Inteligência Artificial

Categorias de Aprendizado de Máquina

Prof. Saulo Popov Zambiasi
saulopz@gmail.com

Aprendizado de Máquina

- 1959, Arthur Lee Samuel, engenheiro do MIT: **“um campo de estudo que dá aos computadores a habilidade de aprender sem terem sido programados para tal”**.
- Vertente da IA que fornece aos sistemas a capacidade de aprendizagem, a partir de dados e experiências (*machine learning*).
- Um sistema com aprendizado de máquina é capaz de identificar padrões, oferecer respostas e tomar decisões com o mínimo de participação humana.
- Permite que o sistema:
  - faça o processamento de dados
  - aprenda como se comportar
  - aprenda quais respostas dar a comandos específicos.

Aprendizado de Máquina

- O sistema, baseado em sua própria experiência, aperfeiçoa seu comportamento de forma autônoma, sem a necessidade de um profissional humano.
- Quanto mais dados são inseridos, mais “bem treinado” e inteligente ele fica.
- As respostas do sistema não são previamente programadas;
- De forma independente, o sistema consegue:
  - reconhecer e analisar dados
  - identificar padrões
  - tomar decisões.

Categorias de Aprendizado de Máquina

- **Aprendizado Supervisionado**
  - Os algoritmos são treinados a partir de exemplos rotulados
    - Já vimos em nossa aula do neurônio artificial para as portas lógicas
  - Uma pessoa insere dados mostrando o que é “certo” e o que é “errado”
  - O sistema aprende comparando a resposta dada pela que deveria ter saído, fazendo ajustes em suas variáveis caso tenha dado a resposta errada

Categorias de Aprendizado de Máquina

- **Aprendizado Não-Supervisionado**
  - O sistema age totalmente por si só
  - Ele não recebe exemplos rotulados para fazer comparações
  - buscam descobrir padrões ocultos que agrupam as informações de acordo com semelhanças ou diferenças
  - O “certo” e o “errado” não são informados ao sistema, cabendo a ele identificar padrões e características em comum dos dados inseridos

Categorias de Aprendizado de Máquina

- **Aprendizado Não-Supervisionado**

- Na imagem não há informações sobre as características que cada animal possui, não sendo possível categorizá-los.
- Porém, um algoritmo deve ser responsável por descobrir semelhanças, padrões e/ou diferenças que permitam diferenciar cães e gatos.
- Nesse exemplo pode-se utilizar uma técnica de agrupamento (*clustering*), mas existem outras técnicas.

Categorias de Aprendizado de Máquina

- **Aprendizado Não-Supervisionado**
  - Agrupamento (*clustering*)
    - consiste em agrupar dados não rotulados com base em suas semelhanças ou diferenças.
    - Algoritmos de agrupamento podem ser subdivididos em:
      - agrupamentos exclusivos
      - Sobrepostos
      - hierárquicos e
      - probabilísticos.

Categorias de Aprendizado de Máquina

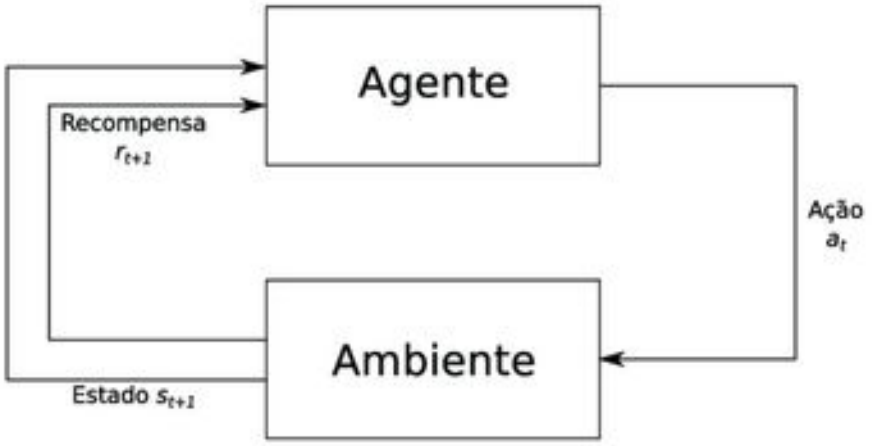
- **Aprendizado Semi-Supervisionado**
  - O sistema consegue lidar com dados rotulados ou não-rotulados
  - Técnica usada quando o custo para rotular os dados é muito elevado.

Categorias de Aprendizado de Máquina

- **Aprendizado Por Reforço**
  - Utiliza-se o sistema de “tentativa e erro”
  - O sistema aprende quais ações são as “melhores” a serem tomadas
    - Se houve benefício, reforça os dados que levaram à isso
    - Se a situação piorou, faz ajustes para evitar esse caminho de solução
  - Em suma, o ensinamento da máquina é feito com base na experiência. Ou seja, ela aprende com os próprios erros.

Categorias de Aprendizado de Máquina

- **Aprendizado Por Reforço**



- **Agente:** aquele que toma as decisões com base nas recompensas e punições e pode realizar uma ação que varia conforme o contexto.
- **Ambiente:** mundo físico ou virtual onde o agente está situado
- **Recompensa:** *feedback* do ambiente baseado na ação tomada
- **Estado:** situação atual do agente.

Aprendizado por Reforço

Categorias de Aprendizado de Máquina

- **Aprendizado Por Reforço**

- A imagem demonstra um exemplo de como o aprendizado por reforço pode ser utilizado.
- Nosso agente (azul) está no estado inicial do ambiente (labirinto)
- O objetivo é alcançar os doces evitando os inimigos (tartarugas).

Categorias de Aprendizado de Máquina

- **Aprendizado Por Reforço**

- Definido o objetivo, o agente deve buscar pelo melhor caminho possível para pegar todos os doces.
- Em cada ação do agente, ele poderá caminhar em uma determinada direção
- Caso ele escolha corretamente, ele irá inserir pesos diferentes, para diferentes respostas.
- Assim, espera-se que ao final o agente consiga realizar seu objetivo de forma que obtenha a maior recompensa cumulativa.

Agrupamentos

Agrupamentos

- Nas técnicas de agrupamento, o sistema recebe um conjunto de dados (objetos) e tenta agrupá-los de forma que os elementos que compõem cada grupo sejam mais parecidos entre si do que parecidos com os elementos dos outros grupos.
- Ou seja, coloca os iguais (ou quase iguais) juntos num mesmo grupo e os desiguais em grupos distintos.

Agrupamentos

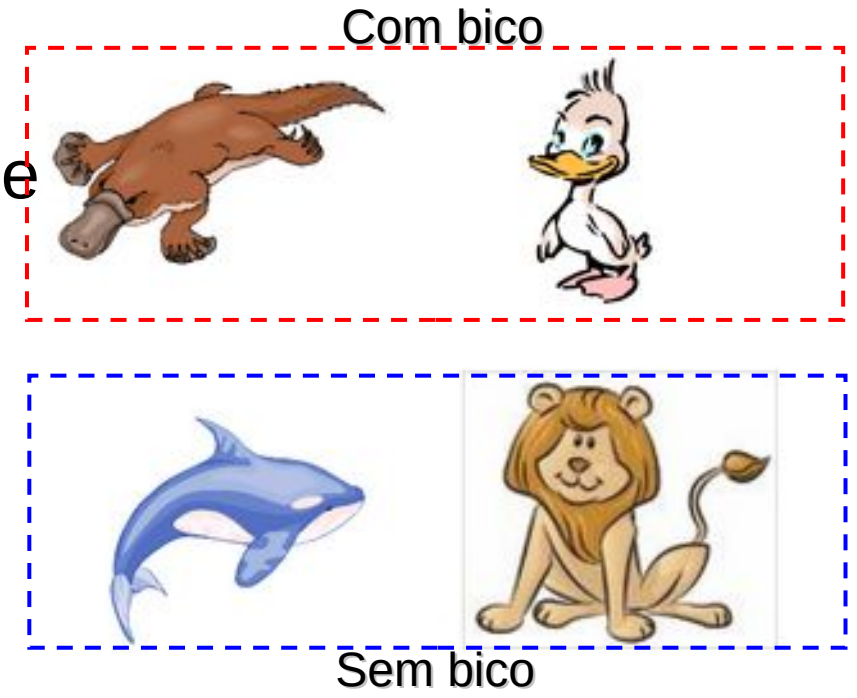
- Tem por objetivo estudar as relações de similaridades entre os dados, determinando quais dados formam quais grupos.
- Os grupos são formados de maneira a maximizar a similaridade entre os elementos de um grupo (similaridade intragrupo) e minimizar a similaridade entre elementos de grupos diferentes (similaridade intergrupos).

Formação de Agrupamentos

- Dado um conjunto de objetos, colocar os objetos em grupos baseados na similaridade entre eles
- É um problema não definido claramente
- Como agrupar os animais ao lado?

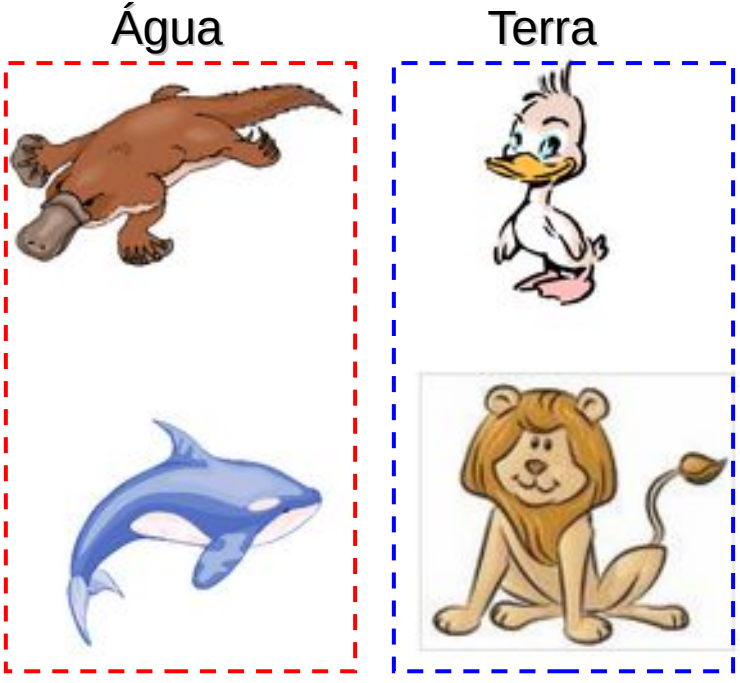
Formação de Agrupamentos

- Dado um conjunto de objetos, colocar os objetos em grupos baseados na similaridade entre eles
- É um problema não definido claramente
- Como agrupar os animais ao lado?



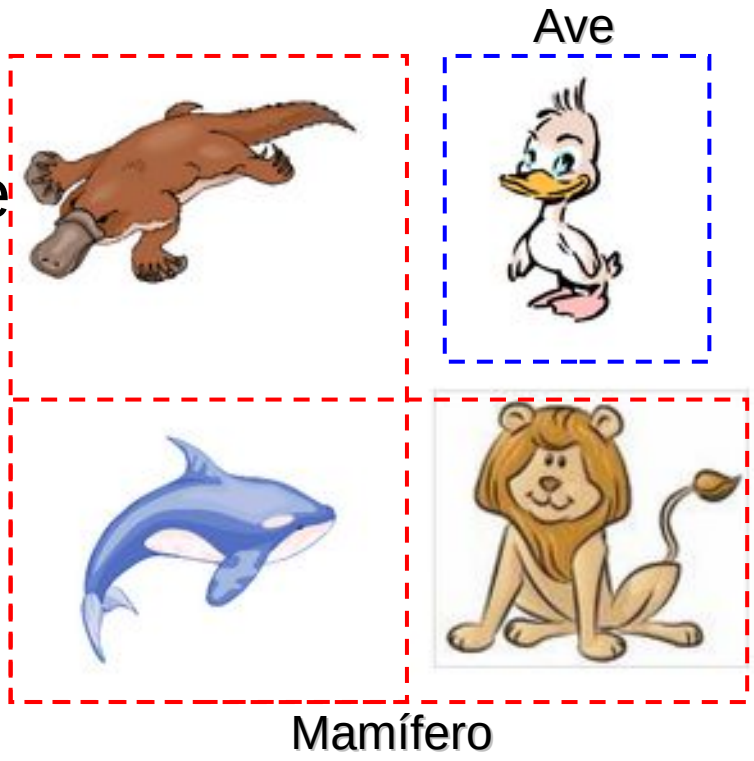
Formação de Agrupamentos

- Dado um conjunto de objetos, colocar os objetos em grupos baseados na similaridade entre eles
- É um problema não definido claramente
- Como agrupar os animais ao lado?



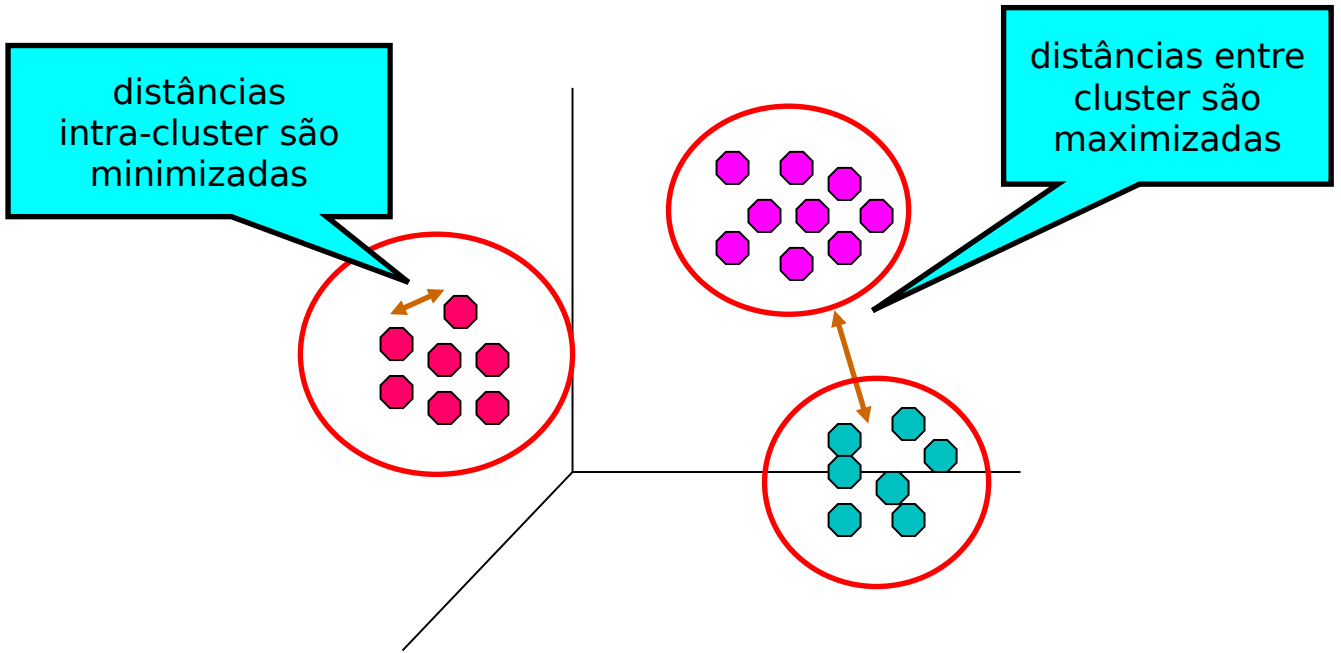
Formação de Agrupamentos

- Dado um conjunto de objetos, colocar os objetos em grupos baseados na similaridade entre eles
- É um problema não definido claramente
- Como agrupar os animais ao lado?

