Curso 2 – CD, AM e DM

Mineração de Dados

Parte 8
Extração de Padrões
Análise de Grandes Volumes de Dados

Prof. Ricardo M. Marcacini
ricardo.marcacini@icmc.usp.br

Grandes Bases de Dados

- Até o momento, estudamos soluções que exploram medidas de proximidade e vizinhos mais próximos
- Limitações
  - **Tempo**
    - O custo computacional depende do número de objetos e quantidade de atributos
  - **Memória**
    - Conjunto de dados não pode ser carregado completamente em memória

Grandes Bases de Dados

- Acelerar o cálculo da medida de distância
- Amostragem de dados para reduzir uso de memória e custo computacional
- Representações condensadas do conjunto de dados

Grandes Bases de Dados

- Acelerar o cálculo da medida de distância
- Amostragem de dados para reduzir uso de memória e custo computacional
- Representações condensadas do conjunto de dados

Medidas de Proximidade

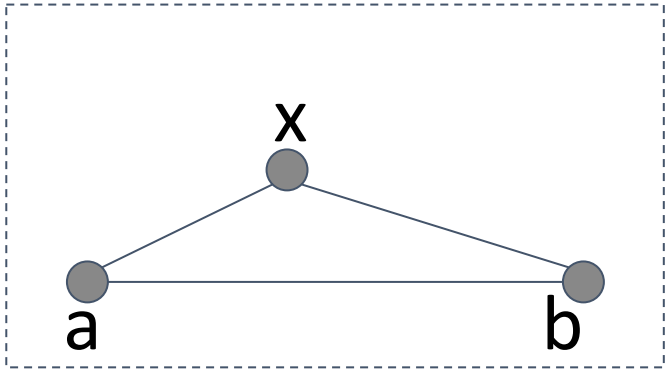
- Base para métodos de agrupamento e de classificação baseada em instâncias

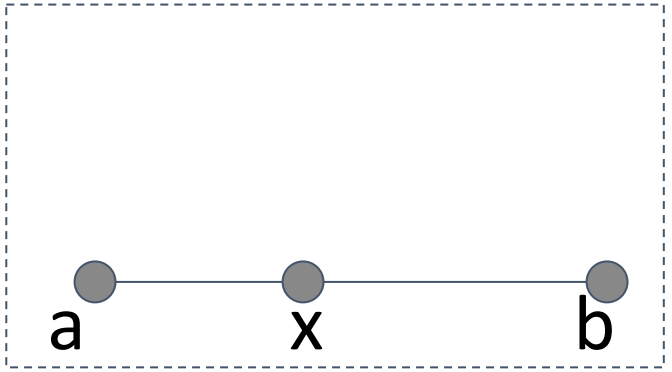
- Propriedades desejáveis (para dissimilaridade)
  - Simetria $d(\mathbf{x}_i, \mathbf{x}_j) = d(\mathbf{x}_j, \mathbf{x}_i) \quad \forall \mathbf{x}_i, \mathbf{x}_j$ *Métrica!*
  - Positividade $d(\mathbf{x}_i, \mathbf{x}_j) \geq 0 \quad \forall \mathbf{x}_i, \mathbf{x}_j$
  - Reflexividade $d(\mathbf{x}_i, \mathbf{x}_j) = 0$ se, e somente se, $\mathbf{x}_i = \mathbf{x}_j$
  - **Desigualdade Triangular**

Medida de Proximidade (Métrica)

- Desigualdade Triangular
  - O comprimento de um dos lados de um triângulo não excede a soma dos outros dois.

$$d(a,b) \leq d(x,b) + d(x,a)$$

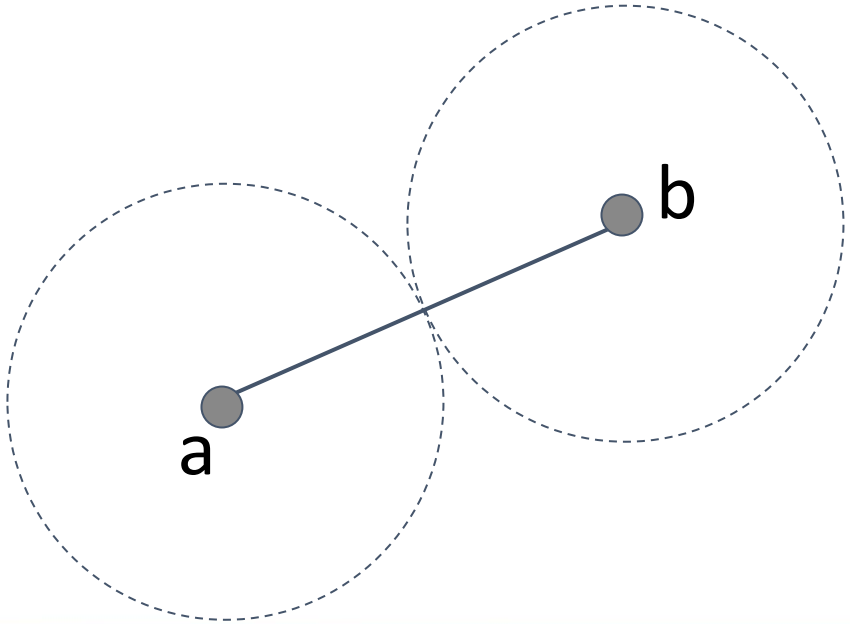




Medida de Proximidade (Métrica)

- Desigualdade Triangular
  - O comprimento de um dos lados de um triângulo não excede a soma dos outros dois.

$$d(a,b) \leq d(x,b) + d(x,a)$$

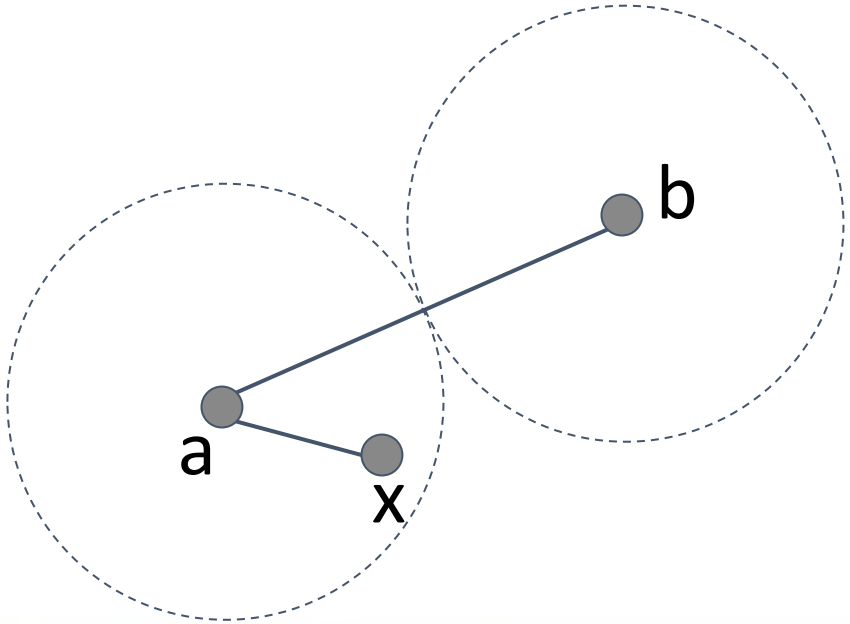


Propriedade pode ser explorada para evitar computar uma parcela distâncias, o que promove uma aceleração do processo.

Medida de Proximidade (Métrica)

- Desigualdade Triangular
  - O comprimento de um dos lados de um triângulo não excede a soma dos outros dois.

$$d(a,b) \leq d(x,b) + d(x,a)$$

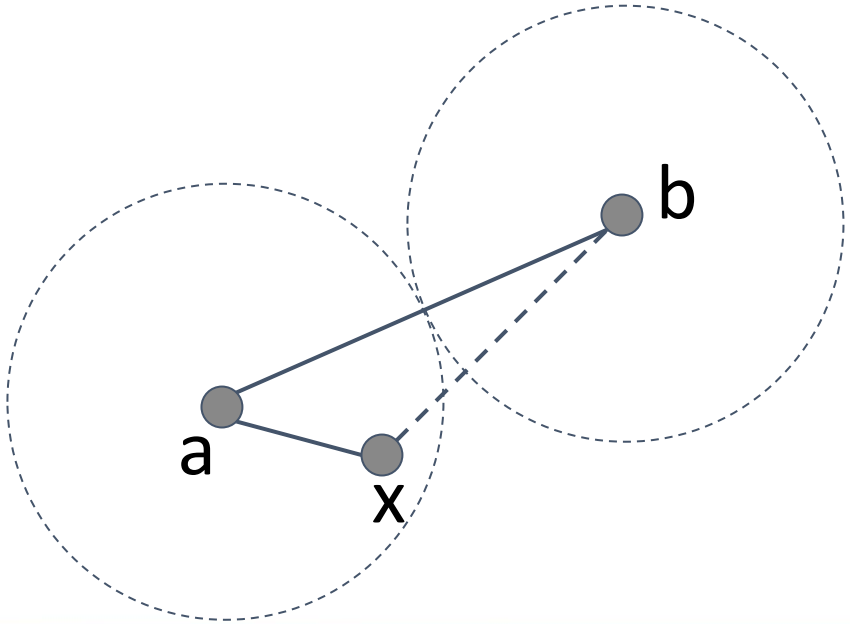


Propriedade pode ser explorada para evitar computar uma parcela distâncias, o que promove uma aceleração do processo.

