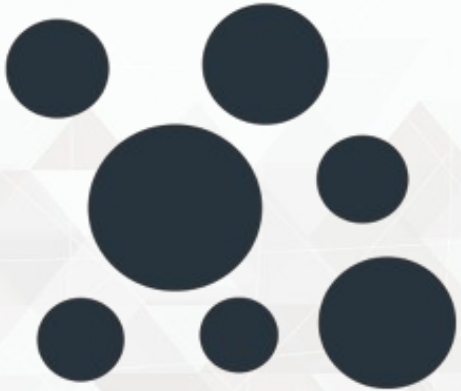
BigML
Clustering

2023

<lapti>

Clustering

- Técnica não-supervisionada de ML
  - Sem a necessidade de labels
  - Útil para encontrar instâncias semelhantes
  - Smart sampling/labelling
- Busca grupos de instâncias “auto-similares”
  - Clientes
    - Grupos com comportamento similar
  - Área médica
    - Pacientes com métricas de diagnóstico semelhantes
- Define cada grupo por um “centróide”
  - Centro geométrico do grupo
  - Representa um membro “médio”
  - Número de centróides (k) pode ser especificado ou determinado



Clustering

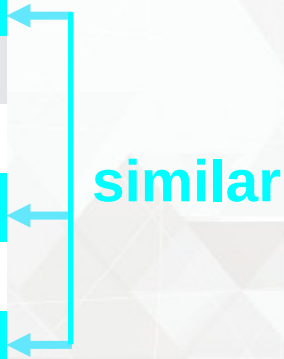
- BigML implementa dois principais algoritmos de clustering
- K-means
  - Algoritmo clássico para segmentação
  - Precisa do parâmetro K
    - Número de clusters
- G-means
  - Técnica para se determinar um K ótimo baseado nas features do dataset
    - Executa k-means em sequência adicionando centróides de maneira hierárquica
    - Paper “Learning the K in K-means”

Centróides de Cluster

date	customer	account	auth	class	zip	amount
Mon	Bob	3421	pin	clothes	46140	135
Tue	Bob	3421	sign	food	46140	401
Tue	Alice	2456	pin	food	12222	234
Wed	Sally	6788	pin	gas	26339	94
Wed	Bob	3421	pin	tech	21350	2459
Wed	Bob	3421	pin	gas	46140	83
Thr	Sally	6788	sign	food	26339	51

Centróides de Cluster

date	customer	account	auth	class	zip	amount
Mon	Bob	3421	pin	clothes	46140	135
Tue	Bob	3421	sign	food	46140	401
Tue	Alice	2456	pin	food	12222	234
Wed	Sally	6788	pin	gas	26339	94
Wed	Bob	3421	pin	tech	21350	2459
Wed	Bob	3421	pin	gas	46140	83
Thr	Sally	6788	sign	food	26339	51



Casos de Uso

- Segmentação de clientes
  - Quais clientes são similares?
  - Quantos grupos naturais existem?
- Descoberta de itens
  - Quais outros itens são similares a este?
- Similaridade
  - Que outras instâncias compartilham uma propriedade em específico?
- Recomendação (parcialmente)
  - Se um cliente gosta de um item, quais outros ele pode gostar?
- Active learning
  - Labelling eficiente de dados

Configurações de clusters

- Algoritmo
  - K-means
    - Permite especificar o número de clusters (k)
  - G-means
    - Usa uma distribuição Gaussiana para encontrar fronteiras de clusters
    - Determina um número de clusters ótimo
    - Valor crítico entre 1 e 20 (default 5)
      - Determina quão rigoroso o algoritmo será ao identificar clusters
- Valores default
  - Max/média/mediana/min/zero (default None)

Configuração de clusters

- Escalas
  - Como calcula distâncias, escala é importante
    - Auto: escala todas as features para que o desvio padrão seja 1
    - Configuração individual de escalas
- Pesos
  - Ajusta o impacto de cada instância no cluster
- Summary fields
  - Excluídos do cálculo de cluster
  - Ainda incluídos na saída, útil para labels
- Centróides

Clustering manual

Clustering manual

Clustering manual

- Foi usado conhecimento anterior para selecionar possíveis features que separam os objetos
- “redondo”, “fino”, “cantos”, “duro”, etc
- Itens foram então agrupados baseados nas features escolhidas
- Qualidade da separação foi então testada para garantir:
  - Chegar ao critério de k=3
  - Grupos eram suficientemente distantes
  - Sem crossover

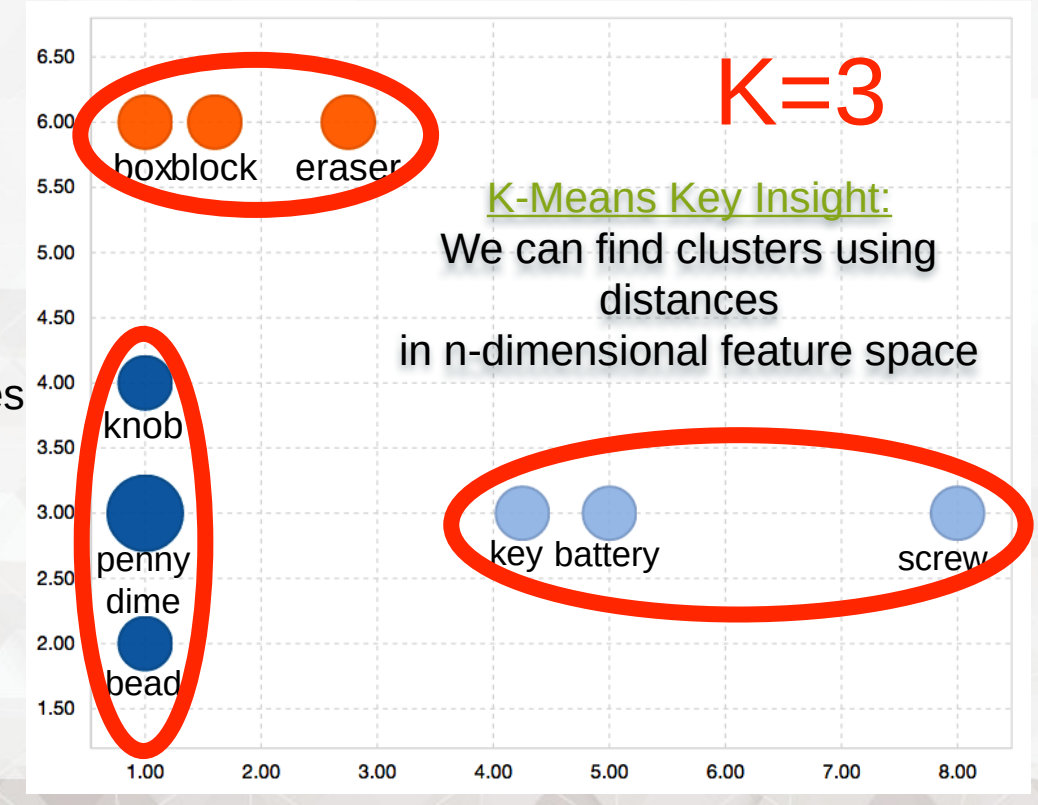
Clustering manual

- Criar features que capturem as diferenças de objetos
  - Comprimento / largura
    - Maior de 1, então “fino”
    - Igual a 1, então “redondo”
  - Número de superfícies
    - Superfícies distintas precisam de “bordas” que tem cantos
    - Fáceis de contar

