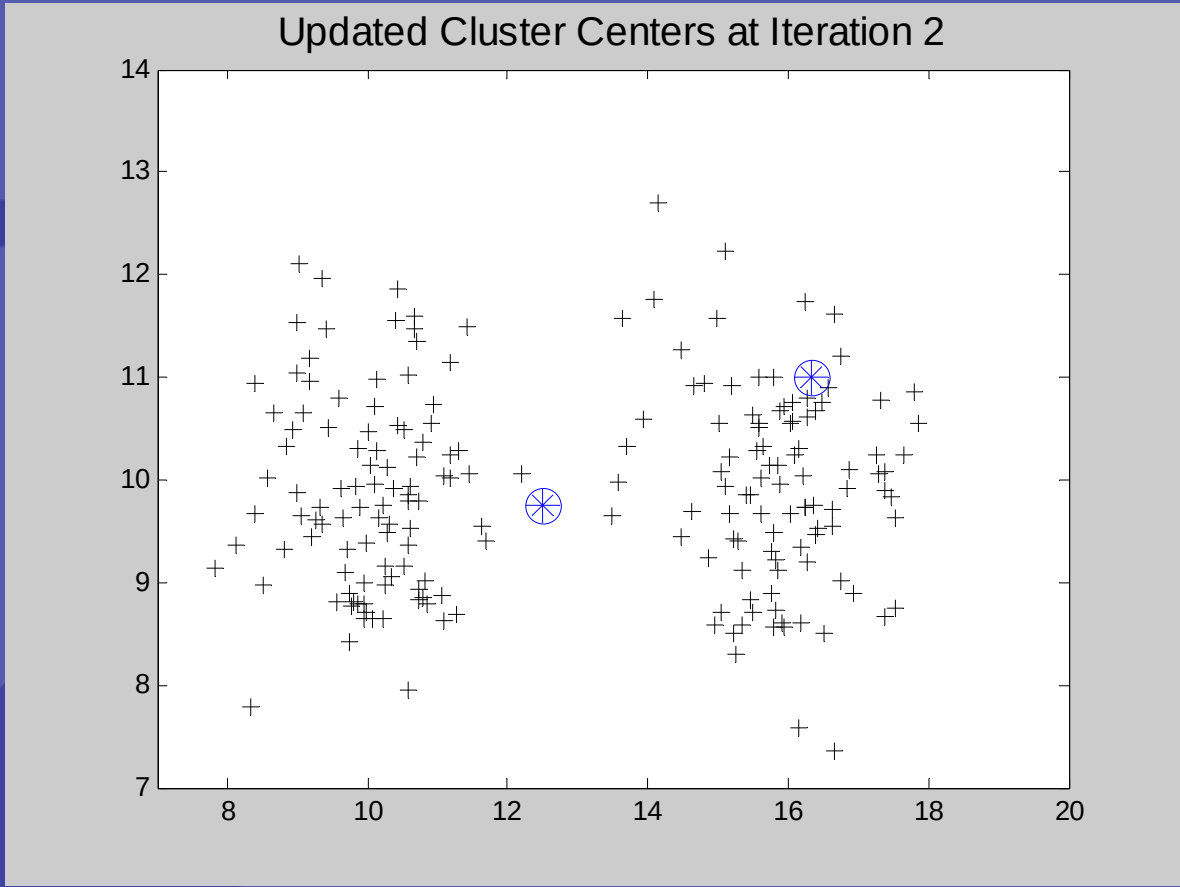
Algoritmo *k*–Means: Exemplo

- Atualizando afiliações (iteração 1)



