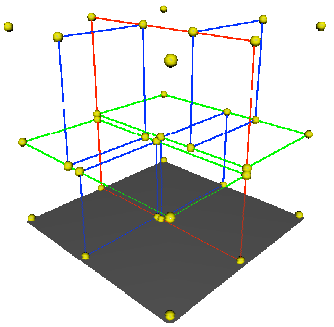
K-MEANS COM ÁRVORES KD

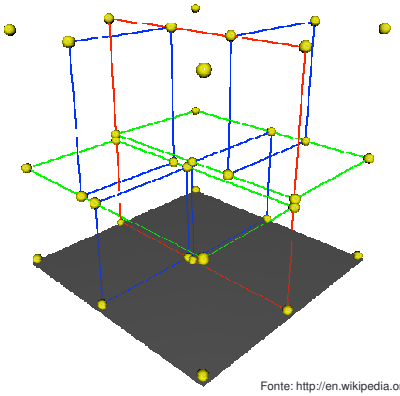
Rodrigo Coelho Barros



Roteiro

- Árvores KD
  - O que são?
  - Algoritmo de Bentley (1975)
  - Métodos de divisão
- K-Means com árvores KD
  - Algoritmo de filtragem (Kanungo et al., 2002)
    - Exemplo
    - Notas sobre desempenho
- Referências

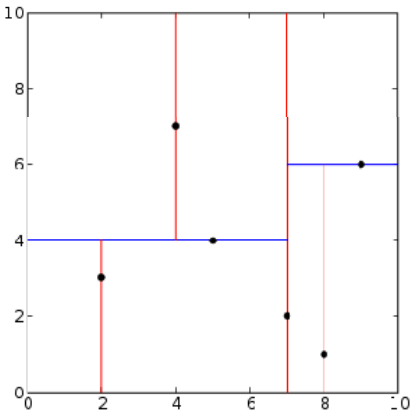
O que é uma árvore KD?

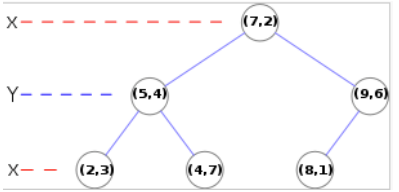


Fonte: http://en.wikipedia.org/wiki/Kd-tree

Exemplo do algoritmo de Bentley (1975)

```
pointList = [(2,3), (5,4), (9,6), (4,7), (8,1), (7,2)]
tree = kdtree(pointList)
```





Fonte: http://en.wikipedia.org/wiki/Kd-tree

Métodos de divisão

- Tradicional ("*optimal kd-tree*") (Friedman et al., 1977)
  - Escolha do eixo mais longo (max-min)
  - Escolha da coordenada mediana
- Ponto médio (Maneewongvatana and Mount,1999)
  - Escolha do eixo com maior extensão (independente dos dados)
  - Escolha do ponto central (extensão/2)
- Ponto médio deslizante (Mount and Arya, 1997)
  - Opera como o método de ponto médio
  - Porém, para divisões vazias, desliza o ponto médio até incluir um elemento

K-Means com árvores KD

- **Idéia geral**
  - Construir árvore KD
  - Árvore é fixa, só precisa ser construída uma única vez
  - Cada nó contém: ponteiros para os objetos, soma destes e quantidade
  - Filtrar protótipos pela árvore até que:
    - Nó não-terminal possua um único protótipo
    - Nó folha é atingido

    } **Passo de atribuição**
- **Vantagens**
  - Árvore subdivide os dados em "vizinhos mais próximos"
  - Vizinhos próximos possivelmente serão associados a um mesmo protótipo
  - Análise geométrica para **podar** protótipos

K-Means com árvores KD

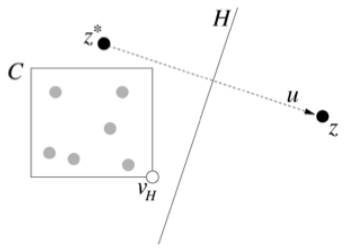
- Vários descobriram de forma independente
  - Alsabti et al. (1998)
  - Pelleg and Moore (1999)
  - **Kanungo et al. (1999, 2002)**
- Diferença
  - Método de divisão da árvore KD
  - Método de **poda** dos protótipos

IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE, VOL. 24, NO. 7, JULY 2002 881

An Efficient *k*-Means Clustering Algorithm: Analysis and Implementation

Tapas Kanungo, *Senior Member, IEEE*, David M. Mount, *Member, IEEE*, Nathan S. Netanyahu, *Member, IEEE*, Christine D. Piatko, Ruth Silverman, and Angela Y. Wu, *Senior Member, IEEE*

- Kanungo et al. (2002)
  - Primeiro constrói árvore KD
  - Cada iteração, roda o algoritmo de filtragem

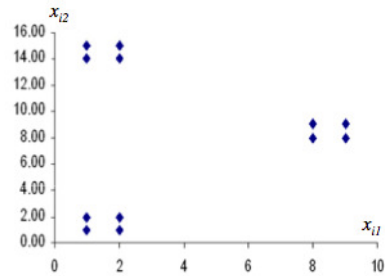



```
Filter(kdNode u, CandidateSet Z) {
    C ← u.cell;
    if (u is a leaf) {
        z* ← the closest point in Z to u.point;
        z*.wgtCent ← z*.wgtCent + u.point;
        z*.count ← z*.count + 1;
    }
    else {
        z* ← the closest point in Z to C's midpoint;
        for each (z ∈ Z \ {z*})
            if (z.isFarther(z*, C)) Z ← Z \ {z};
        if (|Z| = 1) {
            z*.wgtCent ← z*.wgtCent + u.wgtCent;
            z*.count ← z*.count + u.count;
        }
        else {
            Filter(u.left, Z);
            Filter(u.right, Z);
        }
    }
}
```

Exemplo 1º Passo – Construir Árvore-KD

Objeto $\mathbf{x}_i$	x_{i1}	x_{i2}
1	1	2
2	2	1
3	1	1
4	2	2
5	**8**	**9**
6	**9**	**8**
7	**9**	**9**
8	**8**	**8**
9	1	15
10	2	15
11	1	14
12	2	14

Prof. Eduardo R. Hruschka

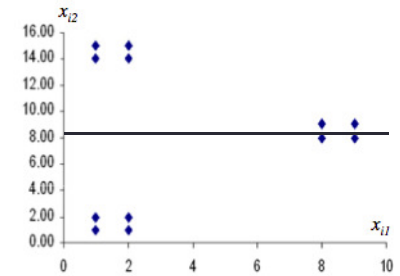


Exemplo 1º Passo – Construir Árvore-KD

Objeto $\mathbf{x}_i$	x_{i1}	x_{i2}
1	1	2
2	2	1
3	1	1
4	2	2
5	**8**	**9**
6	**9**	**8**
7	**9**	**9**
8	**8**	**8**
9	1	15
10	2	15
11	1	14
12	2	14

Prof. Eduardo R. Hruschka

Y<=8



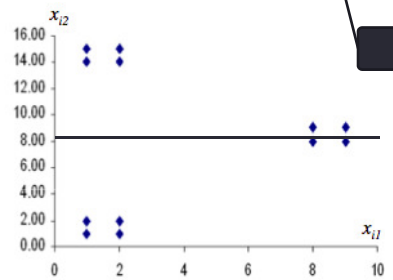
Exemplo 1º Passo – Construir Árvore-KD

Objeto $\mathbf{x}_i$	x_{i1}	x_{i2}
1	1	2
2	2	1
3	1	1
4	2	2
5	**8**	**9**
6	**9**	**8**
7	**9**	**9**
8	**8**	**8**
9	1	15
10	2	15
11	1	14
12	2	14