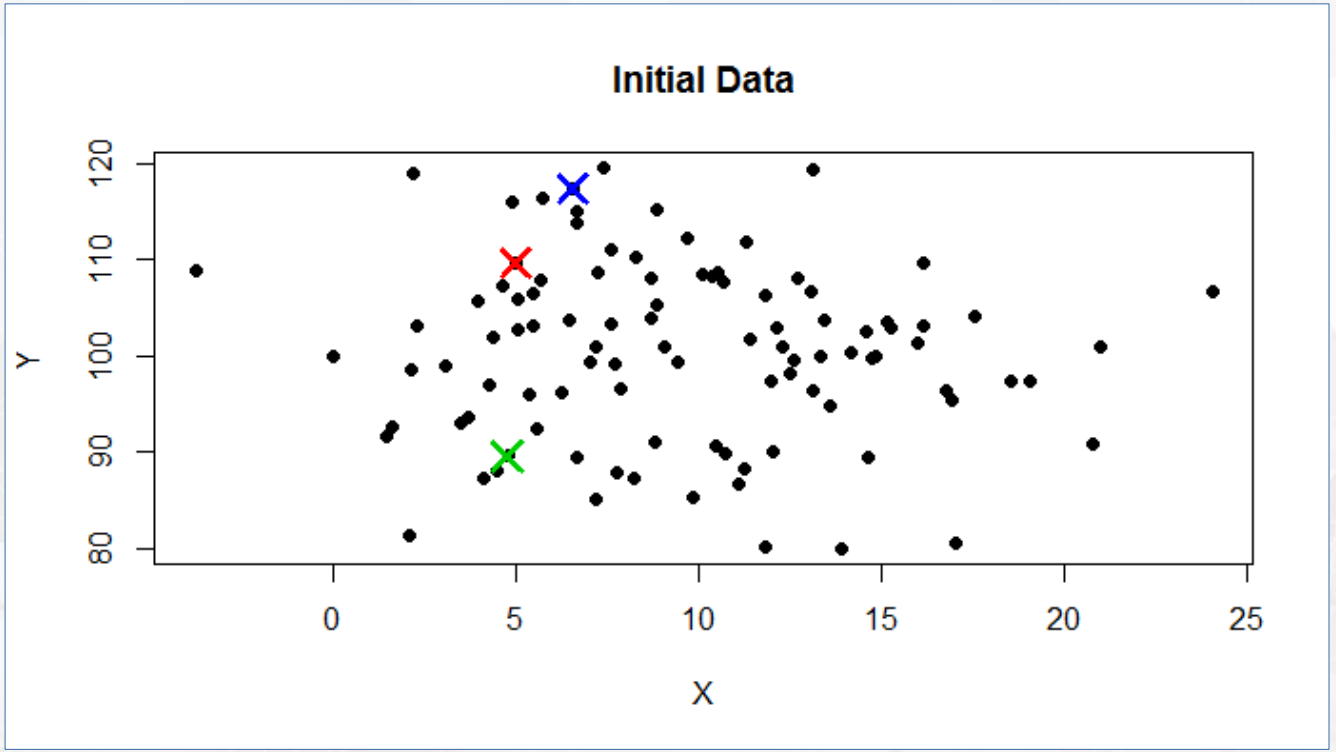
Features de clustering

Object	Length / Width	Num Surfaces
penny	1	3
dime	1	3
knob	1	4
eraser	2,75	6
box	1	6
block	1,6	6
screw	8	3
battery	5	3
key	4,25	3
bead	1	2