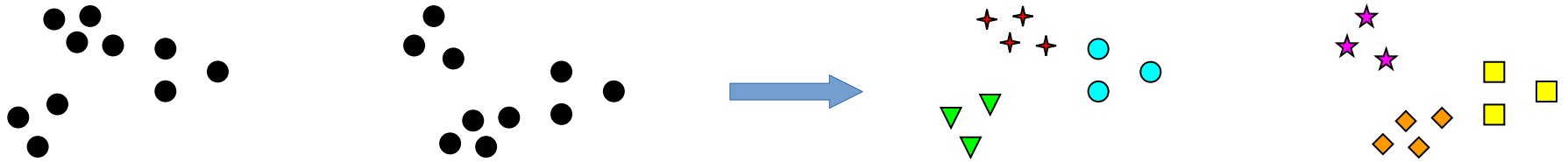


Formação de Agrupamentos

- Dado um conjunto de objetos, colocar os objetos em grupos baseados na similaridade entre eles



A noção de clusters pode ser ambígua



Quantos clusters?

Seis Clusters

Dois Clusters

Quatro Clusters

Dificuldades

- Encontrar o melhor agrupamento para um conjunto de objetos não é uma tarefa simples
  - a não ser que o número de objetos (n) e número de *clusters* (k) sejam extremamente pequenos
  - o número de partições distintas em que podemos dividir n objetos em k *clusters* é aproximadamente
    - Kn / k!
  - Exemplo:
    - K = 2
    - N = 5
    - Logo, são 16 formas de dividir 5 elementos em 2 grupos.

Dificuldades

- Para agrupar 25 objetos em 5 grupos, existem 2.436.684.974.110.751 maneiras possíveis.
- E se o número de *clusters* é desconhecido?
  - Em geral não se sabe qual é o melhor número
  - Então precisamos somar todas as partições possíveis para cada número de clusters entre 2 e 5
    - Isso desconsiderando um só cluster

Dificuldades

- Porque a efetividade dos algoritmos de *Clustering* é um problema:

1) Quase todos os algoritmos de Clustering requerem valores para os parâmetros de entrada que são difíceis de determinar, especialmente para conjuntos de dados do mundo real contendo objetos com muitos atributos.

2) Os algoritmos são muito sensíveis a estes valores de parâmetros, frequentemente produzindo partições muito diferentes do conjunto de dados mesmo para ajustes de parâmetros significativamente pouco diferentes.

3) Conjuntos de dados reais de alta dimensão (muitos atributos) têm uma distribuição muito ampla o que dificulta a análise.

Medidas de Similaridade

- Fornecem valores numéricos que expressam a “distância” entre dois objetos.
- Quanto menor a “distância”, mais semelhantes serão os objetos, e tenderão a ficar no mesmo *cluster*.
- Quanto maior a “distância”, menos similares serão os objetos e deverão estar em grupos distintos.

Medidas de Similaridade

- Uma função de distância deve ser tal que:
  - não possua valores negativos (valores >= 0);
  - seja simétrica
    - a distância do objeto i ao j deve ser igual à distância do objeto j ao i;
  - forneça o valor 0 quando calculada a distância do objeto a si mesmo ou quando dois objetos são idênticos;
  - respeite a desigualdade triangular
    - dados 3 objetos, a distância entre dois deles deve ser menor ou igual a soma das distâncias entre esses dois objetos e o terceiro.

Medidas de Similaridade

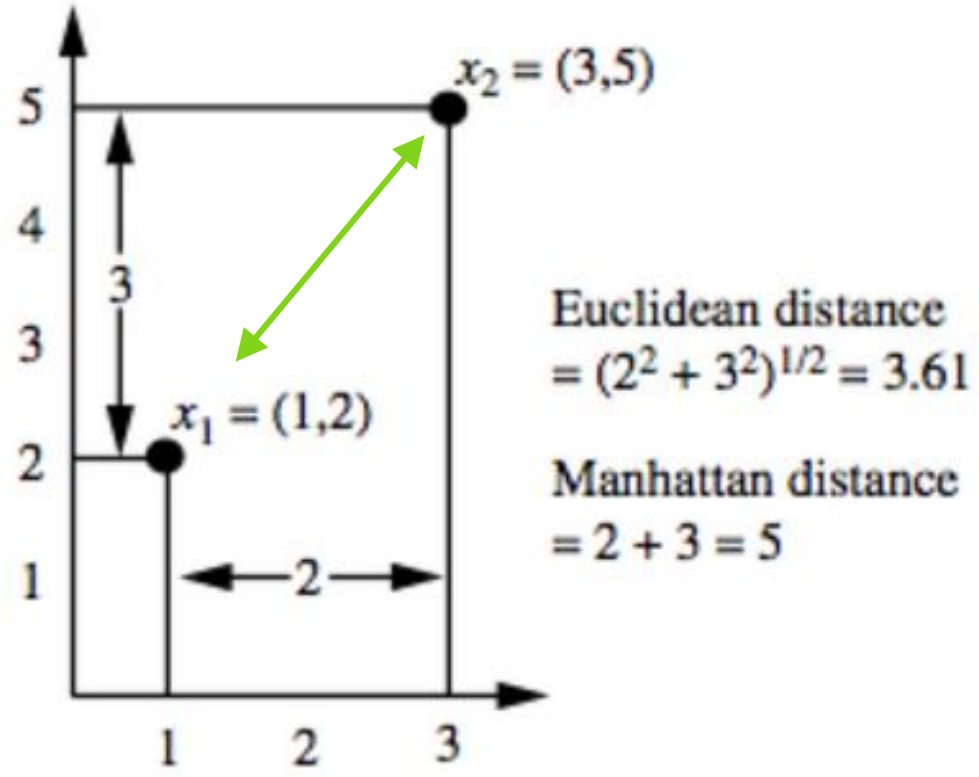
- Distância Euclidiana

$$d(x,y) = \sqrt{(x_1 - y_1)^2 + (x_2 - y_2)^2 + \ldots + (x_p - y_p)^2}$$

- City block (Manhattan, taxicab, L1 norm, Hamming)
  - Um exemplo comum é a distância de Hamming, que é o número de bits que é diferente entre dois vetores binários

$$d(x,y) = |x_1 - y_1| + |x_2 - y_2| + \ldots + |x_p - y_p|$$

Medidas de Similaridade



Medidas de Similaridade

- Não há uma medida de similaridade que sirva para todos os tipos de variáveis que podem existir numa base de dados.
- Variáveis numéricas:
  - A medida que é normalmente usada para computar as dissimilaridades de objetos descritos por variáveis numéricas é a **Distancia Euclidiana**
  - A **normalização** faz com que todas as variáveis tenham um peso igual e deve ser efetuada para todos os atributos.

Medidas de Similaridade

- Exemplo para Strings
  - Cores = {Branco, Amarelo, Vermelho, Marrom, Preto}
    - x1 = Branco
    - y1 = Amarelo
- Opção1: medida binária de similaridade

$$d(x1,y1) = \begin{cases} 0, & \text{se} \quad xi = yi \\ 1, & \text{se} \quad xi \neq yi \end{cases}$$

Medidas de Similaridade

- Opção 2: transformar o string em um valor numérico
  - mais usado quando há ordem entre os valores nominais

Branco	Amarelo	Vermelho	Marrom	Preto
0	0,25	0,5	0,75	1

usar a distância Euclidiana

Principais Abordagens de Agrupamentos

- Particionamento (K-médias e variantes)
  - Divide os pontos (dados) em conjuntos disjuntos (clusters) tal que cada ponto pertence a um único cluster
- Hierárquica
  - Um conjunto de clusters aninhados organizados como uma árvore
- Baseadas em densidade
  - Encontra clusters baseado na densidade de regiões
- Baseadas em grade (Grid-based)
  - Encontra clusters baseado no número de pontos em cada célula

Mapas Auto-organizáveis
Self-Organizing Map – SOM
Kohonen

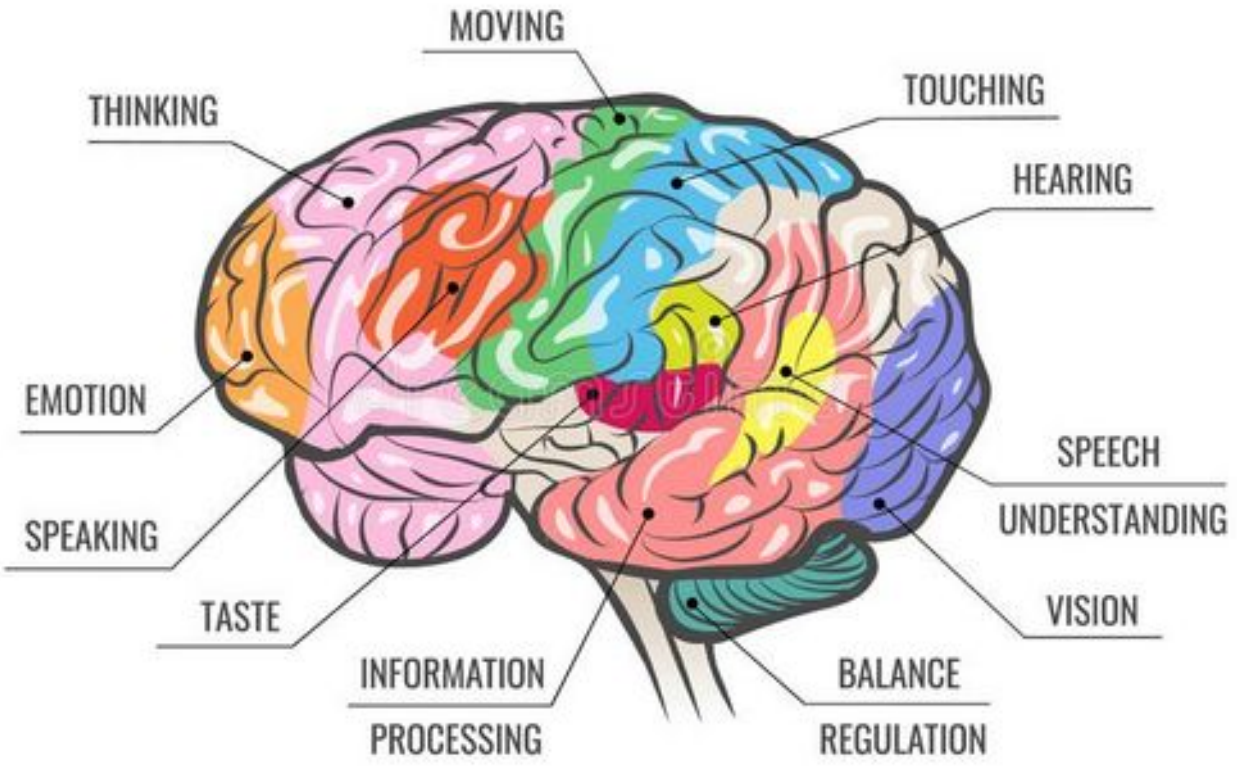
Mapas Auto-organizáveis - Kohonen

- Os Mapas Auto-organizáveis (*Self-Organizing Map – SOM*) foram desenvolvidos por **Kohonen** a partir de 1982
- Aprendizado não-supervisionado
  - diferente de todas as redes neurais artificiais desenvolvidas até a época
  - Existia um descrédito das redes neurais da época por não resolverem problemas não-linearmente separáveis
- Forte inspiração neurofisiológica
- Baseado em Aprendizagem Competitiva

SOM – Inspiração Neurofisiológica

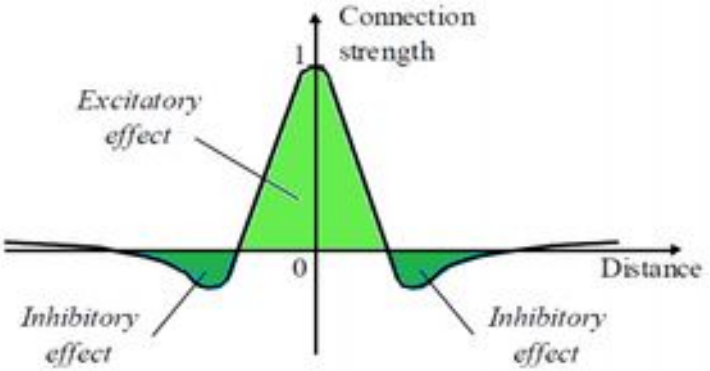
- Estudos neurobiológicos indicam que diferentes entradas sensoriais (motoras, visuais, auditivas, etc.) são mapeadas em áreas correspondentes no córtex cerebral.
- Observação de imagens
  - Ressonância magnética (MRI)
  - Tomografia Computadorizada (CT)
- Diferentes estímulos geram
  - Regiões de excitação
  - Organização topográfica

