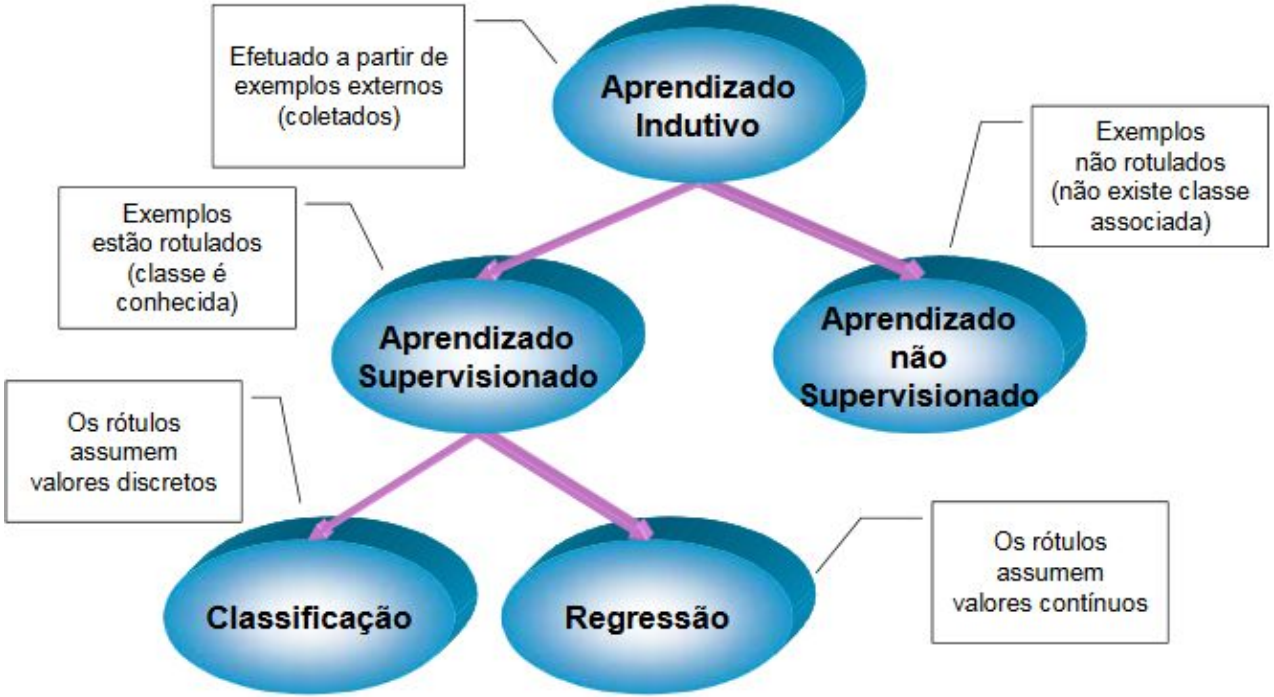
K-Means

Diego Addan

Unibrasil 2019

Aprendizado de máquina

Hierarquia do Aprendizado



Algoritmos de Aprendizagem de Máquinas (ML)

Algoritmos supervisionados: o conjunto de dados de treinamento tem entradas, bem como a saída desejada. Durante a sessão de treinamento, o modelo ajustará suas variáveis para mapear as entradas para a saída correspondente.

Algoritmos não supervisionados: nesta categoria, não há um resultado alvo. Os algoritmos agruparão o conjunto de dados para diferentes grupos.

Algoritmos de reforço: esses algoritmos são treinados para tomar decisões. O algoritmo irá treinar com base no sucesso / erro de saída. Eventualmente, o algoritmo de experiência poderá dar boas previsões.

Algoritmos de Aprendizagem de Máquinas (ML)

Árvore de decisão

Floresta aleatória

KNN (K-vizinhos mais próximos)

SVM (Support Vector Machine)