Prof. Eduardo R. Hruschka

Y<=8

1,2,3,4,6, 8



Exemplo 1º Passo – Construir Árvore-KD

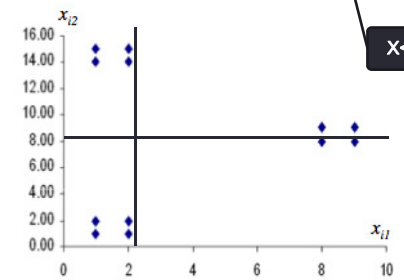
Objeto $\mathbf{x}_i$	x_{i1}	x_{i2}
1	1	2
2	2	1
3	1	1
4	2	2
5	**8**	**9**
6	**9**	**8**
7	**9**	**9**
8	**8**	**8**
9	1	15
10	2	15
11	1	14
12	2	14

Prof. Eduardo R. Hruschka

Y<=8

X<=2

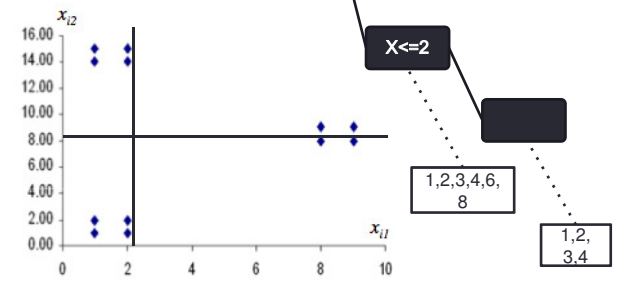
1,2,3,4,6, 8



Exemplo 1º Passo – Construir Árvore-KD

Objeto $\mathbf{x}_i$	x_{i1}	x_{i2}
1	1	2
2	2	1
3	1	1
4	2	2
5	**8**	**9**
6	**9**	**8**
7	**9**	**9**
8	**8**	**8**
9	1	15
10	2	15
11	1	14
12	2	14

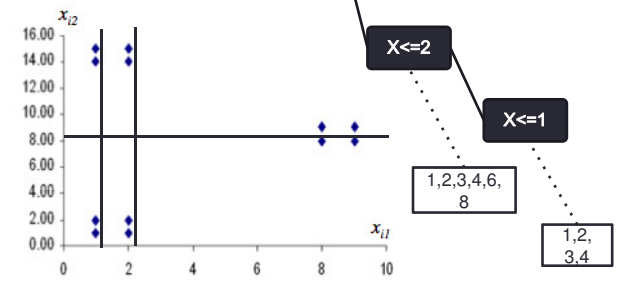
Prof. Eduardo R. Hruschka



Exemplo 1º Passo – Construir Árvore-KD

Objeto $\mathbf{x}_i$	x_{i1}	x_{i2}
1	1	2
2	2	1
3	1	1
4	2	2
5	**8**	**9**
6	**9**	**8**
7	**9**	**9**
8	**8**	**8**
9	1	15
10	2	15
11	1	14
12	2	14

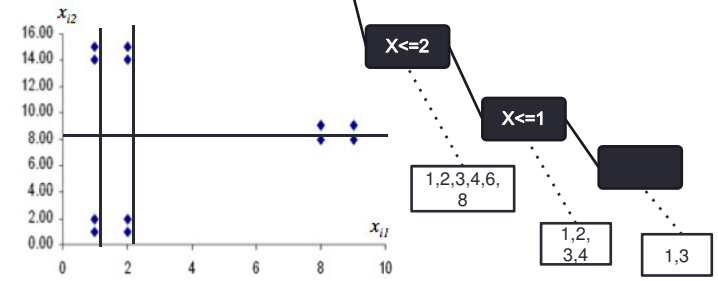
Prof. Eduardo R. Hruschka



Exemplo 1º Passo – Construir Árvore-KD

Objeto $\mathbf{x}_i$	x_{i1}	x_{i2}
1	1	2
2	2	1
3	1	1
4	2	2
5	**8**	**9**
6	**9**	**8**
7	**9**	**9**
8	**8**	**8**
9	1	15
10	2	15
11	1	14
12	2	14

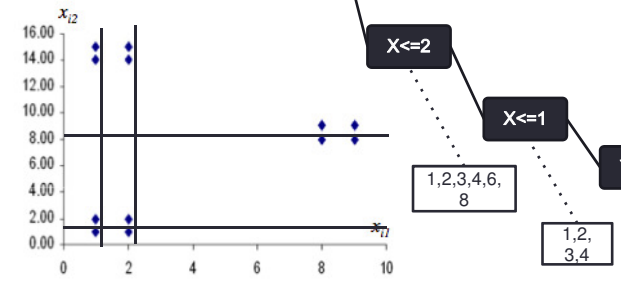
Prof. Eduardo R. Hruschka



Exemplo 1º Passo – Construir Árvore-KD

Objeto $\mathbf{x}_i$	x_{i1}	x_{i2}
1	1	2
2	2	1
3	1	1
4	2	2
5	**8**	**9**
6	**9**	**8**
7	**9**	**9**
8	**8**	**8**
9	1	15
10	2	15
11	1	14
12	2	14

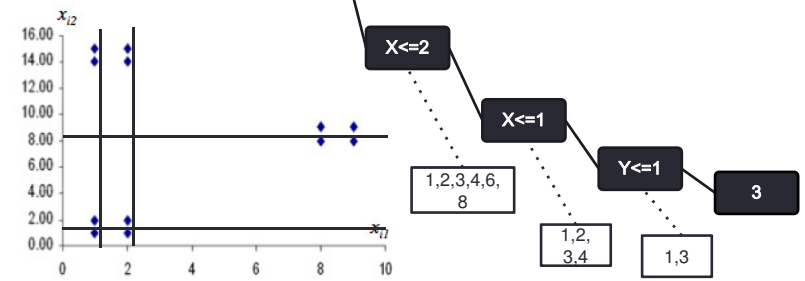
Prof. Eduardo R. Hruschka



Exemplo 1º Passo – Construir Árvore-KD

Objeto $\mathbf{x}_i$	x_{i1}	x_{i2}
1	1	2
2	2	1
3	1	1
4	2	2
5	**8**	**9**
6	**9**	**8**
7	**9**	**9**
8	**8**	**8**
9	1	15
10	2	15
11	1	14
12	2	14

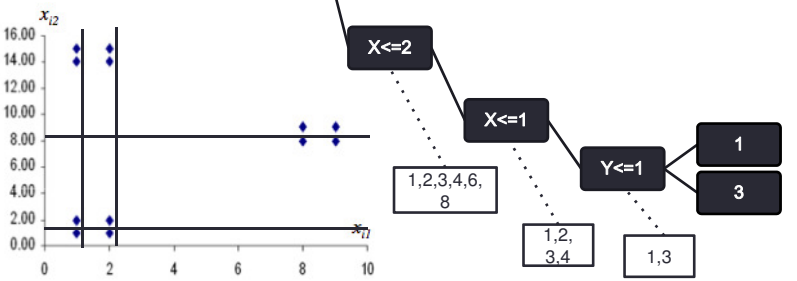
Prof. Eduardo R. Hruschka



Exemplo 1º Passo – Construir Árvore-KD

Objeto $\mathbf{x}_i$	x_{i1}	x_{i2}
1	1	2
2	2	1
3	1	1
4	2	2
5	**8**	**9**
6	**9**	**8**
7	**9**	**9**
8	**8**	**8**
9	1	15
10	2	15
11	1	14
12	2	14

