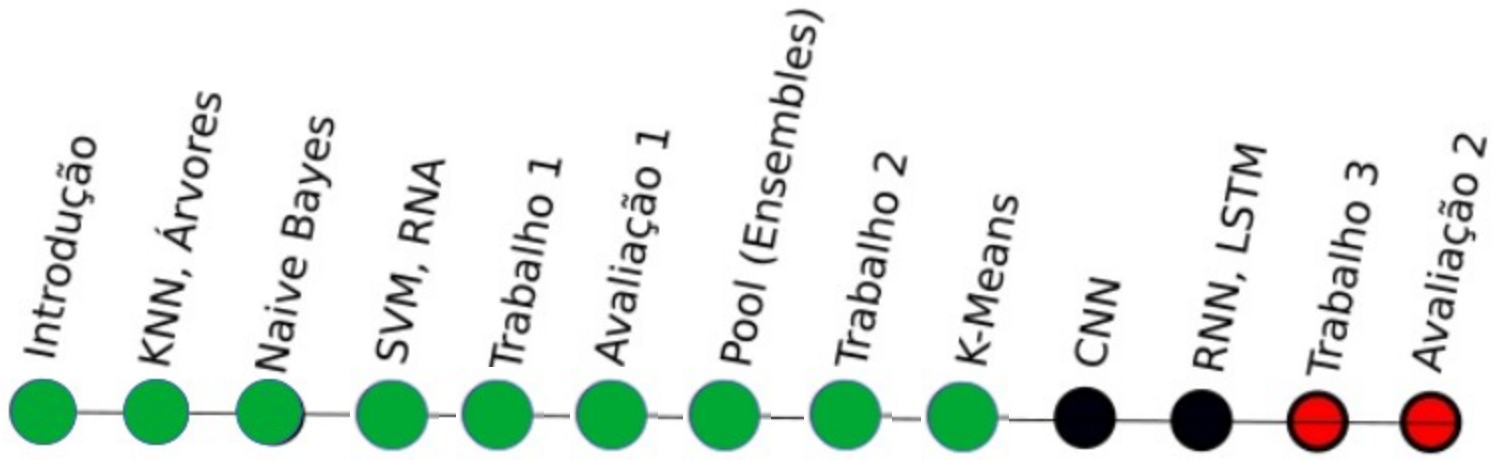
Aprendizado Não-Supervisionado
Agrupamentos

Prof. André Gustavo Hochuli

gustavo.hochuli@pucpr.br
aghochuli@ppgia.pucpr.br
github.com/andrehochuli/teaching

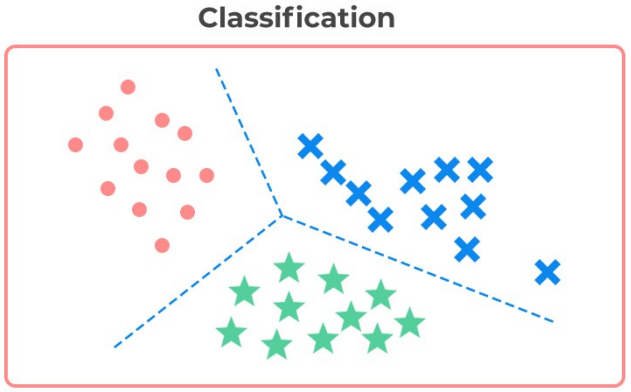
Plano de Aula

- Discussões Iniciais
- Supervisionado vs Não Supevisionado
- Agrupamentos
  - K-Means
- Métricas
- Exercícios