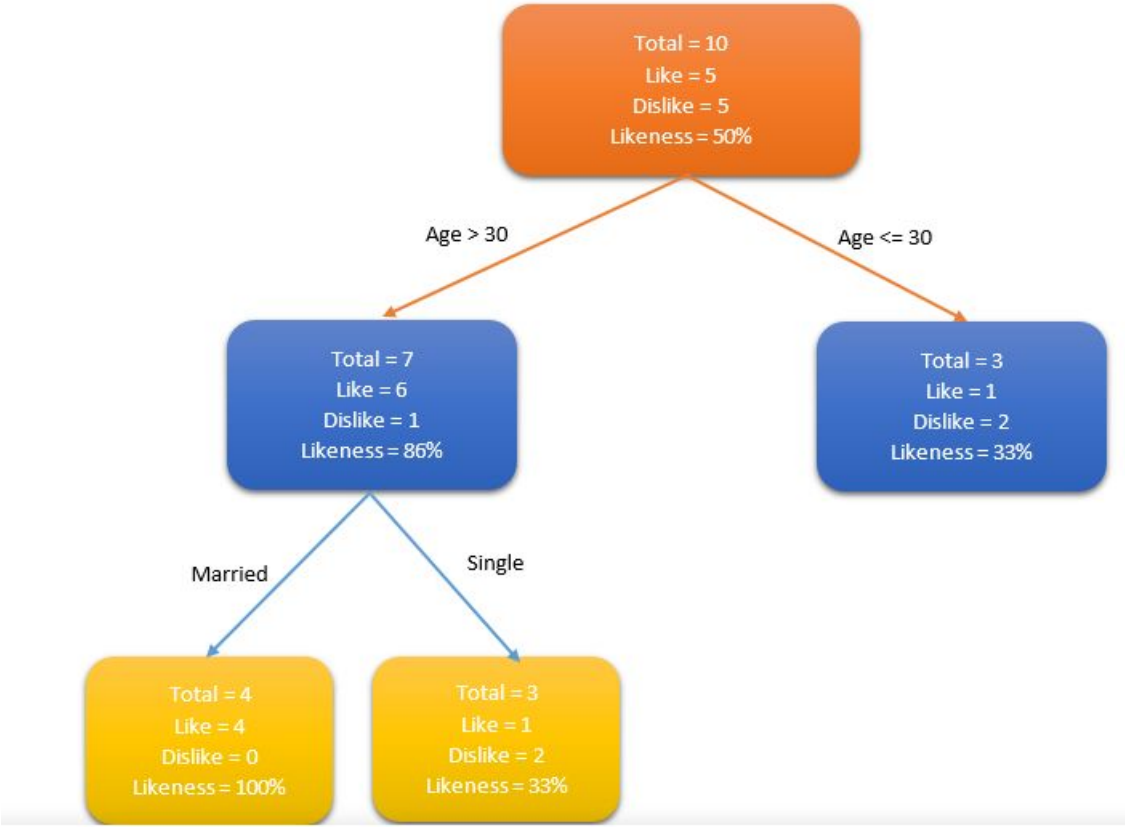
K-Means

Algoritmos de redução dimensional

Algoritmos de aumento de gradiente

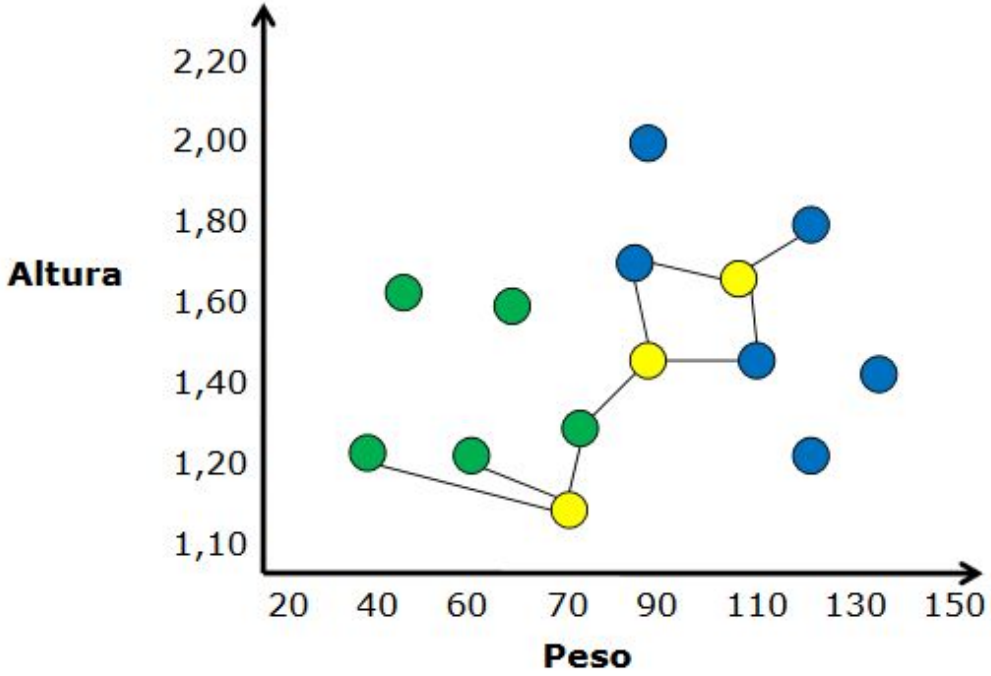
Algoritmos de Aprendizagem de Máquinas (ML)

Árvore de decisão



Algoritmos de Aprendizagem de Máquinas (ML)

KNN: K-vizinhos mais próximos



Algoritmo K-Means

Para os problemas que não possuem uma base de dados já rotulada ou com valores numéricos que identifiquem nossos dados de treino, precisamos de um agente baseado em **Aprendizado Não-Supervisionado**.

Aprendizado não-supervisionado é a forma de aprendizado de máquina em que o agente recebe um conjunto de dados de treino **não-rotulados** e precisa aprender algo sobre os dados em questão.

Algoritmo K-Means

Uma das maneiras mais populares de se implementar o aprendizado não-supervisionado é por meio do aprendizado de **agrupamentos** (do inglês, **clustering**).

Após a convergência de algoritmos de aprendizado de agrupamentos, cada agrupamento é um bloco de uma partição do conjunto de dados de treino que agrupa dados considerados próximos entre si.

Após o agrupamento, se é fornecido um dado de teste, o algoritmo é capaz de informar a qual agrupamento pertence o dado de teste.

Algoritmo K-Means

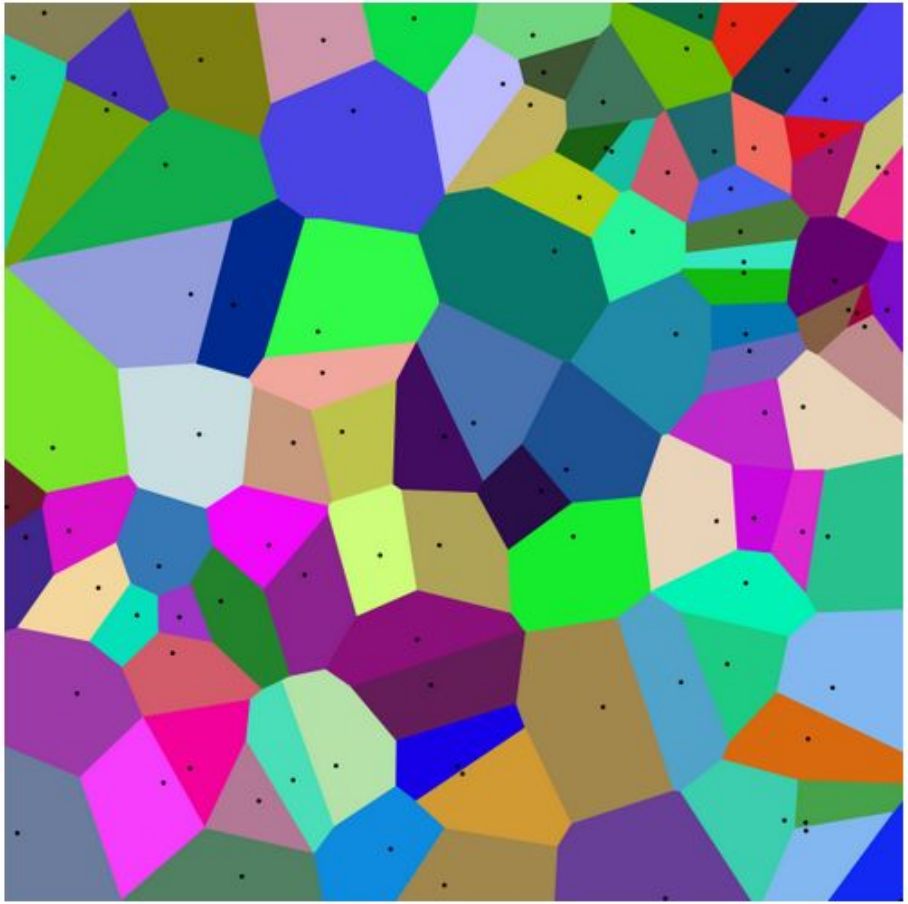
k-means é um popular algoritmo de aprendizado não-supervisionado que gera k agrupamentos a partir de um conjunto de dados de treino.