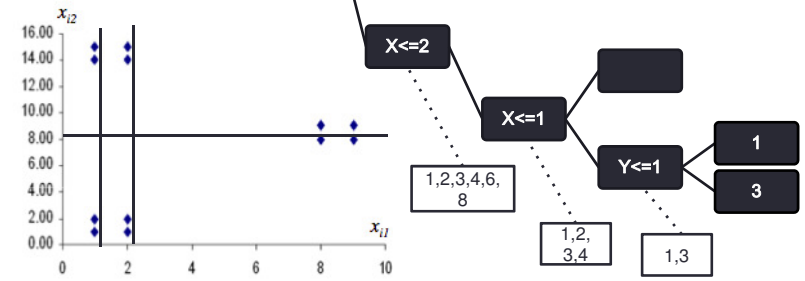
Prof. Eduardo R. Hruschka



Exemplo 1º Passo – Construir Árvore-KD

Objeto $\mathbf{x}_i$	x_{i1}	x_{i2}
1	1	2
2	2	1
3	1	1
4	2	2
5	**8**	**9**
6	**9**	**8**
7	**9**	**9**
8	**8**	**8**
9	1	15
10	2	15
11	1	14
12	2	14

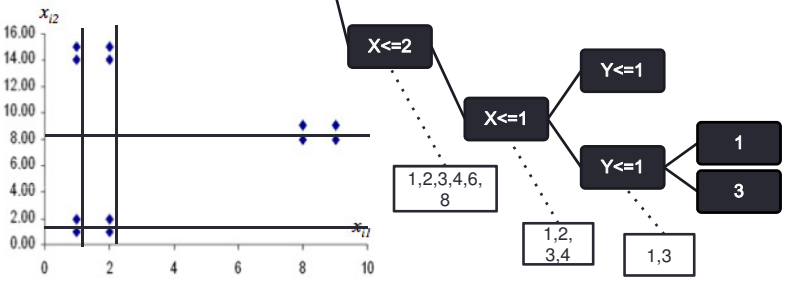
Prof. Eduardo R. Hruschka



Exemplo 1º Passo – Construir Árvore-KD

Objeto $\mathbf{x}_i$	x_{i1}	x_{i2}
1	1	2
2	2	1
3	1	1
4	2	2
5	**8**	**9**
6	**9**	**8**
7	**9**	**9**
8	**8**	**8**
9	1	15
10	2	15
11	1	14
12	2	14

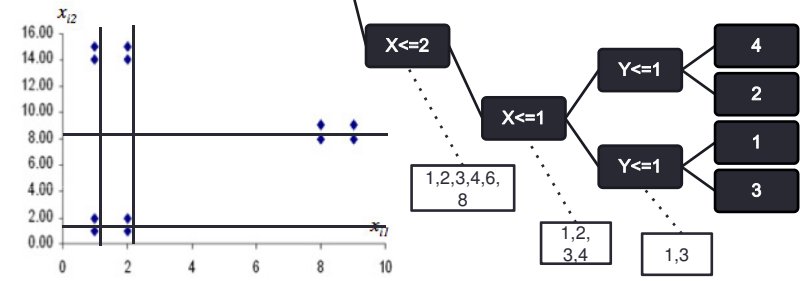
Prof. Eduardo R. Hruschka



Exemplo 1º Passo – Construir Árvore-KD

Objeto x_i	x_{i1}	x_{i2}
1	1	2
2	2	1
3	1	1
4	2	2
5	8	9
6	9	8
7	9	9
8	8	8
9	1	15
10	2	15
11	1	14
12	2	14

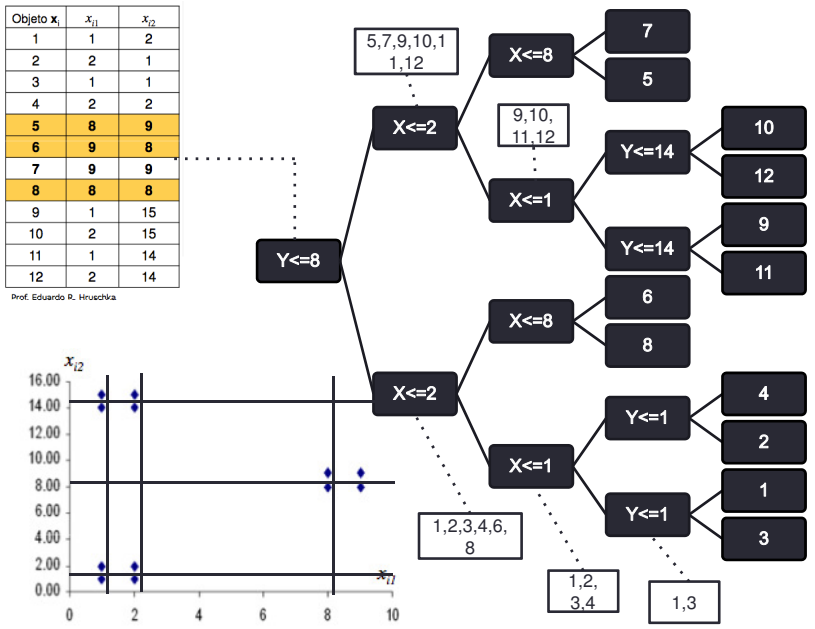
Prof. Eduardo R. Hruschka



Exemplo 1º Passo – Construir Árvore-KD

Objeto x_i	x_{i1}	x_{i2}
1	1	2
2	2	1
3	1	1
4	2	2
5	8	9
6	9	8
7	9	9
8	8	8
9	1	15
10	2	15
11	1	14
12	2	14

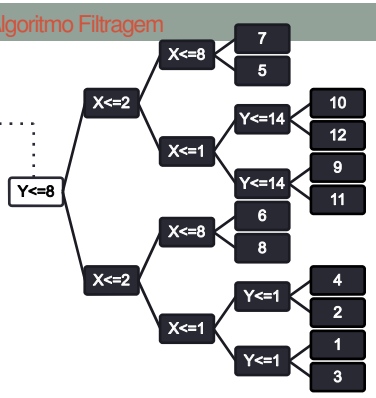
Prof. Eduardo R. Hruschka



Exemplo 2º Passo – Algoritmo Filtragem

Objeto x_i	x_{i1}	x_{i2}
1	1	2
2	2	1
3	1	1
4	2	2
5	8	9
6	9	8
7	9	9
8	8	8
9	1	15
10	2	15
11	1	14
12	2	14

Prof. Eduardo R. Hruschka

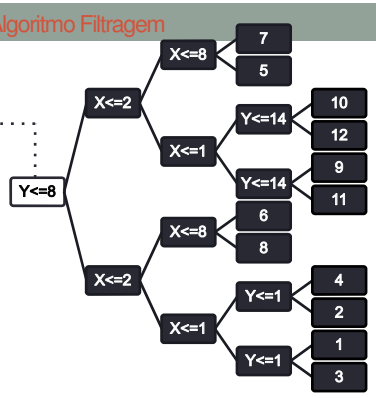



```
Filter(kdNode u, CandidateSet Z) {
    C ← u.cell;
    if (u is a leaf) {
        z* ← the closest point in Z to u.point;
        z*.wgtCent ← z*.wgtCent + u.point;
        z*.count ← z*.count + 1;
    }
    else {
        z* ← the closest point in Z to C's midpoint;
        for each (z ∈ Z \ {z*})
            if (z.isFarther(z*,C)) Z ← Z \ {z};
        if (|Z| = 1) {
            z*.wgtCent ← z*.wgtCent + u.wgtCent;
            z*.count ← z*.count + u.count;
        }
        else {
            Filter(u.left, Z);
            Filter(u.right, Z);
        }
    }
}
```