Discussões Iniciais

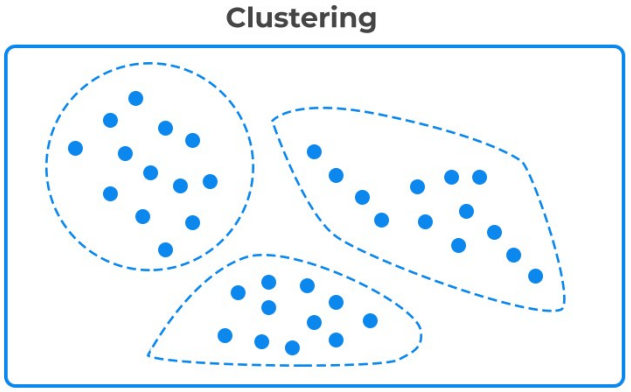
- Supervisionado vs Não Supervisionado



Supervised learning

X_1	X_2	X_p	Y

Target



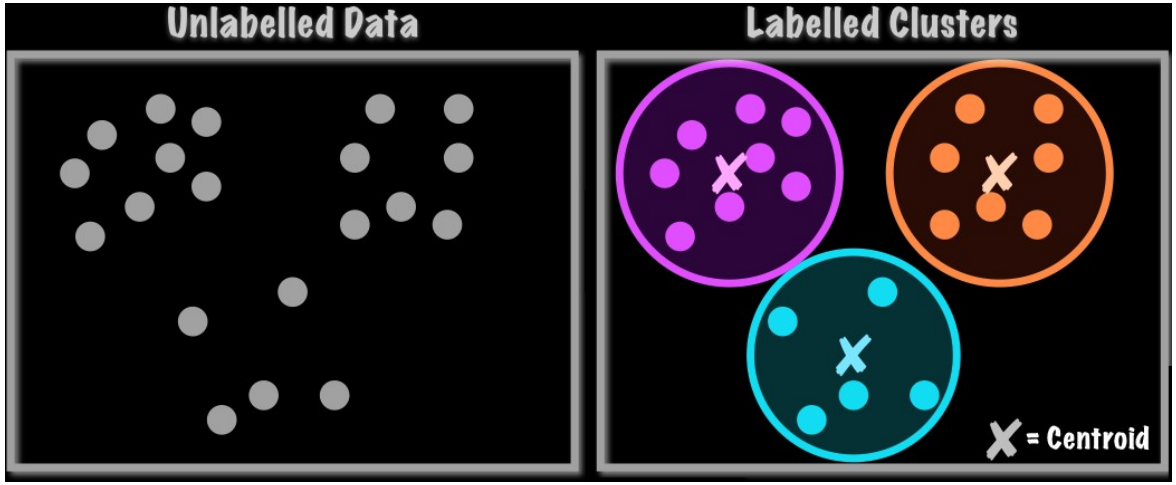
Unsupervised learning

X_1	X_2	X_p	Y

No Target

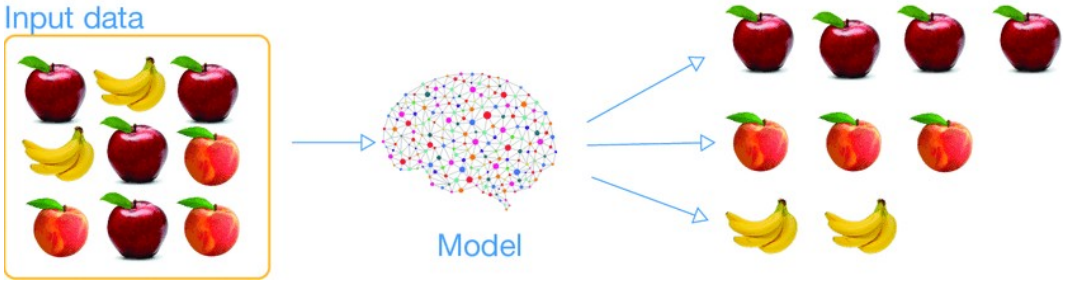
Aprendizado Não Supervisado

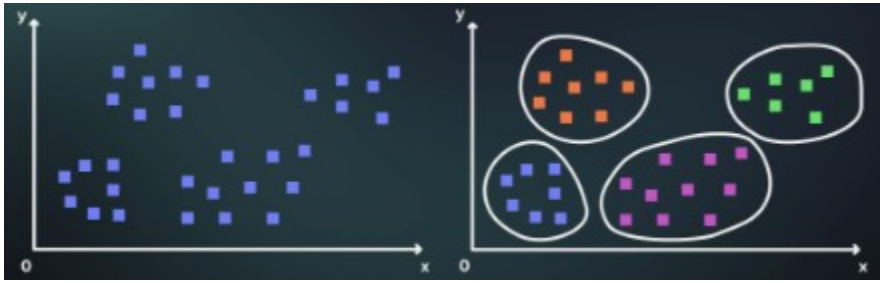
- Rotular dados é custoso
- Agrupamento (Clustering):
  - Encontrar relacionamentos intrínseco dos atributos
  - Métrica de agrupamento
  - Avaliar os grupos (clusters)



Clustering

- Exemplos:
  - Segmentação de consumidores
  - Classificação de Anomalias
    - Defeitos, Doenças, Fraudes
  - Classificação de espécies
  - Organização de Documentos
  - Mineração de dados
  - Redução de Dimensionalidade

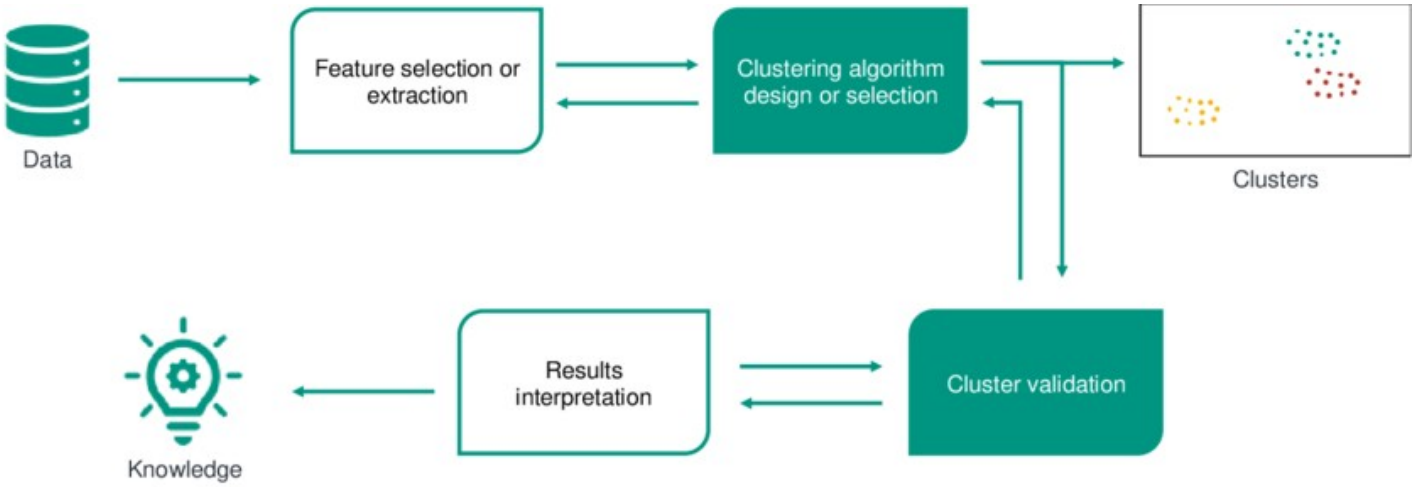




  - Casos reais:
    - Determinação de Recall baseado em histórico de reparos
    - Características de produtos não vendidos (i.e 220v em cidades 110v)