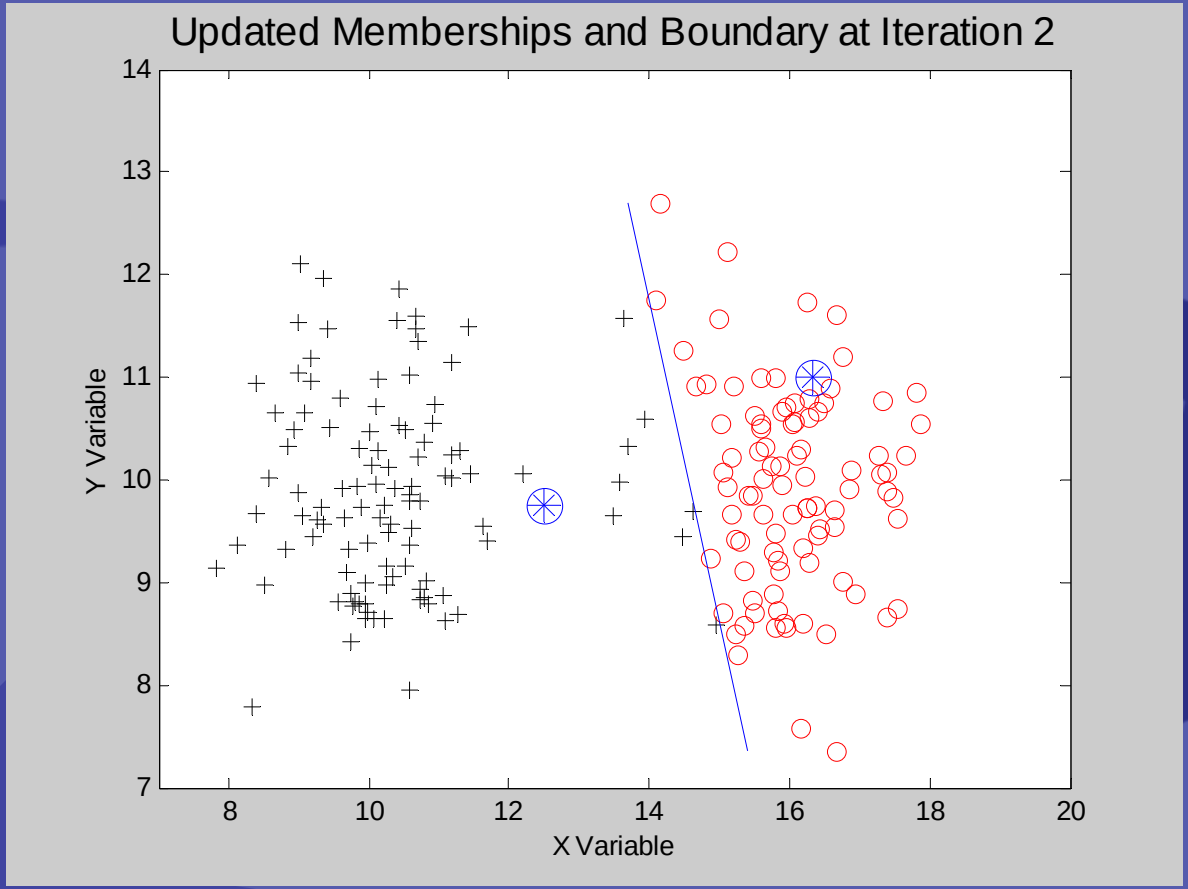
Algoritmo *k*–Means: Exemplo

- Atualizando centros (iteração 2)