A saída gerada pelo k-means pode ser enxergada como um diagrama de Voronoi, que é um particionamento do conjunto de dados com alguns pontos centrais (também chamados de **centroides**).

Os centroides são os dados que melhor representam o agrupamento.

Algoritmo K-Means

Diagrama de Voronoi.



Algoritmo K-Means

A proximidade entre os pares de dados de treino pode ser calculada com diferentes noções de distância.

Por exemplo, podemos usar distância Euclidiana ou distância de Manhattan. Diferentes fórmulas de distância utilizadas provavelmente irão gerar diferentes agrupamentos já que os cálculos não são os mesmos.

Algoritmo K-Means

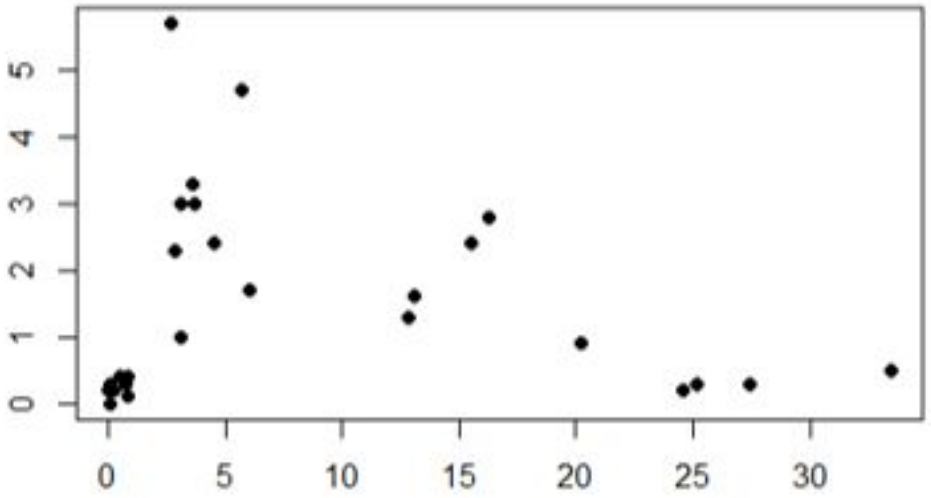
O algoritmo k-means gera agrupamentos da seguinte maneira:

- Recebe um inteiro positivo **k** e um conjunto de **n** dados de treino, cada um deles descrito por um vetor numérico com comprimento **d**.
- Define **k** pontos aleatórios (centroids) no espaço dimensional **d**.
- Enquanto os **k** pontos não convergirem:
  - Para cada dado de treino, o algoritmo encontra qual é o centroide mais próximo e define-o como representante do dado de treino.
  - Para cada agrupamento, o algoritmo encontra quem são os dados de treino agrupados por esse ponto k, calcula a média aritmética desses dados de treino e define o resultado como novo valor para representar o agrupamento, atualiza o valor do centroide.

Algoritmo K-Means

Agrupando os dados

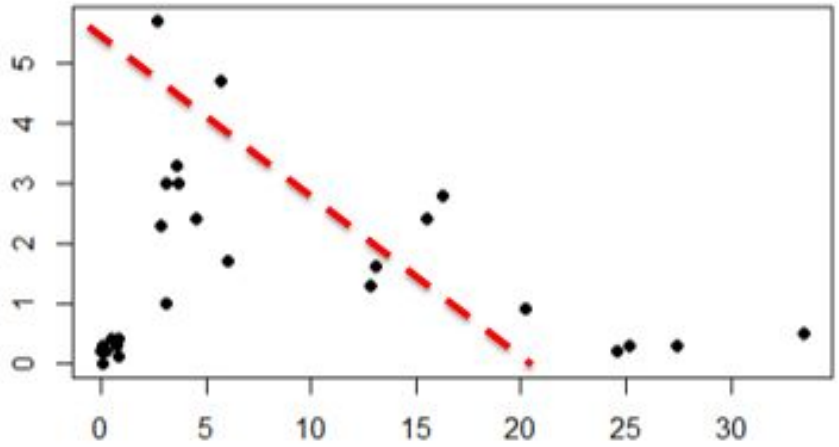
Para exemplificar, pense em um dataset com algumas amostras dispostas nos eixo X e Y, como o gráfico abaixo. Seu objetivo é agrupar estes dados baseado em sua similaridades.