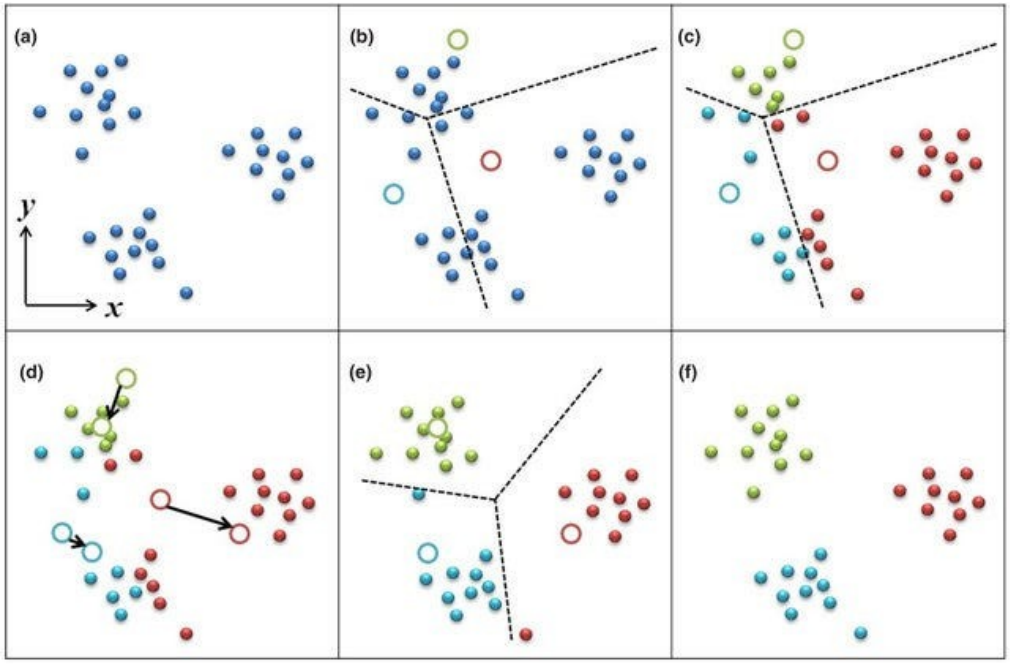
Clustering

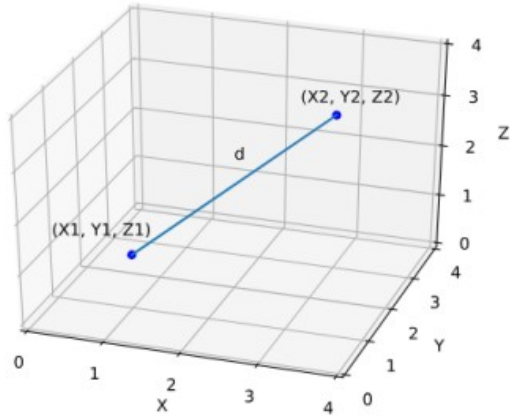
- Workflow:



K-Means

- Define grupos aproximando dados próximos aos ‘centroides’
- Cada iteração ajusta os centroides e os dados percentes a cada grupo
- Ao fim, o centroides representam o ponto médio de cada cluster