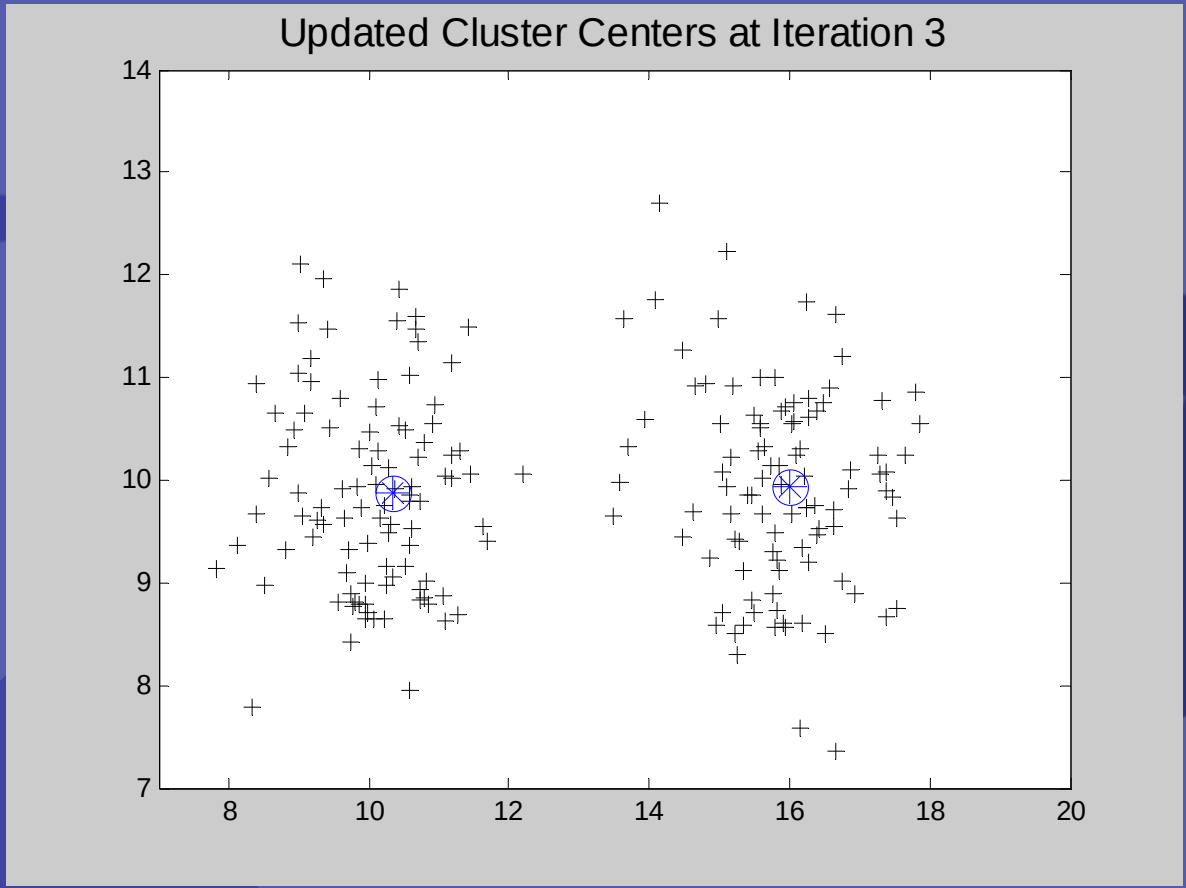
Algoritmo *k*–Means: Exemplo

- Atualizando afiliações (iteração 2)