Medida de Proximidade (Métrica)

- Desigualdade Triangular
  - O comprimento de um dos lados de um triângulo não excede a soma dos outros dois.

$$d(a,b) \leq d(x,b) + d(x,a)$$

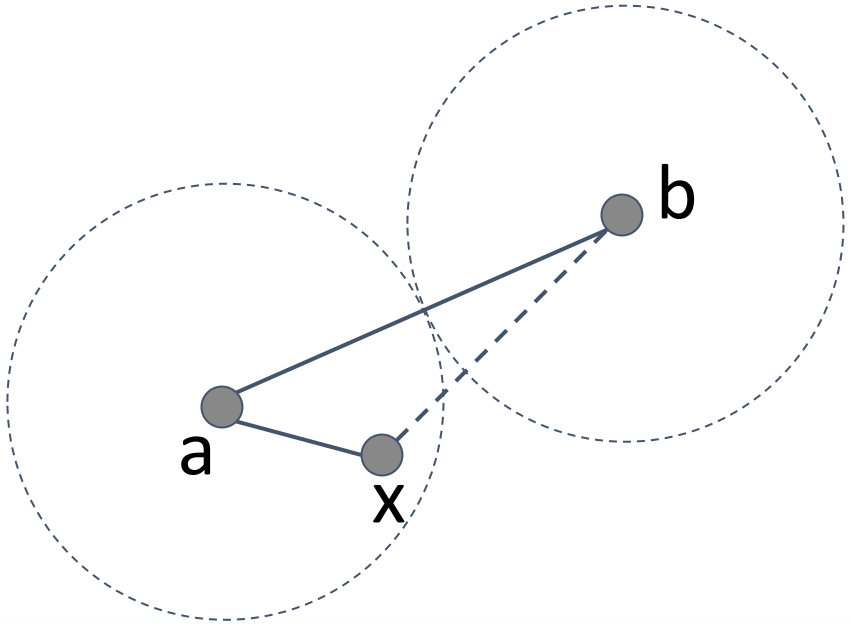


Propriedade pode ser explorada para evitar computar uma parcela distâncias, o que promove uma aceleração do processo.

Medida de Proximidade (Métrica)

- Desigualdade Triangular
  - O comprimento de um dos lados de um triângulo não excede a soma dos outros dois.

$$d(a,b) \leq d(x,b) + d(x,a)$$



Propriedade pode ser explorada para evitar computar uma parcela distâncias, o que promove uma aceleração do processo.

Limitações: seu uso eficiente depende do contexto, como clustering e classificação kNN

Grandes Bases de Dados

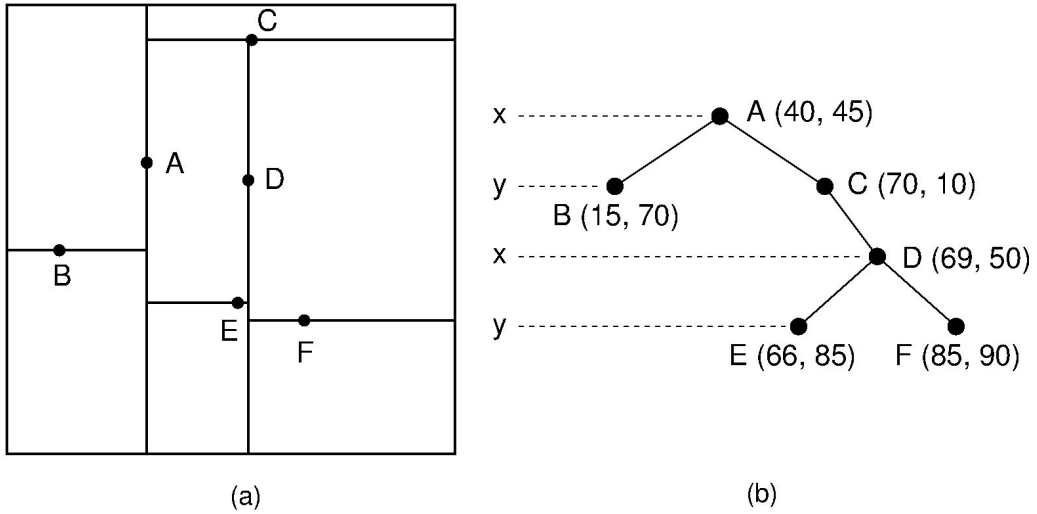
- Acelerar o cálculo da medida de distância
- Amostragem de dados para reduzir uso de memória e custo computacional
- Representações condensadas do conjunto de dados

Grandes Bases de Dados

- Particionamento do Espaço
  - Dividir um espaço (e.g. euclidiano) em dois ou mais subconjuntos disjuntos
  - Qualquer objeto (ponto) no espaço pode ser alocado em uma das regiões
  - Permite buscas mais rápidas por “podar” regiões inteiras do espaço de busca

Grandes Bases de Dados

- Particionamento do Espaço com KD-Tree
  - KD-Tree = Árvore k-dimensional
  - Árvore binária em que cada nó é um ponto k-dimensional
  - Cada nó não folha divide o espaço em duas partes



(a) (b)

- Pontos na subárvore com um valor menor do que o nó aparecerão na subárvore esquerda, caso contrário aparecerão na subárvore direita.
- Os níveis da árvore representam ciclos para alternar o atributo utilizado na divisão do espaço

Fonte: https://opendsa-server.cs.vt.edu/ODSA/Books/CS3/html/KDtree.html

KD-Tree

- Exemplo

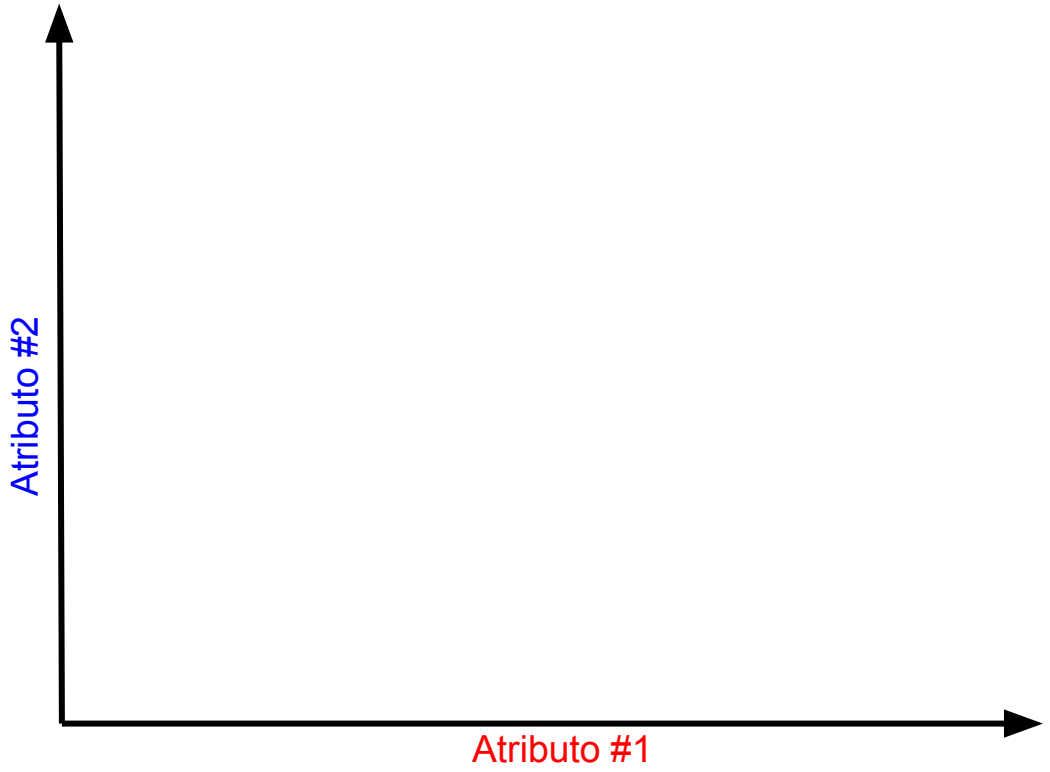
Objeto	Atributo #1	Atributo #2
1	5	4
2	2	6
3	13	3
4	8	7
5	3	1
6	10	2