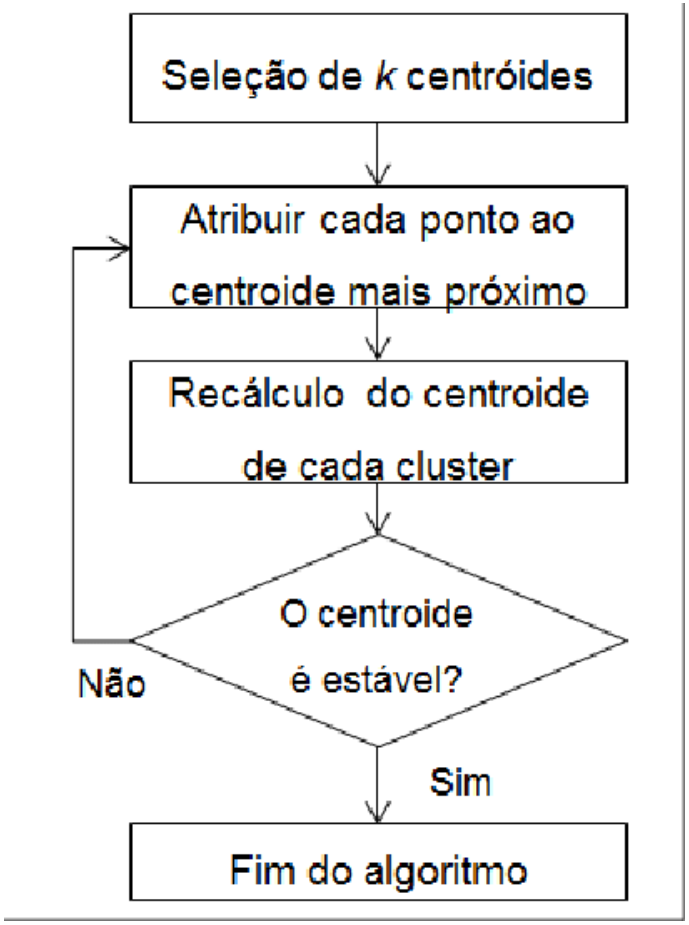


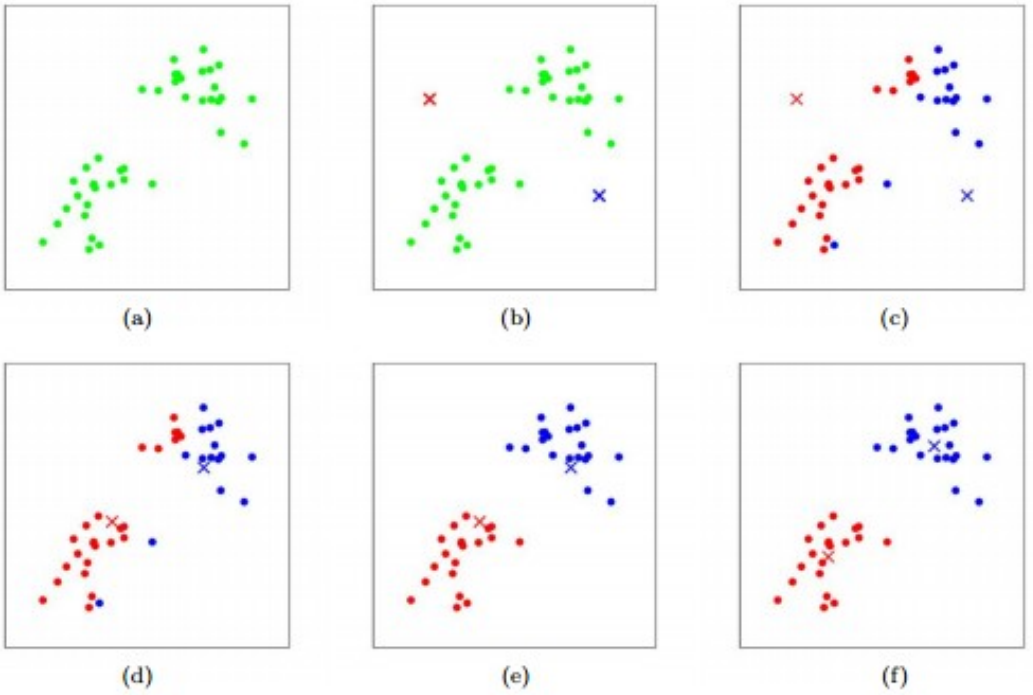
$$d = \sqrt{(x_1 - x_0)^2 + (y_1 - y_0)^2 + (z_1 - z_0)^2}$$

$$d(x,\ y)\ = \sqrt{\sum_{i=1}^{n}(y_i - x_i)^2}$$

K-Means

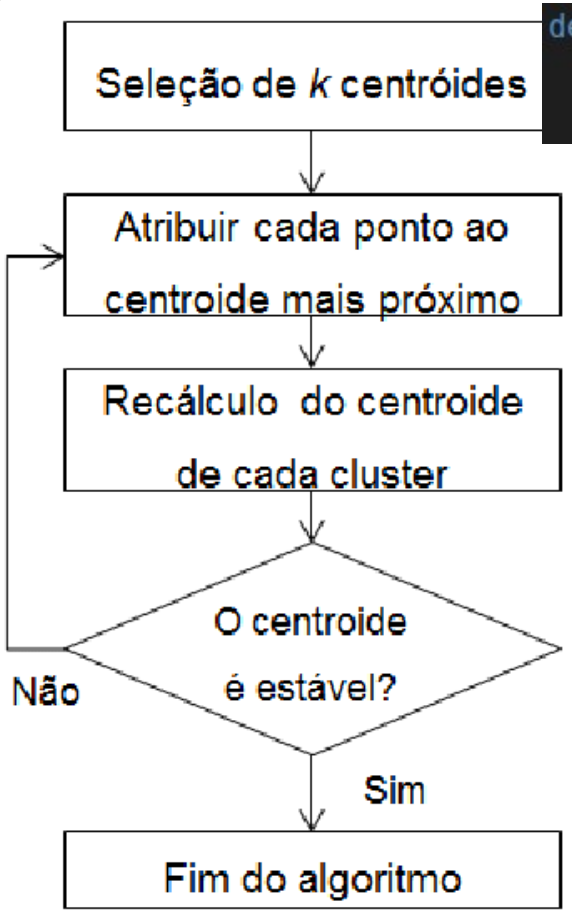
- Algoritmo:





K-Means

- Algoritmo:




```python
def fit(self, X, plot=False):
    # Inicializa os centros dos clusters aleatoriamente
    self.centers = X[np.random.choice(len(X), self.k, replace=False), :]
```

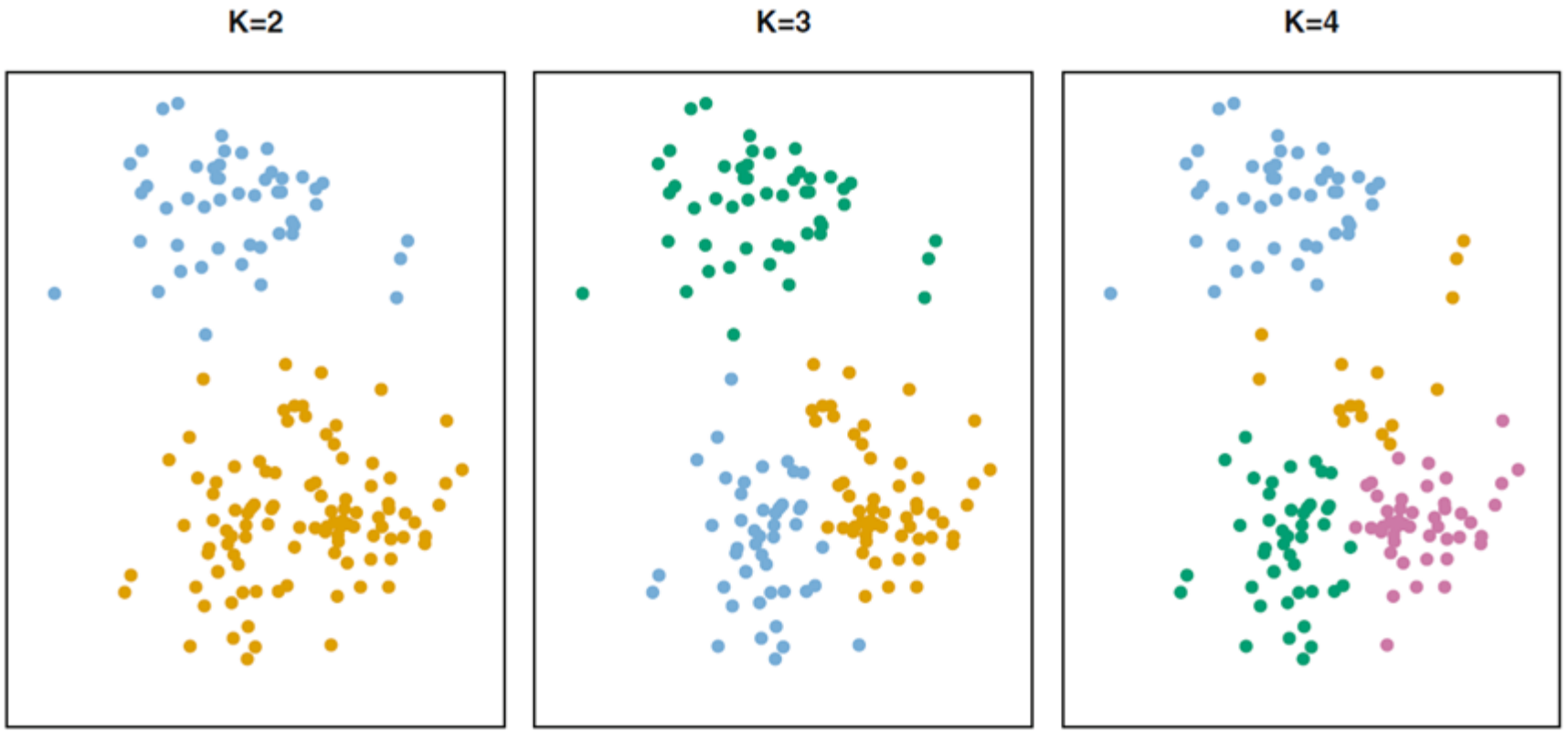
```python
# Executa o algoritmo K-Means
for i in range(self.max_iters):
    # Atribui cada ponto ao cluster mais próximo
    distances = np.sqrt(np.sum((X - self.centers[:, np.newaxis]) ** 2, axis=2))
    labels = np.argmin(distances, axis=0)

# Verifica se o algoritmo convergiu
old_centers = self.centers.copy()
for k in range(self.k):
    self.centers[k] = np.mean(X[labels == k], axis=0)
```

```python
if np.allclose(old_centers, self.centers, rtol=0, atol=self.tol):
    print(f"Algoritmo convergiu após {i+1} iterações.")
    break
```