Exemplo 2º Passo – Algoritmo Filtragem

Objeto x_i	x_{i1}	x_{i2}
1	1	2
2	2	1
3	1	1
4	2	2
5	8	9
6	9	8
7	9	9
8	8	8
9	1	15
10	2	15
11	1	14
12	2	14

Prof. Eduardo R. Hruschka



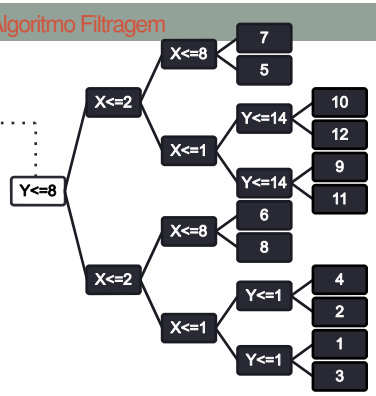

```
Filter(kdNode u, CandidateSet Z) {
    C ← u.cell;
    if (u is a leaf) {
        z* ← the closest point in Z to u.point;
        z*.wgtCent ← z*.wgtCent + u.point;
        z*.count ← z*.count + 1;
    }
    else {
        z* ← the closest point in Z to C's midpoint;
        for each (z ∈ Z \ {z*})
            if (z.isFarther(z*,C)) Z ← Z \ {z};
        if (|Z| = 1) {
            z*.wgtCent ← z*.wgtCent + u.wgtCent;
            z*.count ← z*.count + u.count;
        }
        else {
            Filter(u.left, Z);
            Filter(u.right, Z);
        }
    }
}
```

Z1 = [8 9] Z2 = [9 8] Z3 = [8 8]

Exemplo 2º Passo – Algoritmo Filtragem

Objeto x_i	x_{i1}	x_{i2}
1	1	2
2	2	1
3	1	1
4	2	2
5	8	9
6	9	8
7	9	9
8	8	8
9	1	15
10	2	15
11	1	14
12	2	14

Prof. Eduardo R. Hruschka




```
Filter(kdNode u, CandidateSet Z) {
    C ← u.cell;
    if (u is a leaf) {
        z* ← the closest point in Z to u.point;
        z*.wgtCent ← z*.wgtCent + u.point;
        z*.count ← z*.count + 1;
    }
    else {
        z* ← the closest point in Z to C's midpoint;
        for each (z ∈ Z \ {z*})
            if (z.isFarther(z*,C)) Z ← Z \ {z};
        if (|Z| = 1) {
            z*.wgtCent ← z*.wgtCent + u.wgtCent;
            z*.count ← z*.count + u.count;
        }
        else {
            Filter(u.left, Z);
            Filter(u.right, Z);
        }
    }
}
```

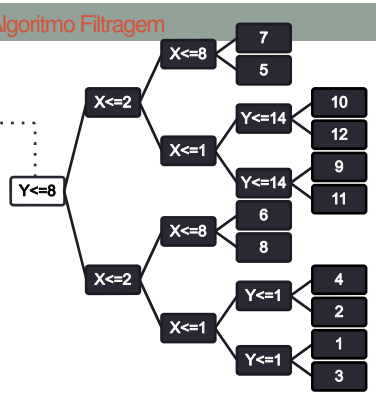
Z1 = [8 9] Z2 = [9 8] Z3 = [8 8]

Midpoint = [3,83 8,16]
Z* ← Z3

Exemplo 2º Passo – Algoritmo Filtragem

Objeto x_i	x_{i1}	x_{i2}
1	1	2
2	2	1
3	1	1
4	2	2
5	8	9
6	9	8
7	9	9
8	8	8
9	1	15
10	2	15
11	1	14
12	2	14

Prof. Eduardo R. Hruschka




```
Filter(kdNode u, CandidateSet Z) {
    C ← u.cell;
    if (u is a leaf) {
        z* ← the closest point in Z to u.point;
        z*.wgtCent ← z*.wgtCent + u.point;
        z*.count ← z*.count + 1;
    }
    else {
        z* ← the closest point in Z to C's midpoint;
        for each (z ∈ Z \ {z*})
            if (z.isFarther(z*,C)) Z ← Z \ {z};
        if (|Z| = 1) {
            z*.wgtCent ← z*.wgtCent + u.wgtCent;
            z*.count ← z*.count + u.count;
        }
        else {
            Filter(u.left, Z);
            Filter(u.right, Z);
        }
    }
}
```

Z1 = [8 9] Z2 = [9 8] Z3 = [8 8]

Midpoint = [3,83 8,16]
Z* ← Z3

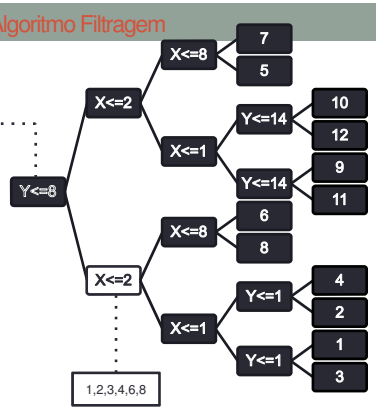
Para Z1:
u(Z1) = Z1 – Z* = [0 1]
v(H) = [9 15]
d(v(H),Z1) = sqrt(37)
d(v(H),Z*) = sqrt(50)

Para Z2:
u(Z2) = Z2 – Z* = [1 0]
v(H) = [9 15]
d(v(H),Z2) = sqrt(49)
d(v(H),Z*) = sqrt(50)

Exemplo 2º Passo – Algoritmo Filtragem

Objeto x_i	x_{i1}	x_{i2}
1	1	2
2	2	1
3	1	1
4	2	2
5	8	9
6	9	8
7	9	9
8	8	8
9	1	15
10	2	15
11	1	14
12	2	14

Prof. Eduardo R. Hruschka



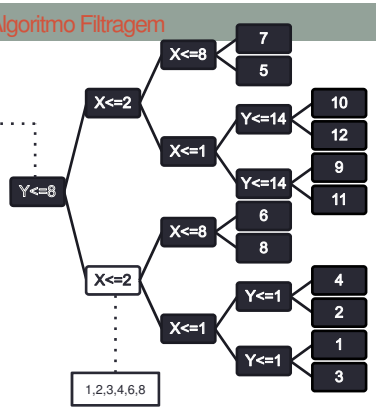

```
Filter(kdNode u, CandidateSet Z) {
    C ← u.cell;
    if (u is a leaf) {
        z* ← the closest point in Z to u.point;
        z*.wgtCent ← z*.wgtCent + u.point;
        z*.count ← z*.count + 1;
    }
    else {
        z* ← the closest point in Z to C's midpoint;
        for each (z ∈ Z \ {z*})
            if (z.isFarther(z*,C)) Z ← Z \ {z};
        if (|Z| = 1) {
            z*.wgtCent ← z*.wgtCent + u.wgtCent;
            z*.count ← z*.count + u.count;
        }
        else {
            Filter(u.left, Z);
            Filter(u.right, Z);
        }
    }
}
```

Z1 = [8 9] Z2 = [9 8] Z3 = [8 8]

Exemplo 2º Passo – Algoritmo Filtragem

Objeto x_i	x_{i1}	x_{i2}
1	1	2
2	2	1
3	1	1
4	2	2
5	8	9
6	9	8
7	9	9
8	8	8
9	1	15
10	2	15
11	1	14
12	2	14