Exemplo em vídeo disponível em: https://www.youtube.com/watch?v=Glp7THUpGow

KD-Tree

- Exemplo

Objeto	Atributo #1	Atributo #2
1	5	4
2	2	6
3	13	3
4	8	7
5	3	1
6	10	2

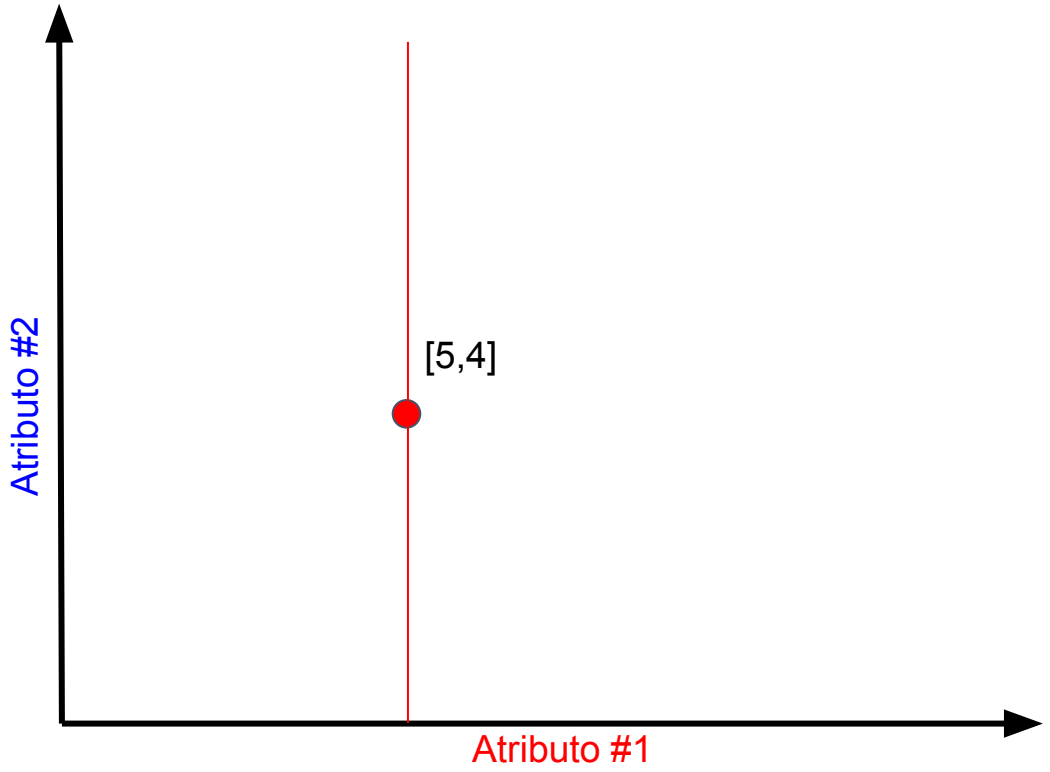


KD-Tree

[5,4]

- Exemplo

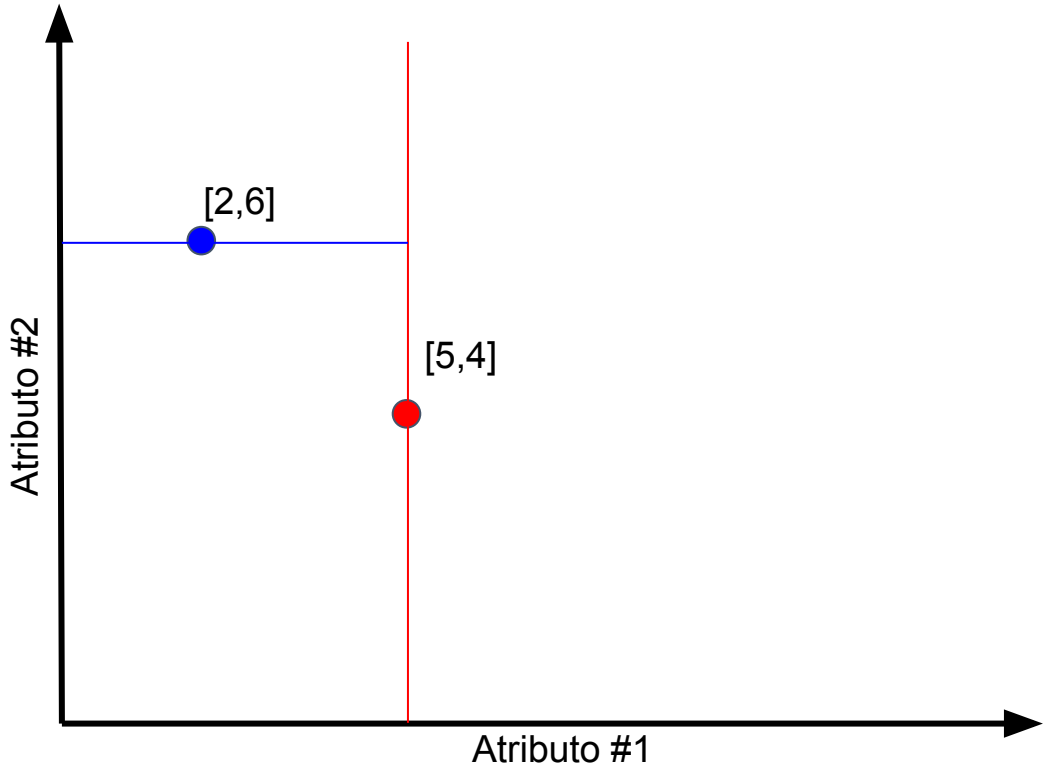
Objeto	Atributo #1	Atributo #2
1	5	4
2	2	6
3	13	3
4	8	7
5	3	1
6	10	2