K-Means

- E como avaliar os clusteres ? Como determinar K?



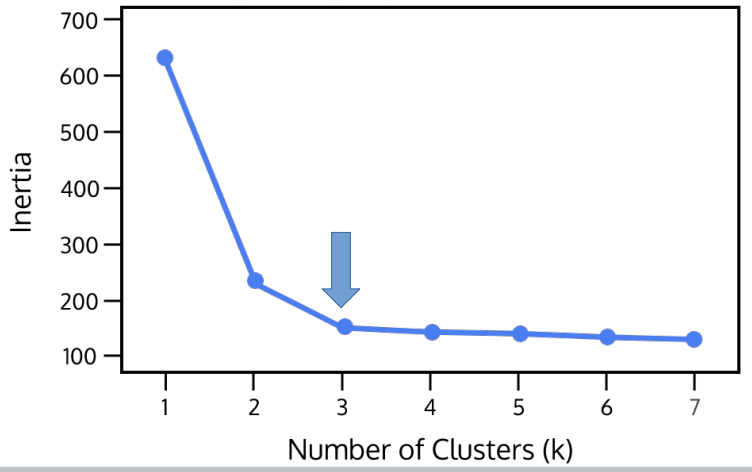
K-Means

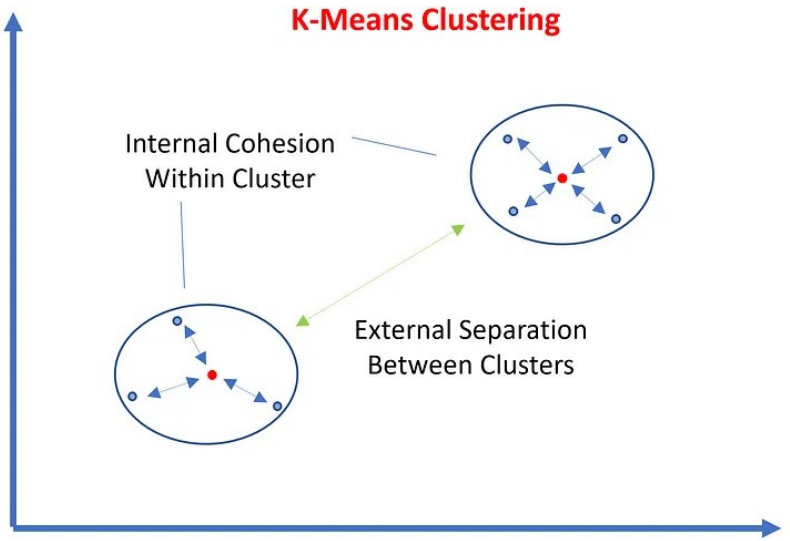
- Inertia (Soma dos erros quadrados)

$$\sum_{i=1}^{N}(x_i - C_k)^2$$

- Elbow Method (Inertia global)

