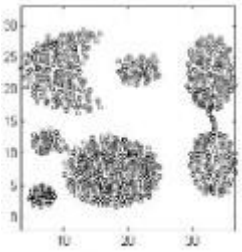
IA – 1° SEMESTRE DE 2022

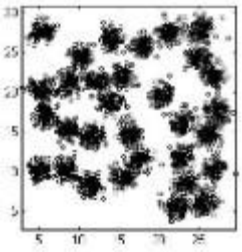
11. EXERCÍCIO PRÁTICO – APRENDIZADO NÃO SUPERVISIONADO

Nome: Luiz Gustavo Alves Assis da Silva
RA: 149115

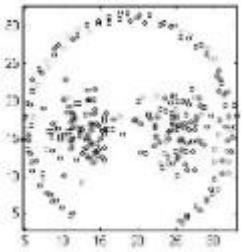
1) Execute os algoritmos *k-means* e *single linkage* nos datasets a seguir. Qual o melhor agrupamento que vc obteve (pode plotar os grupos por cores para facilitar a identificação). Pode usar bibliotecas ou implementar seu próprio código.



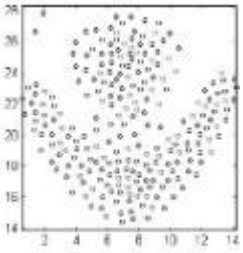
Aggregation http://cs.joensuu.fi/sipu/datasets/Aggregation.txt



D31 http://cs.joensuu.fi/sipu/datasets/D31.txt



Path-based1 http://cs.joensuu.fi/sipu/datasets/pathbased.txt



Flame http://cs.joensuu.fi/sipu/datasets/flame.txt

```
'''


#
#   AULA 11
#   APRENDIZADO NÃO SUPERVISIONADO
#   LINGUAGEM DE PROGRAMAÇÃO: Python
#
#   NOME: LUIZ GUSTAVO ALVES ASSIS DA SILVA
#   RA: 149115
#

'''

# BIBLIOTECAS UTILIZADAS

import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.cluster import KMeans
from sklearn.cluster import AgglomerativeClustering
```

Temos um programa que implementa os seguintes algoritmos: K-Means, Single Linkage.

O objetivo deste programa é utilizar os datasets citados e realizar uma análise de acordo com a plotagem de grupos por cores dos algoritmos.

```
def menu():
    valid_input = ['1', '2', '3', '4']

    while True:
        print("Selecione a opção: 1 - Aggregation | 2 - D31 | 3 - Path Based | 4 - Flame | 5 - Sair do programa")
        input_op = input()

        if (input_op == "1"): # LER DATASET: AGGREGATION
            data = pd.read_csv("Aggregation.txt", sep = "\t", header = None, names=["Atr1", "Atr2", "Classe"])

        elif (input_op == "2"): # LER DATASET: D31
            data = pd.read_csv("D31.txt", sep = "\t", header = None, names=["Atr1", "Atr2", "Classe"])

        elif (input_op == "3"): # LER DATASET: PATHBASED
            data = pd.read_csv("pathbased.txt", sep = "\t", header = None, names=["Atr1", "Atr2", "Classe"])

        elif (input_op == "4"): # LER DATASET: FLAME
            data = pd.read_csv("flame.txt", sep = "\t", header = None, names=["Atr1", "Atr2", "Classe"])

        if (input_op in valid_input):
            executar_algoritmos(data)
        else:
            return

if __name__ == '__main__':
    menu()
```

Na função main, é feita uma chamada de função *menu* com objetivo de implementar uma interface para auxiliar o usuário ler os datasets e observar os gráficos de acordo com a opção escolhida. Caso o usuário deseja sair do programa, basta digitar qualquer outra tecla que não esteja listada no vetor *valid_input*.

```python
def executar_algoritmos(data):

    k = len(data['Classe'].unique())

    labels_k_means, centroids = K_means(data, k)
    labels_single = single_linkage(data, k)
    plotar_graficos(data, labels_k_means, labels_single, centroids)

def K_means(data, k):

    kmeans = KMeans(n_clusters = k, random_state = 0)
    labels = kmeans.fit_predict(data)
    centroids = kmeans.cluster_centers_
    return labels, centroids

def single_linkage(data, k):

    sk = AgglomerativeClustering(n_clusters = k, linkage = 'single')
    labels = sk.fit_predict(data)
    return labels
```