Prof. Eduardo R. Hruschka




```
Filter(kdNode u, CandidateSet Z) {
    C ← u.cell;
    if (u is a leaf) {
        z* ← the closest point in Z to u.point;
        z*.wgtCent ← z*.wgtCent + u.point;
        z*.count ← z*.count + 1;
    }
    else {
        z* ← the closest point in Z to C's midpoint;
        for each (z ∈ Z \ {z*})
            if (z.isFarther(z*,C)) Z ← Z \ {z};
        if (|Z| = 1) {
            z*.wgtCent ← z*.wgtCent + u.wgtCent;
            z*.count ← z*.count + u.count;
        }
        else {
            Filter(u.left, Z);
            Filter(u.right, Z);
        }
    }
}
```

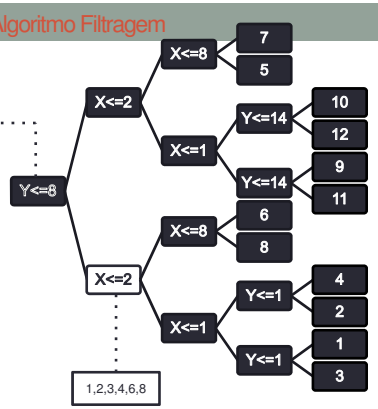
Z1 = [8 9] Z2 = [9 8] Z3 = [8 8]

Midpoint = [3,83 3,66]
Z* ← Z3

Exemplo 2º Passo – Algoritmo Filtragem

Objeto x_i	x_{i1}	x_{i2}
1	1	2
2	2	1
3	1	1
4	2	2
5	8	9
6	9	8
7	9	9
8	8	8
9	1	15
10	2	15
11	1	14
12	2	14

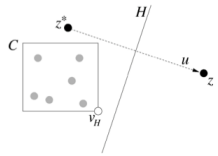
Prof. Eduardo R. Hruschka




```
Filter(kdNode u, CandidateSet Z) {
    C ← u.cell;
    if (u is a leaf) {
        z* ← the closest point in Z to u.point;
        z*.wgtCent ← z*.wgtCent + u.point;
        z*.count ← z*.count + 1;
    }
    else {
        z* ← the closest point in Z to C's midpoint;
        for each (z ∈ Z \ {z*})
            if (z.isFarther(z*,C)) Z ← Z \ {z};
        if (|Z| = 1) {
            z*.wgtCent ← z*.wgtCent + u.wgtCent;
            z*.count ← z*.count + u.count;
        }
        else {
            Filter(u.left, Z);
            Filter(u.right, Z);
        }
    }
}
```

Z1 = [8 9] Z2 = [9 8] Z3 = [8 8]

Midpoint = [3,83 3,66]
Z* ← Z3



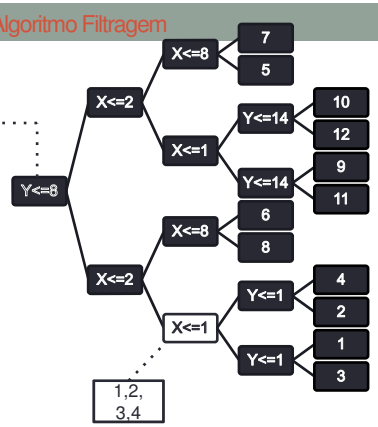
Para Z1:
u(Z1) = Z1 – Z* = [0 1]
v(H) = [9 8]
d(v(H),Z1) = sqrt(2)
d(v(H),Z*) = sqrt(1)
Podemos podar Z1!

Para Z2:
u(Z2) = Z2 – Z* = [1 0]
v(H) = [9 8]
d(v(H),Z2) = sqrt(0)
d(v(H),Z*) = sqrt(1)

Exemplo 2º Passo – Algoritmo Filtragem

Objeto x_i	x_{i1}	x_{i2}
1	1	2
2	2	1
3	1	1
4	2	2
5	8	9
6	9	8
7	9	9
8	8	8
9	1	15
10	2	15
11	1	14
12	2	14

Prof. Eduardo R. Hruschka



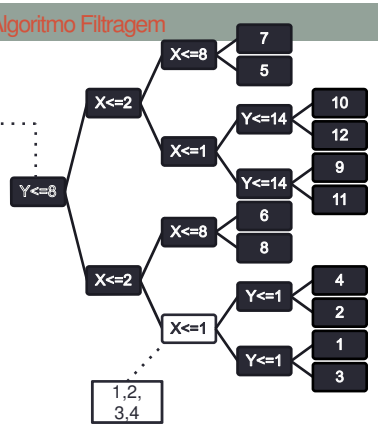

```
Filter(kdNode u, CandidateSet Z) {
    C ← u.cell;
    if (u is a leaf) {
        z* ← the closest point in Z to u.point;
        z*.wgtCent ← z*.wgtCent + u.point;
        z*.count ← z*.count + 1;
    }
    else {
        z* ← the closest point in Z to C's midpoint;
        for each (z ∈ Z \ {z*})
            if (z.isFarther(z*,C)) Z ← Z \ {z};
        if (|Z| = 1) {
            z*.wgtCent ← z*.wgtCent + u.wgtCent;
            z*.count ← z*.count + u.count;
        }
        else {
            Filter(u.left, Z);
            Filter(u.right, Z);
        }
    }
}
```

Z2 = [9 8] Z3 = [8 8]

Exemplo 2º Passo – Algoritmo Filtragem

Objeto x_i	x_{i1}	x_{i2}
1	1	2
2	2	1
3	1	1
4	2	2
5	8	9
6	9	8
7	9	9
8	8	8
9	1	15
10	2	15
11	1	14
12	2	14

Prof. Eduardo R. Hruschka




```
Filter(kdNode u, CandidateSet Z) {
    C ← u.cell;
    if (u is a leaf) {
        z* ← the closest point in Z to u.point;
        z*.wgtCent ← z*.wgtCent + u.point;
        z*.count ← z*.count + 1;
    }
    else {
        z* ← the closest point in Z to C's midpoint;
        for each (z ∈ Z \ {z*})
            if (z.isFarther(z*,C)) Z ← Z \ {z};
        if (|Z| = 1) {
            z*.wgtCent ← z*.wgtCent + u.wgtCent;
            z*.count ← z*.count + u.count;
        }
        else {
            Filter(u.left, Z);
            Filter(u.right, Z);
        }
    }
}
```

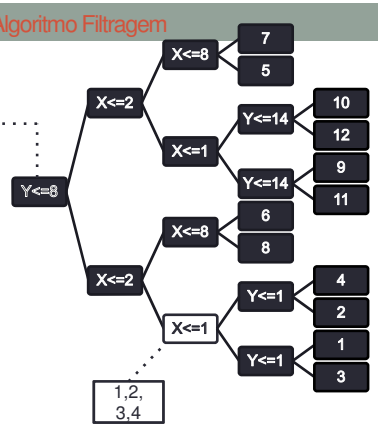
Z2 = [9 8] Z3 = [8 8]

Midpoint = [1,5 1,5]
Z* ← Z3

Exemplo 2º Passo – Algoritmo Filtragem

Objeto x_i	x_{i1}	x_{i2}
1	1	2
2	2	1
3	1	1
4	2	2
5	8	9
6	9	8
7	9	9
8	8	8
9	1	15
10	2	15
11	1	14
12	2	14