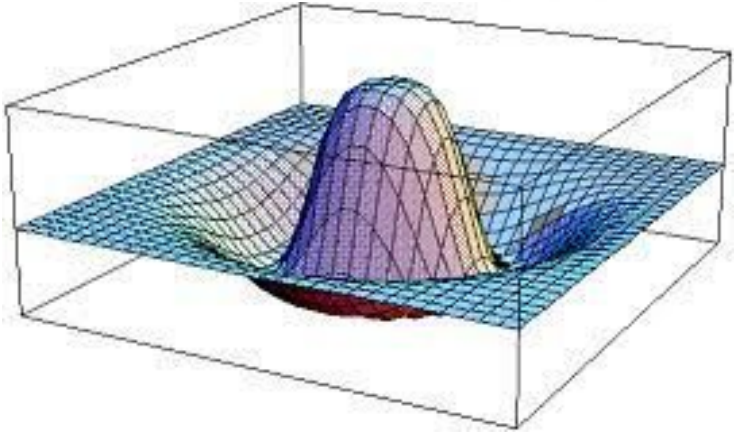
SOM – Inspiração Neurofisiológica



SOM – Inspiração Neurofisiológica

- Quando um neurônio é excitado, ao redor uma área entre 50 e 100 μm também sofre excitação
- Ao redor, uma área sofre inibição para impedir a propagação do sinal a áreas não relacionadas



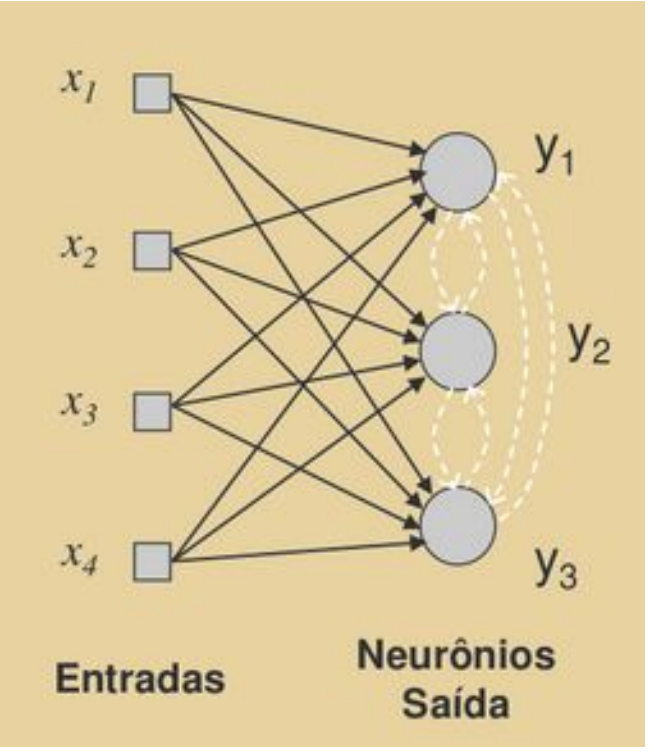


Aprendizagem Competitiva

- **Neurônios** de saída da RNA **competem** entre si para se tornar ativos
- **Apenas um neurônio de saída está ativo** em um determinado instante
- Três elementos básicos:
  1. Neurônios com mesma estrutura, mas diferente pelos valores de pesos, têm respostas diferentes a uma entrada
  2. Um limite imposto sobre a força de cada neurônio
  3. Mecanismo de competição entre neurônios, de forma que um neurônio é vencedor em um dado instante.
- Em cada momento o neurônio vencedor:
  - aprende a se especializar em agrupamentos de padrões similares
  - tornam-se detectores de características para classes diferentes de padrões de entrada
- O número de unidades de entrada define a **dimensionalidade** dos dados

Aprendizagem Competitiva - Exemplo

- Exemplo de estrutura simples:
  - 1 camada de saída
  - Totalmente conectada à entrada
  - Pode incluir realimentação entre neurônios para inibição lateral
- Conexões excitadoras das entradas para os neurônios (setas fechadas com linhas contínuas)
- Conexões laterais inibitórias entre os neurônios (setas abertas em linhas tracejadas)



Aprendizagem Competitiva – Neurônio Vencedor

- O neurônio vencedor é o que possui o maior campo local induzido *υk* para um dado padrão de entrada *x*
- O campo local induzido representa a ação combinada de todas as entradas do neurônio *k*
- A escolha do vencedor maximiza o produto interno entre os pesos do neurônio e o sinal de entrada
- O sinal de saída *yk* do neurônio vencedor é colocado em 1 (um) e dos outros neurônios em 0 (zero).

Aprendizagem Competitiva – Pesos

- Considerando ωkj como o peso sináptico conectando o nó de entrada j ao neurônio k, cada neurônio tem uma quantidade de peso sináptico

$$\sum_j \omega_{kj} = 1 \text{ para todo } k$$

- Cada neurônio tem seus pesos inicializados aleatoriamente
- O aprendizado dá-se então deslocando os pesos do neurônio vencedor, na direção da entrada

Aprendizagem Competitiva – Regra de Aprendizagem

- A regra de aprendizagem competitiva aplica uma variação Δωkj ao peso ωkj, definida por:

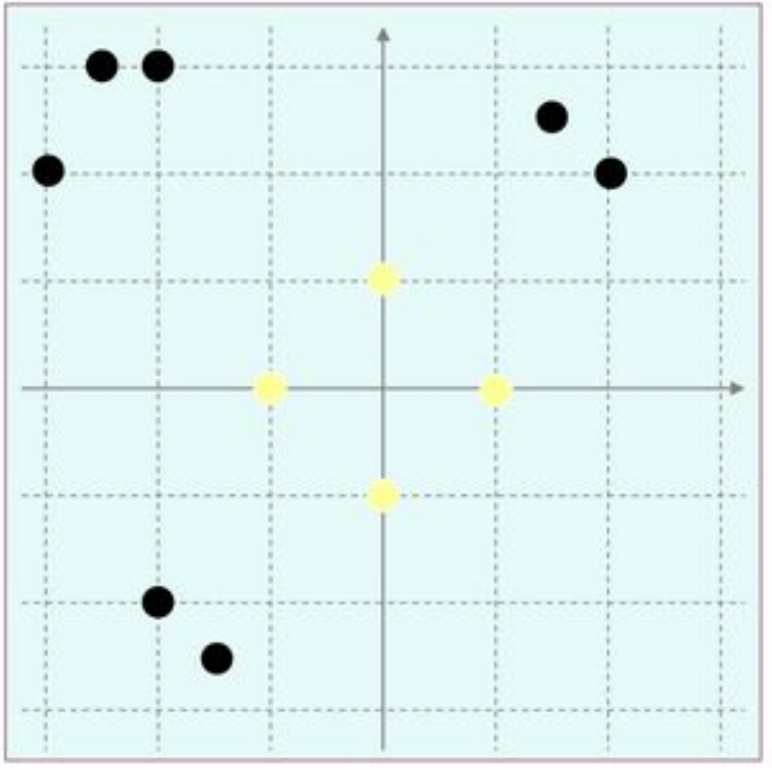
$$\Delta\omega_{kj} = \begin{cases} \alpha(x_j - \omega_{kj}) & \text{se o neurônio } k \text{ for o vencedor} \\ 0 & \text{caso contrario} \end{cases}$$

  onde α é a taxa de aprendizagem.

- Esta regra move o vetor de peso do neurônio vencedor em direção ao padrão de entrada

Aprendizagem Competitiva
Exemplo / Interpretação

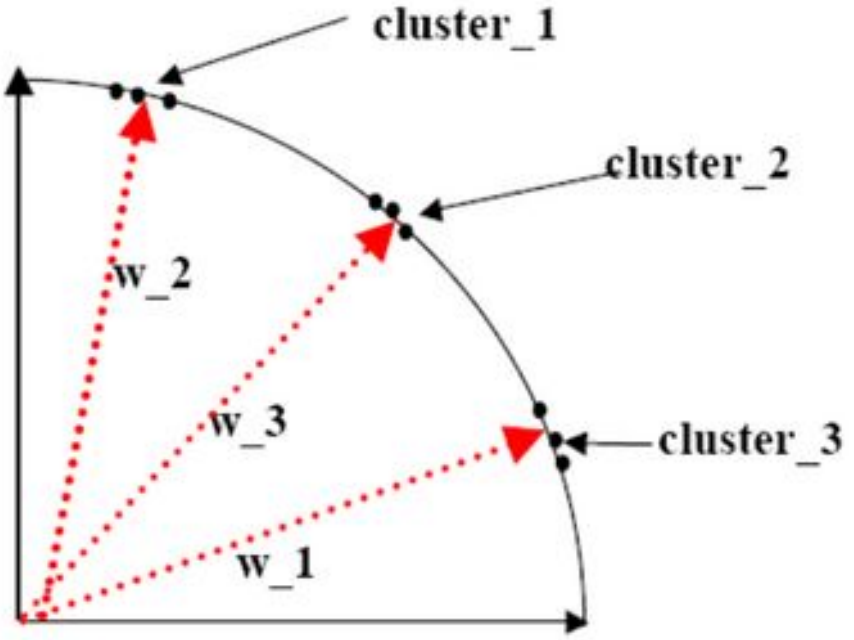
- Entradas com 2 características (espaço 2D)
  - 7 padrões de entrada (pontos pretos)
- 4 neurônios de saída (pontos amarelos)
  - α = 0.5
- 7 iterações



Estado Inicial da Rede

Aprendizagem Competitiva

- Na aprendizagem competitiva, não há ordenação do mapa de neurônios de saída (clusters)
- No exemplo, a ordem topológica seria:
  - w_1, w_2, w_3
- No entanto o mapa de saída está na ordem:
  - w_2, w_3, w_1

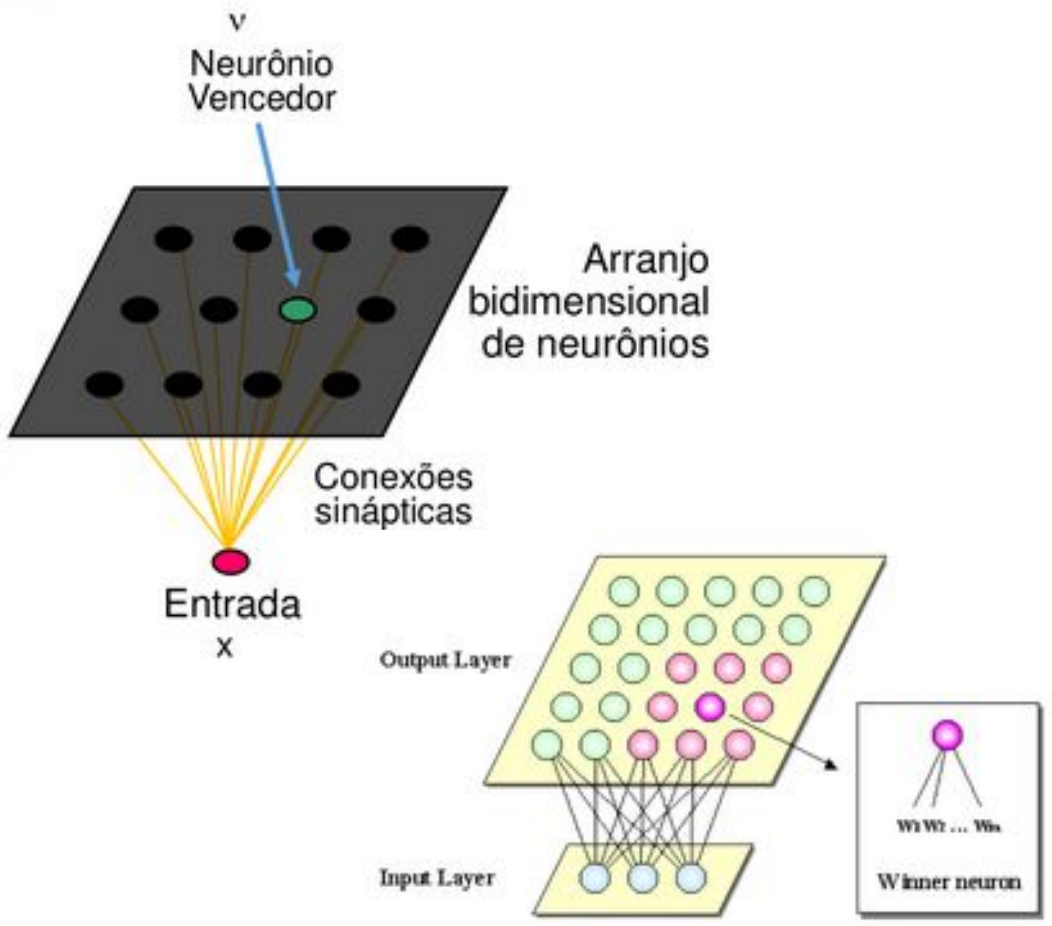


Mapas Auto-organizáveis

- Grades neurais baseadas na aprendizagem competitiva
- Neurônios de saída da grade competem entre si para serem ativados ou disparados
- Os neurônios estão dispostos em nós de uma grade, em geral uni- ou bidimensional
  - Mapas de dimensionalidade mais alta são possíveis porem pouco comuns
- Processo de organização
- Aprendizado não-supervisionado
- Neurônios dispostos em grade
  - Vetores de peso localizam os neurônios no espaço de entrada
- Torna a rede ideal para detecção de agrupamentos (clusters)