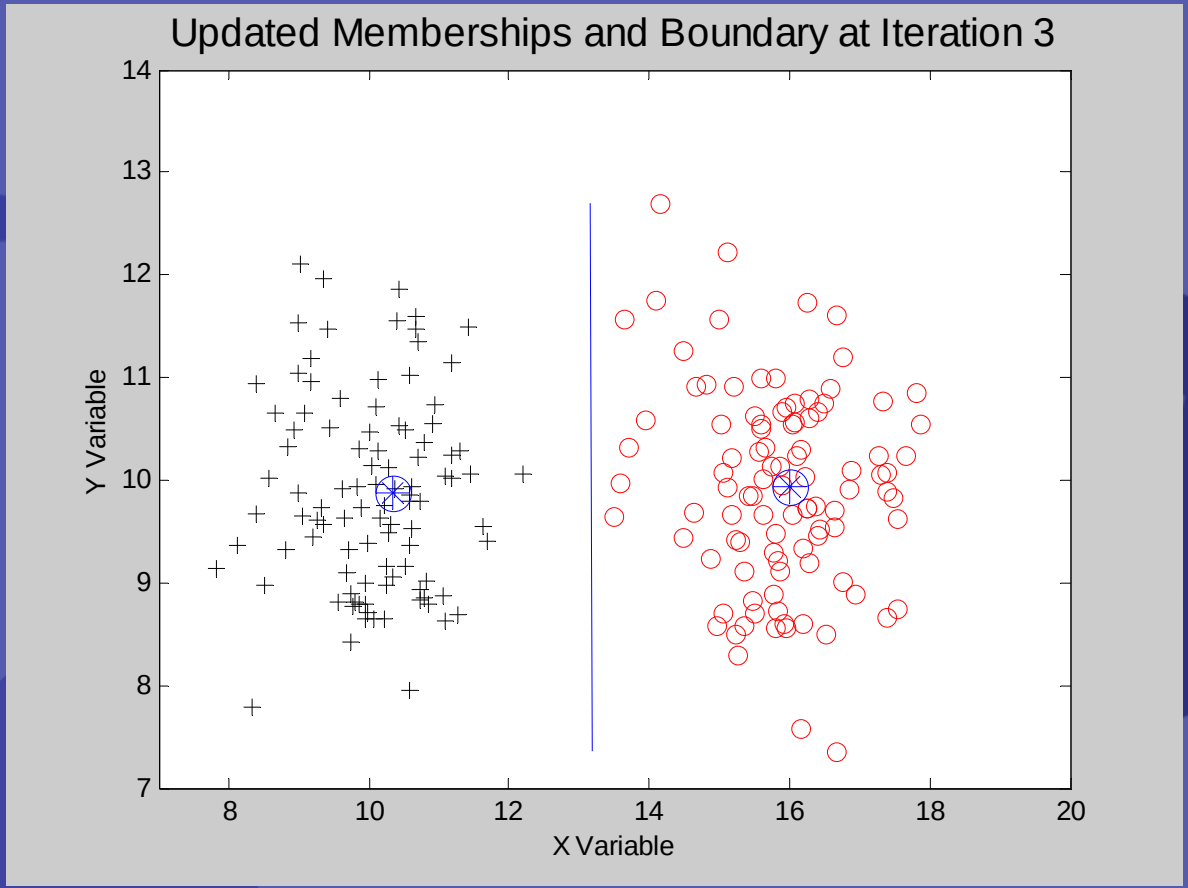
Algoritmo *k*–Means: Exemplo

- Atualizando centros (iteração 3)