Gráfico por features



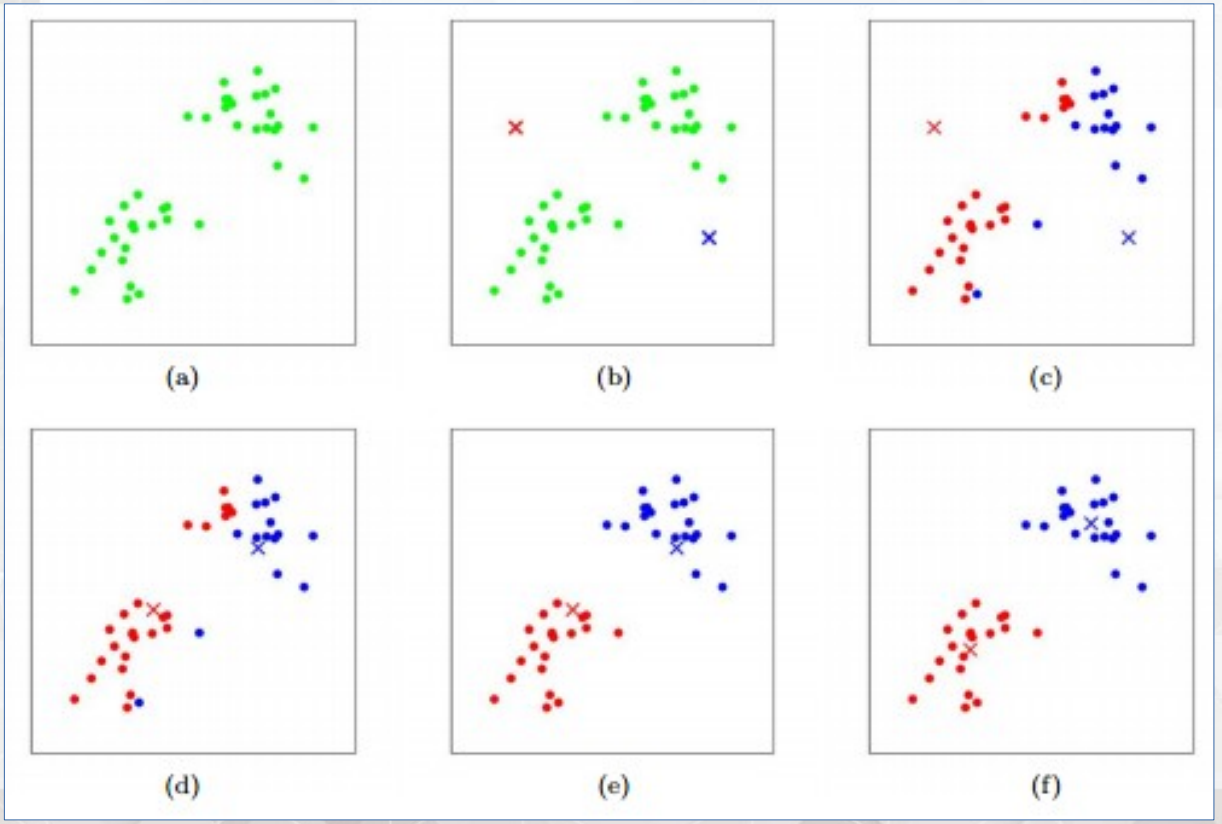
K-means - processo

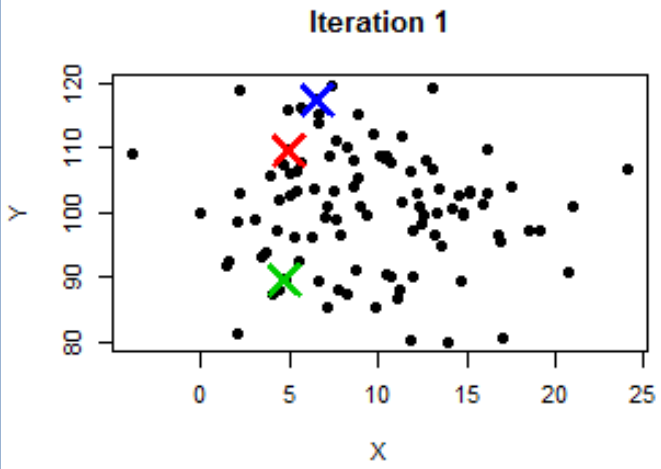


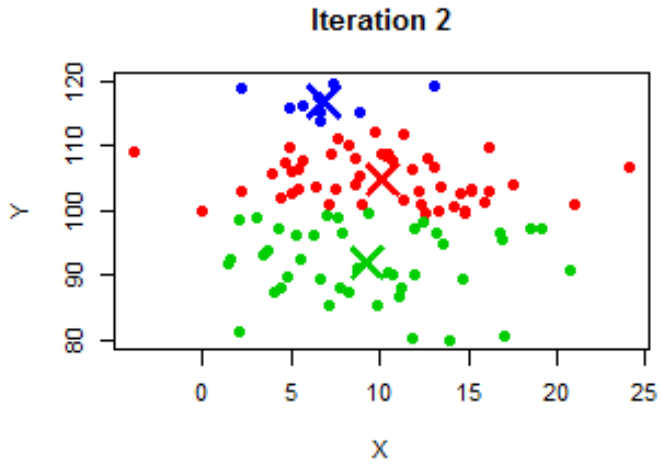
K-means - processo

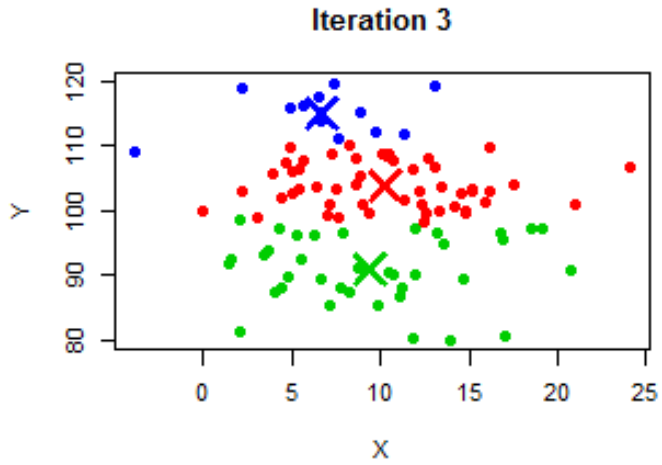
- Selecionar K
- Dispor centróides aleatoriamente
- Calcular os pontos mais próximos do centróide e atribuir ao cluster
- Centralizar o centróide em relação aos pontos
- Recalcular distância dos pontos e atribuir novamente aos clusters
- Repetir até que os centróides estabilizarem posição
  - Convergência