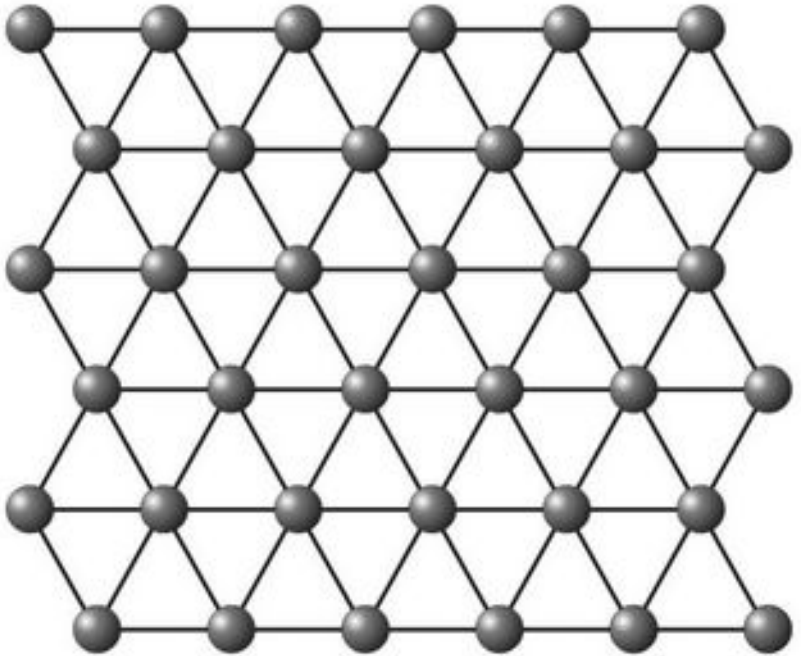
Mapas Auto-organizáveis - Kohonen

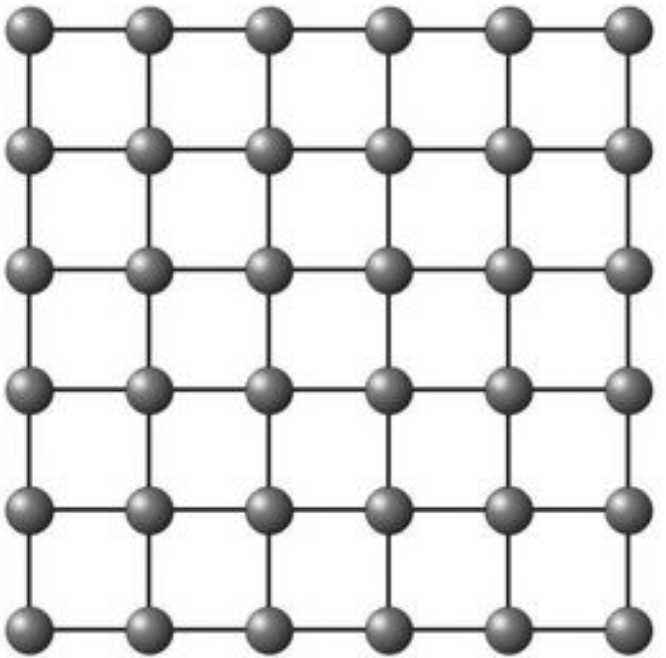
- Produz um mapeamento topológico
- Transforma um padrão de dimensão arbitrária em um mapa discreto uni- ou bidimensional
- Preserva a relação de vizinhança entre os neurônios



Mapa Topológico

- No caso bidimensional, dois tipos de grade são possíveis: hexagonal ou retangular.
- Na hexagonal, cada neurônio possui 6 vizinhos diretos
- Na retangular, cada neurônio possui 4 vizinhos diretos





Arquitetura SOM

- Considere um espaço vetorial de entrada com dimensão d
- Cada amostra é um sinal/padrão representado por um vetor $x = [x_1, x_2, ..., x_d]$
- A arquitetura do SOM consiste de 2 camadas de neurônios
  - Camada de entrada composta por d neurônios
  - Camada de saída (ou de Kohonen) formada por N neurônios denotados por:

    $A = \{ c_1, c_2, ..., c_N \}$

Arquitetura SOM

- O vetor peso sináptico de cada neurônio da grade tem a mesma dimensão que o espaço de entrada
- O vetor peso do neurônio j é representado por:

  $\mathbf{w} = [\ \omega_{j1}, \omega_{j2}, \ldots, \omega_{jd}\ ]\ j = 1, \ldots, N$

  - este vetor indica as coordenadas de sua localização no espaço de entrada *d* dimensional

SOM – Kohonen - Resumo

- Um tipo particular de mapa auto-organizável. Essa rede tem uma estrutura feed-forward com apenas duas camadas.
- O vetor de pesos para um cluster serve como exemplar dos padrões de entrada associado com esse cluster.
- Padrões de entrada são comparados com todos os neurônios, e o mais próximo é considerado o neurônio vencedor.
- Os pesos do neurônio vencedor são atualizados para aproximar-se mais do padrão de dados que representa.

SOM – Algoritmo

1. **Inicialização**: geralmente aleatória, pode ainda ser estimada por análise da representação dos dados
2. **Competição**: para cada padrão de entrada, calcula-se a resposta dos neurônios de saída (grade). O neurônio com a maior resposta é o vencedor da competição.
3. **Cooperação**: o neurônio vencedor define uma vizinhança topológica (em função da grade) de neurônios excitados
4. **Adaptação Sináptica**: aprendizado em relação ao padrão de entrada. Os pesos do neurônio vencedor, e de sua vizinhança, ficam mais próximos do padrão de entrada.

Algoritmo SOM: Inicialização

- Inicialização aleatória: mais simples e mais utilizada
  - nenhum estado de organização é considerado
  - durante as primeiras centenas de passos, os vetores peso passam por processo de auto-organização

Algoritmo SOM: Competição

- O critério do melhor casamento (*best matching*) é baseado na maximização do produto interno como na aprendizagem competitiva
- Isto é matematicamente equivalente a minimizar a distância euclidiana entre w e x
- O neurônio vencedor, v é escolhido por:

$$v = \arg\min_j \left\| \mathbf{x} - \mathbf{w}_j \right\|, \quad j = 1,2,\ldots, N$$

Algoritmo SOM: Cooperação

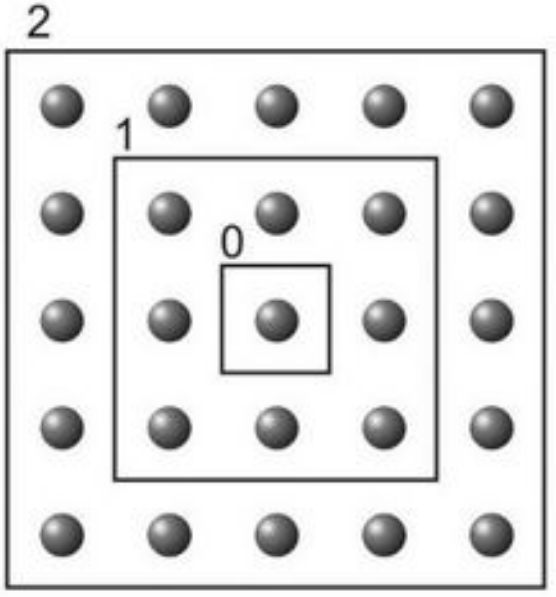
- Compreende a definição de uma função de vizinhança h_{vj} centrada no neurônio vencedor
- Define uma região de neurônios cooperativos, que **terão seus pesos ajustados juntamente com o neurônio vencedor**
- Há diversas formas de implementar a função de vizinhança
  - Mais simples é definir um conjunto $N_v(t)$ de níveis de vizinhança ao redor do neurônio vencedor

Algoritmo SOM: Cooperação

- Então, a função de vizinhança torna-se:

$$h_{vj}(t) = \begin{cases} 1, j \in N_v(t) \\ 0, j \notin N_v(t) \end{cases}$$

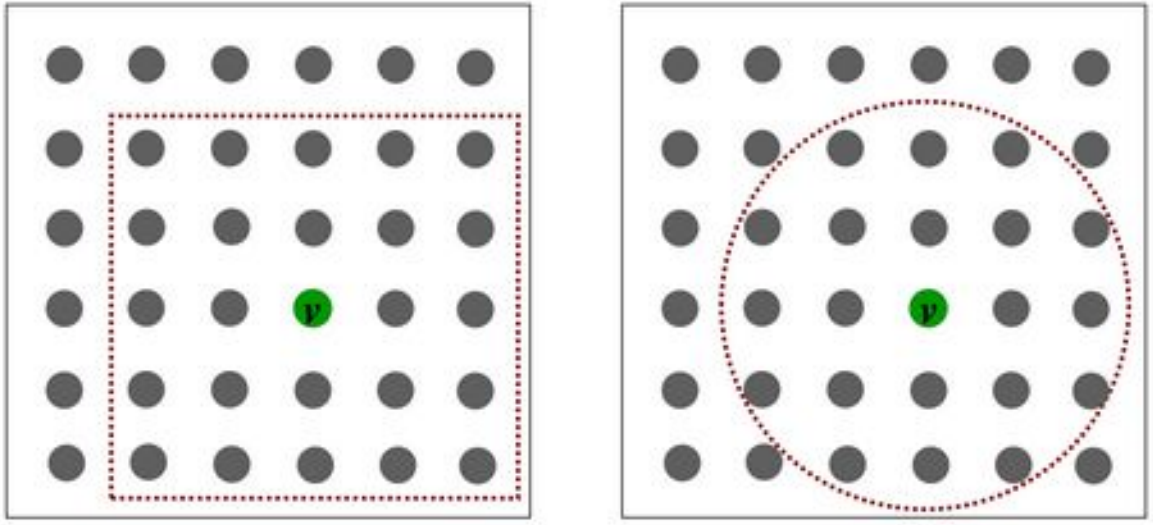
- $N_v(t)$ é função monotônica decrescente no tempo
- A função de vizinhança é mapeada no espaço de saída



Exemplo de níveis de vizinhança para uma grade retangular de neurônios

Algoritmo SOM: Cooperação

- Exemplos de função de vizinhança
- Tipo bolha (todos os neurônios da vizinhança serão atualizados igualmente)
  - Base Quadrada / Base Circular



Algoritmo SOM: Cooperação

- Inicia-se o algoritmo com alto nível de vizinhança e esta é reduzida conforme o tempo avança
- É necessário diminuir a região de vizinhança para obter a auto-organização do mapa
- Para que o algoritmo convirja é necessário que

$$h_{vj}(t) \rightarrow 0 \text{ quando } t \rightarrow \infty$$

Algoritmo SOM: Cooperação

- Uma função muito interessante a ser usada como função de vizinhança é a Gaussiana, dada por:

$$h_{vj}(t) = \exp\left(-\frac{\|\mathbf{r}_v - \mathbf{r}_j\|}{2\sigma^2(t)}\right)$$

- onde o termo $\|\mathbf{r}_v - \mathbf{r}_j\|$

- é a distância entre o neurônio v vencedor e o neurônio j que está sendo atualizado

Algoritmo SOM: Cooperação

- O parâmetro σ define a largura da função e deve ser decrescente no tempo. Pode ser usada uma função linear, mas em geral é usada a exponencial:

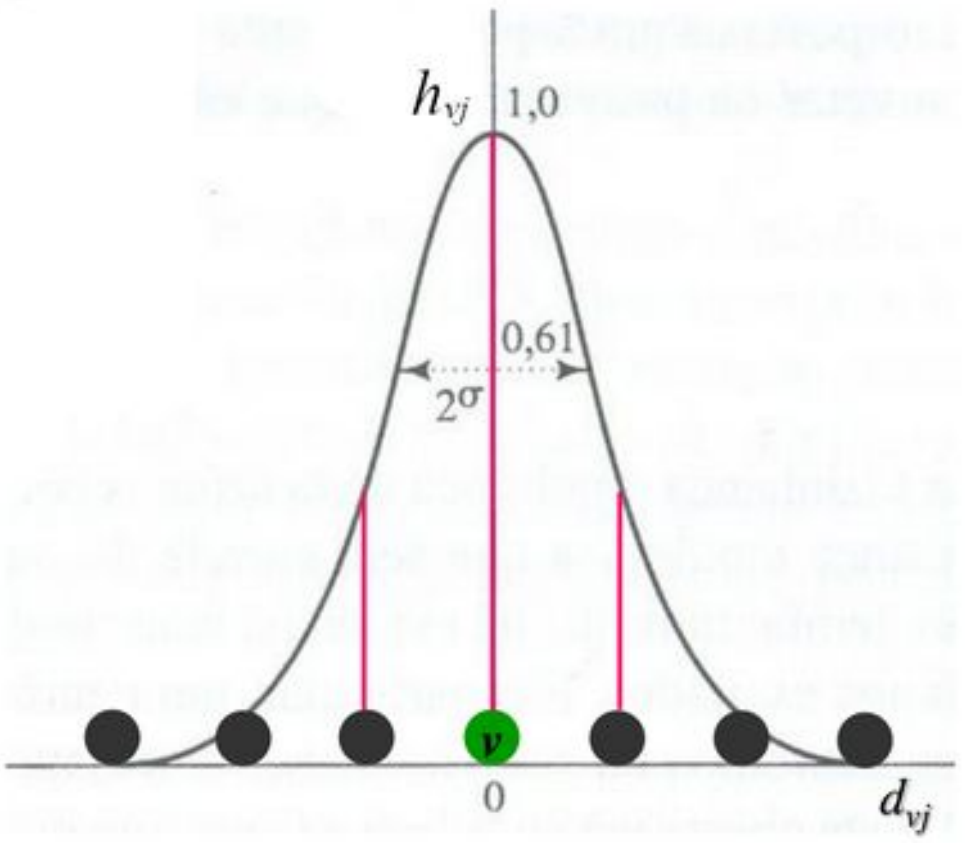
$$\sigma(t) = \sigma(0)\exp\left(-\frac{t}{\tau_1}\right)$$

  - σ(0) é um valor inicial para σ
  - τ1 é uma constante de tempo do SOM definida para que a taxa de aprendizagem nunca decaia para zero

$$\tau_1 = \frac{NIter}{\log \sigma(0)}$$

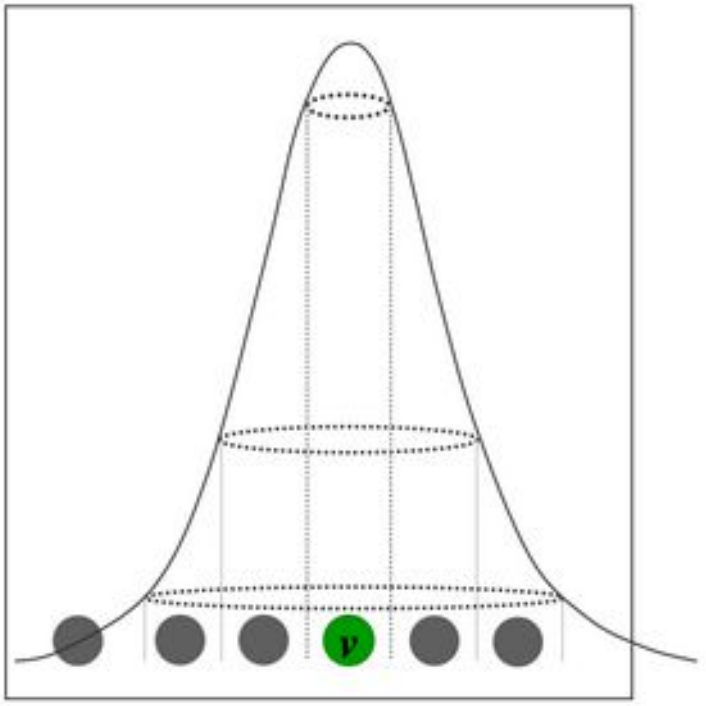
Algoritmo SOM: Cooperação

- Exemplo de função de vizinhança Gaussiana
  - Os neurônios da vizinhança são atualizados de forma ponderada
  - Quanto mais afastados, menor fica a taxa de aprendizado