Algoritmo K-Means

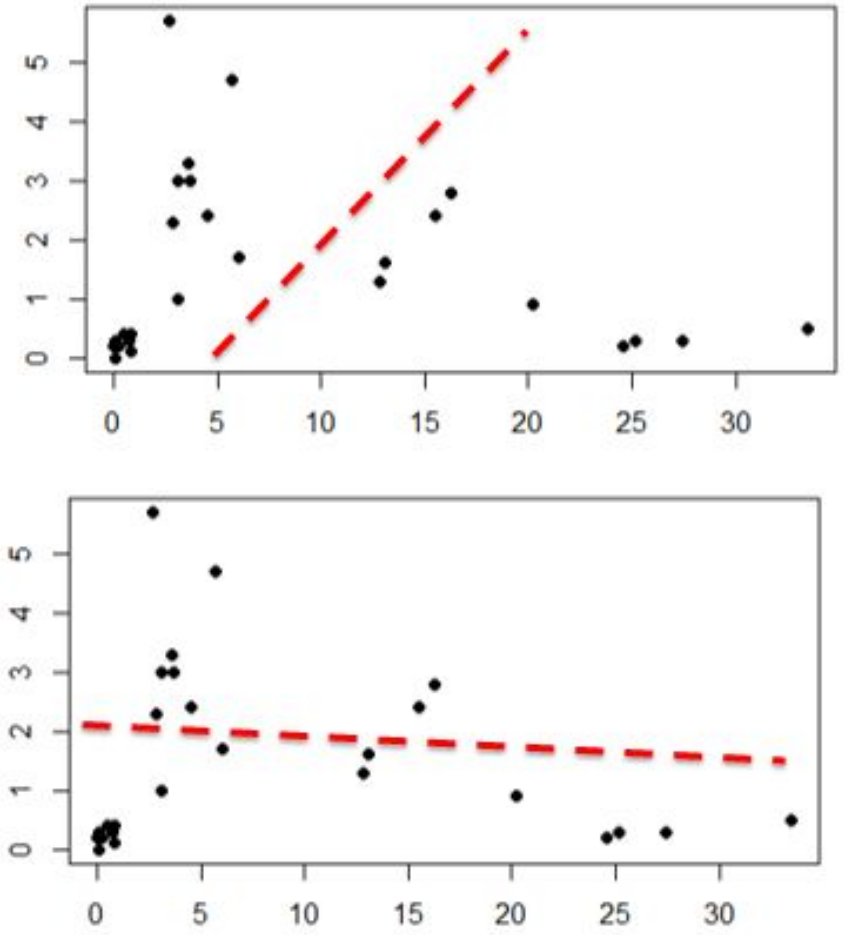
Agrupando os dados

É possível bater o olho neste gráfico e ver a separação em alguns grupos. Observando o gráfico é possível criar um número diferente de cluster. Por exemplo, alguns podem ver a separação com apenas 2 clusters, e o gráfico poderia ser assim:



Algoritmo K-Means

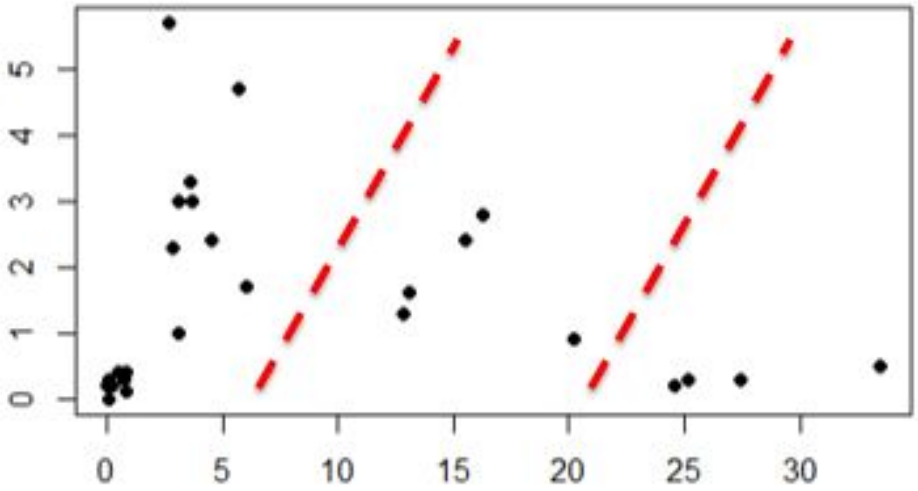
Agrupando os dados

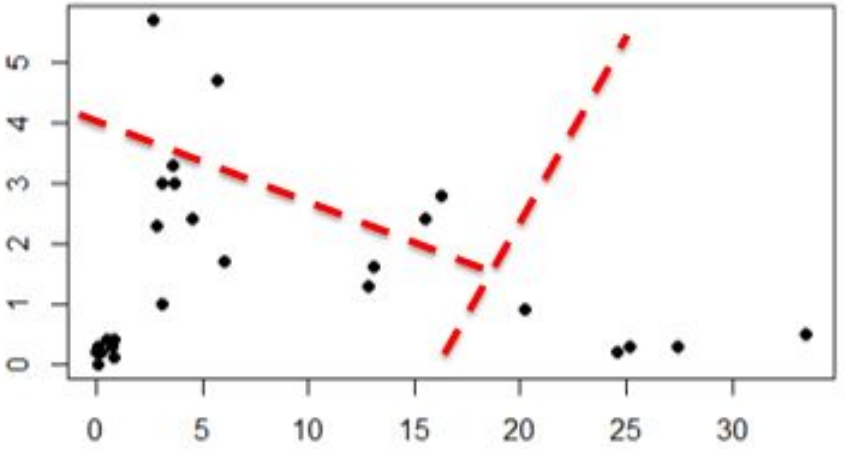


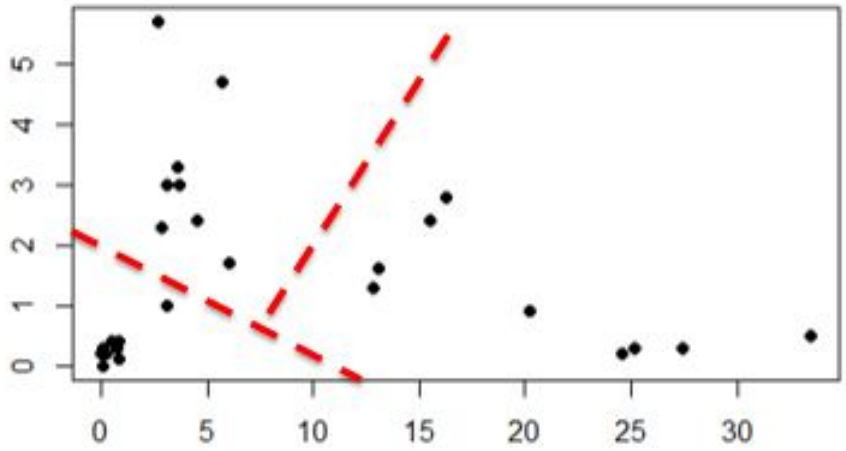
Algoritmo K-Means

Agrupando os dados

Outros podem encontrar 3 grupos,







Algoritmo K-Means

Agrupando os dados

Todos os gráficos estão corretos de acordo com a visão de cada observador.

Isso é determinado por métodos que definem o número de clusters em um dataset.

Ex: Elbow Method, X-Means Clustering, Cross Validation, Silhouette Method, etc.