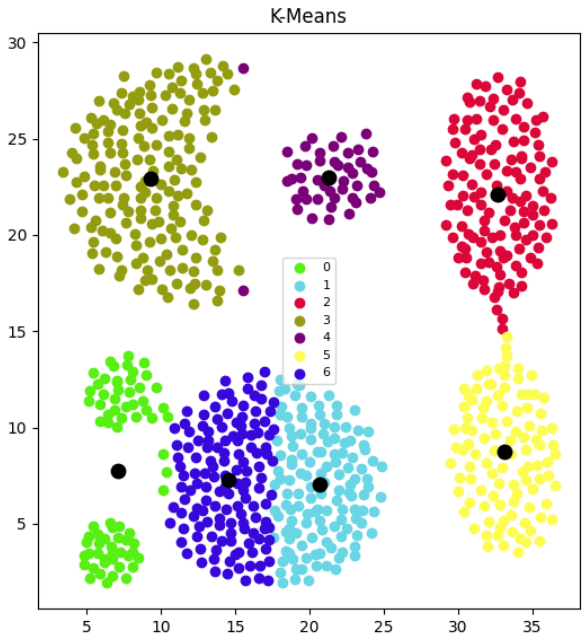
Em uma primeira observação, é criada uma variável *k* que recebe o número total de classes no dataset escolhido pelo usuário, esta variável será utilizada como parâmetro para indicar a quantidade máxima de clusters nos algoritmos K-Means e Single Linkage.

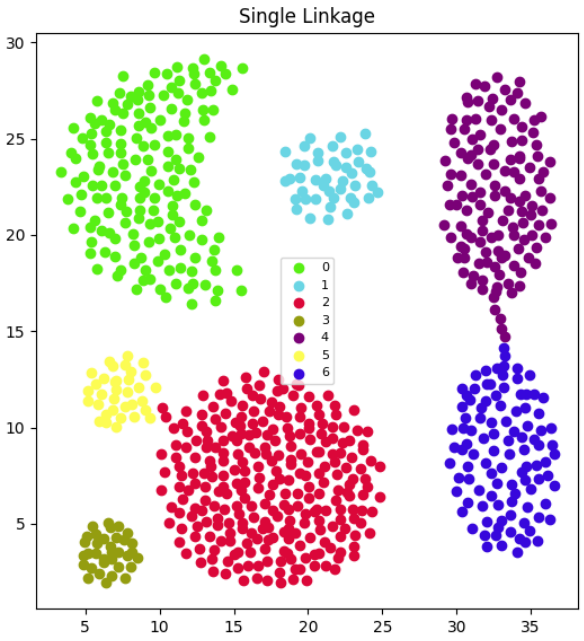
Em seguida, é feita as chamadas de funções para *K_Means* (que executa o algoritmo K-Means e retorna os rótulos "labels" e as centroides) e *Single_linkage (*que executa o algoritmo Single Linkage e retorna os rótulos "labels"). Após isso, é feita a chamada de função de plotagem de gráficos passando os dois rótulos obtidos nas funções anteriores como argumento.

```python
def plotar_graficos(data, labels_k_means, labels_single, centroids):

    colors = ['#58EF14', '#6AD5E4', '#DB0739', '#949D10', '#7A0177', '#FCFC53', '#3707DC', '#2979D9', '#69D926',
              '#83FAD2', '#922E95', '#5257FD', '#0AA4FB', '#E38B52', '#F0E494', '#E70FD4', '#15C6E6', '#312D9C',
              '#A72F1D', '#B45BC7', '#010C68', '#35FE8F', '#F3F63D', '#B388E3', '#A97DE9', '#E9211F', '#9204EB',
              '#F4F943', '#9694C5', '#BFDA1C', '#2E2724', '#220B59', '#C8D630', '#E20ED9']

    u_labels = np.unique(labels_k_means)
    figure, axis = plt.subplots(1, 2)

    for i, color in zip(u_labels, colors):
        label_k = data[labels_k_means == i]
        label_single = data[labels_single == i]

        axis[0].scatter(label_k.iloc[:, 0] , label_k.iloc[:, 1], label = i, color = color)
        axis[1].scatter(label_single.iloc[:, 0] , label_single.iloc[:, 1], label = i, color = color)

    axis[0].scatter(centroids[:, 0] , centroids[:, 1] , s = 80, color = 'k')
    axis[0].legend(fontsize = 8)
    axis[1].legend(fontsize = 8)
    axis[0].set_title("K-Means")
    axis[1].set_title("Single Linkage")

    plt.show()
```

Por fim, a função *plotar_graficos*, como o nome diz, realiza a plotagem de grupos por cores dos algoritmos K-Means e Single Linkage. Ainda, é feita a plotagem das centroides do algoritmo K-Means para facilitar a interpretação da análise.