$$\sum_{j=1}^{k}\sum_{i=1}^{N}(x_i - C_k)^2$$



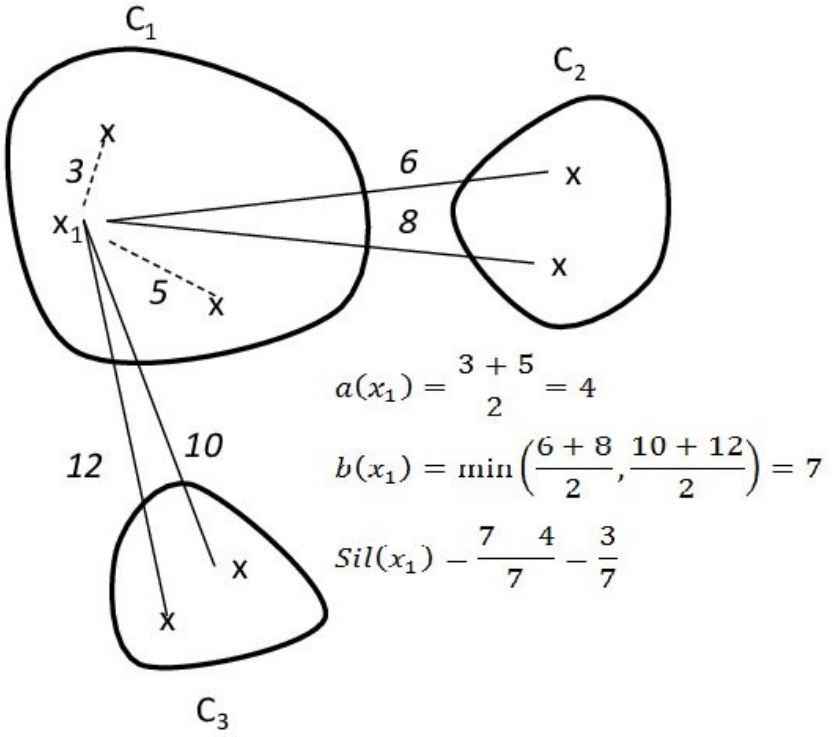


K-Means

- Silhueta (Silhouette)
  - Medida por instâncias [-1,1]
  - Coesão (a)
  - Separação (b)