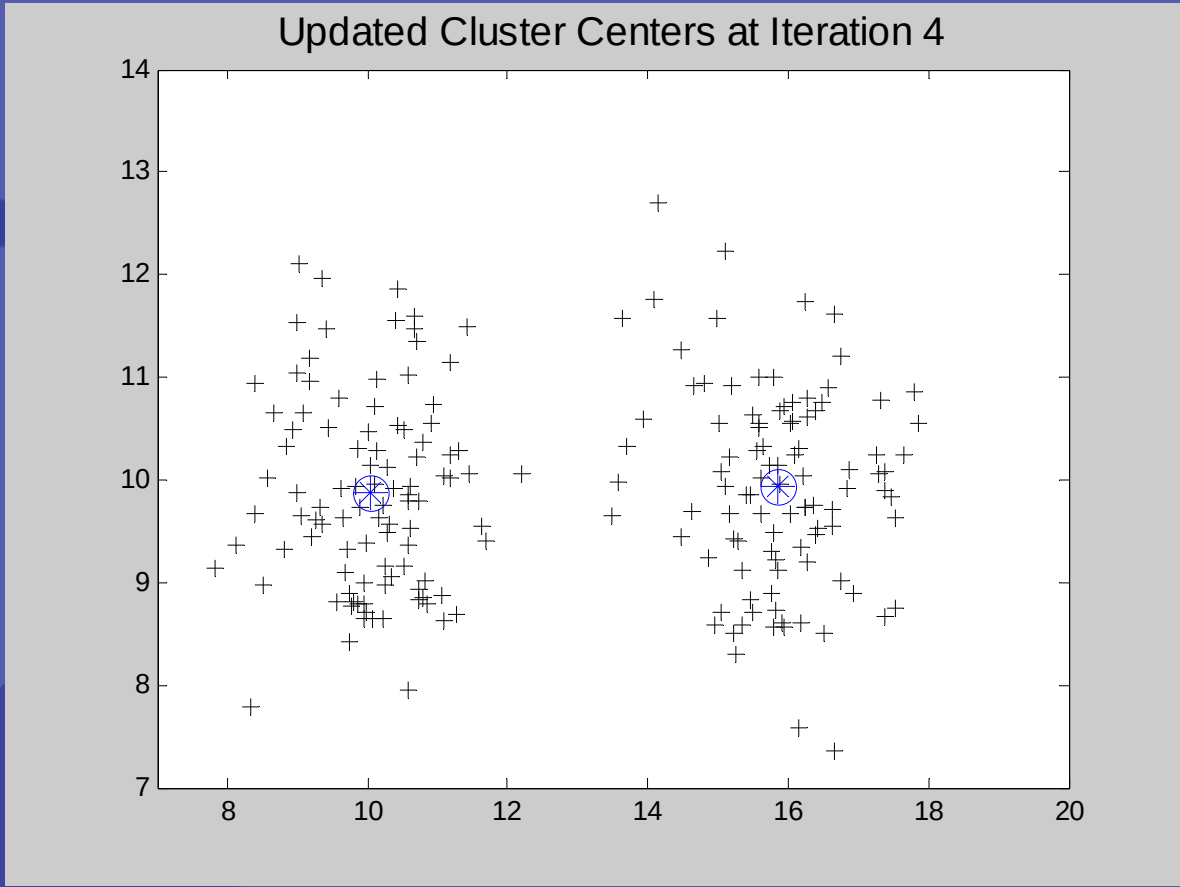
Algoritmo *k*–Means: Exemplo

- Atualizando afiliação



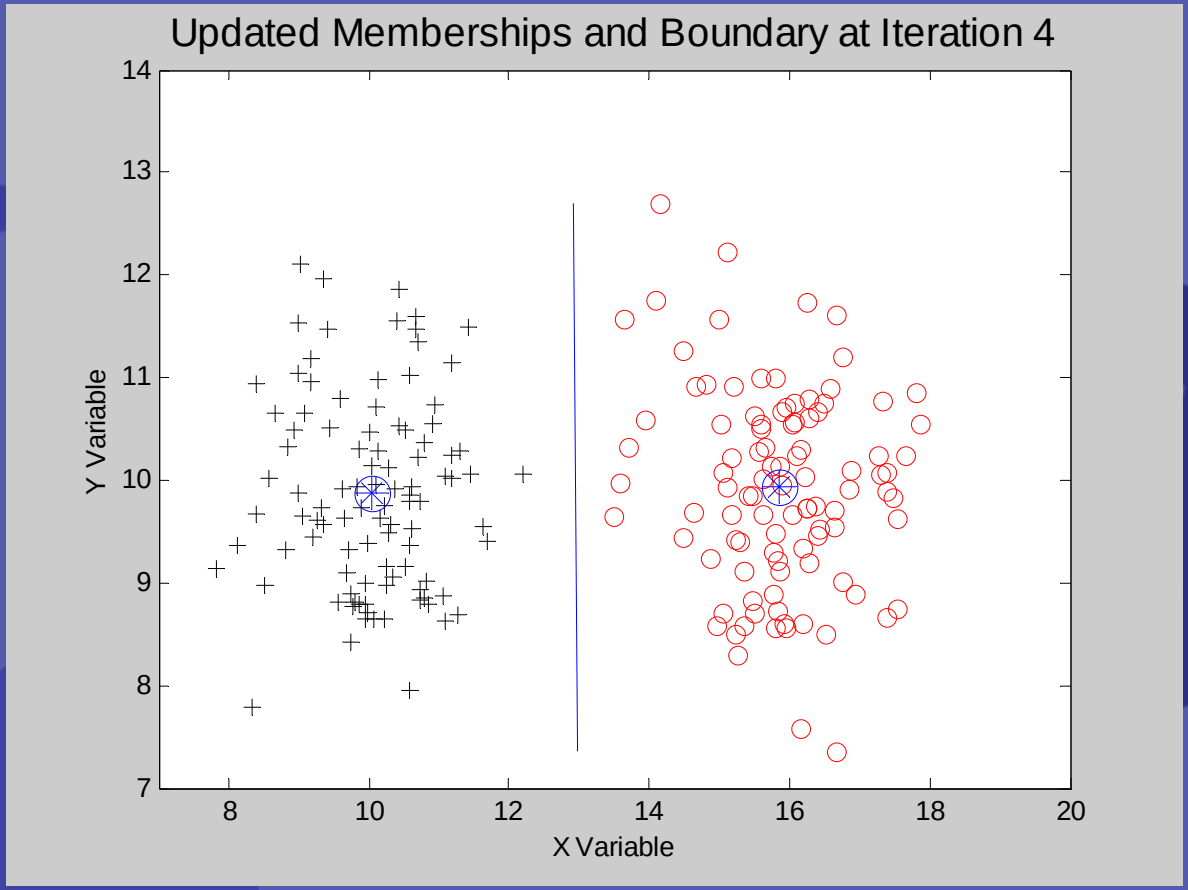
Algoritmo *k*–Means: Exemplo

- Atualizando centros (iteração 4)



Algoritmo *k*–Means: Exemplo

- Atualizando afiliação (iteração 4)



Comentários sobre *k*–Means

- Computacionalmente simples.
- O Erro Quadrático Total (TSE) decresce (ou converge) a cada iteração.
- Ele encontra um TSE mínimo global?
  - Não necessariamente
  - Os resultados são sensíveis ao ponto inicial (inicialização dos centróides)
  - Na prática, podemos executá–lo a partir de múltiplos pontos de partida e pegar a solução com menor erro (TSE).

Exemplo *k*–Means

- Exemplo: Agrupando pixels em uma imagem
  - Podemos usar o algoritmo *k*–Means para agrupar a intensidade dos pixels de uma imagem em *k* clusters.
  - É uma maneira simples de segmentar uma imagem em *k* regiões.
  - É um método mais automático do que um limiar escolhido manualmente.