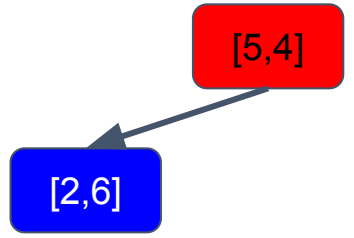


KD-Tree

- Exemplo

Objeto	Atributo #1	Atributo #2
1	5	4
2	2	6
3	13	3
4	8	7
5	3	1
6	10	2

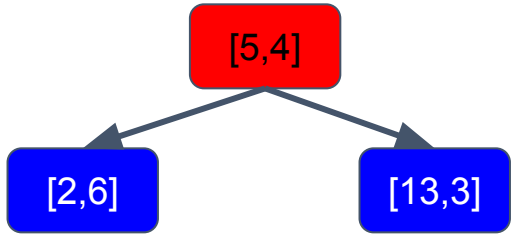


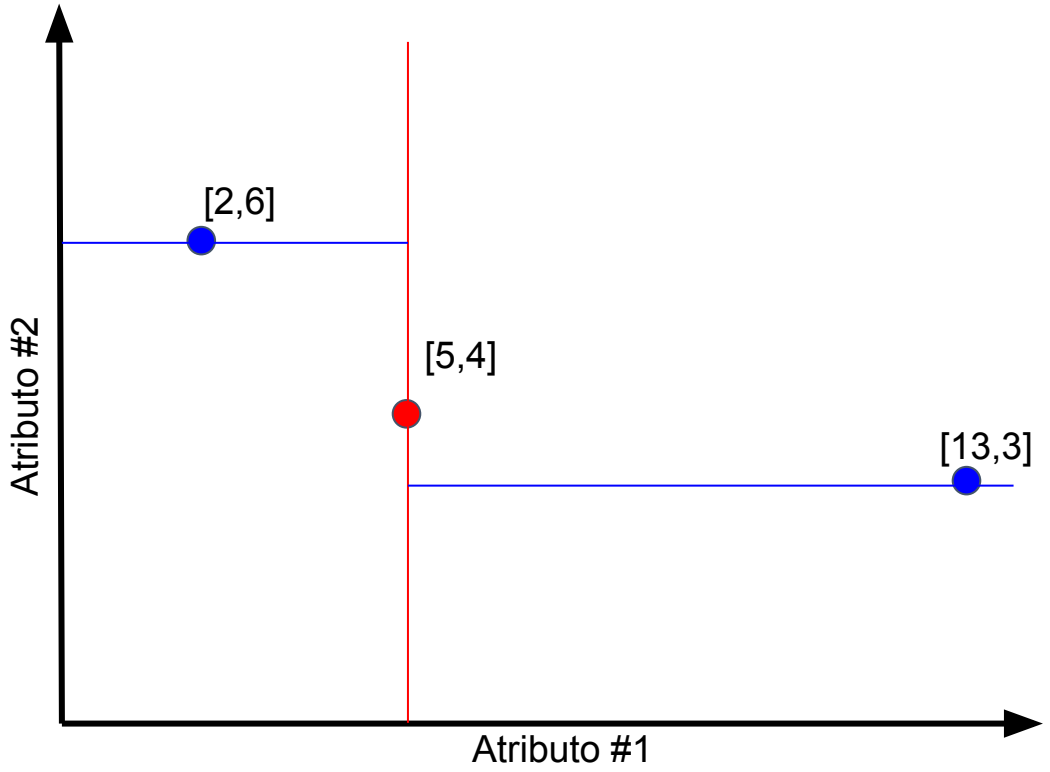


KD-Tree

- Exemplo

Objeto	Atributo #1	Atributo #2
1	5	4
2	2	6
3	13	3
4	8	7
5	3	1
6	10	2

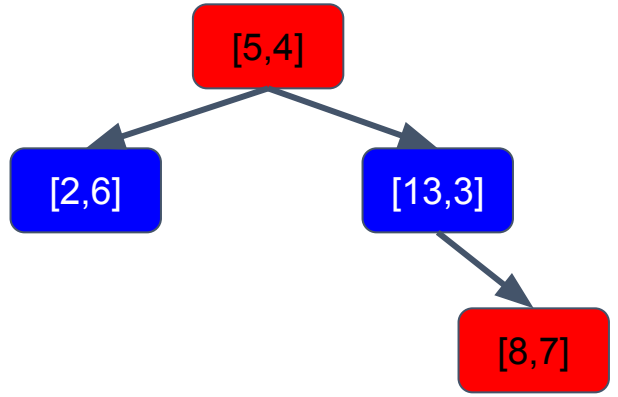


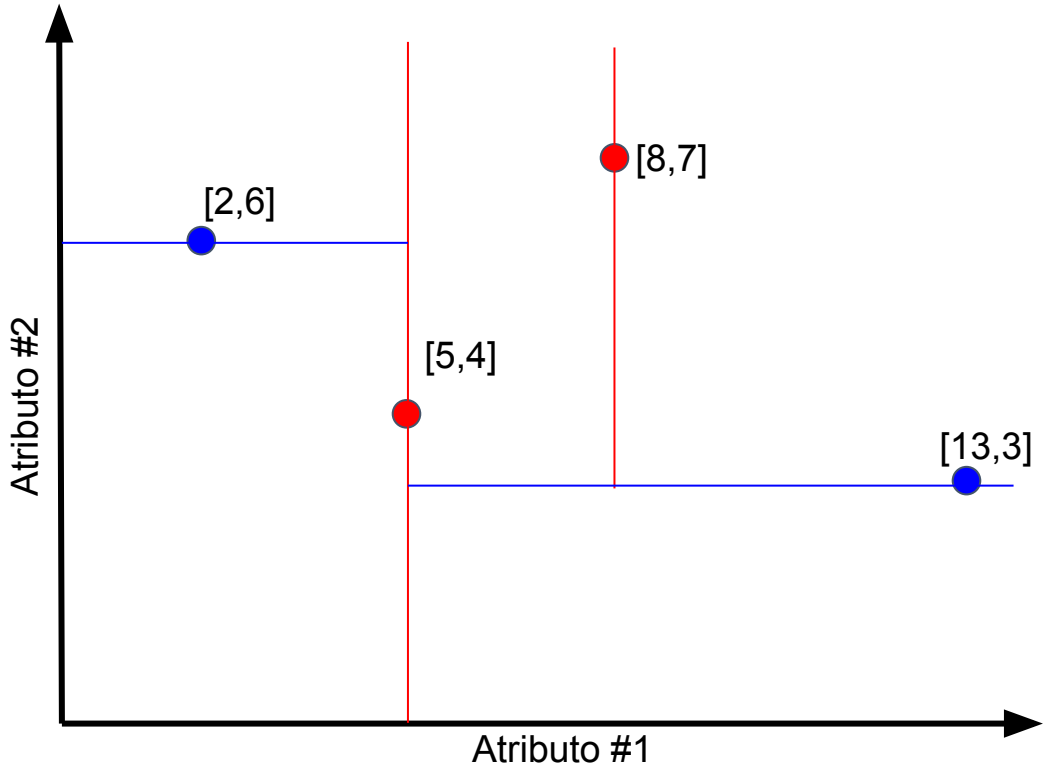


KD-Tree

- Exemplo

Objeto	Atributo #1	Atributo #2
1	5	4
2	2	6
3	13	3
4	8	7
5	3	1
6	10	2

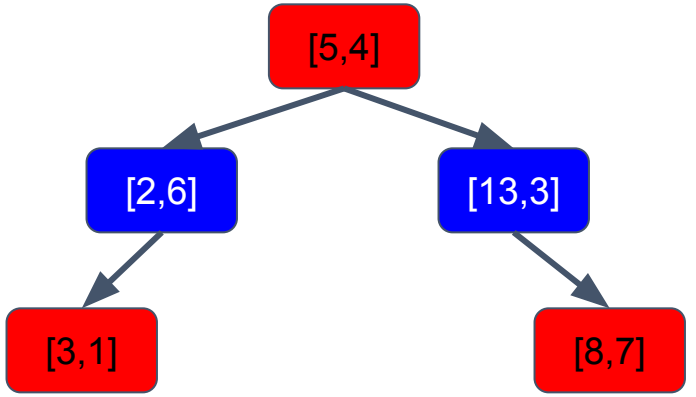


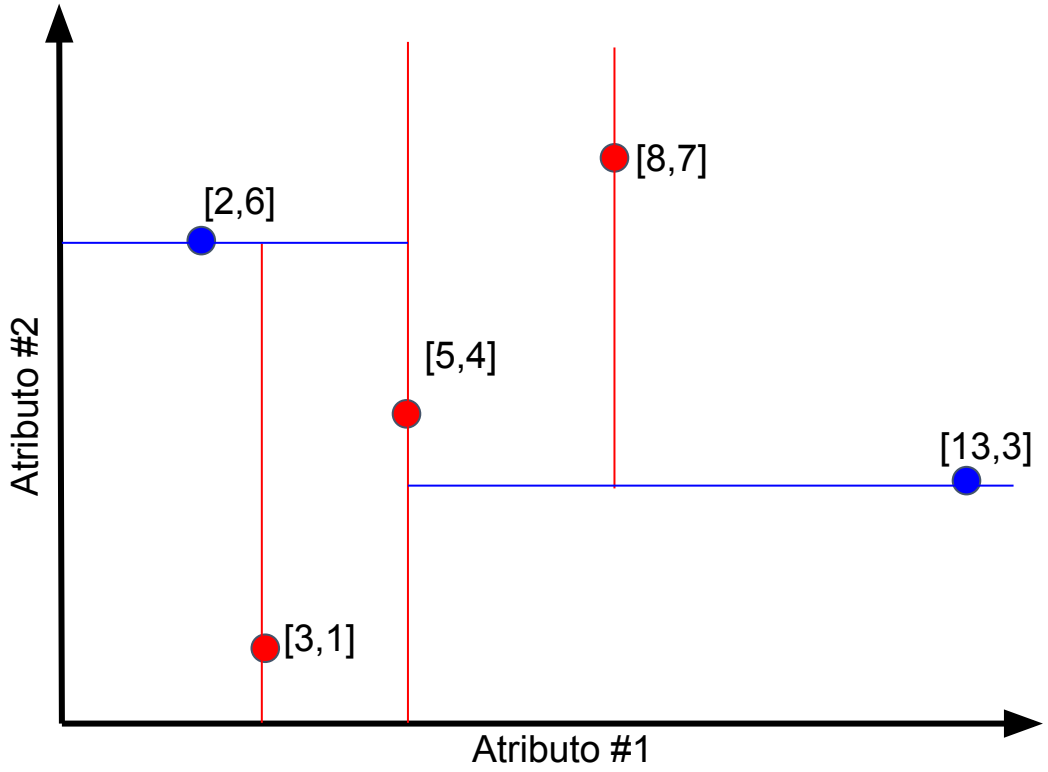


KD-Tree

- Exemplo

Objeto	Atributo #1	Atributo #2
1	5	4
2	2	6
3	13	3
4	8	7
5	3	1
6	10	2

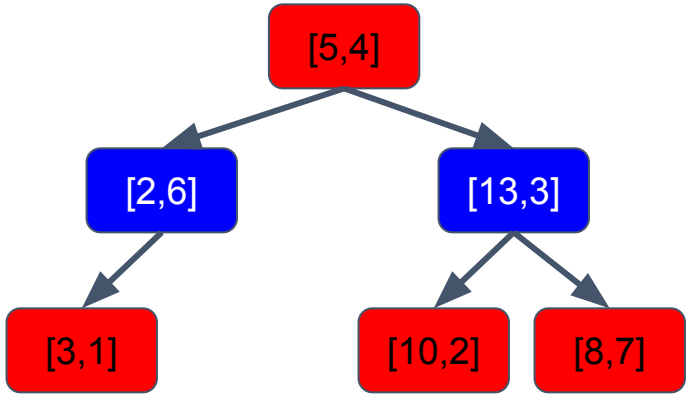


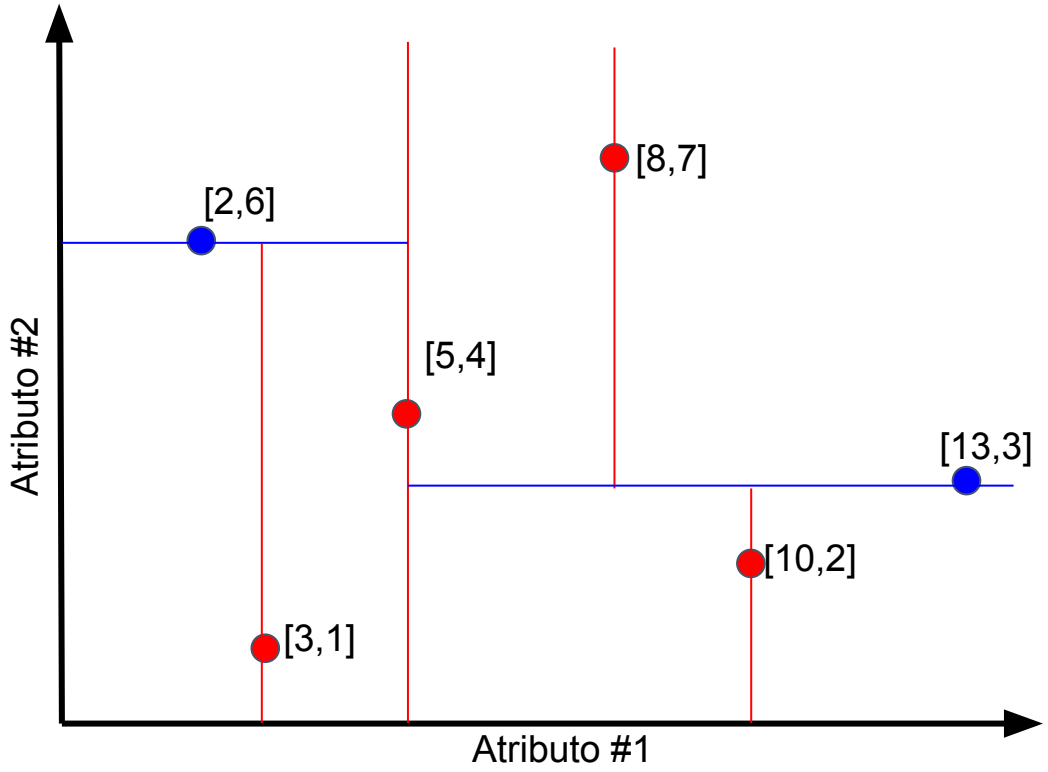


KD-Tree

- Exemplo

Objeto	Atributo #1	Atributo #2
1	5	4
2	2	6
3	13	3
4	8	7
5	3	1
6	10	2





Considerações sobre KD-Tree

- Escolher bons pontos para particionar o espaço é importante balanceamento da KD-Tree
  - Escolher pontos medianos
- A performance é reduzida em altas dimensões
  - Uma grande quantidade de atributos torna a KD-Tree próxima a uma busca linear
  - Alternativa é o uso de busca aproximada de vizinhos mais próximos (Dica: Annoy)
- *Ball-Tree* usa conceitos similares a KD-Tree, mas dividindo o espaço em (hiper)esferas