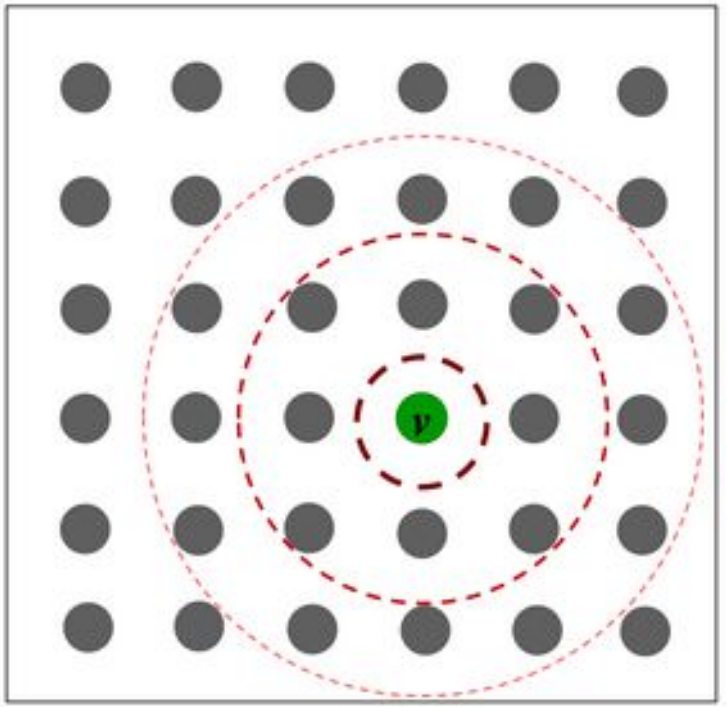


Algoritmo SOM: Cooperação

- Exemplo de função de vizinhança Gaussiana



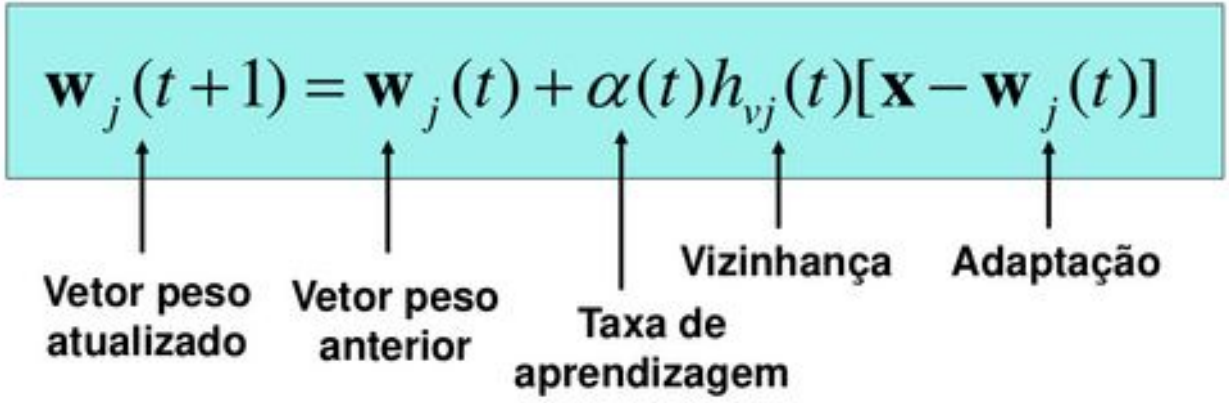
Visão Lateral



Visão Superior

Algoritmo SOM: Adaptação Sináptica

- Modificação dos pesos em relação à entrada, de forma iterativa
- A equação abaixo é aplicada a todos os neurônios da grade dentro da região de vizinhança h_{vj}



Algoritmo SOM: Adaptação Sinaptica

- O parâmetro de aprendizagem α, assim como a função de vizinhança deve decrescer com o tempo, para que as adaptações sejam cada vez mais “finas”
- Pode ser usada uma função linear decrescente, mas é recomendada uma função exponencial decrescente:

$$\alpha(t) = \alpha_0 \exp\left(-\frac{t}{\tau_2}\right),$$

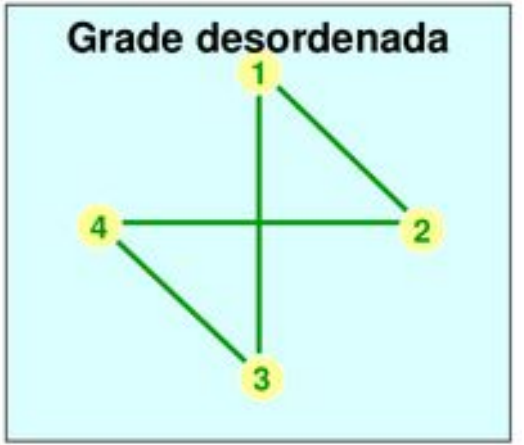
- onde τ_2 é o número total de iterações

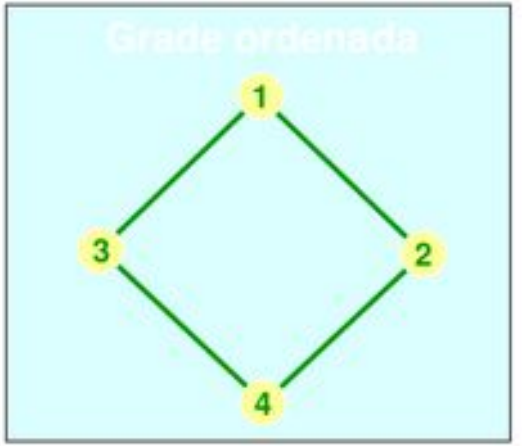
Adaptação Sinaptica: Ordenação

- Assumindo uma inicialização aleatória, é necessário duas fases de adaptação

  1. Fase de Ordenação: devido à inicialização aleatória, a grade está desordenada

  - A ordenação topológica dá-se pela movimentação da vizinhança, o que ocorre em geral nas primeiras 1000 iterações





Adaptação Sinaptica: Ordenação

- Para a Fase de Ordenação, a inicialização dos parâmetros: τ_2 , $\alpha(0)$ e $\sigma(0)$ é importante e depende da aplicação
- Haykin (2000) sugere os seguintes valores iniciais:
  - $\tau_2 = 1000$
  - $\alpha(0) = 0,1$
- O $\sigma(0)$, largura da função de vizinhança, depende do tamanho da rede. Para uma rede de 50 neurônios, por exemplo, uma possibilidade é:
  - $\sigma(0) = 25$

Adaptação Sinaptica: Convergência

- Após a fase de ordenação, inicia-se a adaptação aos padrões de entrada

2. Fase de Convergência: ocorre uma “sintonia fina”, que pode durar muito mais iterações do que a fase de ordenação

  - Neste momento, a função de vizinhança deverá englobar apenas vizinhos próximos do neurônio vencedor
  - Pode ainda ser reduzida à apenas o neurônio vencedor

Adaptação Sinaptica: Convergência

- Para a Fase de Convergência, a seguinte inicialização dos
- parâmetros: τ_2 , $\alpha(0)$ e $\sigma(0)$ é recomendada por Haykin (2000):
  - $\tau_2 = 500 . N$
  - $\alpha(0) = 0,01$
- onde N é o número de neurônios de saída
- é importante não permitir que α chegue a zero

SOM - Algoritmo

- 1. Selecione a topologia da rede (defina os parâmetros)
- 2. Selecione a região de vizinhança inicial
- 3. Inicialize os pesos aleatoriamente e defina o nº de iterações
- 4. Para cada iteração:
  - 4.1. Selecione um padrão de entrada
  - 4.2. Encontre o neurônio vencedor (pela menor distância)
  - 4.3. Atualize os pesos do neurônio vencedor e sua vizinhança
    - $w_j (t + 1) = w_j (t) + \alpha(t) h_{vj} (t) [x - w_j (t)]$
  - 4.4. Decremente a região de vizinhança e a taxa de aprendizagem (caso desejado)
  - 4.5. Incremente o número da iteração
- 5. Fim

SOM - Aplicações

- Organização da representação dos dados
  - Redução de dimensionalidade
  - Reconstrução de Superfícies
- Separação de agrupamentos de dados
  - Segmentação de Imagens
  - Data Mining
  - Classificação de Documentos
- Classificação de dados
  - Reconhecimento de Caracteres
  - Reconhecimento de Fala

K-Means

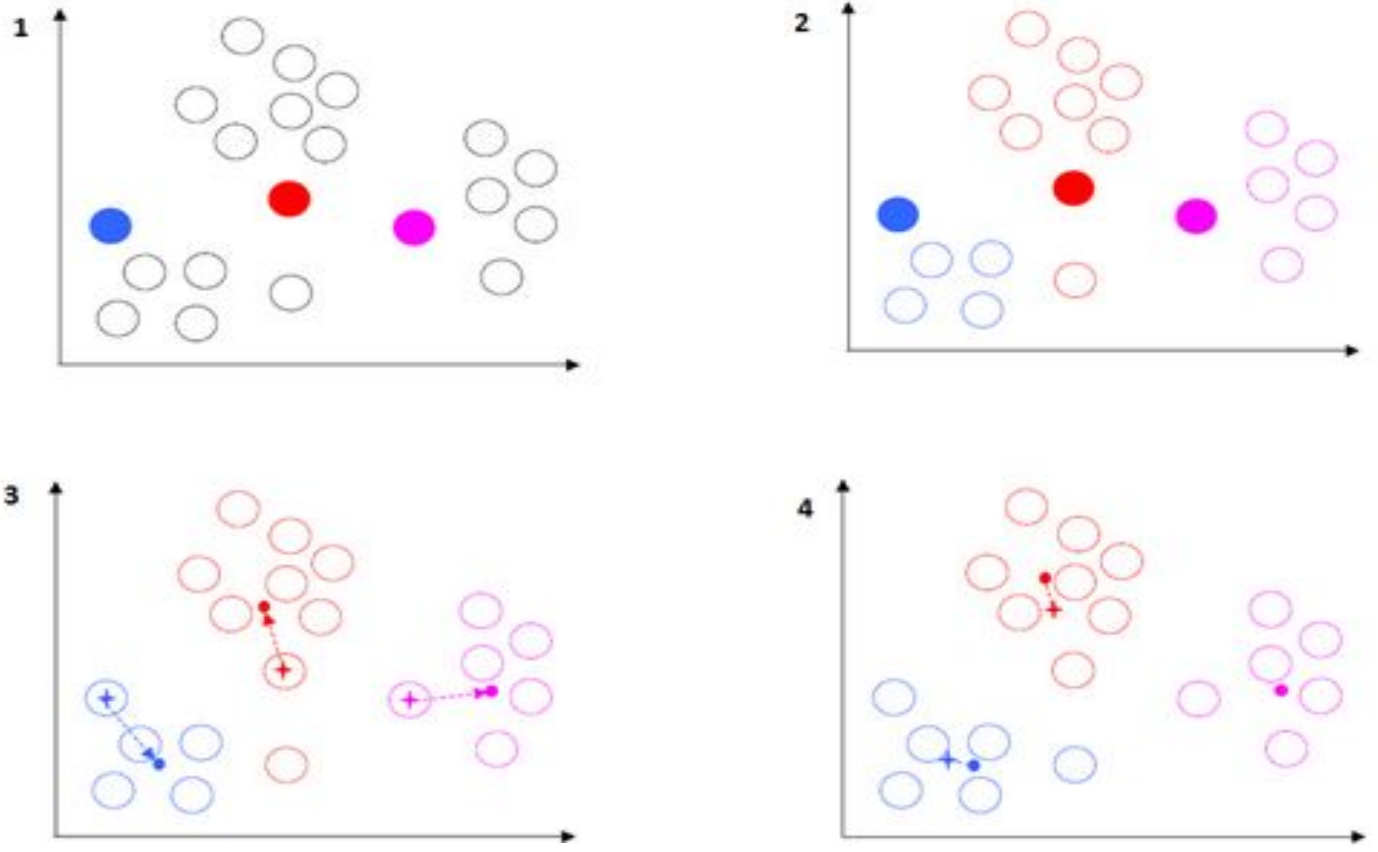
K-médias (K-means)

- Abordagem por particionamento
- Cada cluster está associado a um centroide (ponto central)
- Cada ponto é associado ao cluster cujo centroide está mais próximo
- Número de clusters, K, precisa ser especificado
- O algoritmo básico é bem simples:

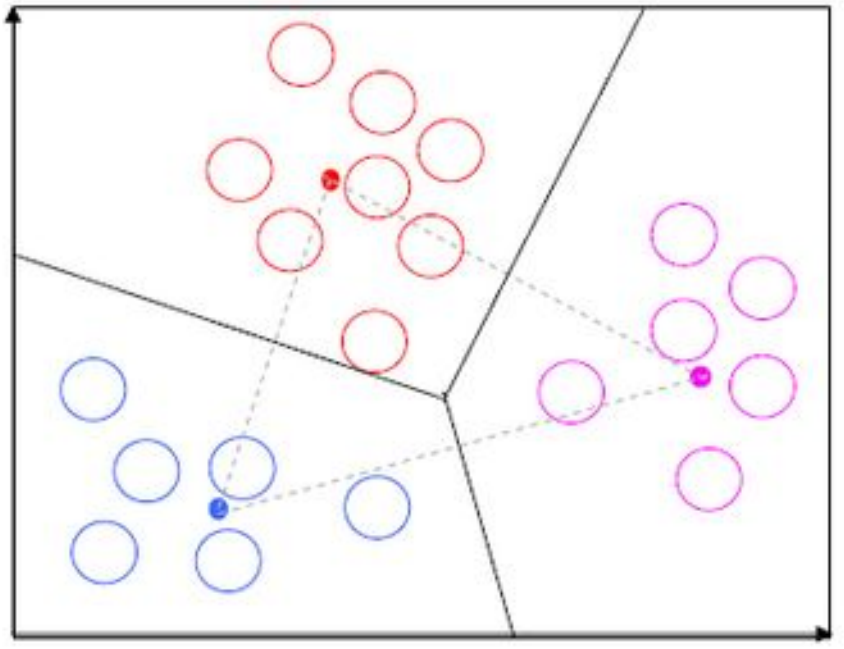
K-means

1: Select K points as the initial centroids.
2: **repeat**
3: Form K clusters by assigning all points to the closest centroid.
4: Recompute the centroid of each cluster.
5: **until** The centroids don't change

