Relatório: implementando o algoritmo de *fuzzy c-means* em Python

Juan Belieni

8 de novembro de 2022

1 Algoritmo de *fuzzy c-means*

O algoritmo de *fuzzy c-means*[1] é um algoritmo baseado no tradicional algoritmo de *k-means*. Nele, cada ponto pode estar associado a mais de um cluster, com uma certa probabilidade, que deve somar 1.

No algoritmo, uma matriz de particionamento $U \in [0,1]^{C \times P}$ é criada, onde C é o número de clusters e P é o número de pontos. Cada elemento u_{ij} da matriz representa a probabilidade do i-ésimo ponto pertencer ao j-ésimo cluster. Também é definido um vetor $\mathbf{v} \in \mathbb{R}^{C \times M}$, que representa os centróides dos clusters, onde M é a dimensão dos pontos. A clusterização é controlada por um parâmetro $m \in [1, \infty)$, chamado de coeficiente de fuzzificação.

1.1 Função objetivo

O *fuzzy c-means* é um algoritmo iterativo, e busca minimizar a função objetivo

$$J = \sum_{i=1}^{P} \sum_{j=1}^{C} u_{ij}^m \|\mathbf{z}_i - \mathbf{v}_j\|^2, \tag{1}$$

onde $\mathbf{z}_i$ é o i-ésimo ponto e $\mathbf{v}_j$ é o centro do j-ésimo cluster. Para que essa minimização seja feita, o algoritmo define, a cada iteração, novos valores para a matriz de particionamento U e para os centros dos clusters $\mathbf{v}$ da seguinte forma:

$$u_{ij} = \frac{1}{\sum_{k=1}^{C} \left(\frac{\|\mathbf{z}_i - \mathbf{v}_j\|}{\|\mathbf{z}_i - \mathbf{v}_k\|} \right)^{\frac{2}{m-1}}}. \tag{2}$$

e

$$\mathbf{v}_j = \frac{\sum_{i=1}^{P} u_{ij}^m \mathbf{z}_i}{\sum_{i=1}^{P} u_{ij}^m}. \tag{3}$$

A cada iteração, o algoritmo verifica se $J^{(k+1)} - J^k < \varepsilon$, onde ε é um parâmetro de tolerância.

2 Implementação

A implementação do algoritmo de *fuzzy c-means* foi feita em Python, utilizando a biblioteca *numpy* para a manipulação de matrizes e vetores.

2.1 Métodos auxiliares

Primeiramente, foi definida a função auxiliar `J`, que calcula o valor da função objetivo J:

```
def J(U: np.ndarray, v: np.ndarray, data: np.ndarray, m: int = 2) -> float:
    return np.sum(
        U ** m,
        * np.linalg.norm(data[:, np.newaxis, :] - v[np.newaxis, :, :], axis=2) ** 2
    )
```

2.2 Método principal

A função principal do algoritmo é a `fuzzy_c_means`, que recebe como parâmetros a matriz de dados, o número de clusters C, o coeficiente de fuzzificação m e a tolerância ε:

```
def fuzzy_c_means(
    data: np.ndarray, C: int, m: int = 2, eps: float = 1e-4
) -> np.ndarray:
    U = np.random.rand(data.shape[0], C)
    U /= U.sum(axis=1, keepdims=True)

    v = np.random.rand(C, data.shape[1]) * data.max()

    J1 = J(U, v, data, m)
    J2 = np.inf

    while np.abs(J1 - J2) > eps:
        for i in range(data.shape[0]):
            for j in range(C):
                d1 = np.linalg.norm(data[i] - v[j])
                d2s = np.linalg.norm(data[i] - v, axis=1)
                U[i, j] = 1 / np.sum(np.power(d1 / d2s, 2 / (m - 1)))

        for j in range(n_clusters):
            s1 = np.sum(U[:, j, np.newaxis] ** m * data, axis=0)
            s2 = np.sum(U[:, j] ** m)
            v[j] = s1 / s2

        J2 = J1
        J1 = J(U, v, data, m)

    return U, v
```

3 Resultados

A implementação foi testada com a base de dados *World Development Indicators*, presente no *Kaggle*[2]. Nela, temos dados de diversos indicadores de países ao longo dos anos. Para a clusterização, foram utilizados alguns indicadores econômicos.

3.1 Primeira clusterização

A primeira clusterização foi feita utilizando dois indicadores: PIB per capita e taxa de crescimento do PIB. Os resultados obtidos são mostrados na figura 1.

Figura 1: Gráfico de dispersão da primeira clusterização

Além disso, foi possível obter um dendrograma (figura 2), que mostra a hierarquia dos clusters, onde a proximidade entre cada país foi calculada utilizando a média do grau de pertencimento de cada país aos clusters.