Grandes Bases de Dados

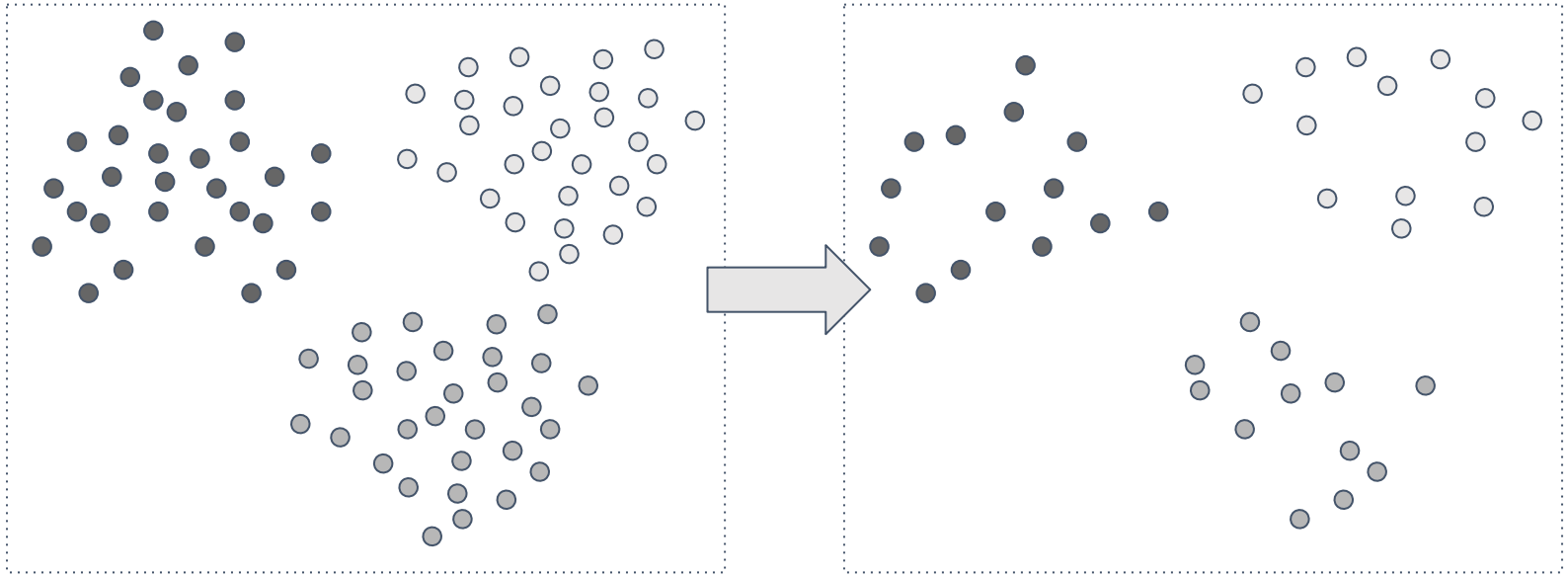
- Acelerar o cálculo da medida de distância

- Amostragem de dados para reduzir uso de memória e custo computacional

- Representações condensadas do conjunto de dados

Amostragem

- Analisar uma parcela de dados que representa os principais padrões da base de dados
  - Como gerar a amostra?

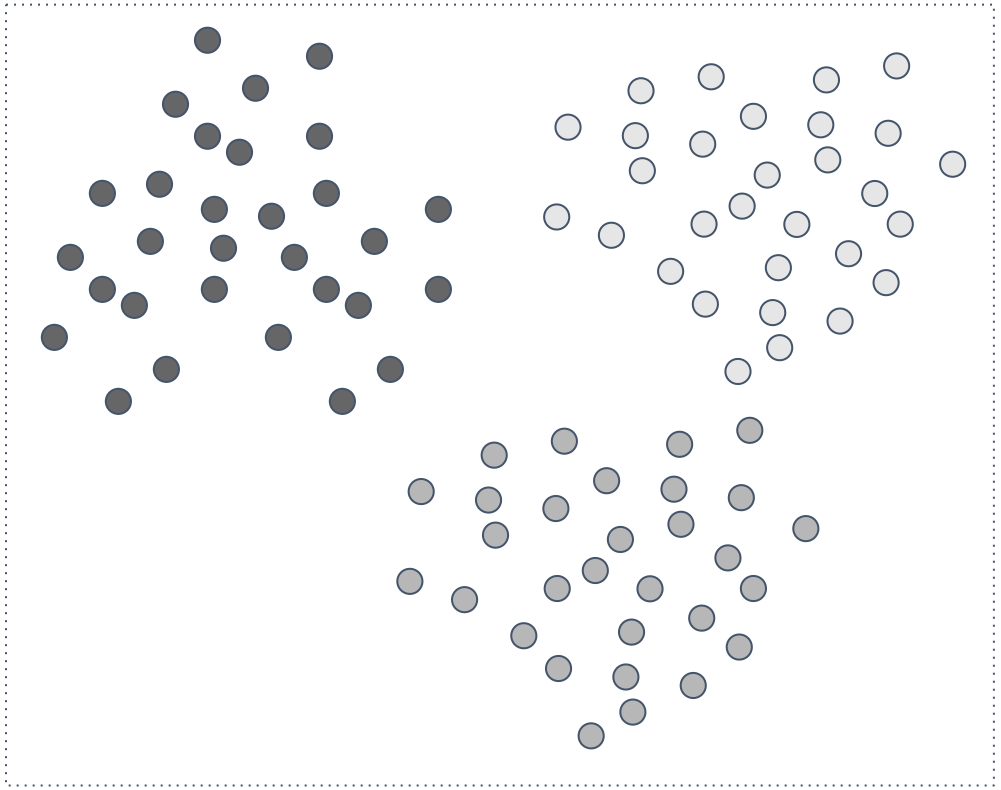


Mini-batch k-Means

- Variante do *K-Means* que usa mini-batch (amostras) para reduzir memória e tempo
- Amostras geradas aleatoriamente
- Duas etapas:
  - 1. Gerar amostras aleatoriamente do conjunto de dados, para formar um mini-batch e atribuir ao centroide mais próximo
  - 2. Atualizar centroides considerando a amostra atual e os centroides anteriores