$$s_i = \frac{b_i - a_i}{\max(a_i, b_i)}$$

  - [-1,+1]:
    - -1: Não coerente
    - +1: Coerente
  -



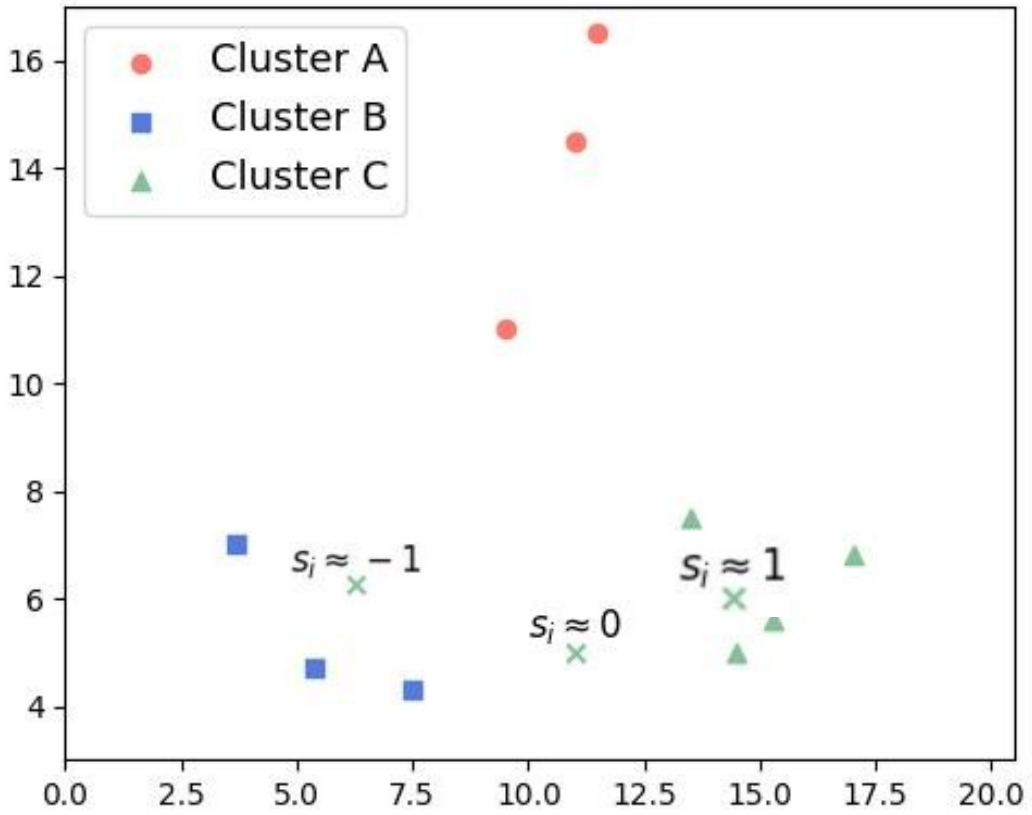
$$a(x_1) = \frac{3+5}{2} = 4$$

$$b(x_1) = \min\left(\frac{6+8}{2}, \frac{10+12}{2}\right) = 7$$

$$Sil(x_1) - \frac{7}{7} - \frac{4}{7} - \frac{3}{7}$$

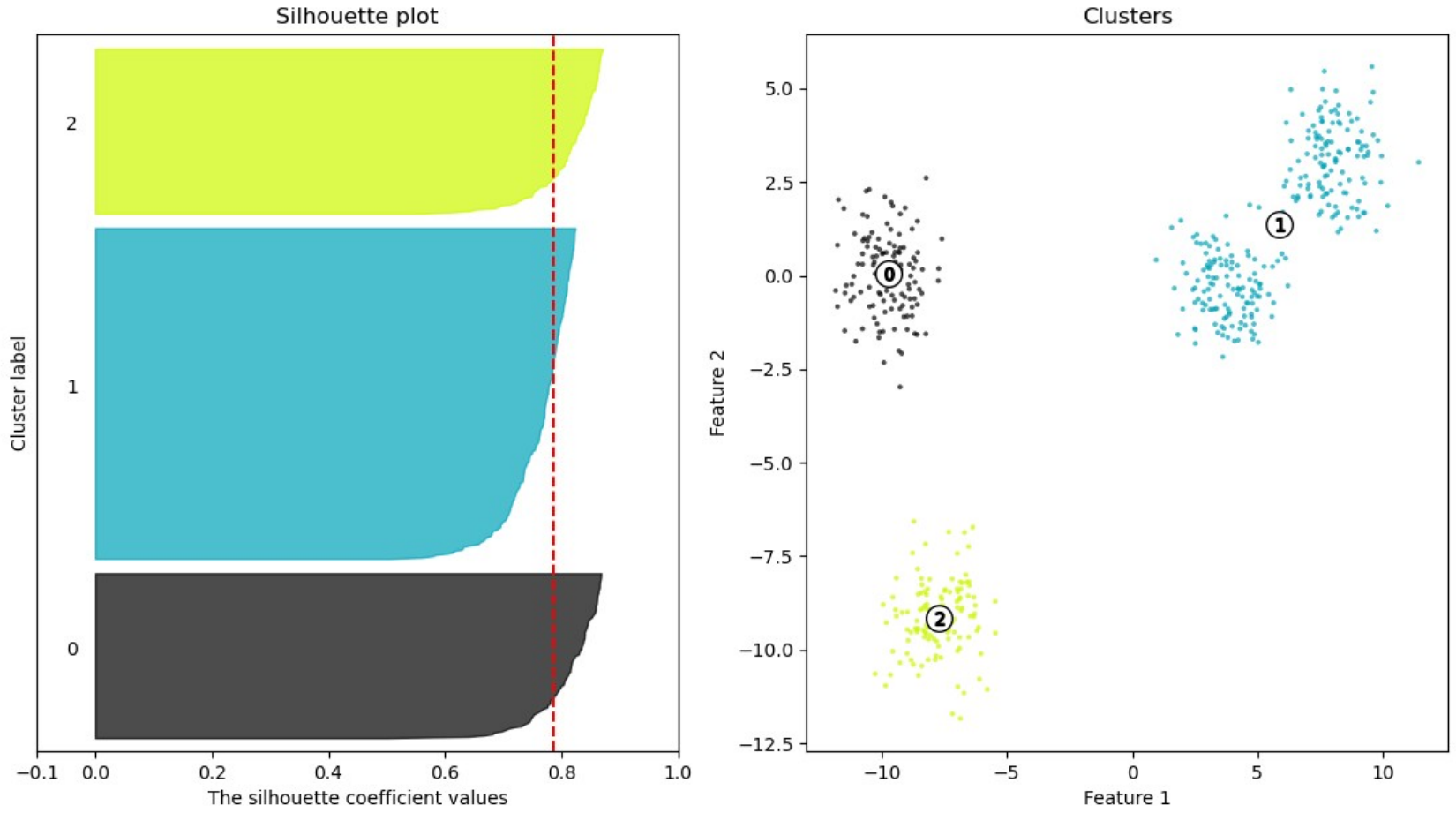
K-Means

- Silhueta (Silhouete)



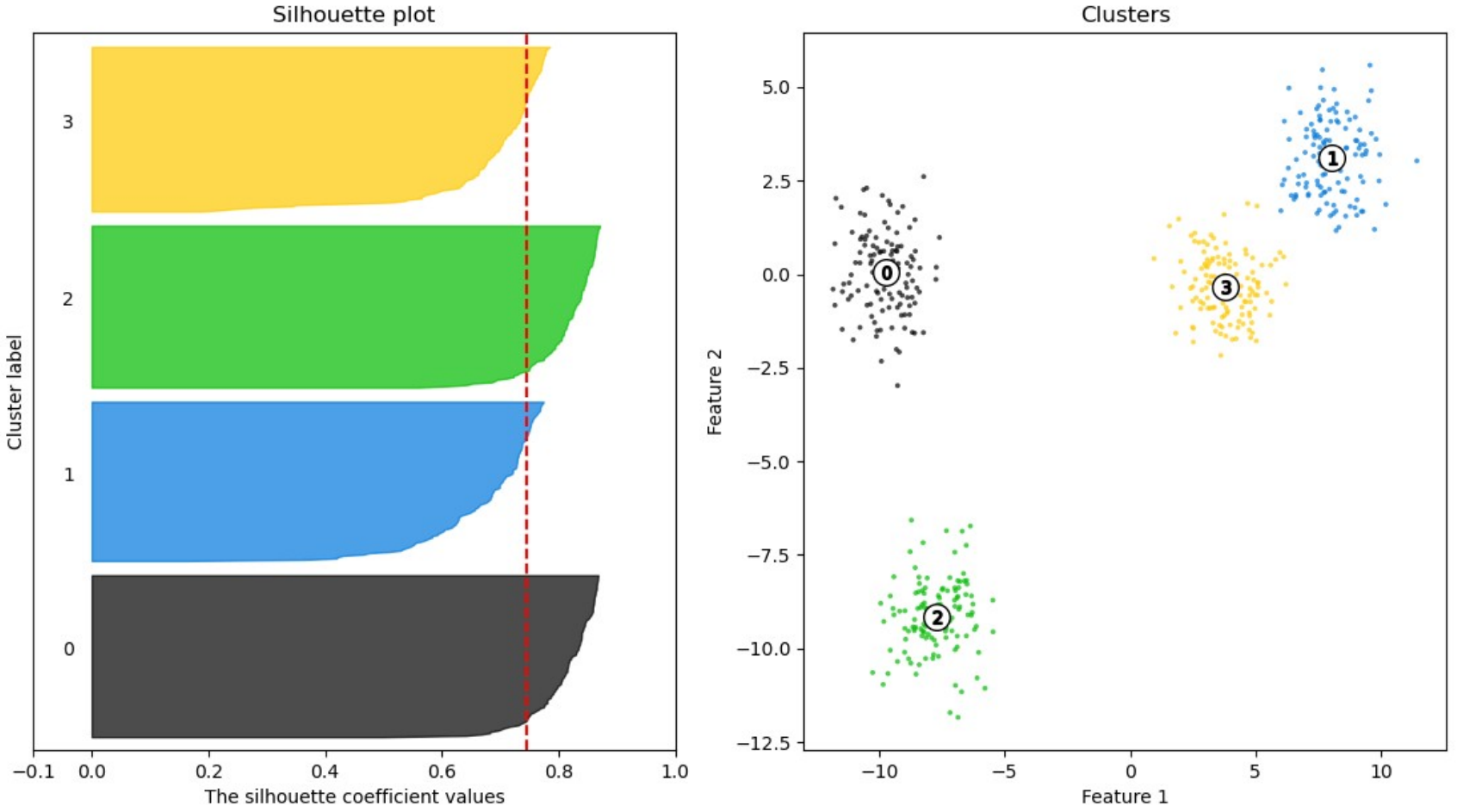
K-Means

- Silhueta (Silhouete)