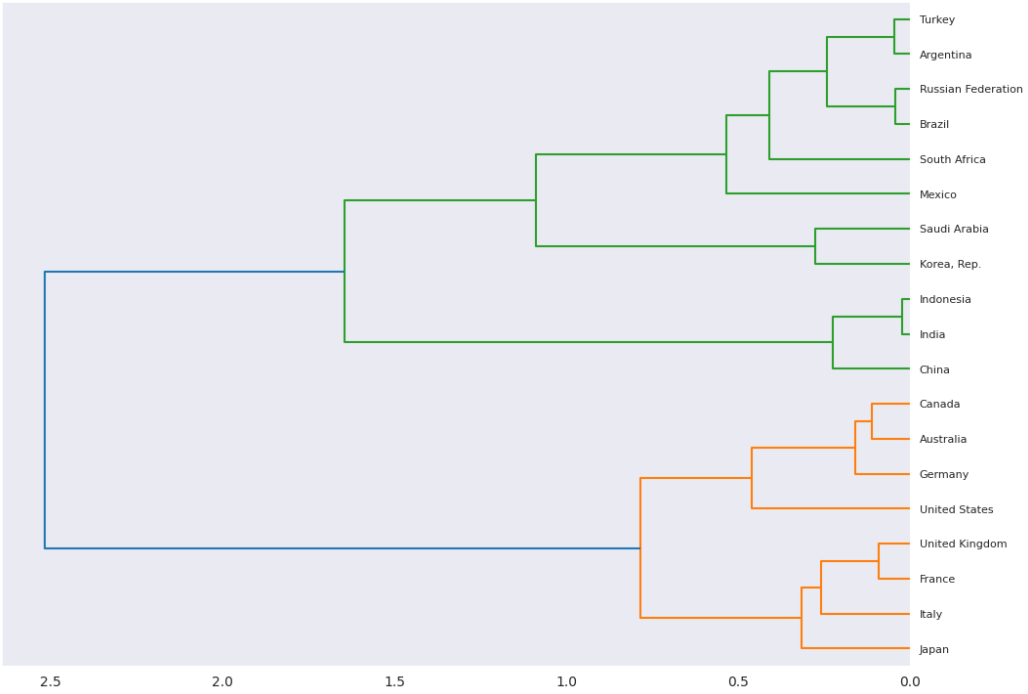

Figura 2: Dendrograma da primeira clusterização

3.2 Segunda clusterização

A segunda clusterização foi feita utilizando vários indicadores: PIB per capita, crescimento econômico, inflação, desemprego e crescimento populacional. Com isso, um segundo dendrograma foi obtido (figura 3).

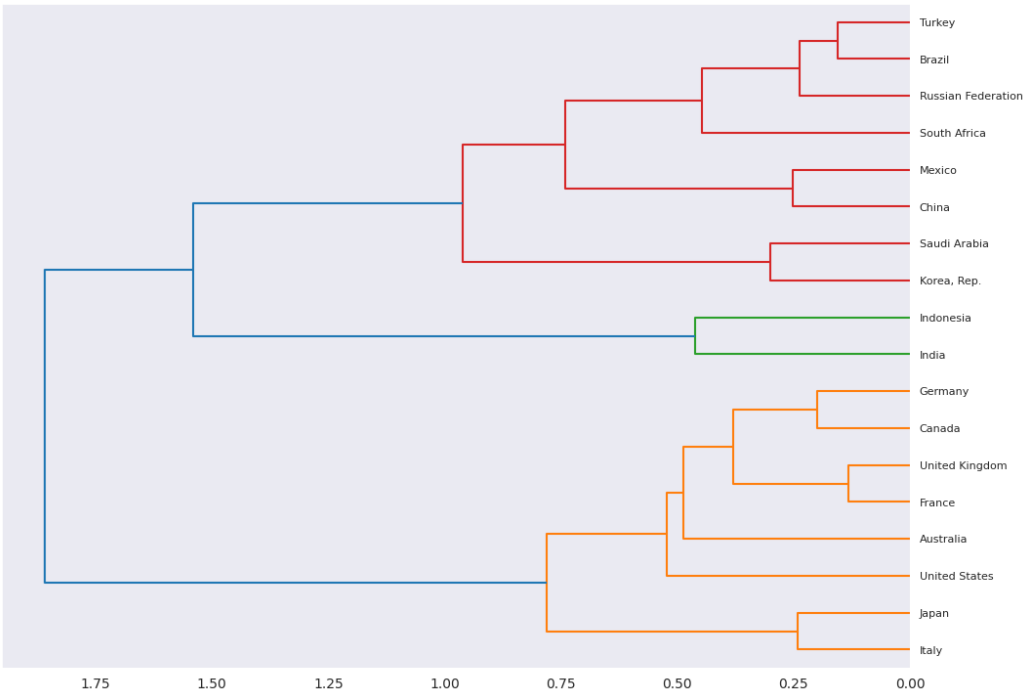

Figura 3: Dendrograma da segunda clusterização

Referências

[1] Jakub Bartak e Agnieszka Jastrzębska. "Mining patterns of transitional growth using multivariate concept-based models". Em: *Quality & Quantity* 56.6 (2022), 4395–4419. DOI: 10.1007/s11135-022-01318-8.

[2] Kaggle. *World development indicators*. 2017. URL: https://www.kaggle.com/datasets/kaggle/world-development-indicators.