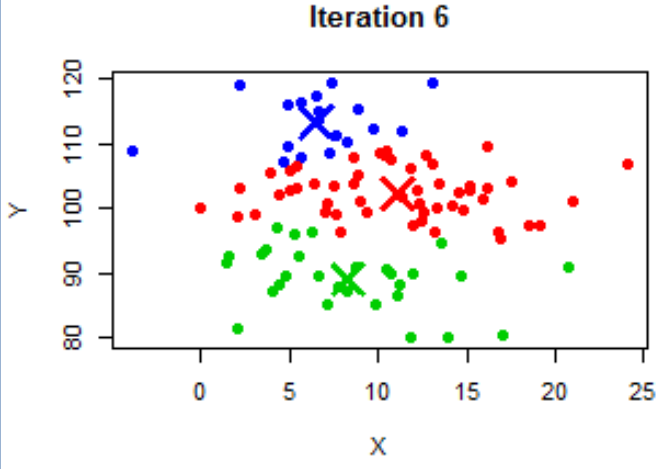
K-means - processo

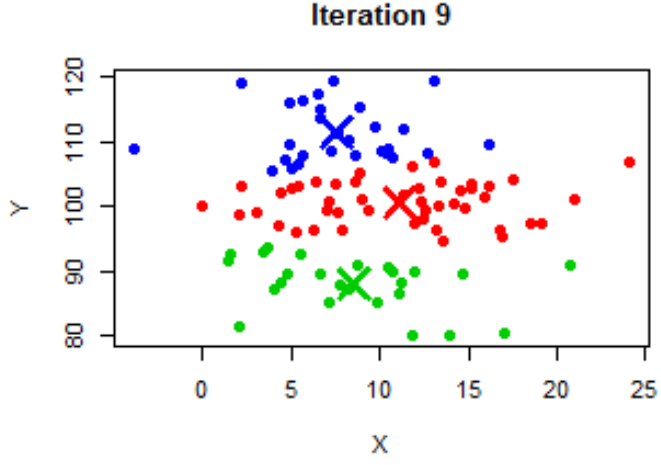


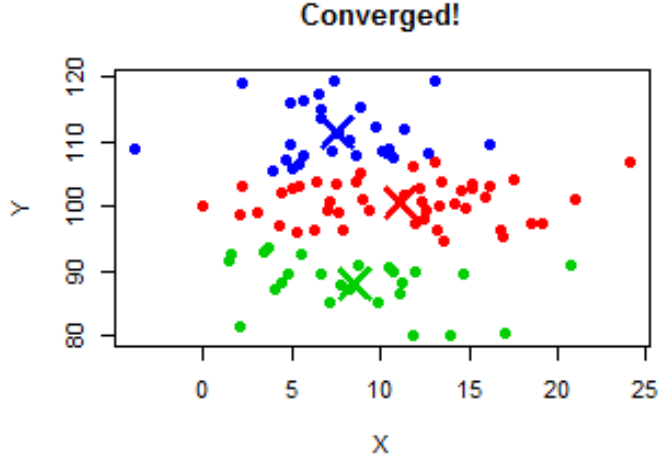

Iteration 1
Y
X
80
90
100
110
120
0
5
10
15
20
25


Iteration 2
Y
X
80
90
100
110
120
0
5
10
15
20
25


Iteration 3
Y
X
80
90
100
110
120
0
5
10
15
20
25


Iteration 6
Y
X
80
90
100
110
120
0
5
10
15
20
25

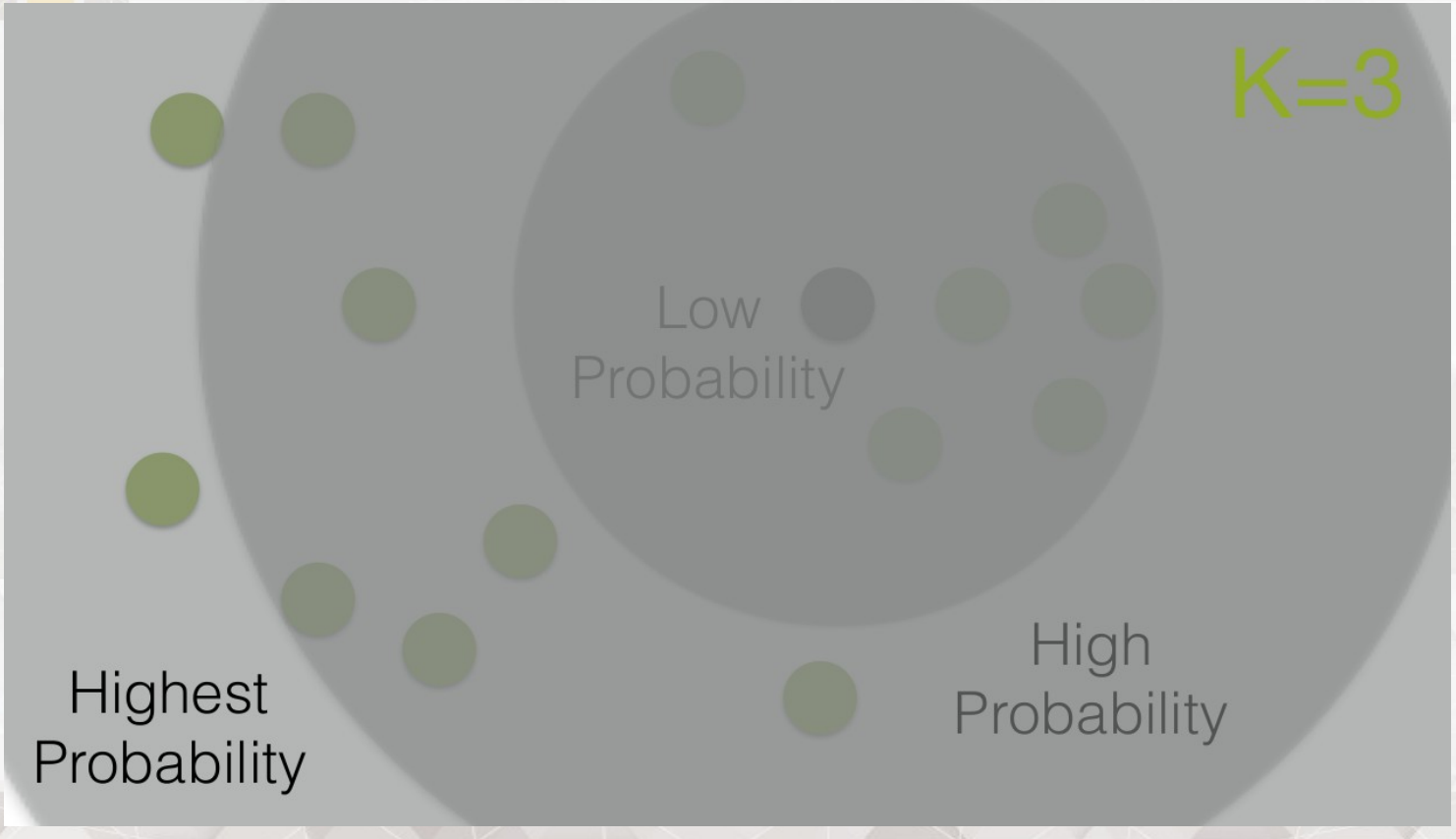

Iteration 9
Y
X
80
90
100
110
120
0
5
10
15
20
25


Converged!
Y
X
80
90
100
110
120
0
5
10
15
20
25