Exemplo *k*–Means

- Como fazer?
  - Tamanho (matriz de pixels) = *m* x *n*
  - Converter para um vetor com (*m* x *n*) linhas e 1 coluna
  - Executar o algoritmo *k*–Means com entrada = vetor de intensidades.
  - Atribuir para cada pixel a “cor ou nível de cinza” do *cluster* a que ele for atribuído.

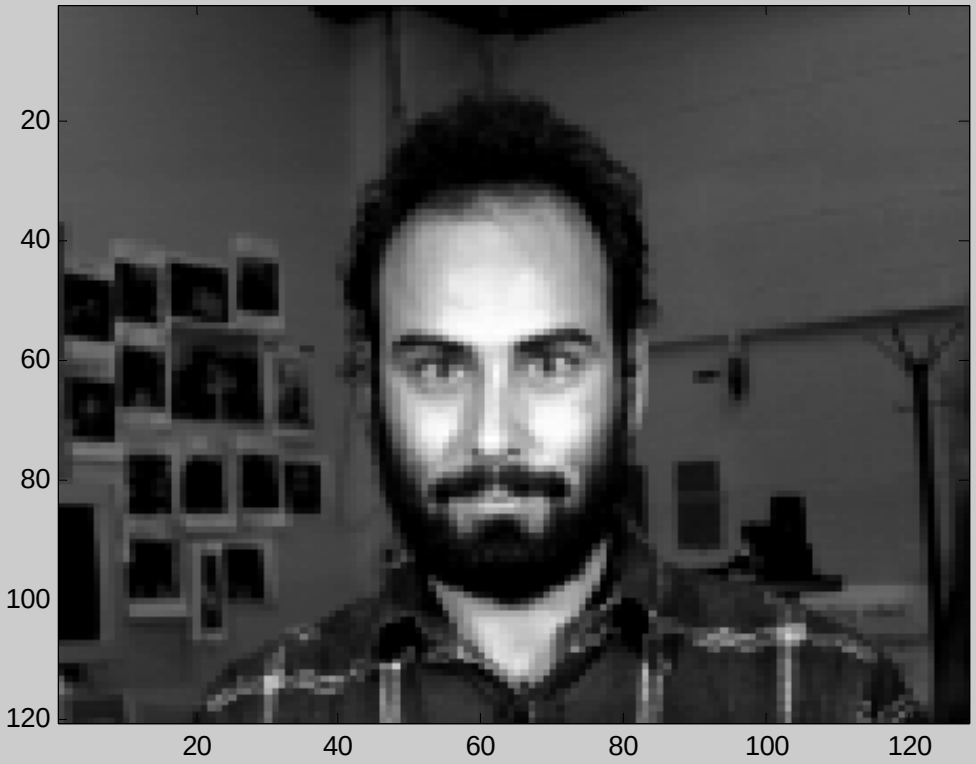
Exemplo: Imagem Original

Exemplo: *k*–Means (*k*=2)

Exemplo: *k*–Means (*k*=3)

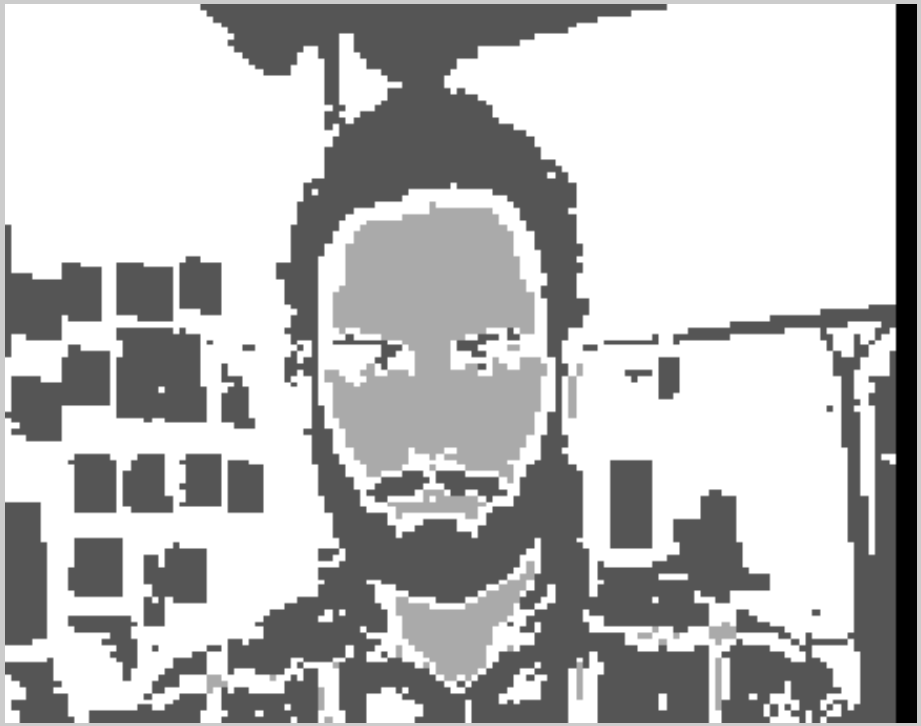
Exemplo: *k*–Means (*k*=5)