Mini-batch k-Means

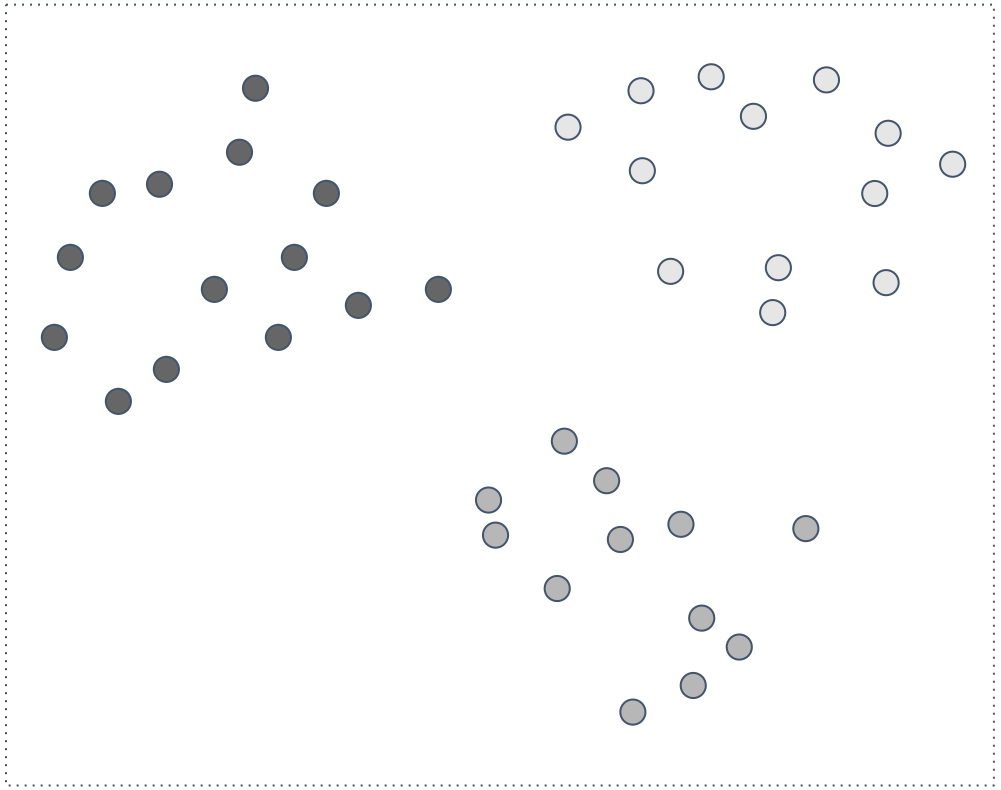
- Exemplo



1. Conjunto de dados (completo)

Mini-batch k-Means

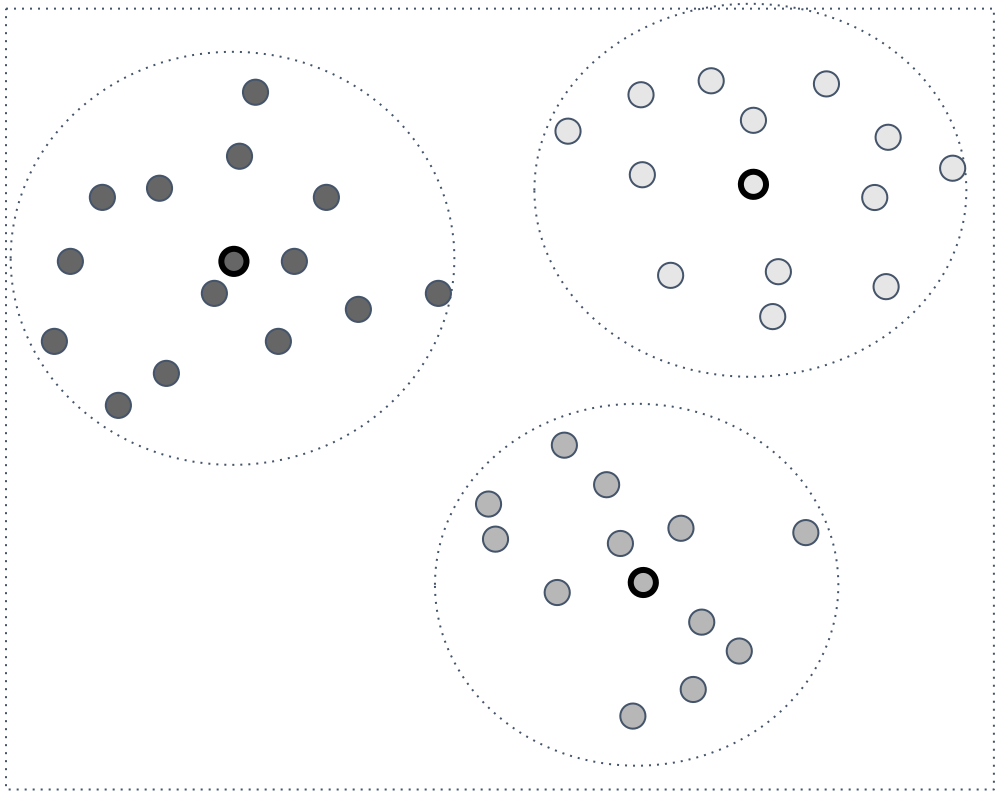
- Exemplo



1. Conjunto de dados (completo)
2. Gerar uma amostra aleatoriamente

Mini-batch k-Means

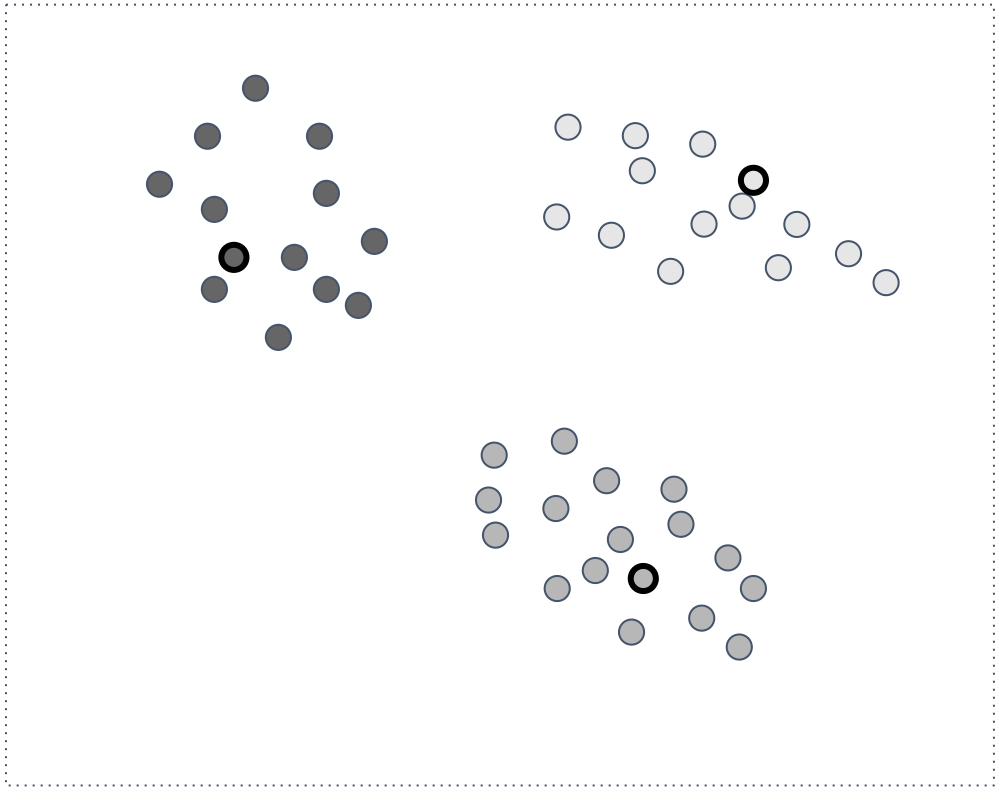
- Exemplo



1. Conjunto de dados (completo)
2. Gerar uma amostra aleatória
3. Atribuir objetos da amostra ao centroide mais próximo

Mini-batch k-Means

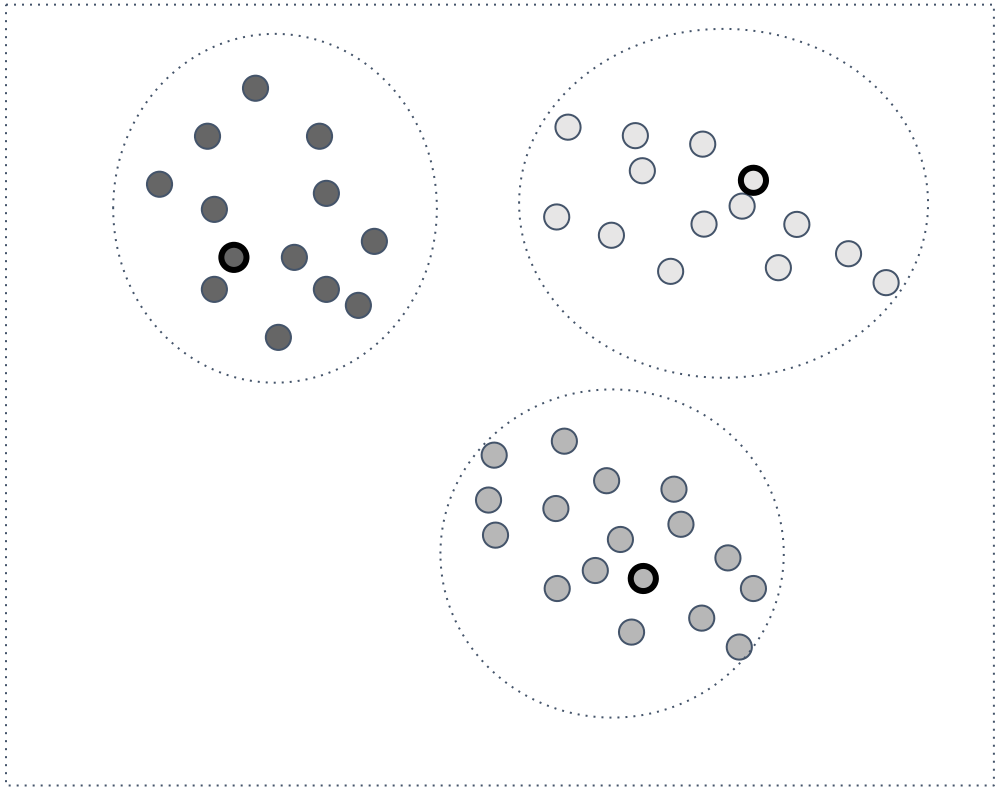
- Exemplo



1. Conjunto de dados (completo)
2. Gerar uma amostra aleatória
3. Atribuir objetos da amostra ao centroide mais próximo
4. Gerar uma amostra aleatoriamente