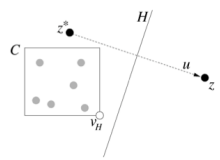
Prof. Eduardo R. Hruschka




```
Filter(kdNode u, CandidateSet Z) {
    C ← u.cell;
    if (u is a leaf) {
        z* ← the closest point in Z to u.point;
        z*.wgtCent ← z*.wgtCent + u.point;
        z*.count ← z*.count + 1;
    }
    else {
        z* ← the closest point in Z to C's midpoint;
        for each (z ∈ Z \ {z*})
            if (z.isFarther(z*,C)) Z ← Z \ {z};
        if (|Z| = 1) {
            z*.wgtCent ← z*.wgtCent + u.wgtCent;
            z*.count ← z*.count + u.count;
        }
        else {
            Filter(u.left, Z);
            Filter(u.right, Z);
        }
    }
}
```

Z2 = [9 8] Z3 = [8 8]

Midpoint = [1,5 1,5]
Z* ← Z3

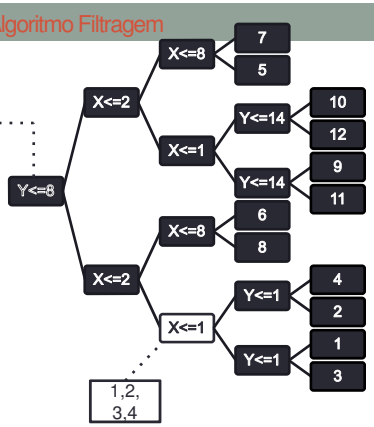


Para Z2:
u(Z2) = Z2 – Z* = [1 0]
v(H) = [2 2]
d(v(H),Z2) = sqrt(85)
d(v(H),Z*) = sqrt(72)
Podemos podar Z2!

Exemplo 2º Passo – Algoritmo Filtragem

Objeto x_i	x_{i1}	x_{i2}
1	1	2
2	2	1
3	1	1
4	2	2
5	8	9
6	9	8
7	9	9
8	8	8
9	1	15
10	2	15
11	1	14
12	2	14

Prof. Eduardo R. Hruschka




```
Filter(kdNode u, CandidateSet Z) {
    C ← u.cell;
    if (u is a leaf) {
        z* ← the closest point in Z to u.point;
        z*.wgtCent ← z*.wgtCent + u.point;
        z*.count ← z*.count + 1;
    }
    else {
        z* ← the closest point in Z to C's midpoint;
        for each (z ∈ Z \ {z*})
            if (z.isFarther(z*,C)) Z ← Z \ {z};
        if (|Z| = 1) {
            z*.wgtCent ← z*.wgtCent + u.wgtCent;
            z*.count ← z*.count + u.count;
        }
        else {
            Filter(u.left, Z);
            Filter(u.right, Z);
        }
    }
}
```

Z2 = [9 8] Z3 = [8 8]

Midpoint = [1,5 1,5]
Z* ← Z3

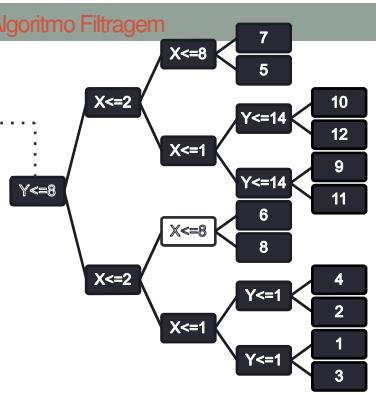
|Z| = 1:
Z3.wgtCent ← Z3.wgtCent + (x1+x2+x3+x4)
Z3.count ← Z3.count + 4

Volta da recursão!

Exemplo 2º Passo – Algoritmo Filtragem

Objeto x_i	x_{i1}	x_{i2}
1	1	2
2	2	1
3	1	1
4	2	2
5	8	9
6	9	8
7	9	9
8	8	8
9	1	15
10	2	15
11	1	14
12	2	14

Prof. Eduardo R. Hruschka



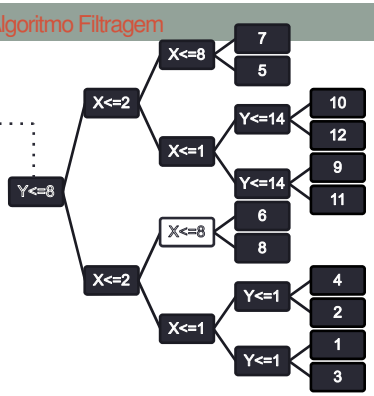

```
Filter(kdNode u, CandidateSet Z) {
    C ← u.cell;
    if (u is a leaf) {
        z* ← the closest point in Z to u.point;
        z*.wgtCent ← z*.wgtCent + u.point;
        z*.count ← z*.count + 1;
    }
    else {
        z* ← the closest point in Z to C's midpoint;
        for each (z ∈ Z \ {z*})
            if (z.isFarther(z*,C)) Z ← Z \ {z};
        if (|Z| = 1) {
            z*.wgtCent ← z*.wgtCent + u.wgtCent;
            z*.count ← z*.count + u.count;
        }
        else {
            Filter(u.left, Z);
            Filter(u.right, Z);
        }
    }
}
```

Z2 = [9 8] Z3 = [8 8]

Exemplo 2º Passo – Algoritmo Filtragem

Objeto x_i	x_{i1}	x_{i2}
1	1	2
2	2	1
3	1	1
4	2	2
5	8	9
6	9	8
7	9	9
8	8	8
9	1	15
10	2	15
11	1	14
12	2	14

Prof. Eduardo R. Hruschka




```
Filter(kdNode u, CandidateSet Z) {
    C ← u.cell;
    if (u is a leaf) {
        z* ← the closest point in Z to u.point;
        z*.wgtCent ← z*.wgtCent + u.point;
        z*.count ← z*.count + 1;
    }
    else {
        z* ← the closest point in Z to C's midpoint;
        for each (z ∈ Z \ {z*})
            if (z.isFarther(z*,C)) Z ← Z \ {z};
        if (|Z| = 1) {
            z*.wgtCent ← z*.wgtCent + u.wgtCent;
            z*.count ← z*.count + u.count;
        }
        else {
            Filter(u.left, Z);
            Filter(u.right, Z);
        }
    }
}
```

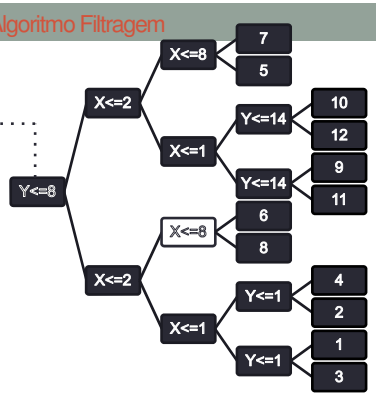
Z2 = [9 8] Z3 = [8 8]

Midpoint = [8,5 8]
Z* ← Z2 (mas poderia ser Z3)

Exemplo 2º Passo – Algoritmo Filtragem

Objeto x_i	x_{i1}	x_{i2}
1	1	2
2	2	1
3	1	1
4	2	2
5	8	9
6	9	8
7	9	9
8	8	8
9	1	15
10	2	15
11	1	14
12	2	14