Algoritmo K-Means

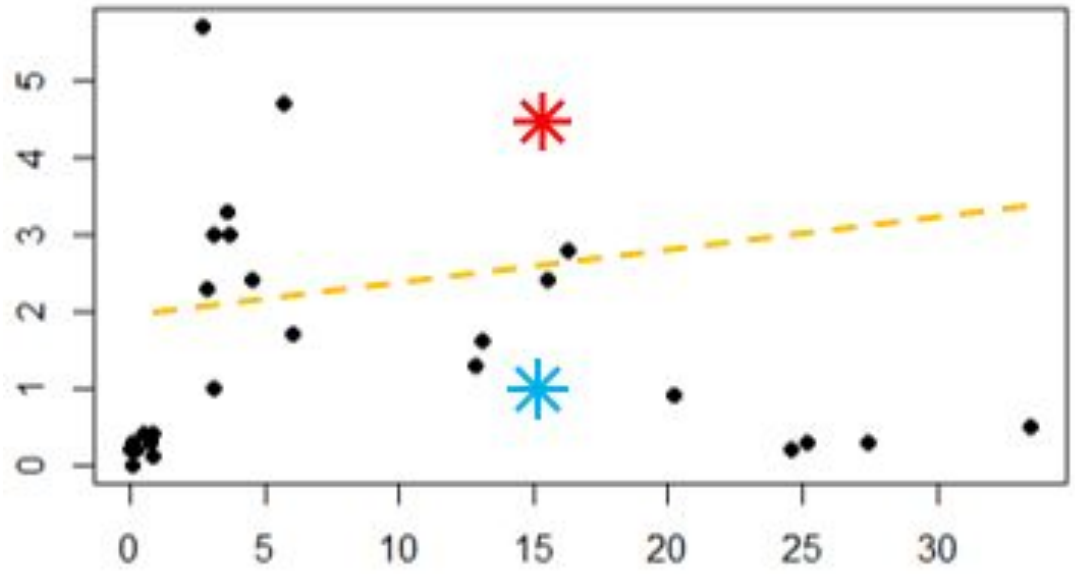
Para entender o funcionamento vamos separar em 2 clusters e entender os passos que o algoritmo K-Means faz como os dados para convergir em um resultado. Neste caso o K será igual a 2, criando os 2 clusters que estamos buscando.

- O K, de K-Means, é a quantidade de centróides (pontos centrais dos grupos) que serão criados e ajudará a encontrar a similaridade dos dados.

- Uma das formas de iniciar o processo é o algoritmo inserir o K pontos (centróides) aleatórios iniciais. Pode ser qualquer lugar do plano, para em seguida começar as iterações e encontrar os resultados.

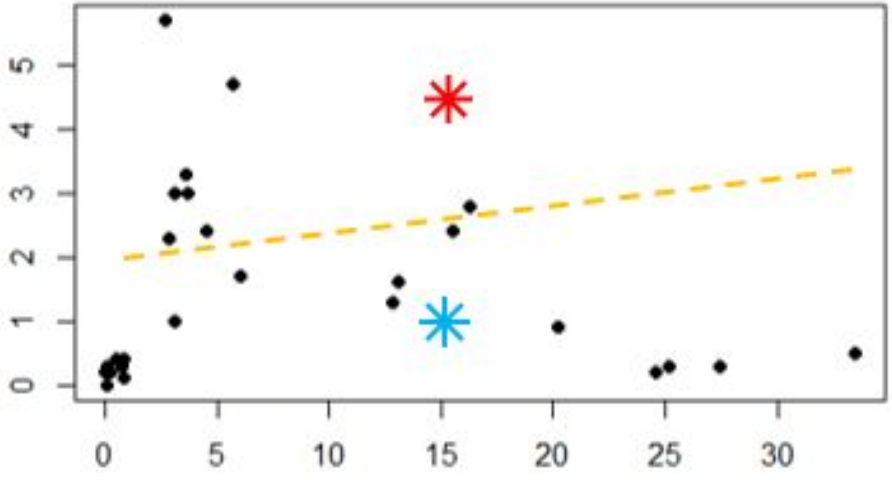
Algoritmo K-Means

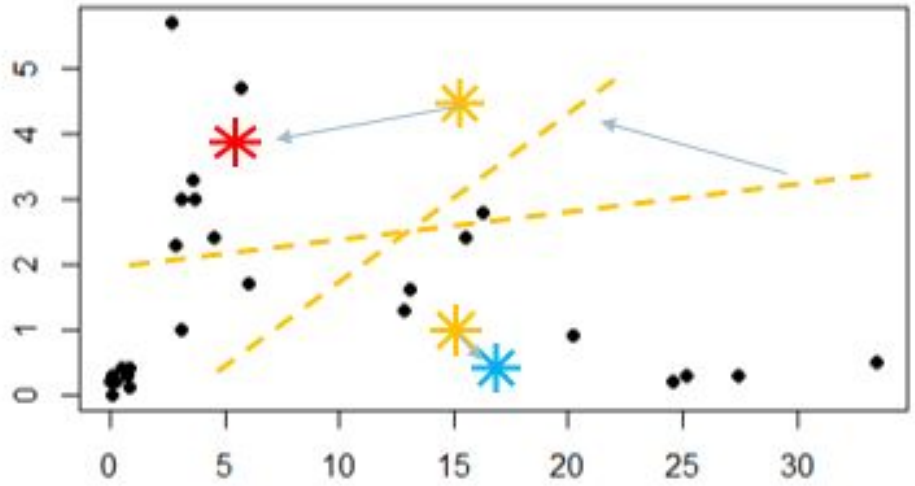
Veja dois pontos aleatórios criados no gráfico, e uma linha tracejada que é calculada aproximadamente na metade da distância dos pontos **Vermelho** e **Azul**.



Algoritmo K-Means

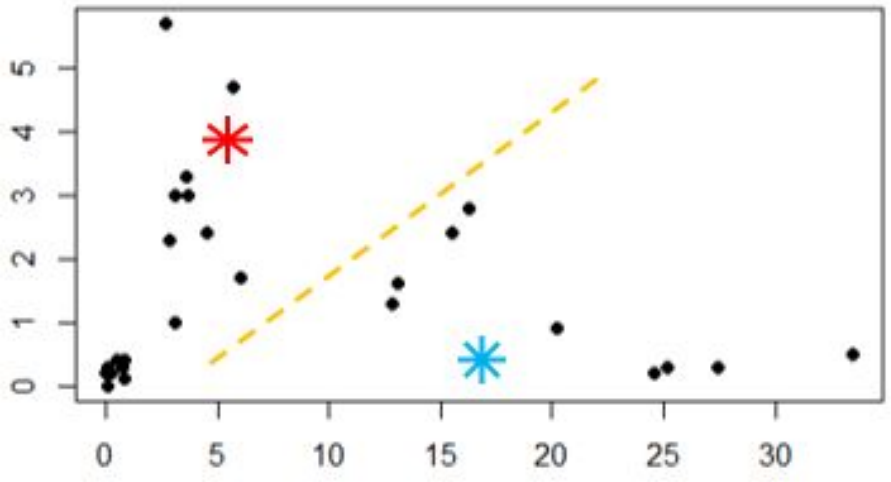
A primeira iteração do algoritmo é calcular a **distância média de todos os pontos** que estão atrelados ao centróide, e então mudar a posição do centróide para o novo ponto que foi calculado. Essa mudança de posição do centróide pode alterar os itens que fazem parte daquele grupo.