Mini-batch k-Means

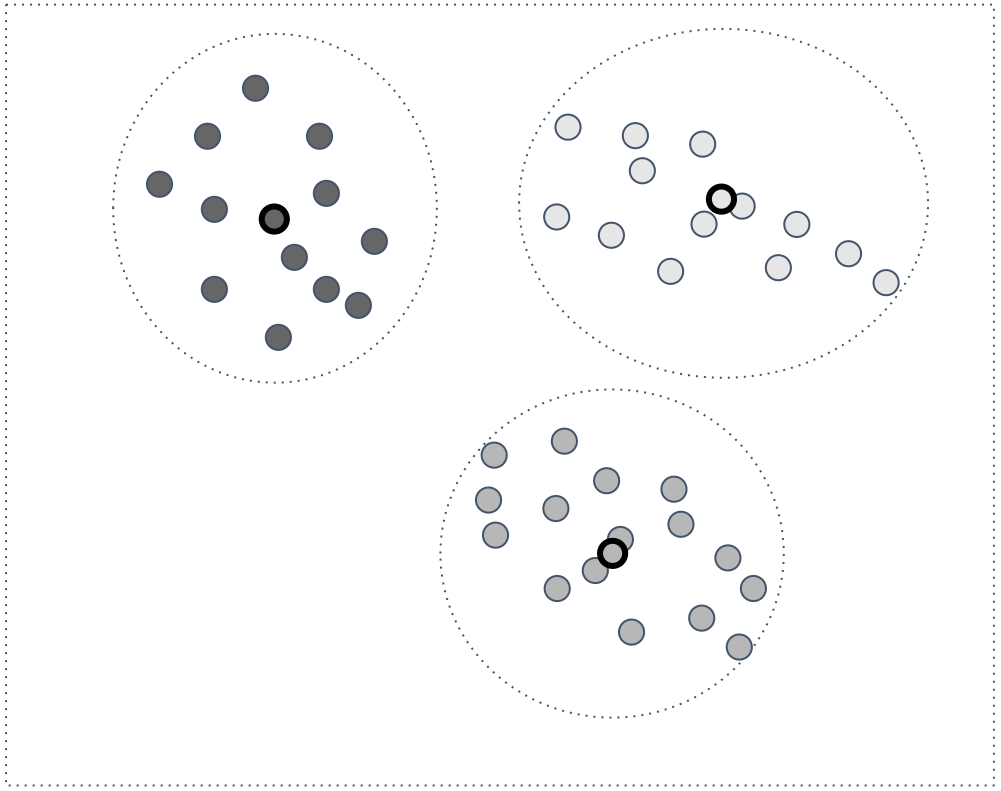
- Exemplo



1. Conjunto de dados (completo)
2. Gerar uma amostra aleatória
3. Atribuir objetos da amostra ao centroide mais próximo e atualizar
4. Gerar uma amostra aleatória
5. Atribuir objetos da amostra ao centroide mais próximo e atualizar

Mini-batch k-Means

- Exemplo



1. Conjunto de dados (completo)
2. Gerar uma amostra aleatória
3. Atribuir objetos da amostra ao centroide mais próximo e atualizar
4. Gerar uma amostra aleatória
5. Atribuir objetos da amostra ao centroide mais próximo e atualizar
6. ...

Considerações: *Mini-batch k-Means*

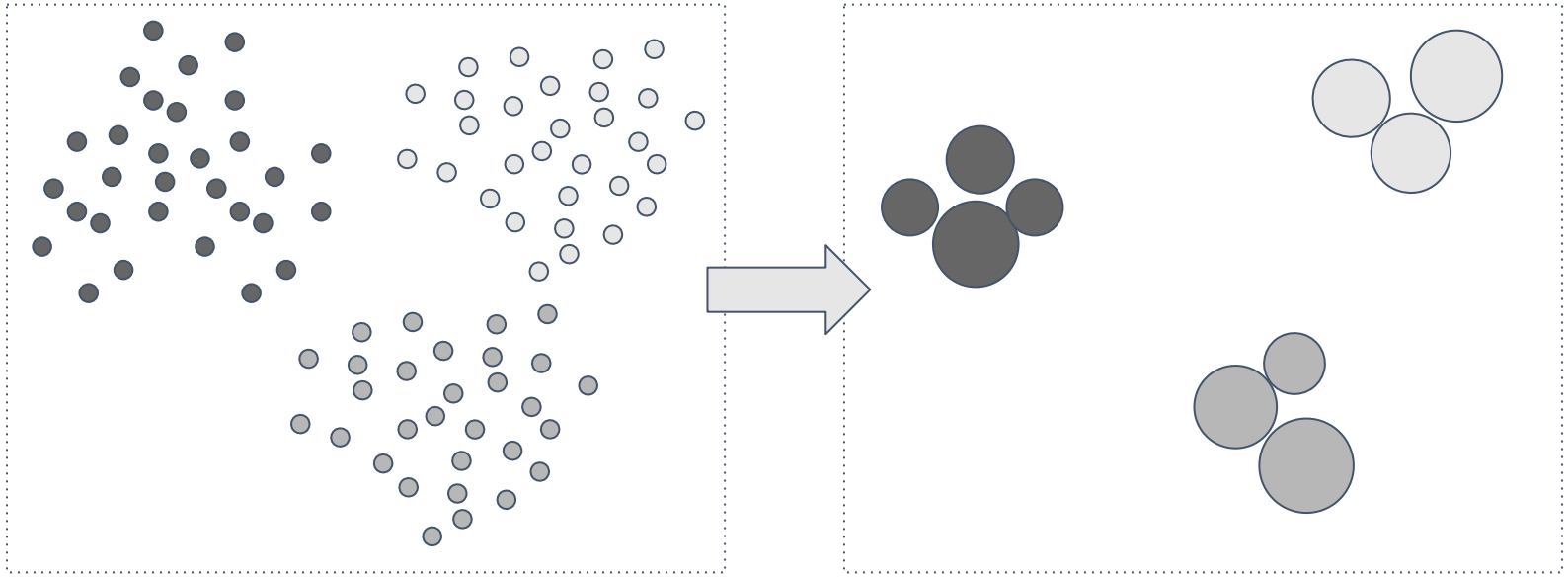
- Converge mais rápido que o *k-Means*
- A qualidade do agrupamento costuma ser ligeiramente pior que o *k-Means*
- Em termos práticos, a (pouca) diferença de qualidade compensa (grandes bases de dados)

Grandes Bases de Dados

- Acelerar o cálculo da medida de distância
- Amostragem de dados para reduzir uso de memória e custo computacional
- Representações condensadas do conjunto de dados

Representações Condensadas

- Ideia geral: substituir pontos originais do conjunto de dados por representantes.
  - Como obter o representantes?



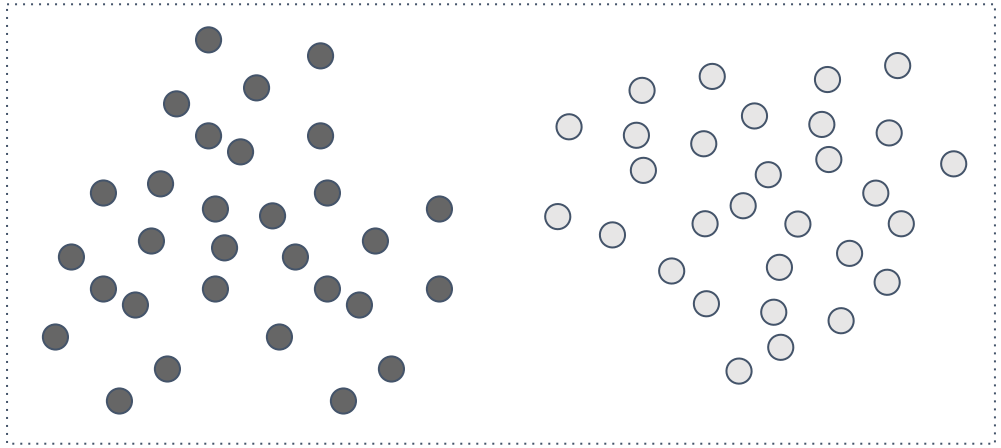
BIRCH ***(Balanced Iterative Reducing and Clustering using Hierarchies)***

- Um método baseado em *multi-level clustering*
  - Micro-clustering:
    - Computa um sumário com estatísticas do conjunto de dados
    - Reduz o custo computacional
  - Macro-clustering
    - Aumenta flexibilidade e integração com outros métodos de agrupamento

Geração de uma CF-TREE (*Cluster-Feature Tree*) com uma simples leitura da base de dados permite a geração de micro-clustering.

BIRCH

- *Cluster Features (CF)*
  - Representação intermediária (condensada) do conjunto de dados



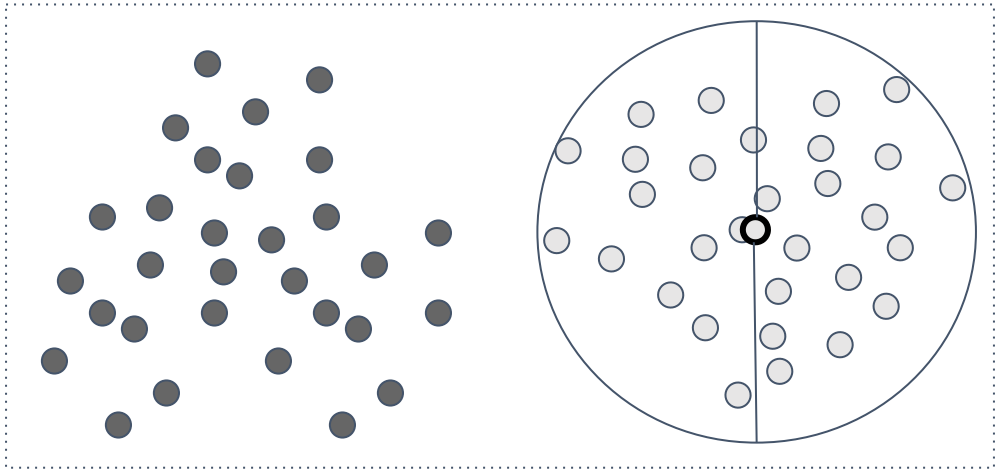
$$CF = (N, LS, SS)$$

- *N* = Quantidade de objetos
- LS = Soma linear dos vetores de cada objeto
- SS = Soma ao quadrado dos vetores de cada objeto