Prof. Eduardo R. Hruschka




```
Filter(kdNode u, CandidateSet Z) {
    C ← u.cell;
    if (u is a leaf) {
        z* ← the closest point in Z to u.point;
        z*.wgtCent ← z*.wgtCent + u.point;
        z*.count ← z*.count + 1;
    }
    else {
        z* ← the closest point in Z to C's midpoint;
        for each (z ∈ Z \ {z*})
            if (z.isFarther(z*,C)) Z ← Z \ {z};
        if (|Z| = 1) {
            z*.wgtCent ← z*.wgtCent + u.wgtCent;
            z*.count ← z*.count + u.count;
        }
        else {
            Filter(u.left, Z);
            Filter(u.right, Z);
        }
    }
}
```

Z2 = [9 8] Z3 = [8 8]

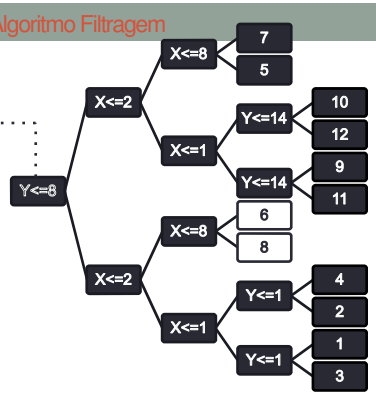
Midpoint = [8,5 8]
Z* ← Z2 (mas poderia ser Z3)

Para Z3:
u(Z3) = Z3 – Z* = [-1 0]
v(H) = [8 8]
d(v(H),Z3) = sqrt(0)
d(v(H),Z*) = sqrt(1)

Exemplo 2º Passo – Algoritmo Filtragem

Objeto x_i	x_{i1}	x_{i2}
1	1	2
2	2	1
3	1	1
4	2	2
5	8	9
6	9	8
7	9	9
8	8	8
9	1	15
10	2	15
11	1	14
12	2	14

Prof. Eduardo R. Hruschka




```
Filter(kdNode u, CandidateSet Z) {
    C ← u.cell;
    if (u is a leaf) {
        z* ← the closest point in Z to u.point;
        z*.wgtCent ← z*.wgtCent + u.point;
        z*.count ← z*.count + 1;
    }
    else {
        z* ← the closest point in Z to C's midpoint;
        for each (z ∈ Z \ {z*})
            if (z.isFarther(z*,C)) Z ← Z \ {z};
        if (|Z| = 1) {
            z*.wgtCent ← z*.wgtCent + u.wgtCent;
            z*.count ← z*.count + u.count;
        }
        else {
            Filter(u.left, Z);
            Filter(u.right, Z);
        }
    }
}
```

Z2 = [9 8] Z3 = [8 8]

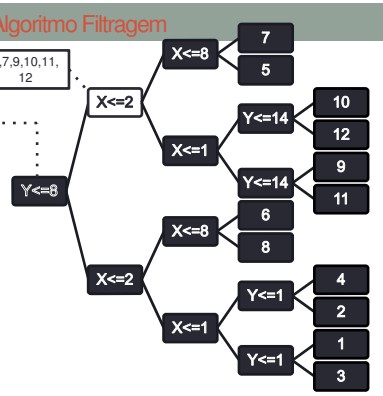
Para X8:
Z* ← Z3
Z3.wgtCent ← Z3.wgtCent + X8
Z3.count ← Z3.count + 1

Para X6:
Z* ← Z2
Z2.wgtCent ← Z2.wgtCent + X6
Z2.count ← Z2.count + 1

Exemplo 2º Passo – Algoritmo Filtragem

Objeto x_i	x_{i1}	x_{i2}
1	1	2
2	2	1
3	1	1
4	2	2
5	8	9
6	9	8
7	9	9
8	8	8
9	1	15
10	2	15
11	1	14
12	2	14

Prof. Eduardo R. Hruschka



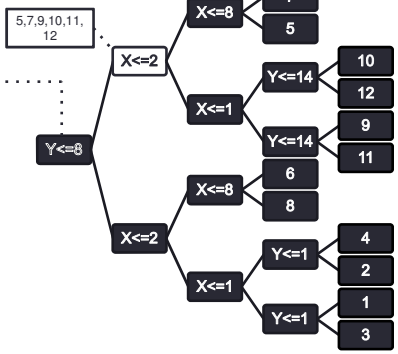

```
Filter(kdNode u, CandidateSet Z) {
    C ← u.cell;
    if (u is a leaf) {
        z* ← the closest point in Z to u.point;
        z*.wgtCent ← z*.wgtCent + u.point;
        z*.count ← z*.count + 1;
    }
    else {
        z* ← the closest point in Z to C's midpoint;
        for each (z ∈ Z \ {z*})
            if (z.isFarther(z*,C)) Z ← Z \ {z};
        if (|Z| = 1) {
            z*.wgtCent ← z*.wgtCent + u.wgtCent;
            z*.count ← z*.count + u.count;
        }
        else {
            Filter(u.left, Z);
            Filter(u.right, Z);
        }
    }
}
```

Z1 = [8 9] Z2 = [9 8] Z3 = [8 8]

Exemplo 2º Passo – Algoritmo Filtragem

Objeto x_i	x_{i1}	x_{i2}
1	1	2
2	2	1
3	1	1
4	2	2
5	8	9
6	9	8
7	9	9
8	8	8
9	1	15
10	2	15
11	1	14
12	2	14

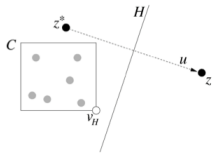
Prof. Eduardo R. Hruschka




```
Filter(kdNode u, CandidateSet Z) {
    C ← u.cell;
    if (u is a leaf) {
        z* ← the closest point in Z to u.point;
        z*.wgtCent ← z*.wgtCent + u.point;
        z*.count ← z*.count + 1;
    }
    else {
        z* ← the closest point in Z to C's midpoint;
        for each (z ∈ Z \ {z*})
            if (z.isFarther(z*,C)) Z ← Z \ {z};
        if (|Z| = 1) {
            z*.wgtCent ← z*.wgtCent + u.wgtCent;
            z*.count ← z*.count + u.count;
        }
        else {
            Filter(u.left, Z);
            Filter(u.right, Z);
        }
    }
}
```

Z1 = [8 9] Z2 = [9 8] Z3 = [8 8]

Midpoint = [3,83 12,66]
Z* ← Z1



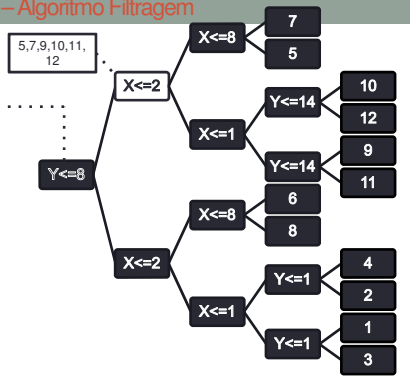
Para Z2:
u(Z2) = Z2 – Z* = [1 -1]
v(H) = [9 9]
d(v(H),Z2) = sqrt(1)
d(v(H),Z*) = sqrt(1)
Podamos Z2!

Para Z3:
u(Z3) = Z3 – Z* = [0 -1]
v(H) = [9 9]
d(v(H),Z3) = sqrt(2)
d(v(H),Z*) = sqrt(1)
Podamos Z3!