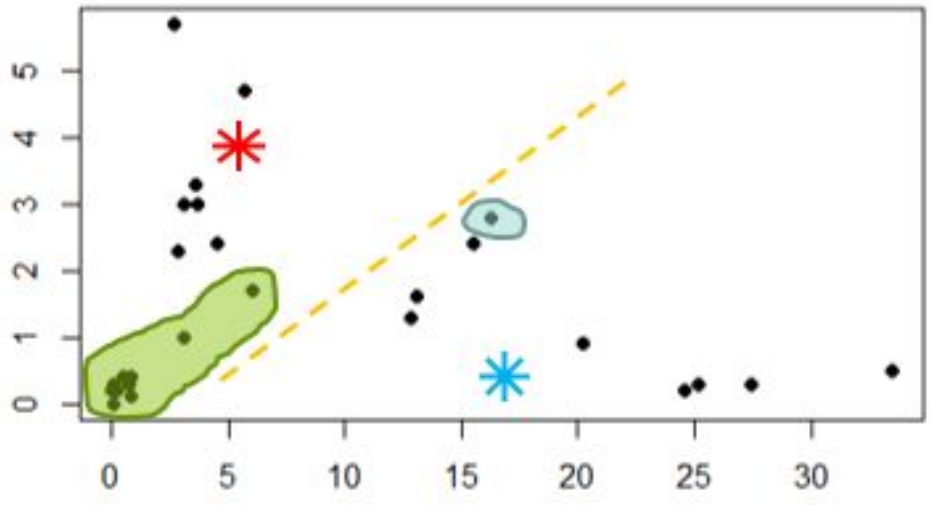




Algoritmo K-Means

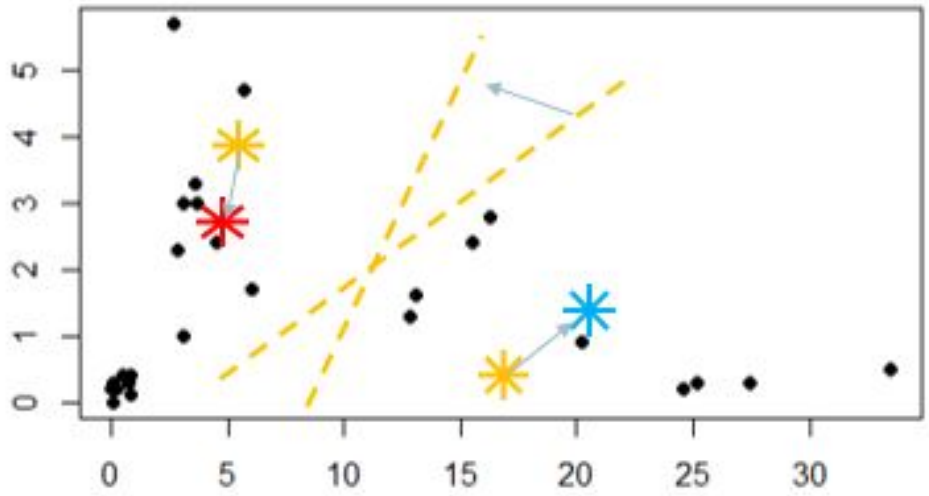
A primeira iteração do algoritmo é calcular a **distância média de todos os pontos** que estão atrelados ao centróide, e então mudar a posição do centróide para o novo ponto que foi calculado. Essa mudança de posição do centróide pode alterar os itens que fazem parte daquele grupo.

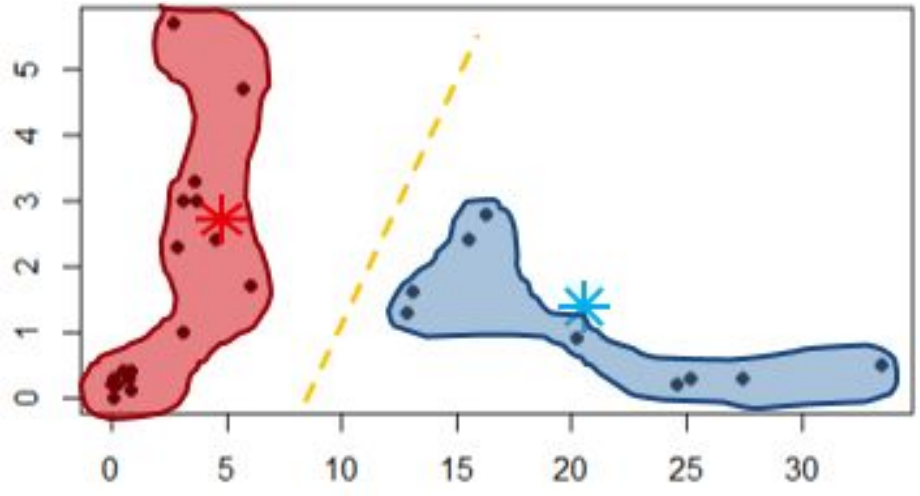




Algoritmo K-Means

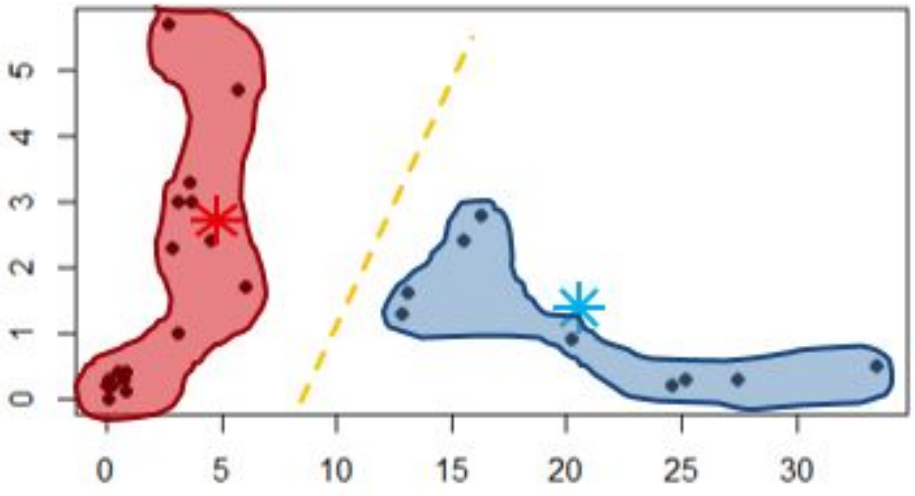
Essa iteração de cálculo da média da distância dos pontos até o centróide ocorre em loop até que nenhum ponto mude de centróide, isso acontece quando os centróides param de se mover porque já estão na posição central da distância entre os pontos.