K-means: Exemplo de Funcionamento



K-means: partição



K-means: Exemplo

Usar k=2 (parâmetro informado pelo usuário)

Dataset a ser clusterizado

	Variáveis	
Item	x1	x2
A	5	3
B	-1	1
C	1	-2
D	-3	-2

K-means: Exemplo

Passo 1: Determina-se os centroides iniciais (normalmente pega-se ao acaso k pontos (registros): por exemplo os registros A e B), isto é:

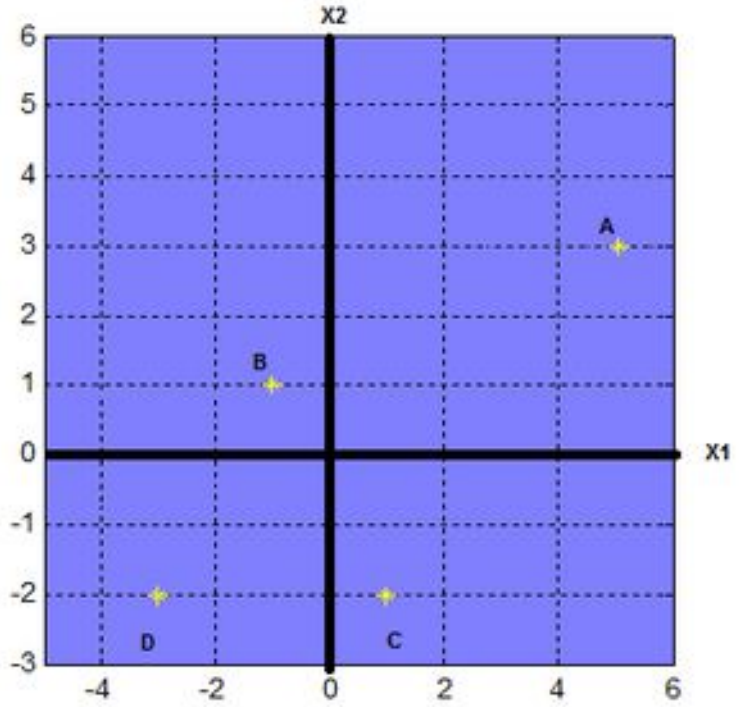
$C_1(1) = (5,3)$

$C_2(1) = (-1,1)$

	Variáveis	
Item	x1	x2
A	5	3
B	-1	1
C	1	-2
D	-3	-2

K-means: Exemplo

Passo 2: Calcula-se as distâncias de cada ponto aos centroides, para definir os cluster iniciais.



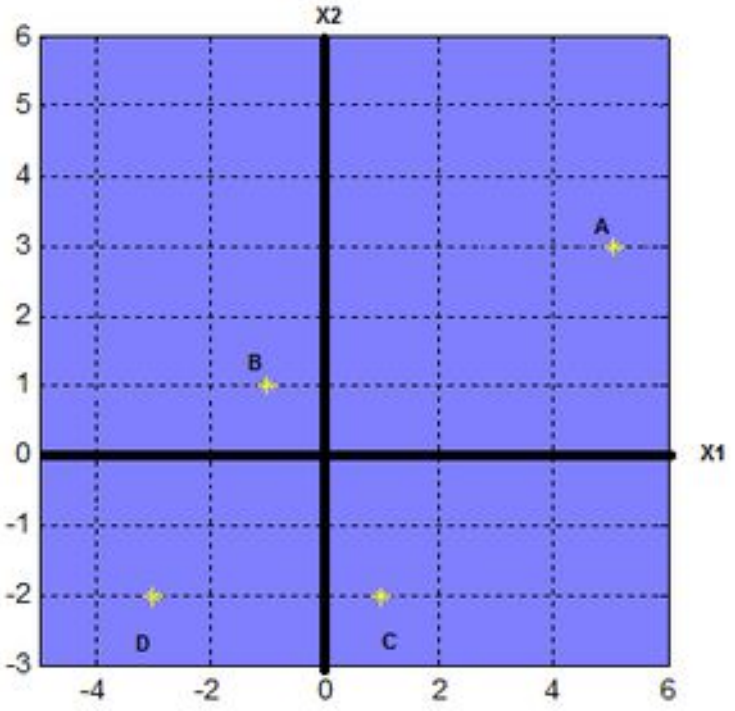
Os clusters são:

$C_1 = \{A\}$
$C_2 = \{B,C,D\}$

Pois C e D estão mais perto de B do que de A

K-means: Exemplo

Passo 3: cálculo dos novos centroides



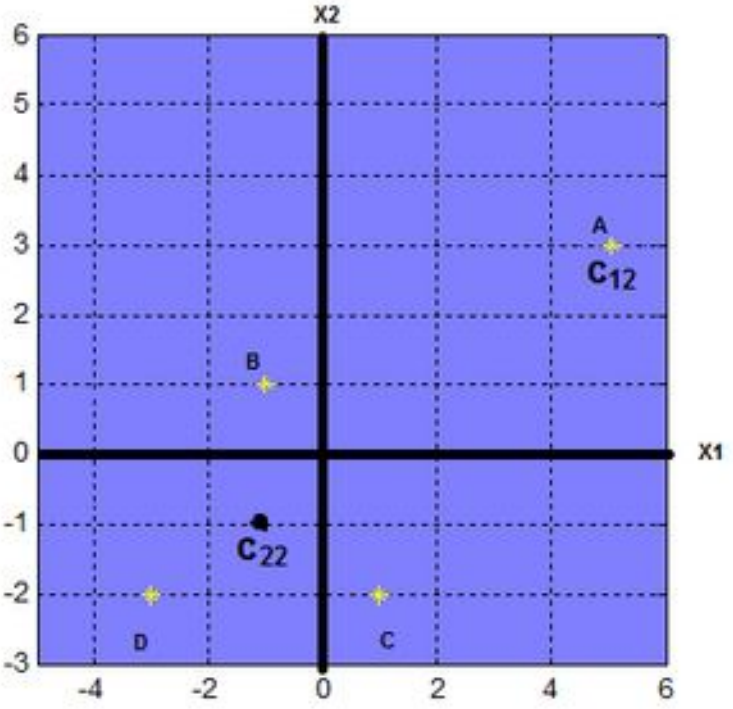
$C_1(2)= (5,3)$

$C_2(2)= \left(\frac{-1+1-3}{3}, \frac{1-2-2}{3}\right) = (-1,-1)$

	Variáveis	
Item	x1	x2
A	5	3
B	-1	1
C	1	-2
D	-3	-2

K-means: Exemplo

Passo 4: novo cálculo dos clusters



Os clusters são:

$C_1 = \{A\}$
$C_2 = \{B,C,D\}$

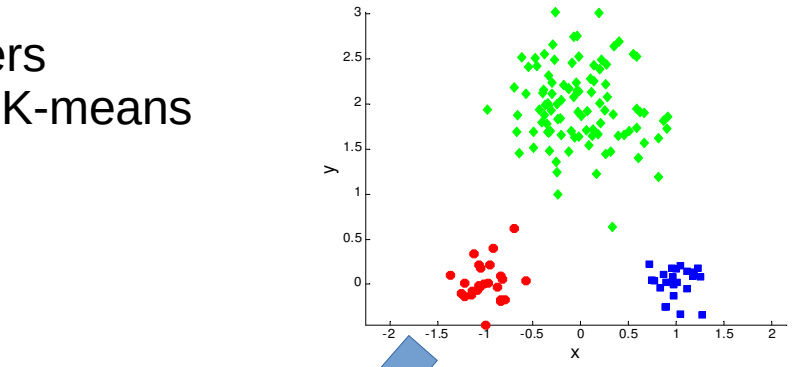
Pois:

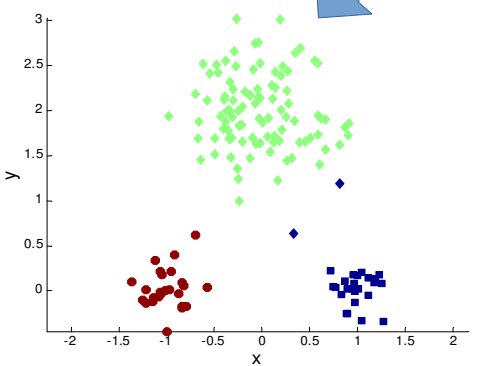
- A está mais perto de C_{12} do que de C_{22}
- B, C e D estão mais perto de C_{22} do que de C_{12}

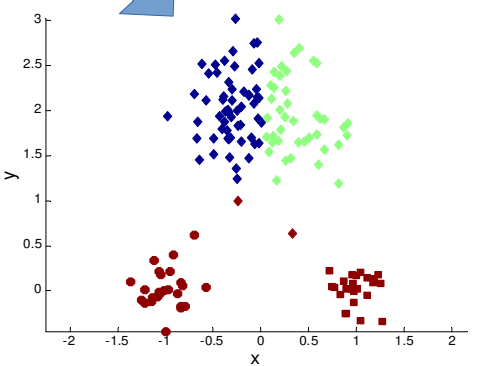
Como os elementos dos clusters não se alteraram, os centroides vão ser os mesmos e com isso o algoritmo para.

K-means

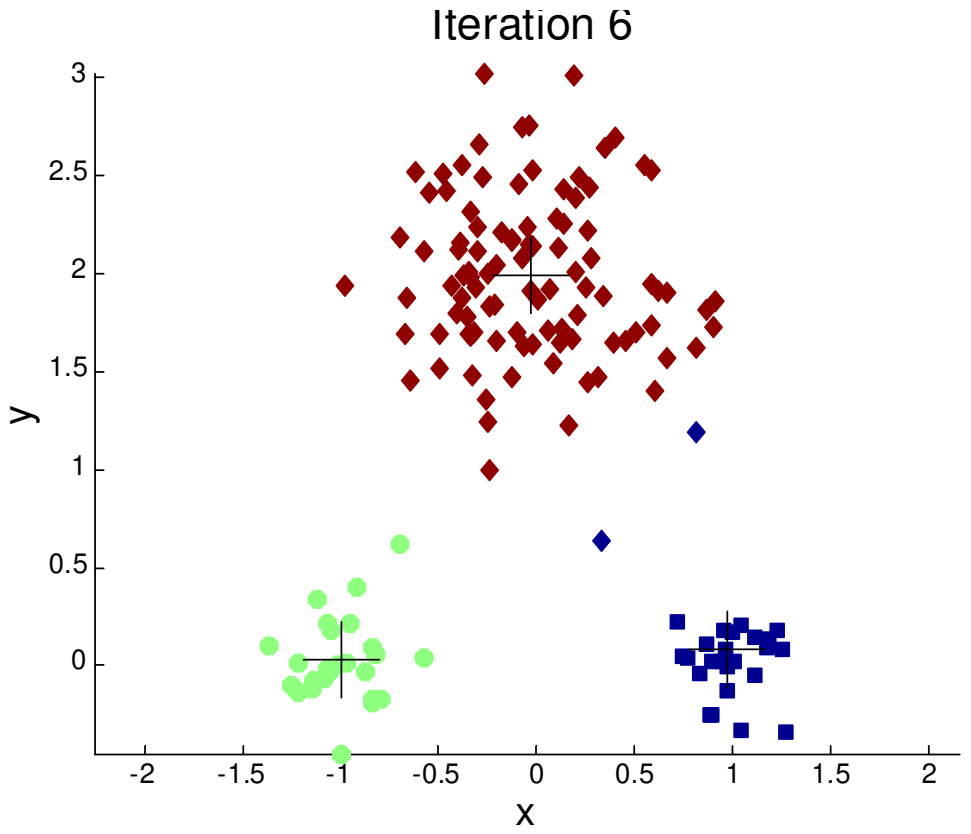
Dois conjuntos de clusters diferentes gerados pelo K-means