Exemplo 2º Passo – Algoritmo Filtragem

Objeto x_i	x_{i1}	x_{i2}
1	1	2
2	2	1
3	1	1
4	2	2
5	8	9
6	9	8
7	9	9
8	8	8
9	1	15
10	2	15
11	1	14
12	2	14

Prof. Eduardo R. Hruschka




```
Filter(kdNode u, CandidateSet Z) {
    C ← u.cell;
    if (u is a leaf) {
        z* ← the closest point in Z to u.point;
        z*.wgtCent ← z*.wgtCent + u.point;
        z*.count ← z*.count + 1;
    }
    else {
        z* ← the closest point in Z to C's midpoint;
        for each (z ∈ Z \ {z*})
            if (z.isFarther(z*,C)) Z ← Z \ {z};
        if (|Z| = 1) {
            z*.wgtCent ← z*.wgtCent + u.wgtCent;
            z*.count ← z*.count + u.count;
        }
        else {
            Filter(u.left, Z);
            Filter(u.right, Z);
        }
    }
}
```

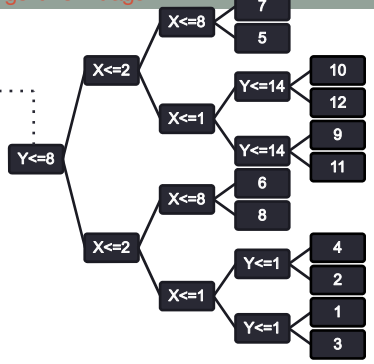
Z1 = [8 9] Z2 = [9 8] Z3 = [8 8]

Midpoint = [3,83 12,66]
Z* ← Z1

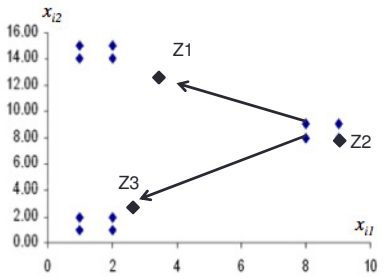
|Z| = 1:
Z1.wgtCent ← Z1.wgtCent + (x5+x7+x9+10+x11+x12)
Z1.count ← Z1.count + 6

Objeto x_i	x_{i1}	x_{i2}
1	1	2
2	2	1
3	1	1
4	2	2
5	8	9
6	9	8
7	9	9
8	8	8
9	1	15
10	2	15
11	1	14
12	2	14

Prof. Eduardo R. Hruschka




```
Filter(kdNode u, CandidateSet Z) {
    C ← u.cell;
    if (u is a leaf) {
        z* ← the closest point in Z to u.point;
        z*.wgtCent ← z*.wgtCent + u.point;
        z*.count ← z*.count + 1;
    }
    else {
        z* ← the closest point in Z to C's midpoint;
        for each (z ∈ Z \ {z*})
            if (z.isFarther(z*, C)) Z ← Z \ {z};
        if (|Z| = 1) {
            z*.wgtCent ← z*.wgtCent + u.wgtCent;
            z*.count ← z*.count + u.count;
        }
        else {
            Filter(u.left, Z);
            Filter(u.right, Z);
        }
    }
}
```



Fim da primeira iteração do algoritmo!

Centroids:
Z1 = [3,83 12,66] (x5,x7,x9,x10,x11,x12)
Z2 = [9 8] (x6)
Z3 = [2,8 2,8] (x1,x2,x3,x4,x8)

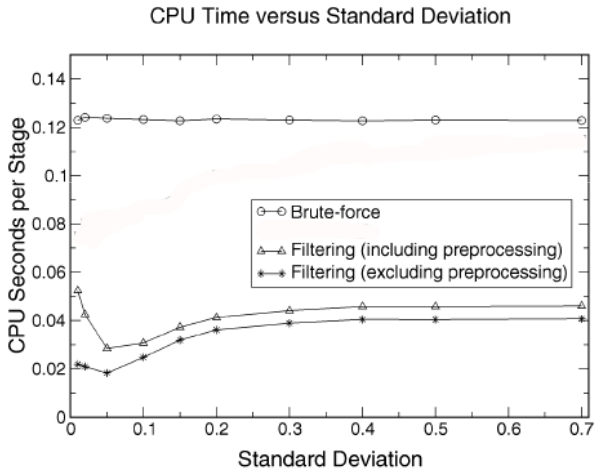
Notas sobre desempenho

- Para construir árvore KD
  - Altura da árvore: O(log*N*)
  - Cada nível: O(*nN*) - se utilizado método linear para cálculo da mediana – ex*: median of medians – Blum et al. (1973)*
  - O(*nN* log*N*)
- Para calcular as distâncias
  - Exemplo de pior caso: centros em esfera unitária centrada na origem e pontos agrupados ao redor da esfera, quase equidistantes dos centros
  - Praticamente não haverá podas de centros
    - O(nKN) = força bruta

Notas sobre desempenho

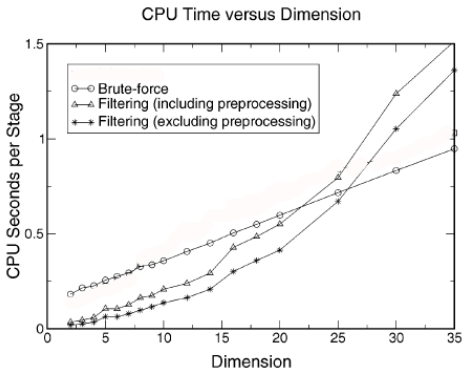
- Quanto maior a distância inter-grupos, mais eficiente é o algoritmo de filtragem
  - Sujeito a suposição de que os centros candidatos estão de fato próximos aos centros reais dos grupos (forte!)
  - Na prática, o tempo de execução cai drasticamente quanto maior a separação dos grupos para quaisquer centros candidatos
- Algoritmo de filtragem mostrou-se eficiente em dados sem grupos naturais
  - Experimentos em dados artificiais

Notas sobre desempenho



Notas sobre desempenho

- Tempo de execução aumenta exponencialmente com o número de atributos (dimensões)
  - Enquanto que na versão tradicional, o aumento é linear
  - Característico em algoritmos dependentes de árvores-KD ou árvores-R



Dúvidas?

Referências

- K. Alsabti, S. Ranka, and V. Singh, aAn Efficient k-means Clustering Algorithm,o Proc. First Workshop High Performance Data Mining, Mar. 1998.
- J. L. Bentley, Multidimensional binary search trees used for associative searching, Communications of the ACM 18 (1975), no. 9, 509–517.
- M. Blum, R.W. Floyd, V. Pratt, R. Rivest and R. Tarjan, "Time bounds for selection," J. *Comput. System Sci.* 7 (1973) 448-461.
- J. H. Friedman, J. L. Bentley, and R. A. Finkel, An algorithm for finding best matches in logarithmic expected time, ACM Trans. Math. Software 3 (1977), no. 3, 209–226.
- T. Kanungo, D.M. Mount, N.S. Netanyahu, C. Piatko, R. Silver- man, and A.Y. Wu, aComputing Nearest Neighbors for Moving Points and Applications to Clustering,o Proc. 10th Ann. ACM-SIAM Symp. Discrete Algorithms, pp. S931-S932, Jan. 1999.
- T. Kanungo, D. M. Mount, N. Netanyahu, C. Piatko, R. Silverman, and A. Y. Wu, An efficient k-means clustering algorithm: Analysis and implementation, *IEEE Trans. Pattern Analysis and Machine Intelligence*, 24 (2002), 881-892.
- S. Maneewongvatana and D. M. Mount. It's okay to be skinny, if your friends are fat. 4th Annual CGC Workshop on Computational Geometry, 1999.
- D. M. Mount and S. Arya, Ann: A library for approximate nearest neighbor searching, CGC 2nd Annual Fall Workship on Computational Geometry
- D. Pelleg and A. Moore, Accelerating Exact k-means Algorithms with Geometric Reasoning,o Proc. ACM SIGKDD Int'l Conf. Knowledge Discovery and Data Mining, pp. 277-281, Aug. 1999.