Algoritmo K-Means

Veja que não houve mais mudança de pontos entre o gráfico e o centróide, fazendo com que o algoritmo K-Means pare sua execução chegando ao resultado esperado e criando dois grupos. Assim, quando um novo item for incluído no gráfico, ele já terá um grupo que atende aquela região e o computador já saberá do que se trata o dado novo.



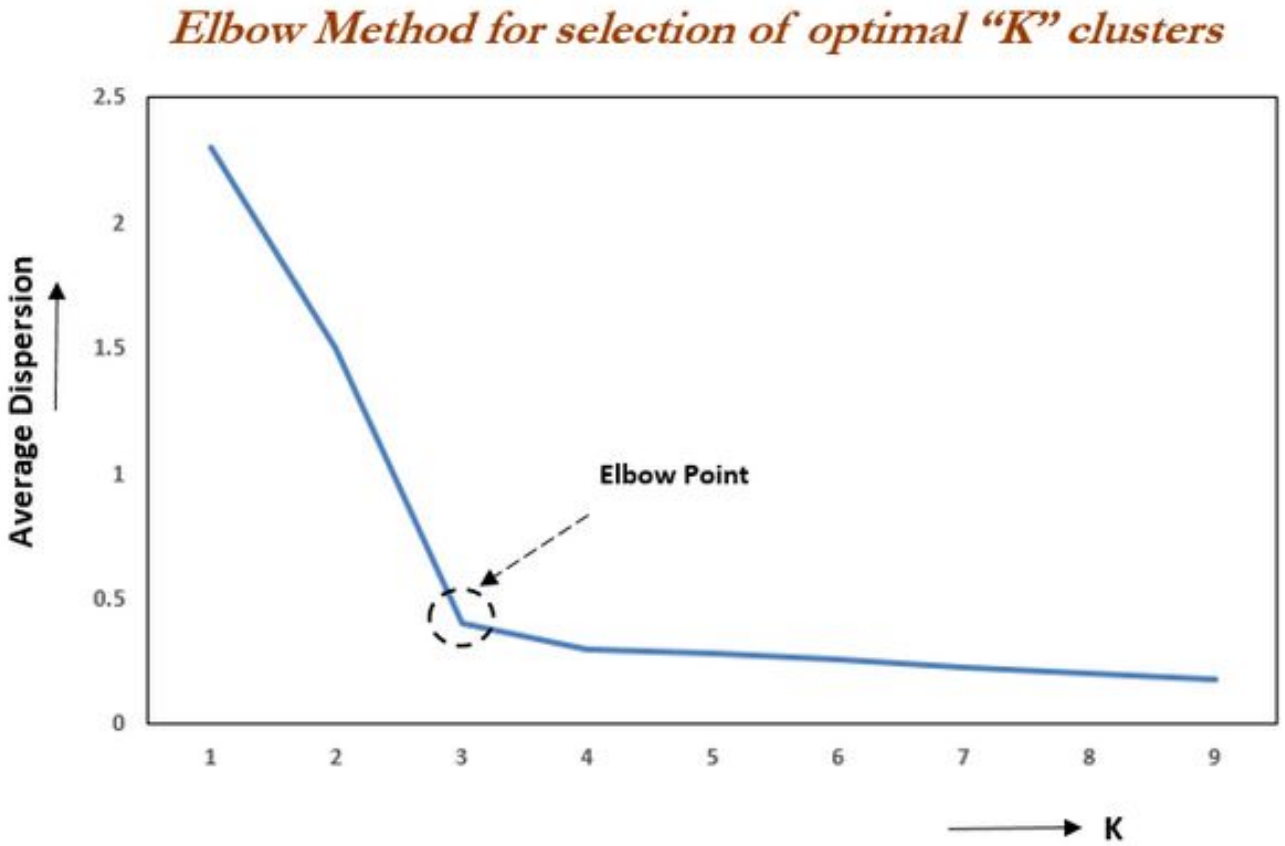
Escolhendo a quantidade de K (clusters) no algoritmo

O **Elbow Method** é uma das formas usadas para definição do parâmetro k.

Este método vai crescendo a quantidade de clusters a partir de 2 e analisando o resultado melhorado a cada incremento.

Quando o benefício parar de ser relevante (um salto entre uma quantidade de cluster e a próxima quantidade) ele entra em um modelo platô, no qual a diferença da distância é quase insignificante. É neste momento que entende-se que o algoritmo é relevante com aquela quantidade de K e então ele deve ser usado pra segmentar os dados do gráfico.

Escolhendo a quantidade de K (clusters) no algoritmo



K-Means Exemplo:

K = 2

Medida do Centroid: Distância Euclideana.

$$d(p,q) = \sqrt{\sum_{i=1}^{n}(p_i - q_i)^2}$$

Exemplo	Atributo 1	Atributo 2
1	**1.0**	**1.0**
2	1.5	2.0
3	3.0	4.0
4	**5.0**	**7.0**
5	3.5	5.0
6	4.5	5.0
7	3.5	4.5

	Exemplo	Centroide
Cluster 1	1	(1.0, 1.0)
Cluster 2	4	(5.0, 7.0)

K-Means Exemplo:

	Exemplos	Centroide
Cluster 1	1	(1.0, 1.0)
Cluster 2	4	(5.0, 7.0)

Exemplos	Distâncias	
	Cluster 1	Cluster 2
2	**1,11**	6,10
3	**3,6**	3,6
5	4,71	**2,5**
6	5,31	**2,06**
7	4,3	**2,9**

	Exemplos	Centroide
Cluster 1	1,2,3	(1.8, 2.3)
Cluster 2	4,5,6,7	(4.1, 5.4)

K-Means Exemplo:

	Exemplos	Centroide
Cluster 1	1,2,3	(1.8, 2.3)
Cluster 2	4,5,6,7	(4.1, 5.4)

Exemplos	Distâncias	
	Centroide Cluster 1	Centroide Cluster 2
1	**1.5**	5.4
2	**0.4**	4.3
3	2.1	**1.8**
4	5.7	**1.8**
5	3.2	**0.7**
6	3.8	**0.6**
7	2.8	**1.1**

	Exemplos	Centroide
Cluster 1	1,2	(1.3, 1.5)
Cluster 2	3,4,5,6,7	(3.9, 5.1)

K-Means

O K-means aprimora de forma iterativa seus resultados até alcançar um resultado final. O algoritmo recebe o número de clusters K e o conjunto de dados a ser analisado.