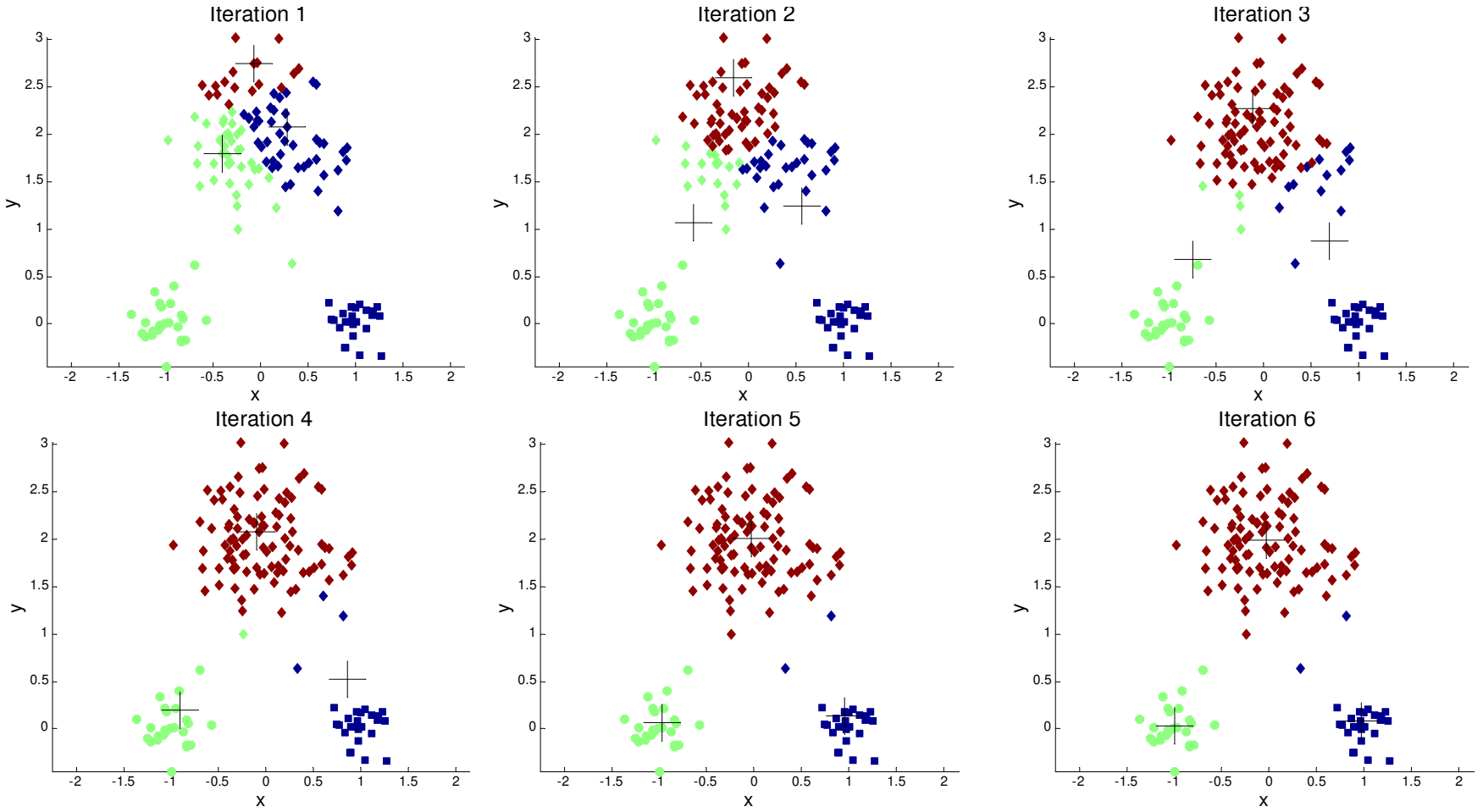




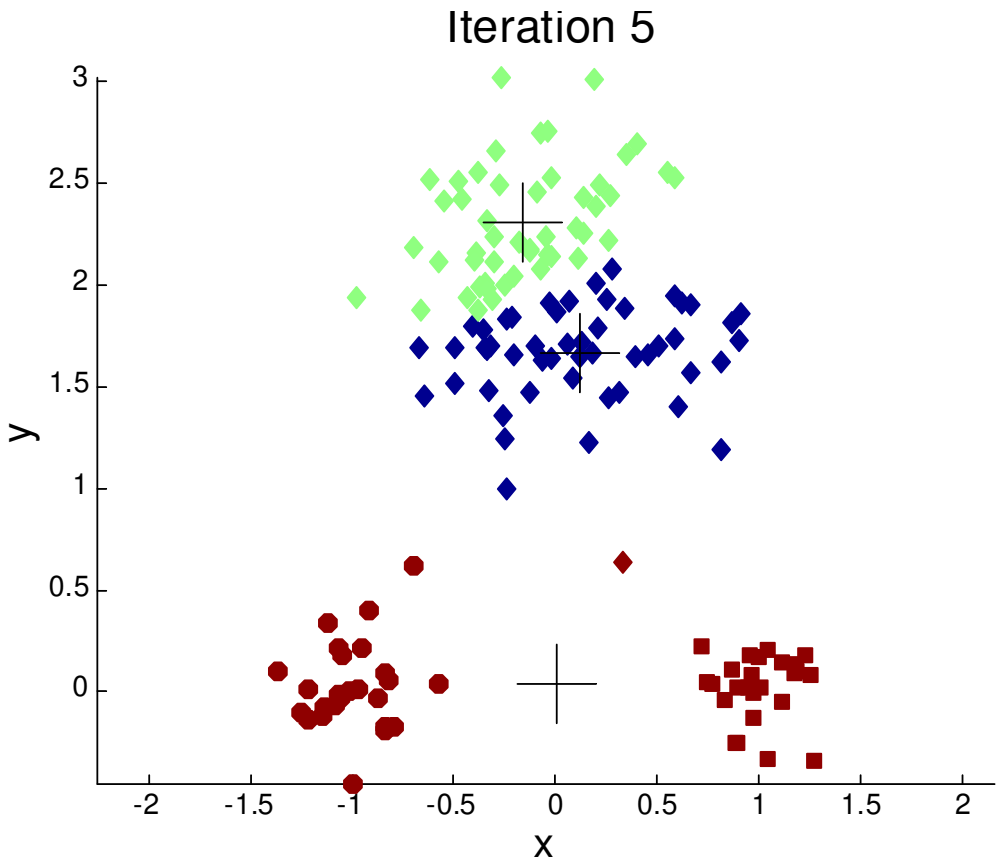
Importância de se escolher os centroides iniciais



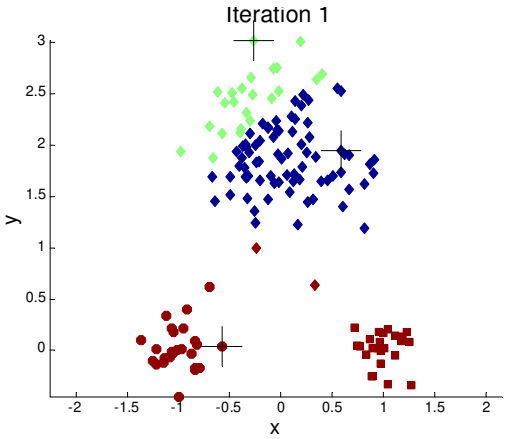
Importância de se escolher os centroides iniciais

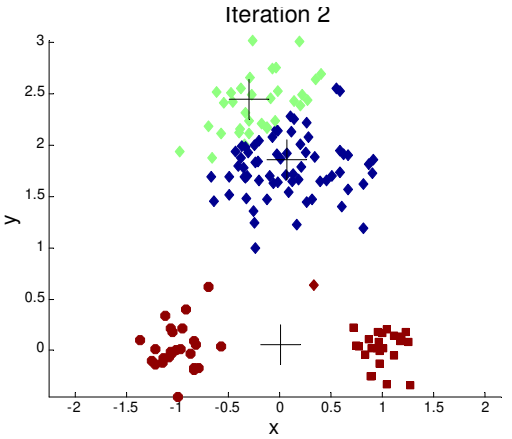


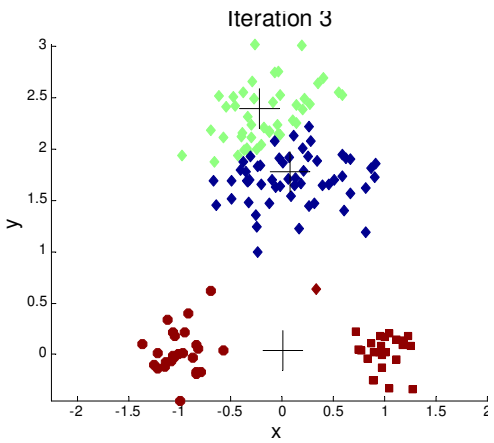
Importância de se escolher os centroides iniciais

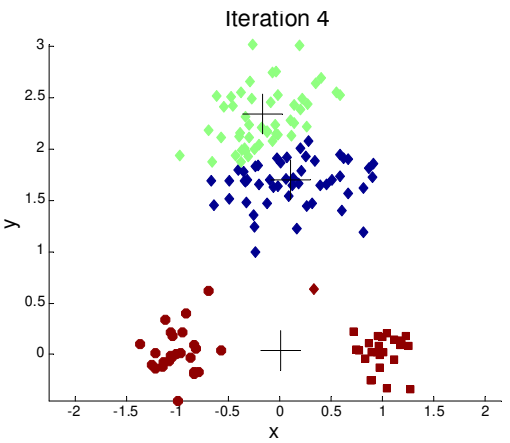


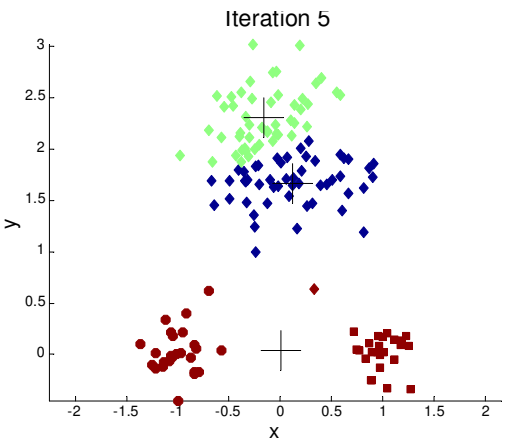
Importância de se escolher os centroides iniciais











Avaliando os Clusters Gerados

- A medida mais comum é a soma dos erros quadrados (*Sum of Squared Error - SSE*)
  - Para cada ponto, o erro é a distância ao centroide mais próximo

$$SSE = \sum_{i=1}^{K} \sum_{x \in C_i} dist^2(m_i, x)$$

  - x é um ponto de dados no cluster Ci e mi é o ponto representativo (centroide) do cluster Ci
  - Uma maneira fácil de reduzir o SSE é aumentar K, o número de clusters
    - Um bom particionamento com um K pequeno pode ter um SSE menor do que um mau particionamento com um K maior
  - Dado o numero k de clusters, executa-se o algoritmo varias vezes para este k. Aquele que gerar os menores SSE será o de melhor resultado

Pré-Processamento

- Normalizar os dados
- Eliminar exceções (outliers)

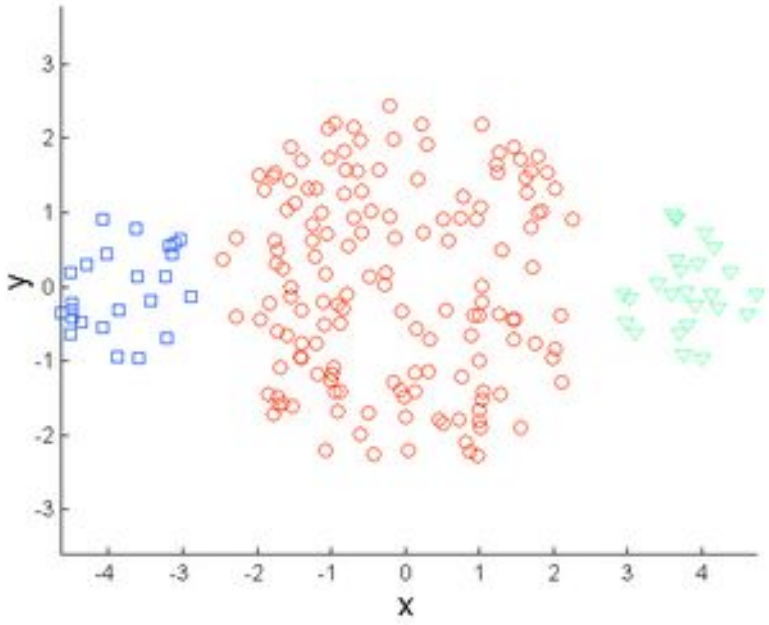
Pós-Processamento

- Eliminar os clusters pequenos que podem representar outliers
- Dividir os clusters “fracos”, ou seja, clusters com SSE relativamente alto
- Juntar os clusters que estão “perto” e que tenham SSE relativamente baixo

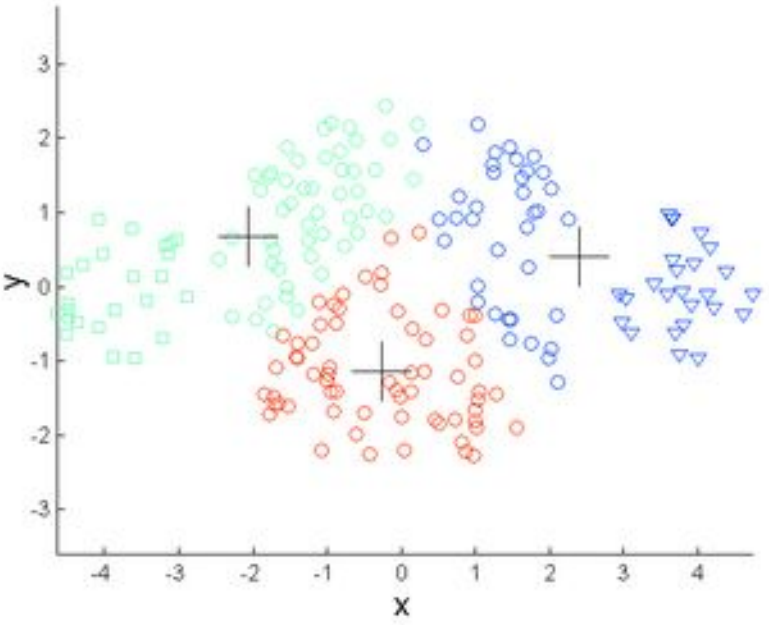
K-means: Limitações

- O K-means tem problemas quando os clusters têm
  - Tamanhos diferentes
  - Densidades diferentes
  - Formato não esférico
- O K-means tem problemas quando os dados contêm outliers.

Tamanhos Diferentes

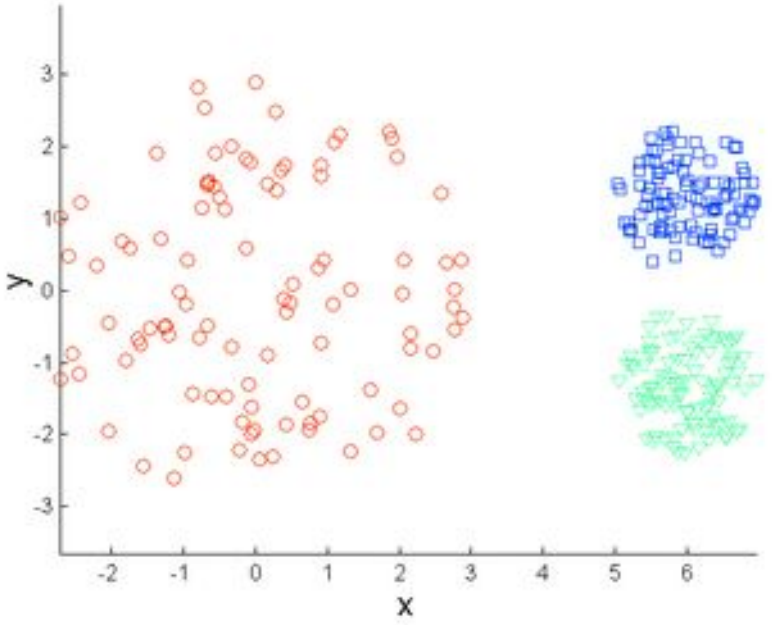


Pontos originais

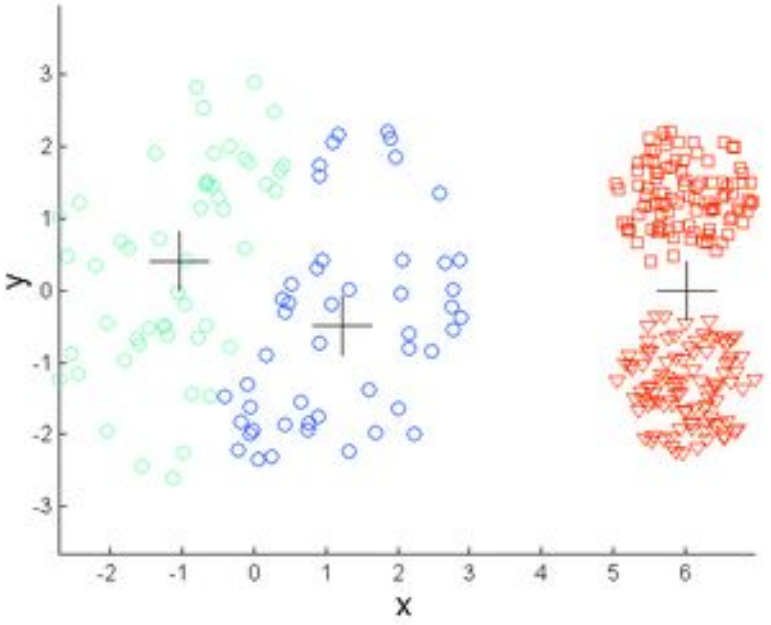


K-means (3 Clusters)