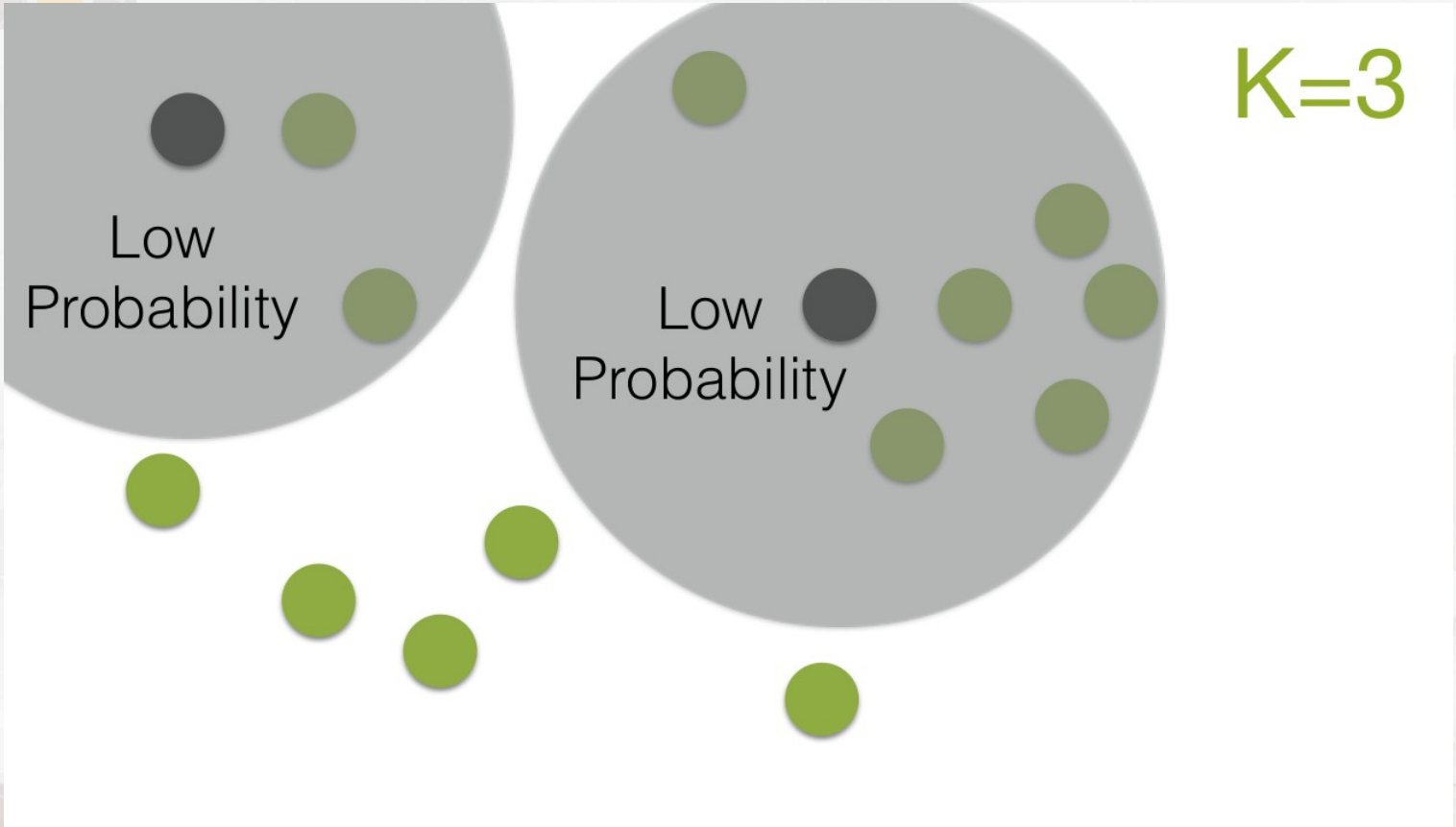
Pontos iniciais

- Pontos aleatórios ou instâncias em um espaço n-dimensional
  - Podem iniciar perto demais
  - Risco de convergência não ótima
- Escolher pontos o mais distantes uns dos outros
  - Pode ser sensível a outliers
- K++
  - O primeiro ponto é escolhido aleatoriamente a partir das instâncias
  - Cada ponto subsequente é escolhido a partir das instâncias restantes, com uma probabilidade proporcional ao quadrado da distância dos pontos próximos ao centro do cluster existente