Exemplo: Imagem Original

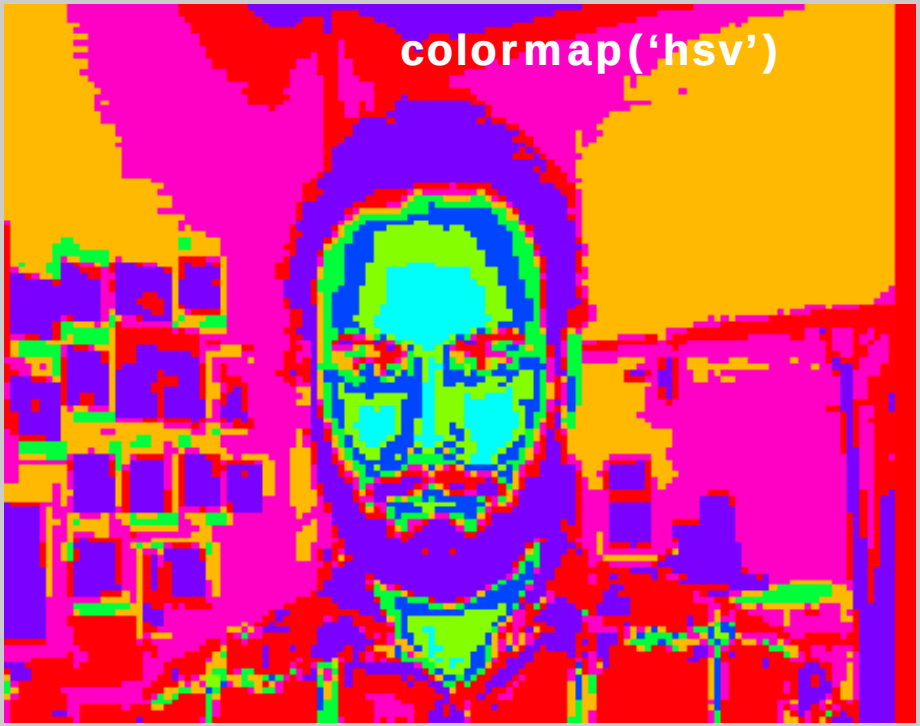


Exemplo: *k*–Means (*k*=2)

Exemplo: *k*–Means (*k*=3)



Exemplo: *k*–Means (*k*=8)