Densidades Diferentes

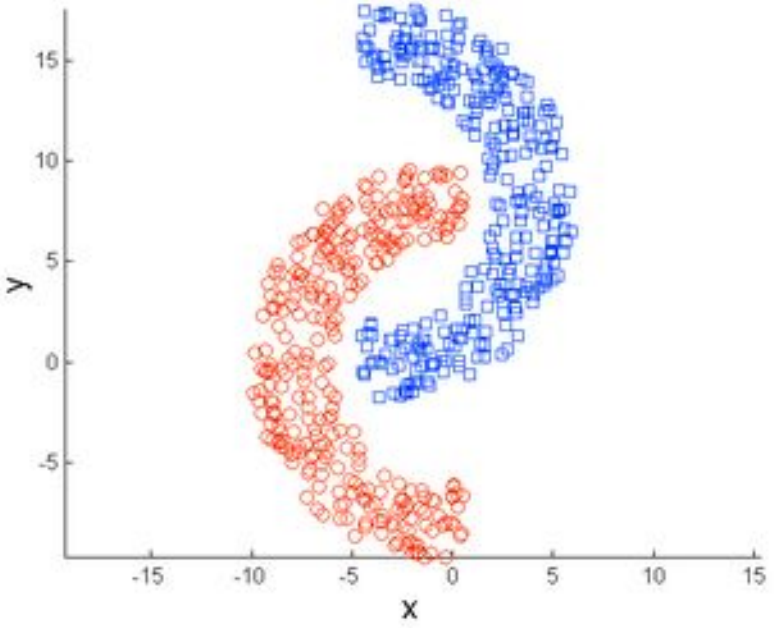


Pontos originais

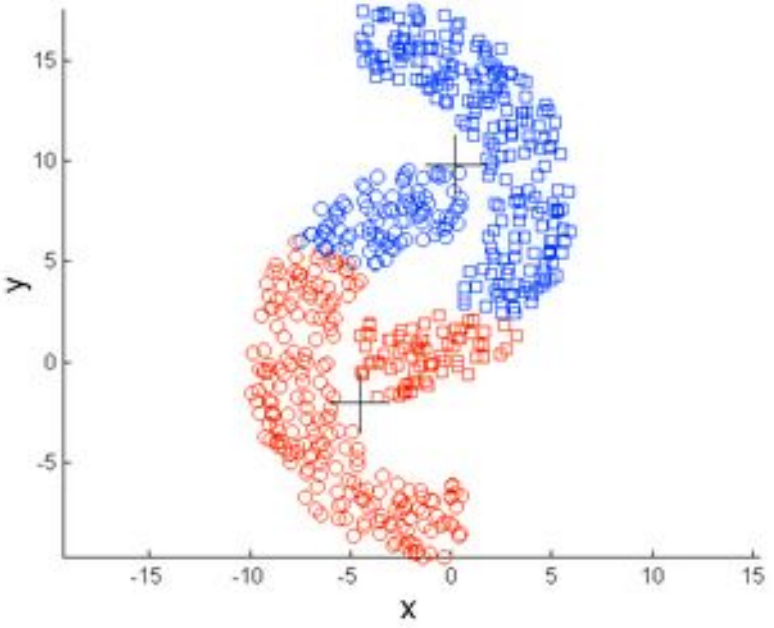


K-means (3 Clusters)

Formatos Não Esféricos



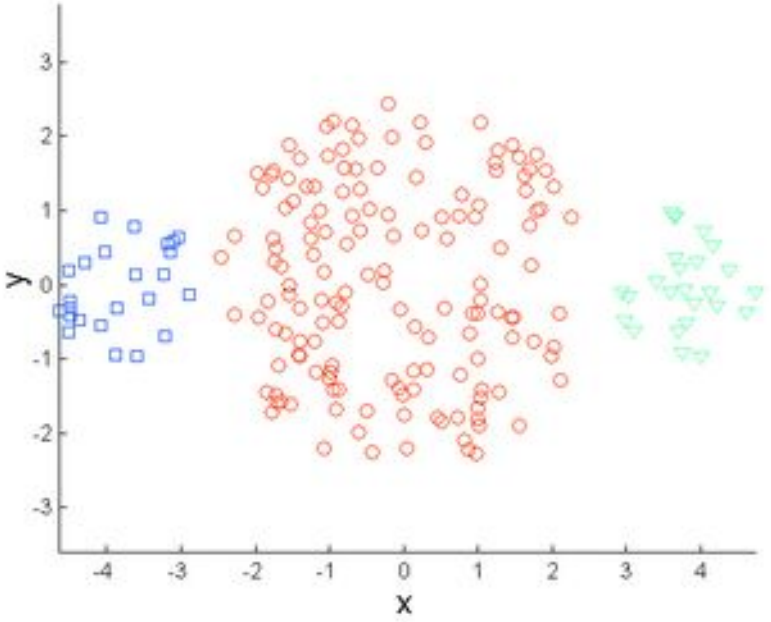
Pontos originais



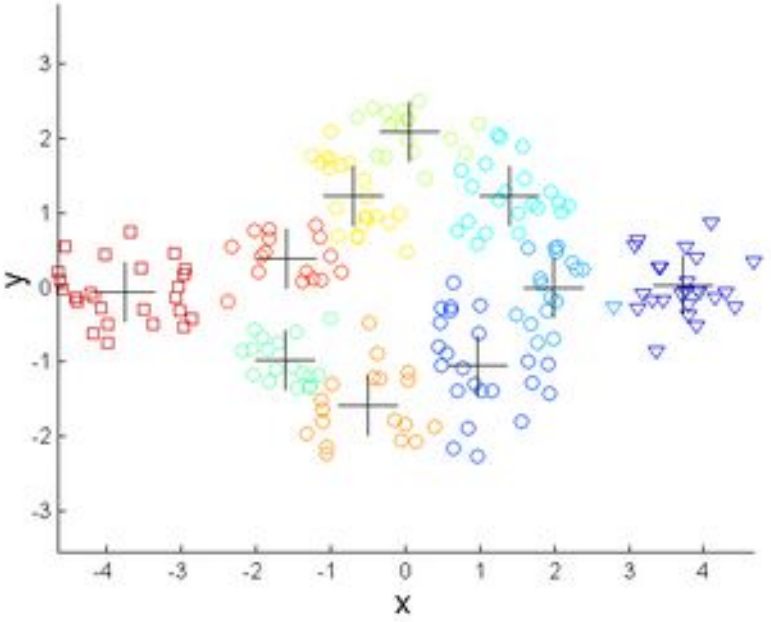
K-means (2 Clusters)

Solução

- Usar muitos clusters. Encontra partes de clusters, mas que precisam ser unidos.

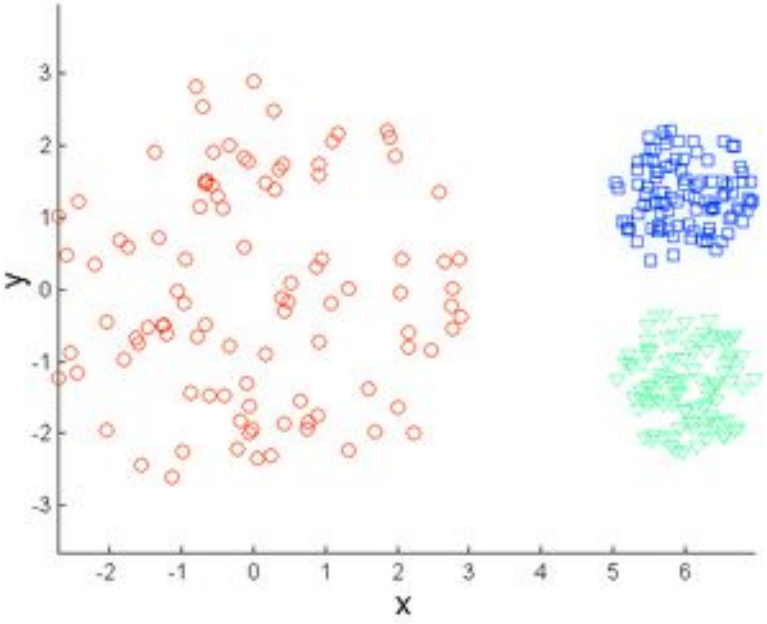


Pontos originais

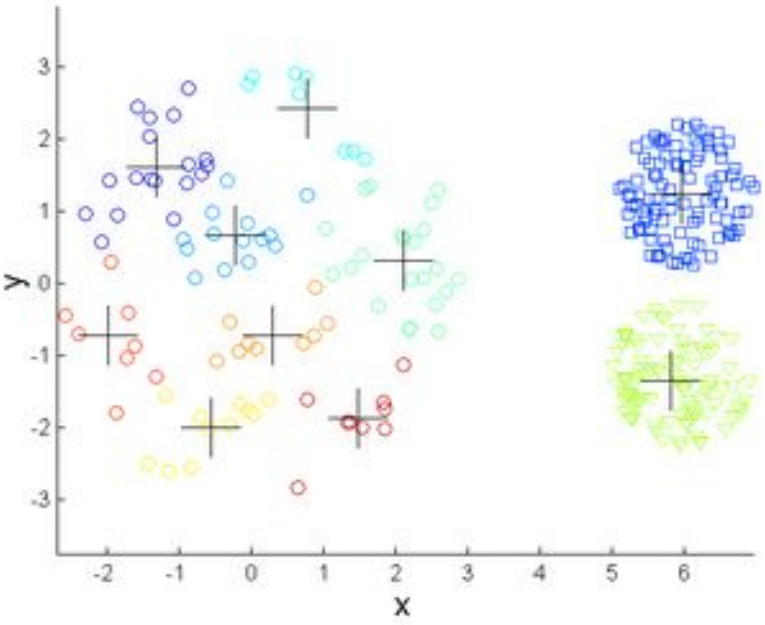


Clusters do K-means

Solução

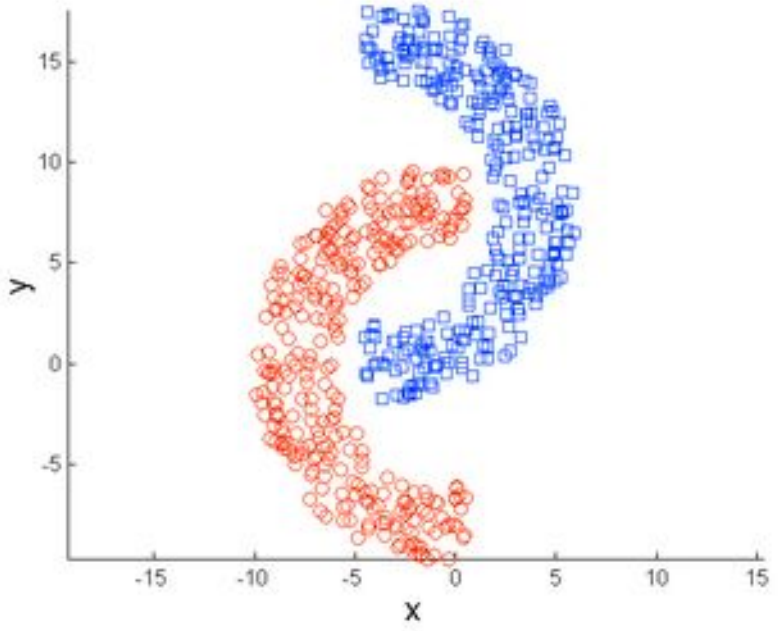


Pontos originais

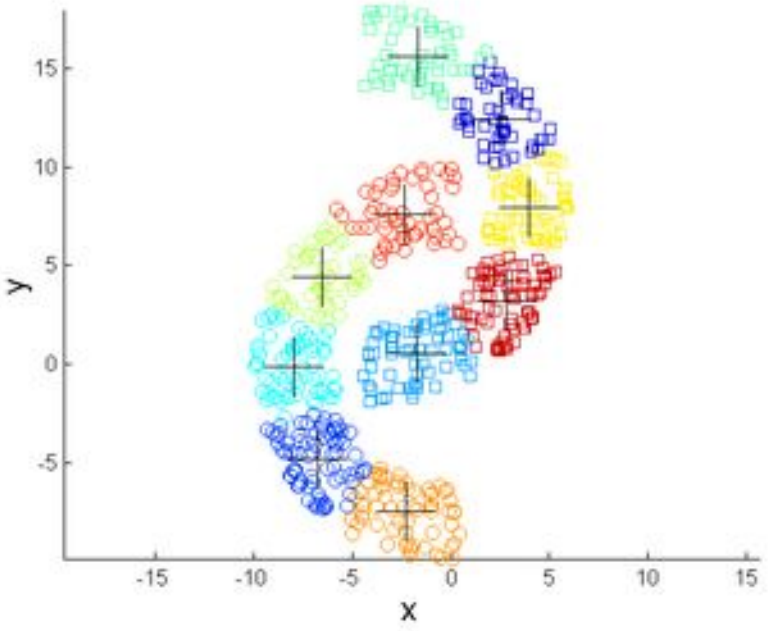


clusters do K-means

Solução



Pontos originais



clusters do K-means

K-means

- O algoritmo K-médias é sensível a ruídos
  - um objeto com um valor extremamente grande pode distorcer a distribuição de dados.
- Para diminuir essa sensibilidade
  - ao invés de utilizar o valor médio dos objetos em um cluster como um ponto referência, usa a mediana, que é o objeto mais centralmente localizado em um cluster.

Inteligência Artificial

Categorias de Aprendizado de Máquina

Prof. Saulo Popov Zambiasi
saulopz@gmail.com