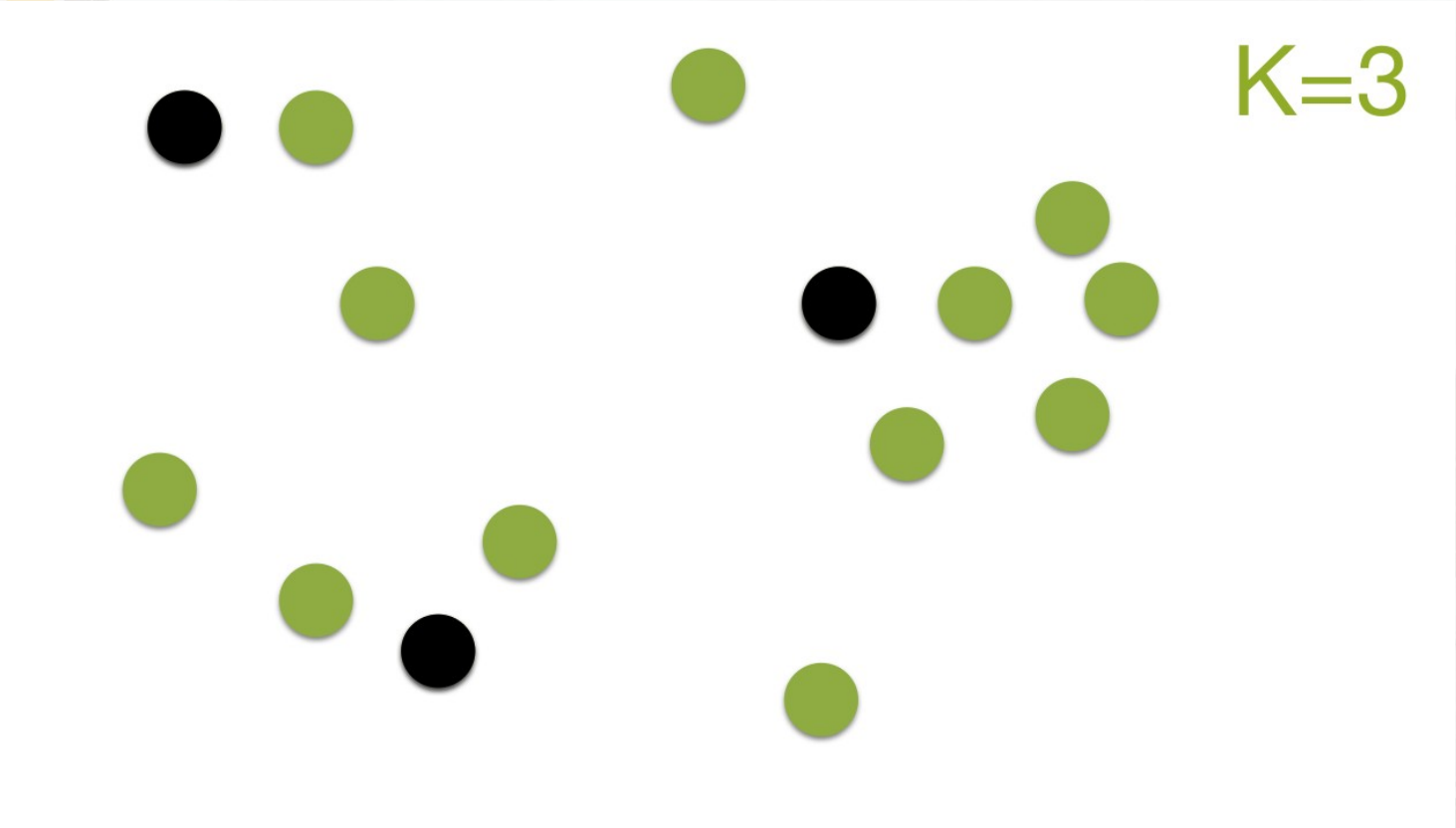
K++ - centros iniciais



K++ - centros iniciais

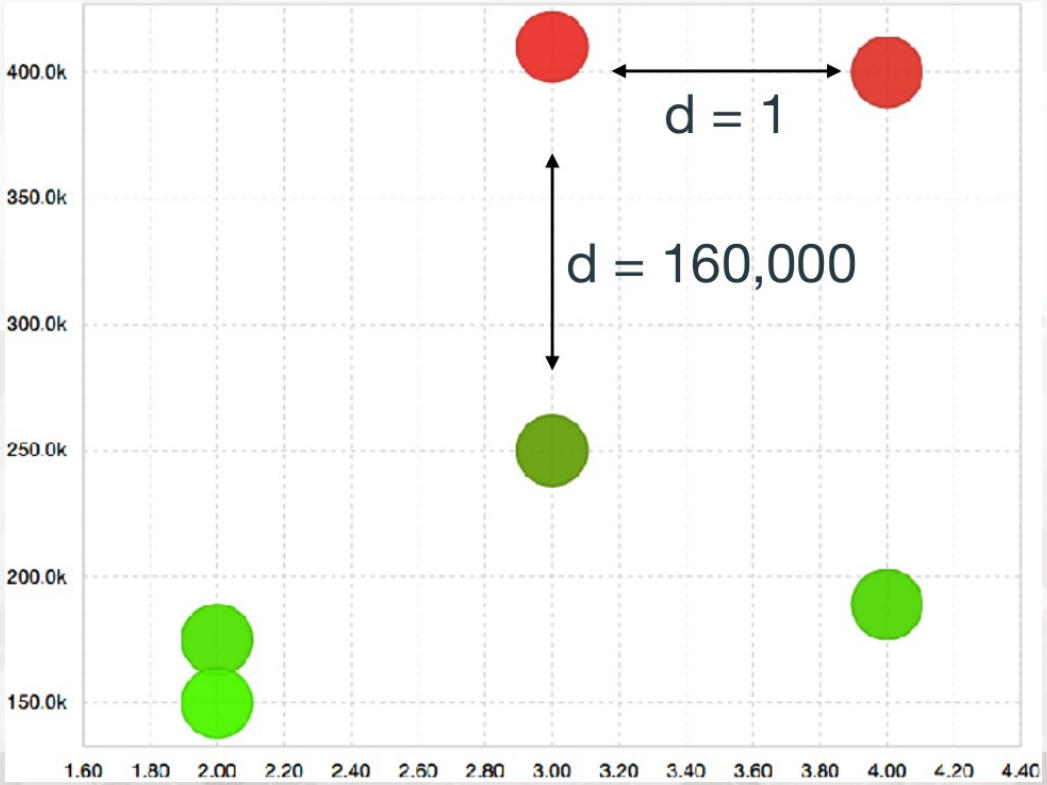


K++ - centros iniciais



Escala dos valores é importante

preço



número de quartos

Outras considerações

- Missing values
  - Qual a distância para um missing value?
  - Como (e se) considerar um missing value?
    - Média, valor fixo, desconsiderar?
- Quais as distâncias entre campos categóricos?
  - Quão distante é “vermelho” de “verde”?
- Quais as distâncias entre campos de texto?
  - Texto livre
- As distâncias precisam ser de um tipo específico?
  - Como Euclidiana?
- Número desconhecido de K ideal

Missing values

- Substituir por:
  - Máximo
  - Média
  - Mediana
  - Mínimo
  - Zero
- Ou ignorar as instâncias com missing values
  - Se um único campo não estiver preenchido, todo o registro é desconsiderado

Campos categóricos

- Abordagem: similar a k-prototipes
- Definir função especial de distância
  - Para duas instâncias x e y, e o campo categórico a:

if $x_a == y_a$ then

$(x,y)_{distância} = 0$ (ou valor que escala o campo)

else

$(x,y)_{distância} = 1$

Campos categóricos

Calcular então a distância Euclidiana entre vetores

animal	favorite toy	toy color
cat	ball	red
cat	ball	green
d=0	d=0	d=1

→ **D = 1**

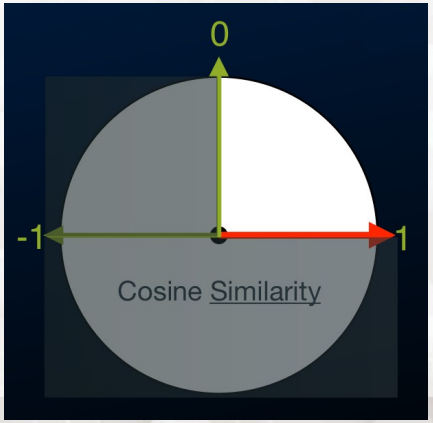
cat	laser	red
dog	squeaky	red
d=1	d=1	d=0

→ **D = √2**

Nota: o centróide é atribuído à categoria mais comum

Vetores de texto

	"hippo"	"safari"	"zebra"	....
Text Field #1 →	1	0	1	…
Text Field #2 →	1	1	0	…
	0	1	1	…



Similaridade do cosseno

cos() entre dois vetores

1 se collinear, 0 se ortogonal

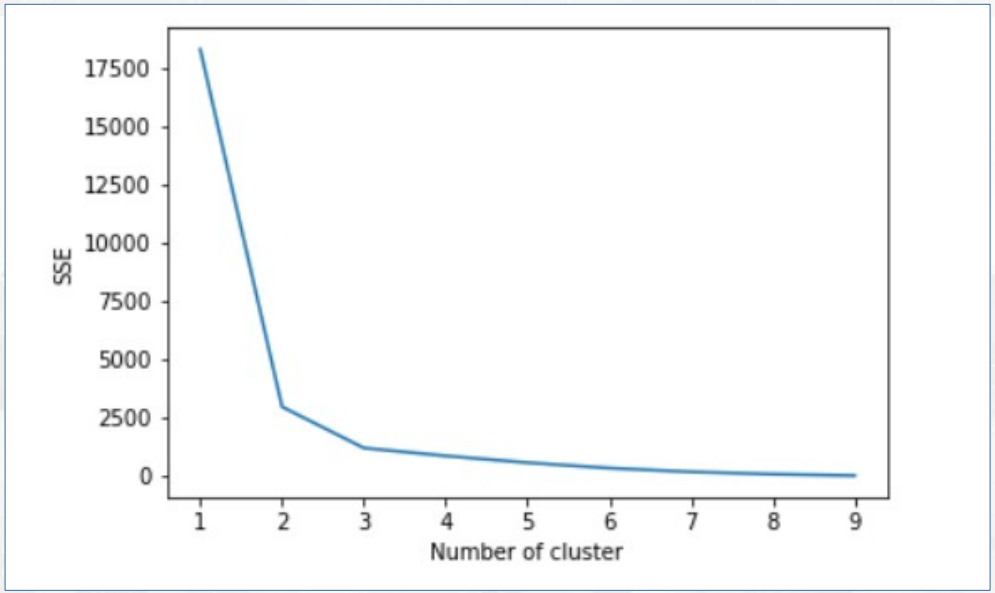
Apenas vetores positivos: $0 \leq CS \leq 1$