Em seguida são estabelecidas posições iniciais para os K centróides, que podem ser gerados ou selecionados aleatoriamente dentro do conjunto de dados. O algoritmo é iterado nos seguintes passos até que se estabilize:

- Associação de cada instância a um centróide - cada centróide define um cluster, então cada instância será associada a seu cluster mais semelhante (centróide mais próximo).
- A distância será calculada por alguma métrica de distância, em geral utiliza-se a distância euclidiana entre as duas instâncias;

K-Means

Implementação (em Python)

- Crie k centroids aleatórios no espaço de dados
- Repita n vezes o processo:
  - Calcule a distância entre cada ponto e cada centroid.
  - Atribua o ponto p ao centroid mais próximo (classificação).
  - Calcule novas medias para os centroids.
  - Reposicione os centroids.

Bibliotecas

```
import numpy as np # 1
import pandas as pd # 2
import matplotlib.pyplot as plt # 3
from sklearn.cluster import KMeans # 4
```

K-Means

Dataset (dados de entrada não-supervisionada)

```
headers = ['sepal length', 'sepal width', 'petal length', 'petal width', 'class']
dataset = pd.read_csv("./iris.data", encoding = "ISO-8859-1", decimal=",", header=None, names=headers)
```

sepal length	sepal width	petal length	petal width	class
5.1	3.5	1.4	0.2	Iris-setosa
4.9	3.0	1.4	0.2	Iris-setosa
4.7	3.2	1.3	0.2	Iris-setosa
...	...	...	...	...
7.0	3.2	4.7	1.4	Iris-versicolor
6.4	3.2	4.5	1.5	Iris-versicolor
6.9	3.1	4.9	1.5	Iris-versicolor

K-Means

Pré-processando os dados: Para garantirmos dados numéricos, vamos definir que as colunas sejam do tipo float:

```
for col in  dataset.columns[0:4]:
    dataset[col] = dataset[col].astype(float)
```

Agora deve ser realizada a divisão dos dados entre variáveis dependentes (X) e independente (y).

```
X = dataset.iloc[:, 0:4]
y = dataset.iloc[:, 4]
```

K-Means

Agora vamos aplicar o kmeans no conjunto de variáveis dependentes - ou seja, não estamos ‘contando’ ao algoritmo quais são as classes de cada instância.

Definimos o número de clusters - k - como 3, uma situação ideal. Existem técnicas para encontrar o melhor k mas isso depende do problema.

```
kmeans = KMeans(n_clusters=3, random_state=0).fit(X)
X_clustered = kmeans.fit_predict(X)
```

K-Means

```
results = dataset[['class']].copy()
results['clusterNumber'] = X_clustered
results
```

class	clusterNumber
Iris-setosa	0
Iris-setosa	0
Iris-setosa	0
...	...
Iris-versicolor	1
Iris-versicolor	1
Iris-versicolor	1
...	...
Iris-virginica	2
Iris-virginica	2
Iris-virginica	2

K-Means

Podemos agora testar uma abordagem mais gráfica. Primeiramente vamos definir cores para os diferentes tipos de clusters e ‘pintar’ pontos em clusteres diferentes de cores diferentes:

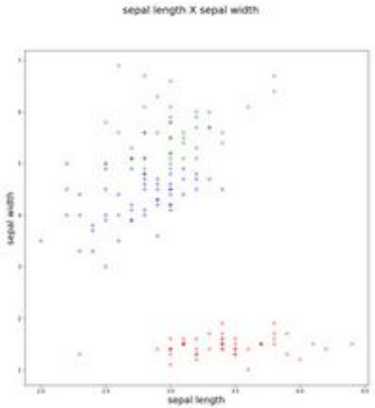
```
LABEL_COLOR_MAP = {0 : 'red', 1 : 'blue', 2: 'green'}
label_color = [LABEL_COLOR_MAP[l] for l in X_clustered]
```

```
c1 = 0 # valor do índice da coluna, pode ser 0, 1 ou 2
c2 = 1
labels = ['sepal length', 'sepal width', 'petal length']
c1label = labels[c1]
c2label = labels[c12]
title = xlabel + ' x ' + ylabel
```

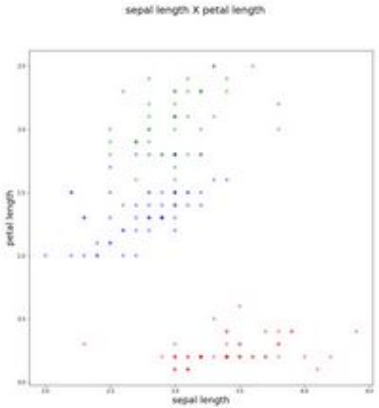
K-Means

```
plt.figure(figsize = (12,12))
plt.scatter(X.iloc[:, c1],X.iloc[:, c2], c=label_color, alpha=0.3)
plt.xlabel(c1label, fontsize=18)
plt.ylabel(c2label, fontsize=18)
plt.suptitle(title, fontsize=20)
plt.savefig(title + '.jpg')
plt.show()
```

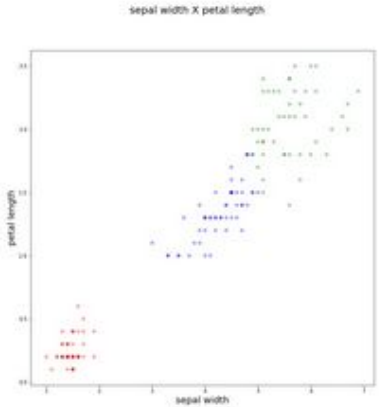
sepal length X sepal width



sepal length X petal length



sepal width X petal length



K-Means

Exercício

Crie um programa que receba valores de coordenada x e y para 5 pontos. Calcule a posição do centroid para estes pontos.