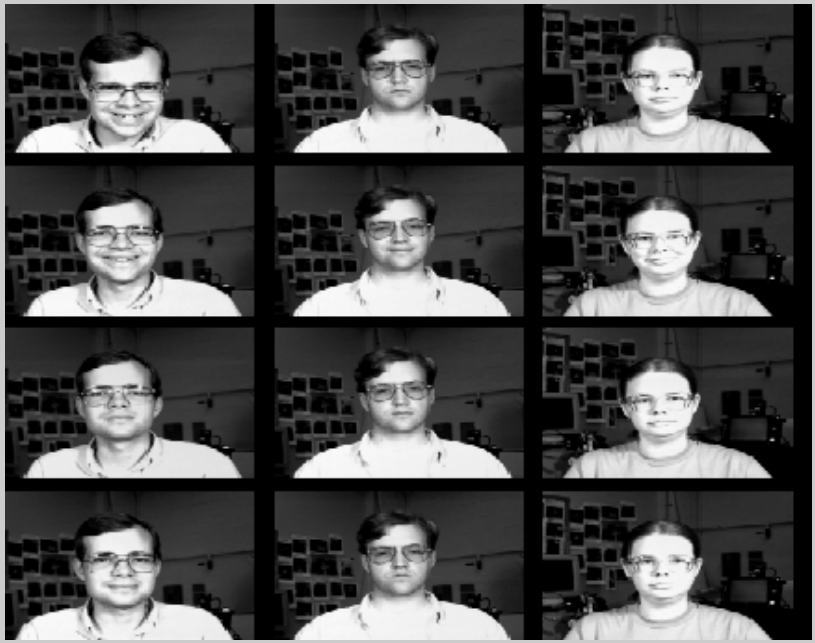
Agrupando Imagens

- Exemplo: Podemos também agrupar conjuntos de imagens
- Cada vetor = uma imagem inteira (dimensão m x n)
- N imagens de tamanho m x n
  - Execute k–*Means*
  - k–*Means* está agora agrupando em um espaço de dimensão m x n
  - k–Means agrupará as imagens em k grupos

Agrupando Imagens

- 5 primeiros indivíduos, $k = 2$

Cluster 1

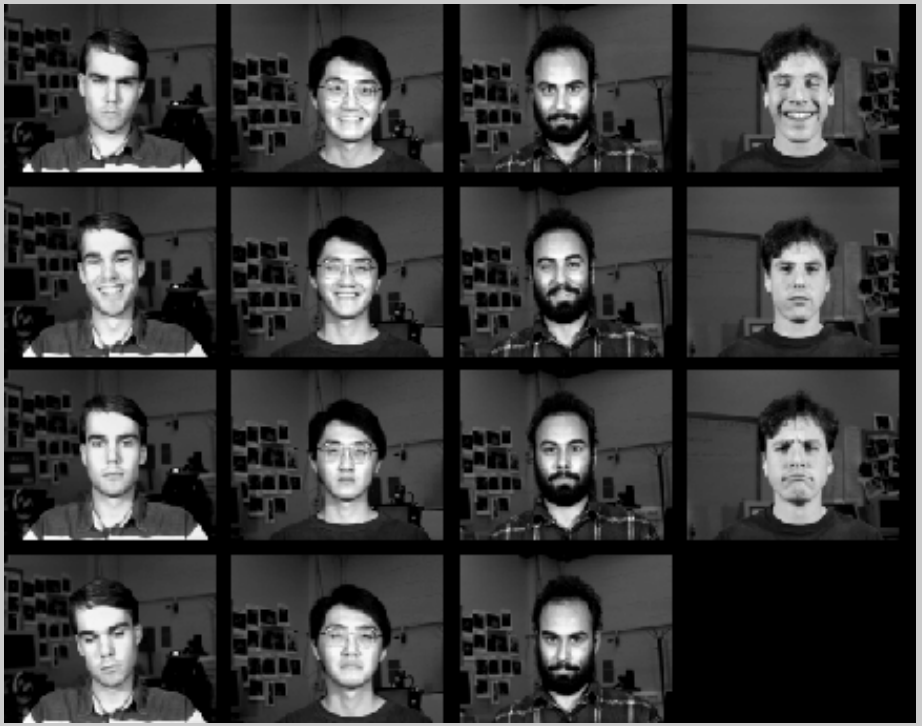


Cluster 2

Agrupando Imagens

- 5 segundos indivíduos, $k = 2$

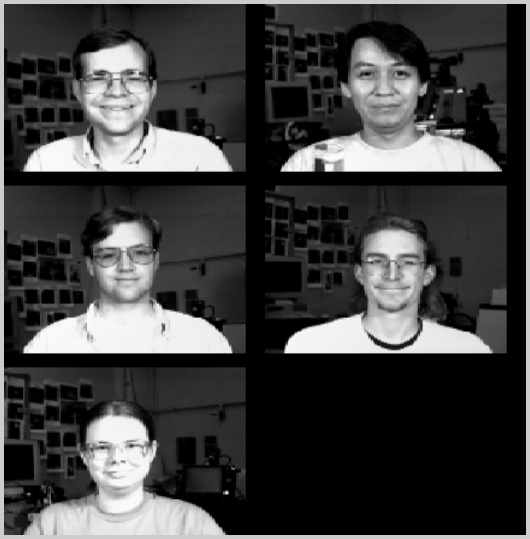
Cluster 1



Cluster 2

Agrupando Imagens

- Todos indivíduos faces alegres, $k = 5$



Resumo

- *k–Means* é um algoritmo de aprendizagem não supervisionada.
- Ele não classifica, mas agrupa vetores de atributos similares, isto é, coloca em um mesmo agrupamento vetores similares.
- Por ser um bastante simples e funcionar bem na prática, ele é um principais e mais usados métodos de agrupamento.