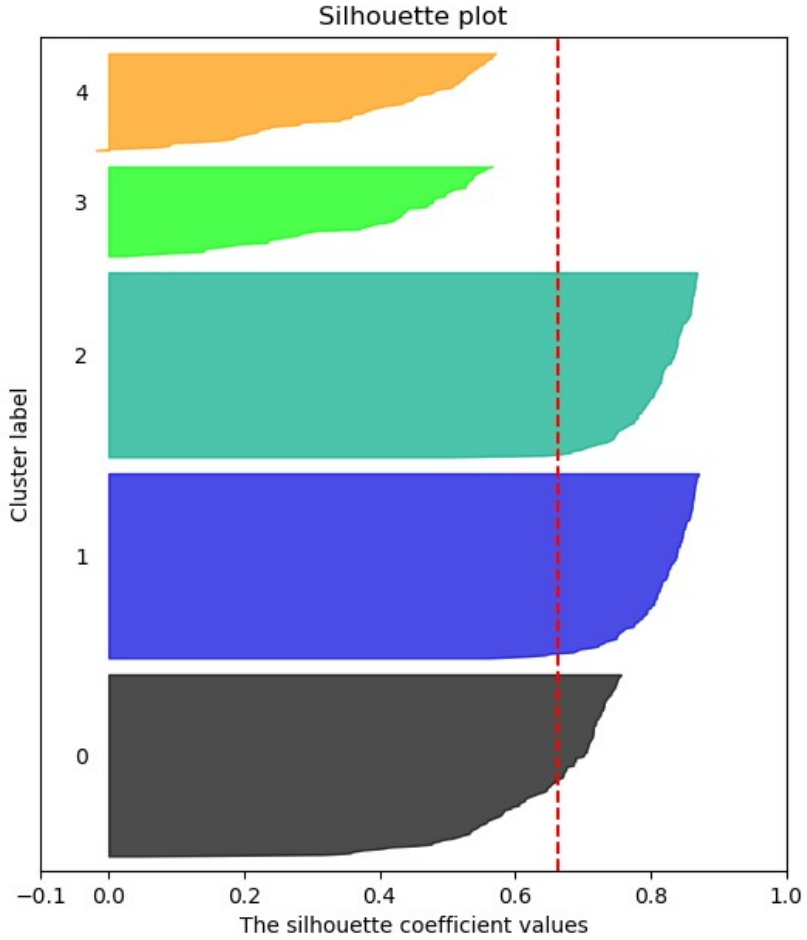
K-Means

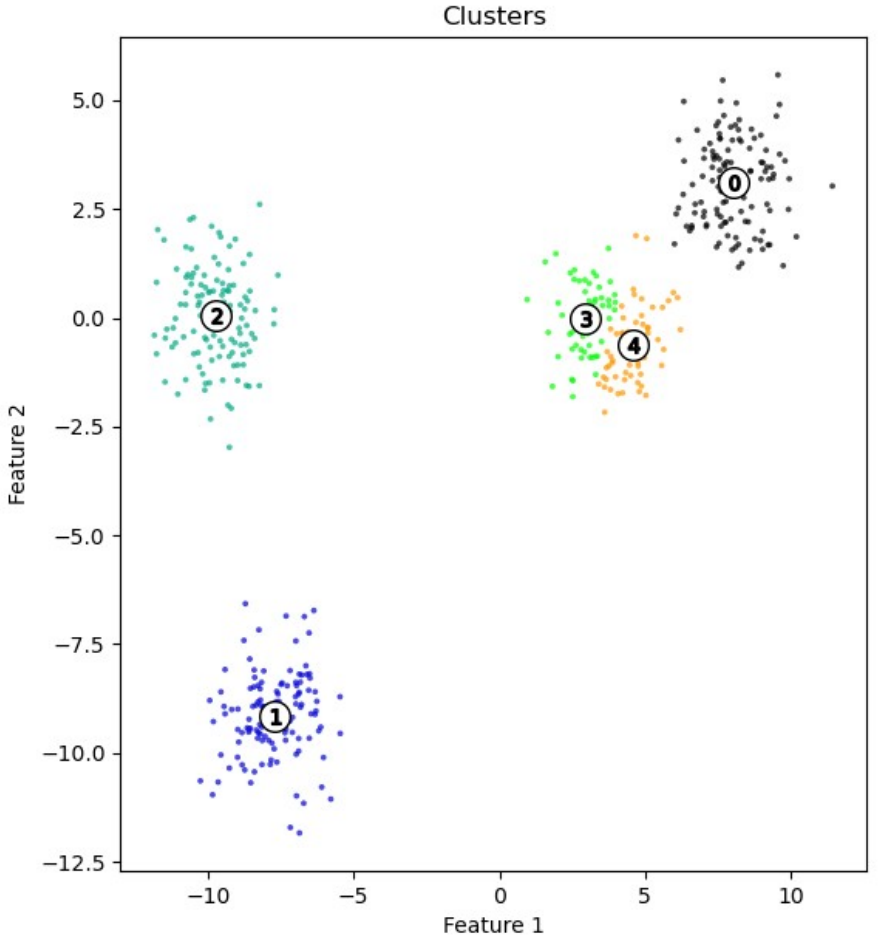
- Silhueta (Silhouete)



K-Means

- Silhueta (Silhouete)





K-Means

Lets Code!

Link → Tópico 03 - Aprendizado - Não Supervisionado - Kmeans