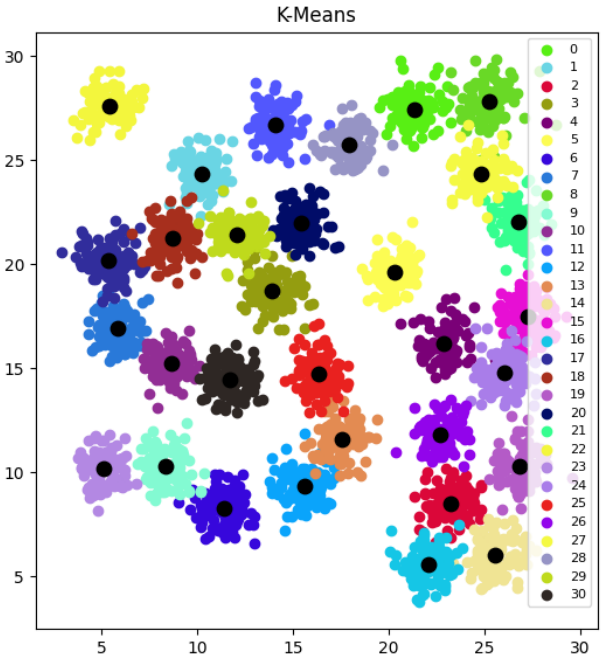
DATASET – AGGREGATION:

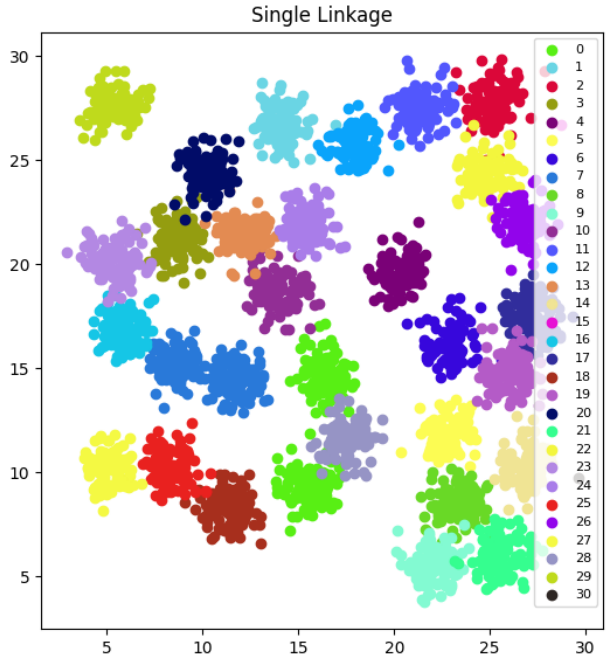




Para o dataset *Aggregation*, é simples identificar que o melhor agrupamento é dado pelo algoritmo Single Linkage, uma vez que o algoritmo K-Means é sensível a *outliers* (conjunto de dados que difere significativamente das outras observções).

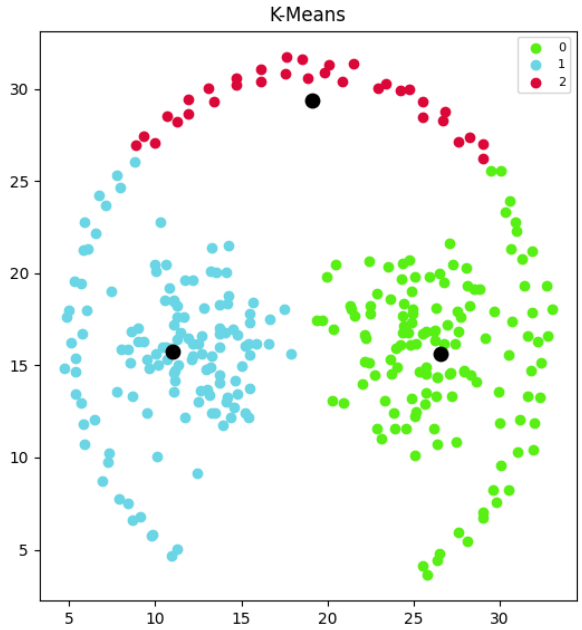
DATASET – D31:

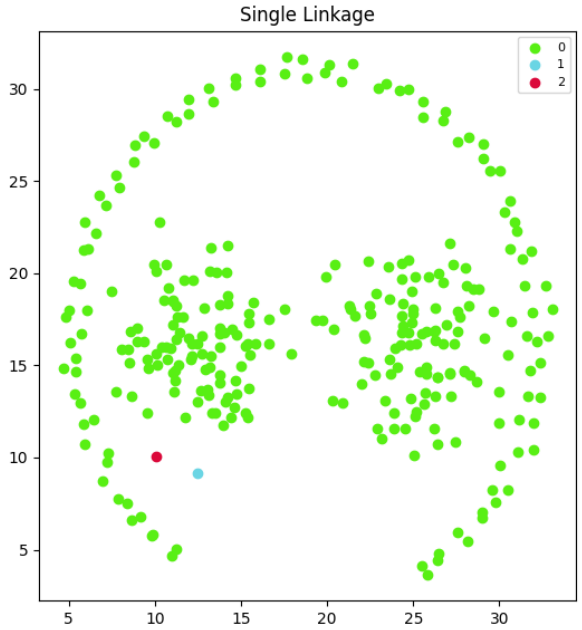




Para o dataset *D31*, como o conjunto de dados estão bem definidos, tanto o algoritmo K-Means quanto Single Linkage implica em ótimos algoritmos para o agrupamento de dados em questão.

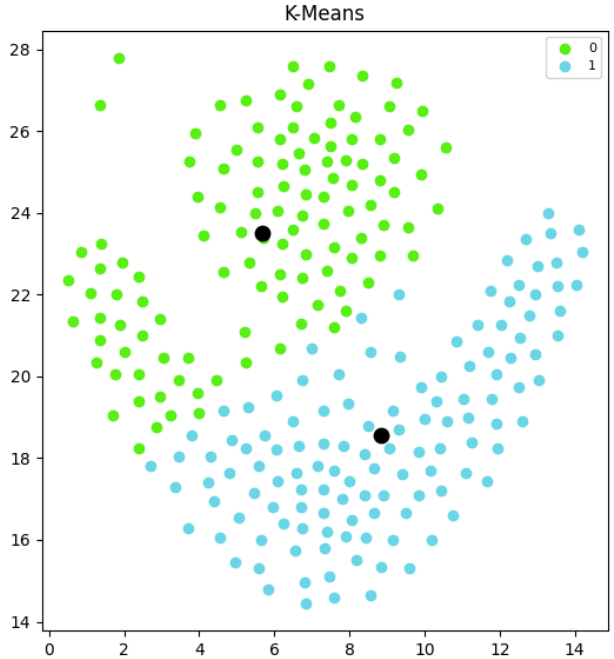
DATASET – PATH BASED:

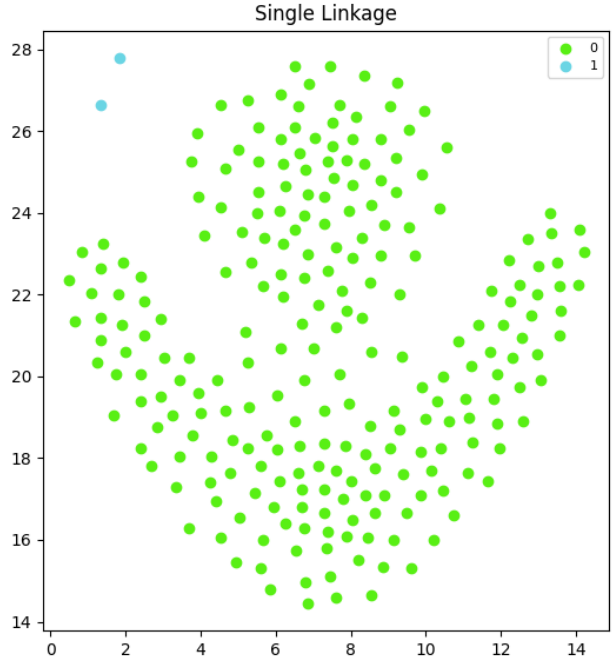




Para o dataset *Path Based*, é simples identificar que o melhor agrupamento é dado pelo algoritmo K-Means, uma vez que o algoritmo Single Linkage é sensível para conjuntos de dados globulares.

DATASET – FLAME:





Para o dataset *Flame*, K-Means representa o melhor algoritmo para o agrupamento de dados em questão. Assim como o dataset *Path Based,* o algoritmo Single Linkage é sensível para dados fortemente agrupados (quando não há uma grande diferenciação de distâncias mínimas entre clusters)