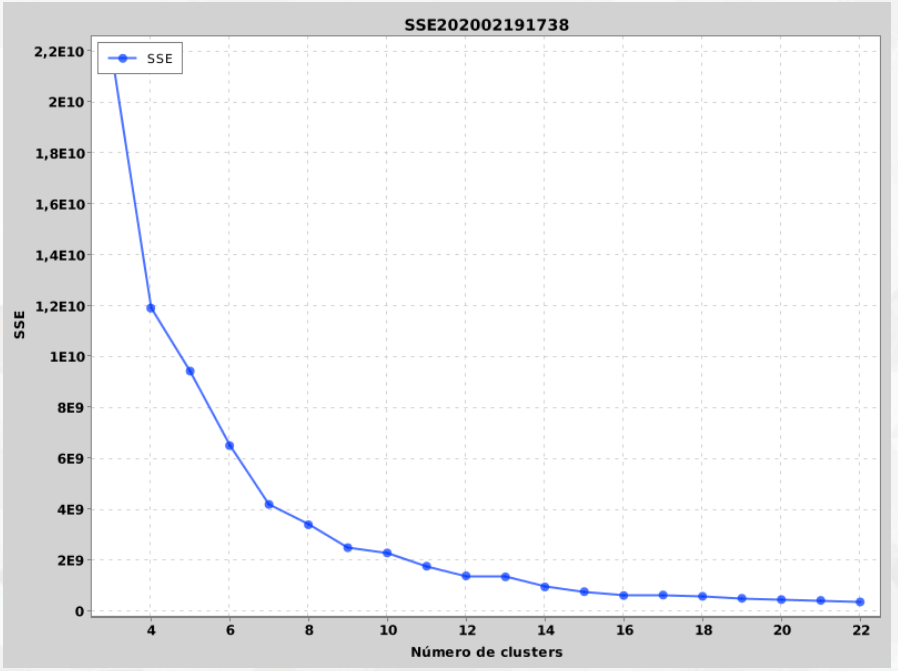
Distância Cos = 1 − Similaridade Cos

CD(TF1, TF2) = 0.5

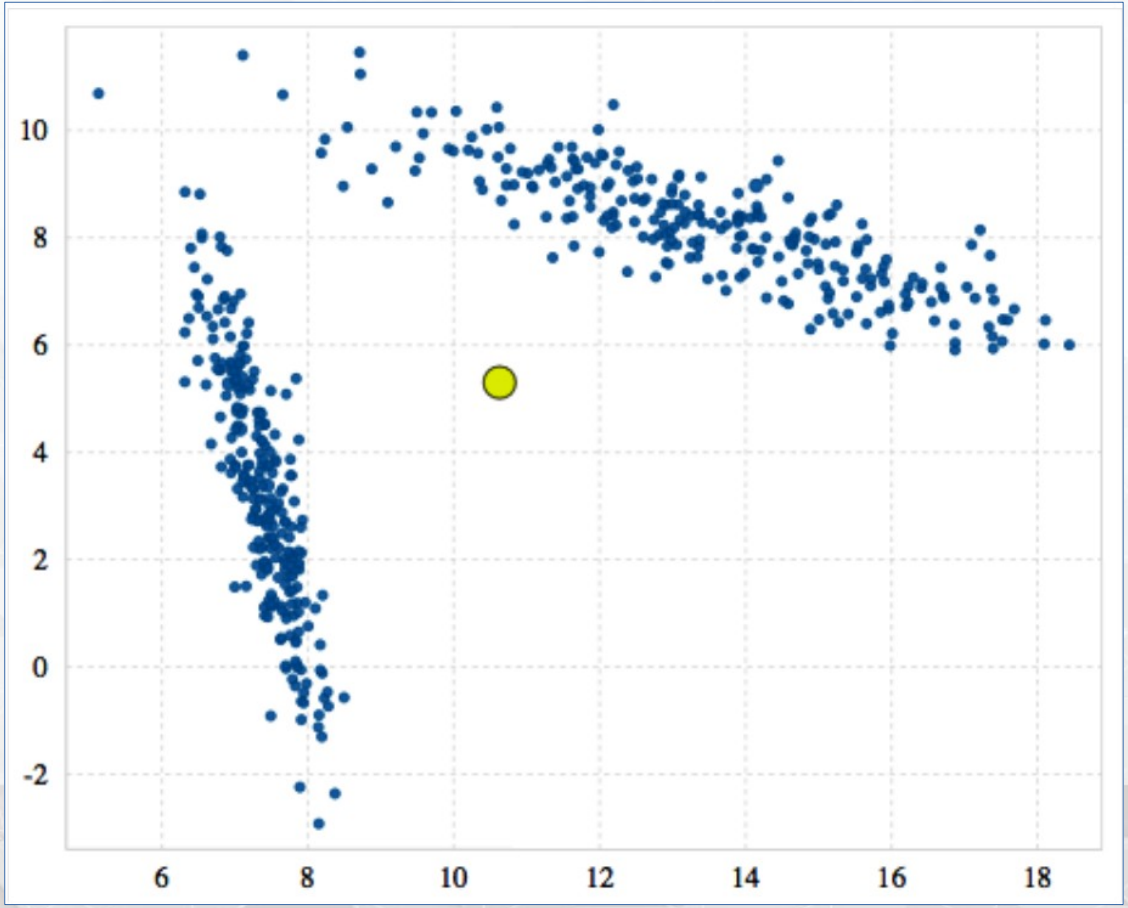
Encontrando K: SSE

Sum of the Squared Distance

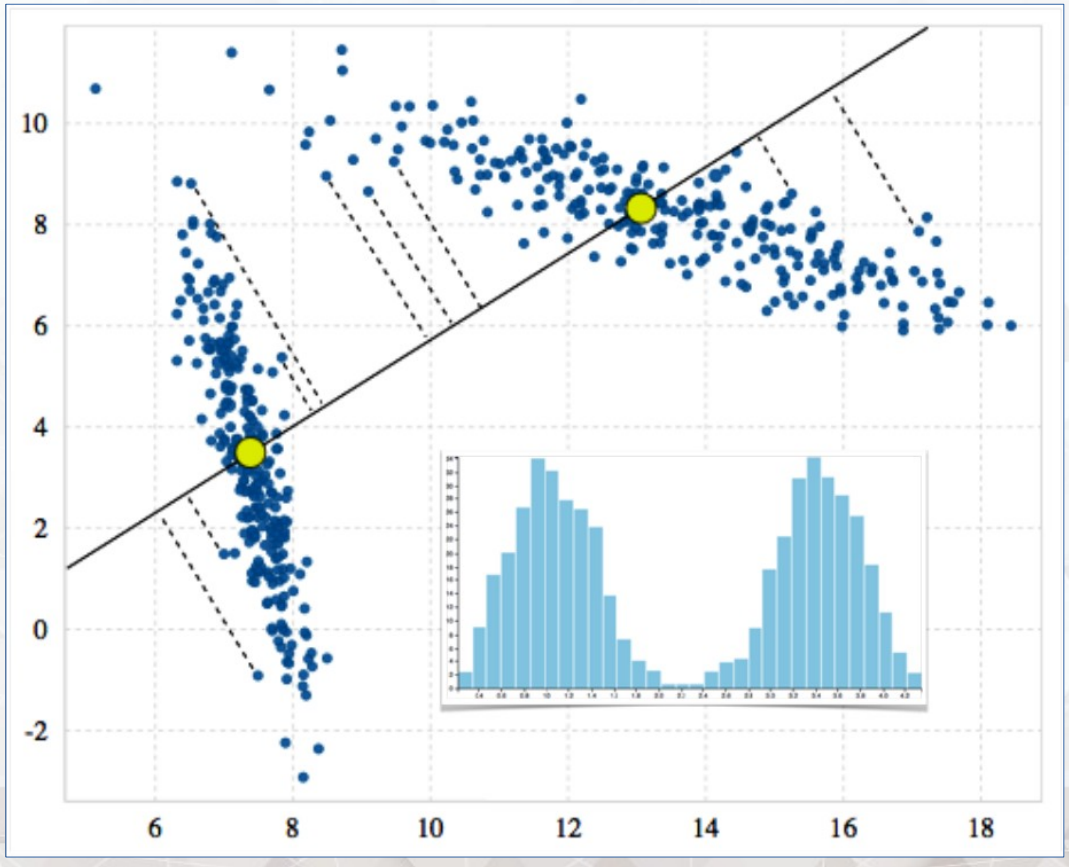




Encontrando K: G-means



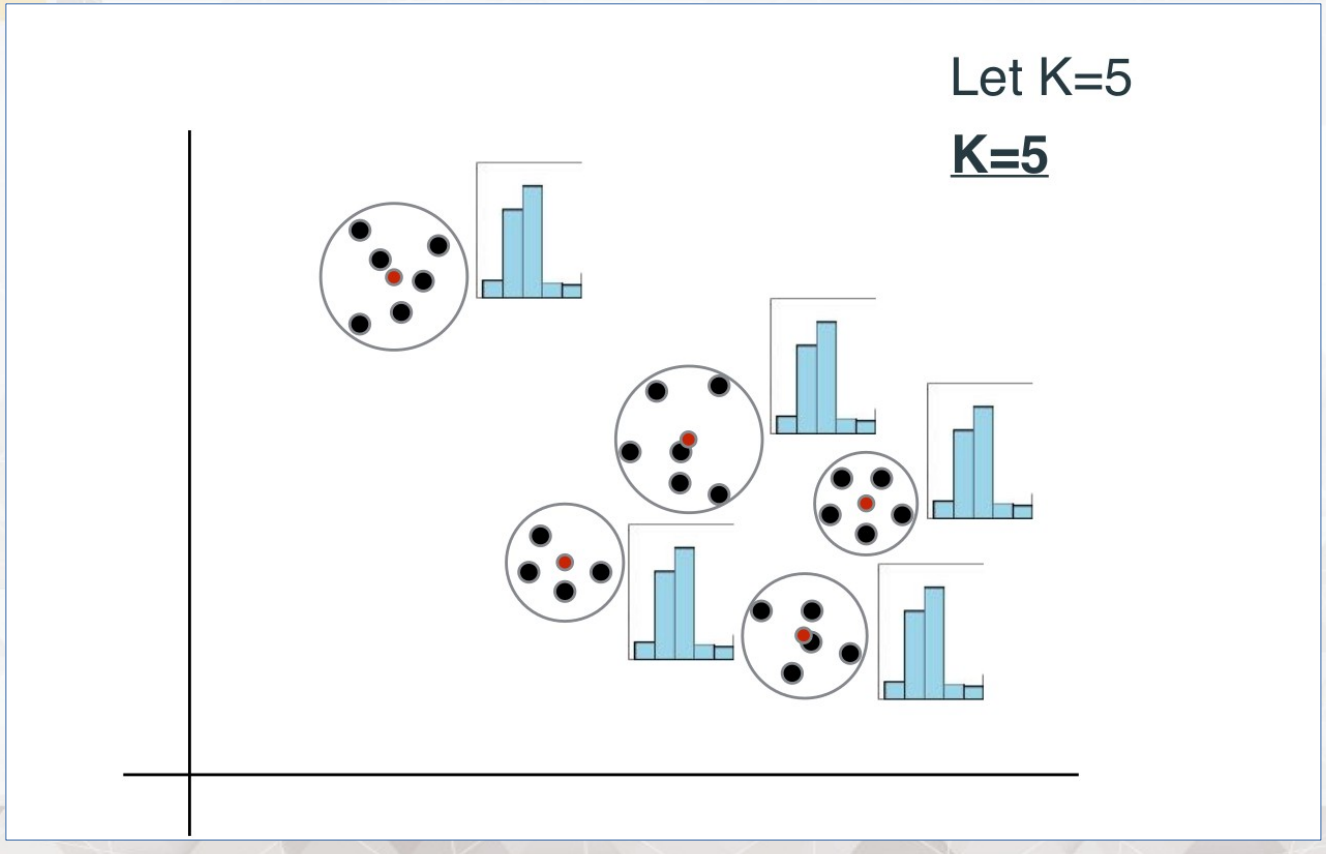
Encontrando K: G-means



Encontrando K: G-means

Encontrando K: G-means

Encontrando K: G-means



Datasets e modelos ativos

- É possível gerar um dataset para cada cluster no modelo
  - Análise, validação, exploração de dados
- Código pode ser gerado para uso dos modelos
  - Usando a API de bindings do BigML
- DecisionTrees podem ser geradas para cada cluster
  - Determinação de elementos do cluster
  - Reasoning com usuários
  - Modelo ativo independente (offline)

Predições

- Centroid
  - Determina o cluster de um indivíduo pelas propriedades locais
  - Local prediction (armazenada na memória do navegador)
- Batch centroid
  - Usa outro dataset como entrada
  - Grava um campo adicional de identificação de cluster
- Ambos acionáveis via API ou bindings

Limites para clusters

- K-means
  - No máximo 300 clusters
- G-means
  - No máximo 128 clusters
- Previsões no dashboard (Centroid)
  - Até 100 campos
  - Acima disso, somente pela API

Obrigado

leandro@utfpr.edu.br
http://lapti.ct.utfpr.edu.br

<lapti>