$$LS = \sum_{i=1}^{N} X_i \qquad SS = \sum_{i=1}^{N} (X_i)^2$$

BIRCH

- *Cluster Features (CF)*
  - Representação intermediária (condensada) do conjunto de dados



$$CF = (N, LS, SS)$$

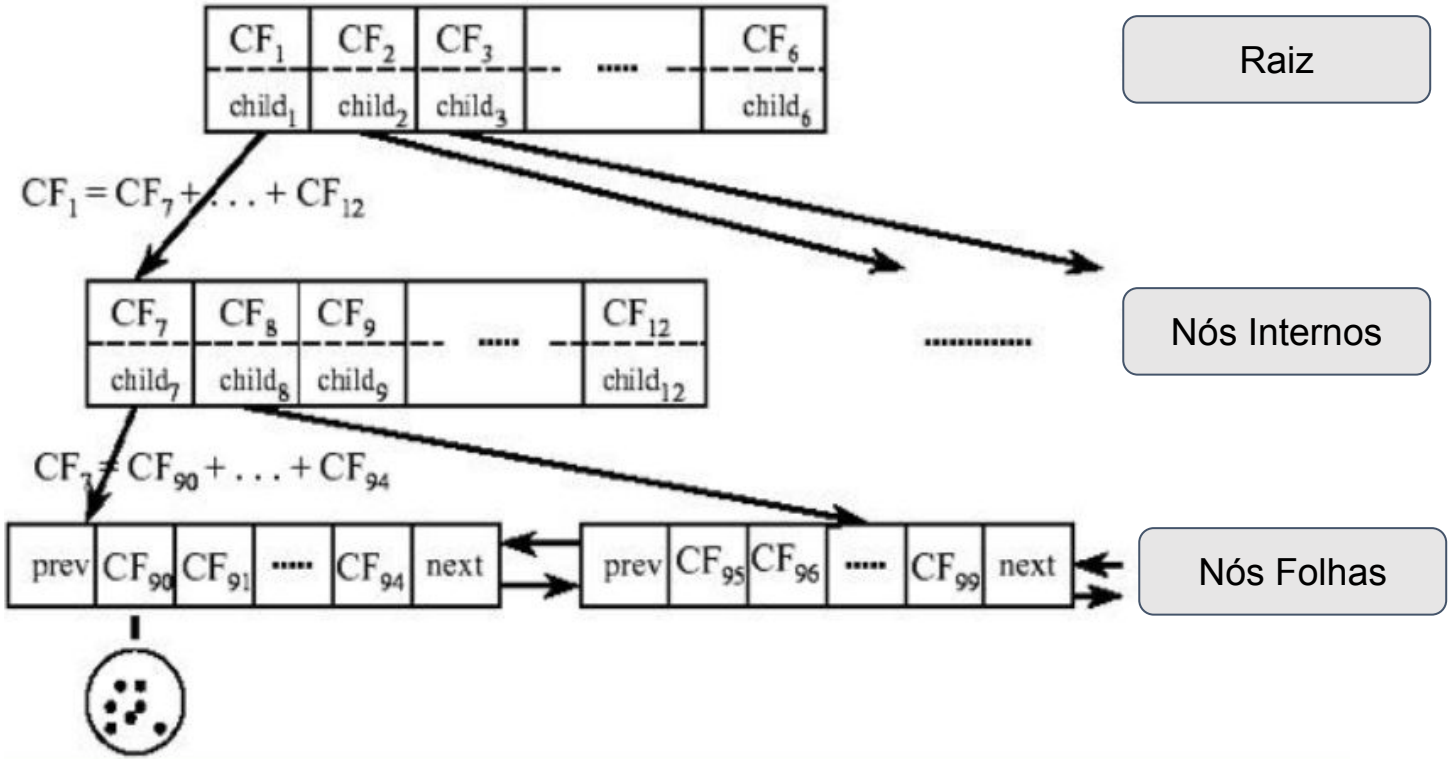
- *N* = Quantidade de objetos
- LS = Soma linear dos vetores de cada objeto
- SS = Soma ao quadrado dos vetores de cada objeto

Permite calcular centroide, raio e diâmetro de um micro-cluster sem acessar os dados originais

$$LS = \sum_{i=1}^{N} X_i \qquad SS = \sum_{i=1}^{N} (X_i)^2$$

BIRCH

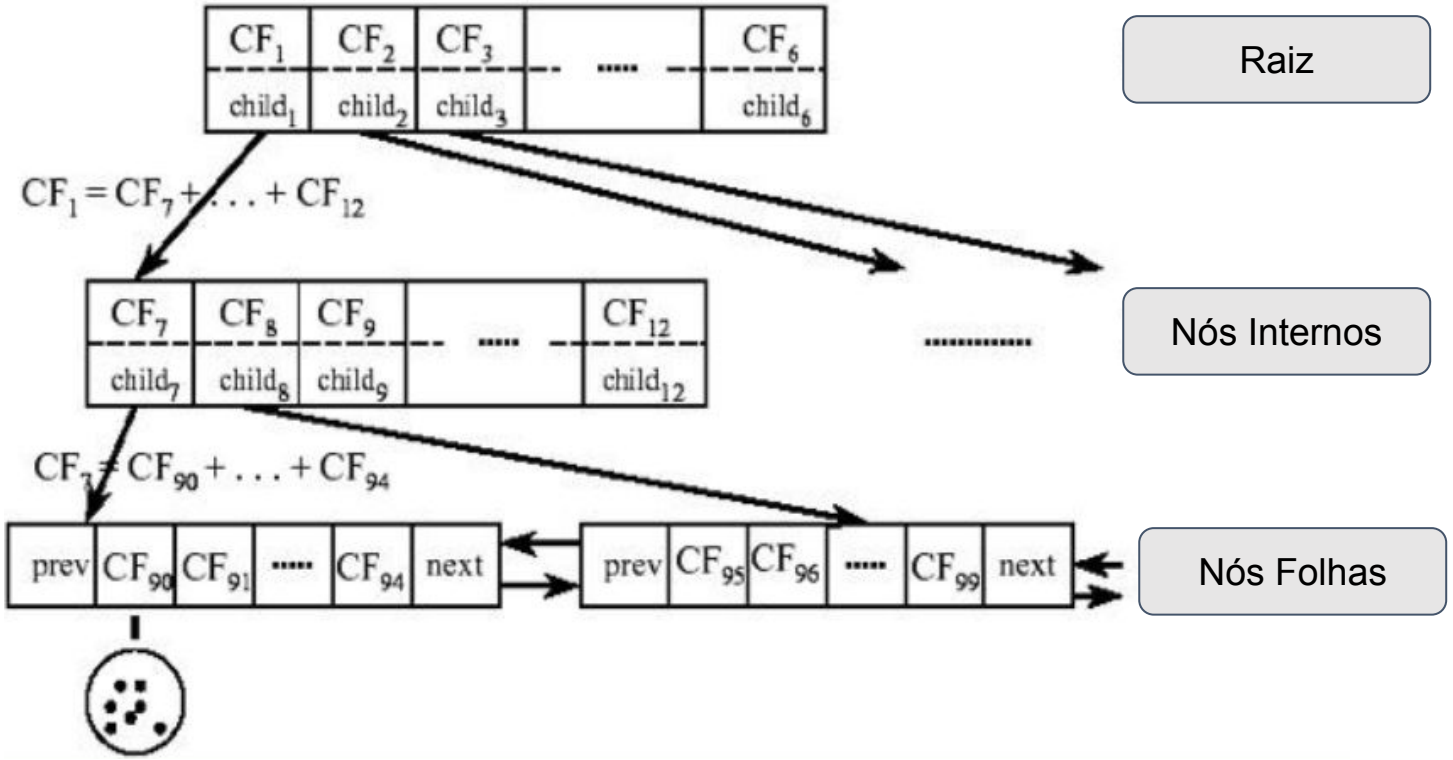
- *CF-Tree*



- Percorrer a base de dados para inserção incremental de objetos
- Para cada objeto
  - Encontrar a CF mais próxima
  - Adicionar na CF e atualizar estatísticas
  - Se diâmetro da CF ultrapassar um limiar, então dividir a CF
- Agrupar as CF dos nós folhas em *k* clusters

BIRCH

- *CF-Tree*



- **Parâmetros:**
  - *B (Branching Factor)*: número máximo de ramificação. Número de CF por nó da árvore.
  - Tamanho Máximo do Diâmetro (ou Raio)
  - Número de clusters (k) desejado

Considerações sobre o BIRCH

- Parâmetros difíceis de calibrar
  - O que é um bom Branching Factor?
  - Como definir o raio ou diâmetro mínimo?

- A ordem de processamento dos dados altera o resultado da *CF-Tree*

- Formato dos *clusters*
  - Tendência para clusters globulares

Grandes Bases de Dados

- Acelerar o cálculo da medida de distância
- Amostragem de dados para reduzir uso de memória e custo computacional
- Representações condensadas do conjunto de dados

Bibliografia

Rezende, S. O. (2003). *Sistemas inteligentes: fundamentos e aplicações*. Editora Manole Ltda.

Tan, P.N.; Steinbach, M.; Karpatne, A.; Kumar, V. (2016). *Introduction to Data Mining (2nd Edition)*. Pearson.

Sculley, David. "Web-scale k-means clustering." In Proceedings of the 19th international conference on World wide web, pp. 1177-1178. 2010.

Zhang, Tian, Raghu Ramakrishnan, and Miron Livny. "BIRCH: an efficient data clustering method for very large databases." ACM sigmod record 25, no. 2 (1996): 103-114.