UNIVERSIDADE FEDERAL DE ITAJUBÁ
ENGENHARIA DE COMPUTAÇÃO

GABRIEL TAUCHEN FILGUEIRAS - 2022003880
ANDRÉ LUIZ MELO DOS SANTOS FRANCO - 2021012740
MATEUS ALEXANDRE MARTINS DE SOUZA - 2021004023

INTELIGÊNCIA ARTIFICIAL
REDUÇÃO DE DIMENSIONALIDADE
CLASSIFICAÇÃO DE DADOS

ITAJUBÁ
2023

1. OBJETIVO

O objetivo deste relatório é demonstrar e analisar o algoritmo PCA para a redução de dimensionalidade e o algoritmo DBSCAN para a classificação de dados.

Para realizar esse estudo foram utilizados datasheets da biblioteca scikit-learn, os programas foram desenvolvidos em Python.

Para fins de experimentação, foi escolhido um datasheet para o algoritmo PCA que reduz a dimensionalidade e imprime a tabela com o resultado. Já para o algoritmo DBSCAN se utilizou outro datasheet sobre classificação para classificar dados e por fim imprime um gráfico com o resultado.

2. CÓDIGO FONTE

A seguir será apresentado o código fonte que contém o algoritmo PCA para a redução de dimensionalidade.

```
!pip install mglearn

import matplotlib.pyplot as plt
import numpy as np
import mglearn
from sklearn.datasets import load_wine
from sklearn.model_selection import train_test_split

from sklearn.datasets import load_wine
from sklearn.model_selection import train_test_split
import matplotlib.pyplot as plt
import mglearn

# Carregando a base de dados
wine = load_wine()
X_train, X_test, y_train, y_test = train_test_split(wine.data,
wine.target, random_state=1)
fig, axes = plt.subplots(2, 2, figsize=(10, 5))
class_0 = wine.data[wine.target == 0]
class_1 = wine.data[wine.target == 1]
class_2 = wine.data[wine.target == 2]
ax = axes.ravel()

# Criação e plot dos gráficos de cada característica, antes da
utilização
# do algoritmo PCA
for i in range(4):
```

```
    _, bins = np.histogram(wine.data[:, i], bins=50)
    ax[i].hist(class_0[:, i], bins=bins, color=mglearn.cm3(0),
alpha=.5)
    ax[i].hist(class_1[:, i], bins=bins, color=mglearn.cm3(2),
alpha=.5)
    ax[i].hist(class_2[:, i], bins=bins, color=mglearn.cm3(1),
alpha=.5)
    ax[i].set_title(wine.feature_names[i])
    ax[i].set_yticks(())
    ax[0].set_xlabel("Feature magnitude")
    ax[0].set_ylabel("Frequency")

ax[0].legend(["class_0", "class_1", "class_2"], loc="best")
fig.tight_layout()
plt.show()

# Aplicação do algoritmo
from sklearn.datasets import load_wine
from sklearn.decomposition import PCA
import matplotlib.pyplot as plt

# Carregando o conjunto de dados
data = load_wine()
X = data.data
y = data.target

# Normalizando os dados (opcional, mas geralmente recomendado)
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)

# Aplicando o PCA para redução de dimensionalidade
pca = PCA(n_components=2)  # Número de componentes principais desejado
X_pca = pca.fit_transform(X_scaled)

# Visualizando os resultados
plt.figure(figsize=(8, 6))
plt.scatter(X_pca[:, 0], X_pca[:, 1], c=y, cmap=plt.cm.Set1,
edgecolor='k')
plt.title("PCA do dataset Wine")
plt.xlabel("Primeiro Componente Principal")
plt.ylabel("Segundo Componente Principal")
```

```
plt.show()
```

A seguir será apresentado o código fonte que contém o algoritmo DBSCAN para a classificação de dados.

```
#https://www.section.io/engineering-education/dbscan-clustering-in-python/
from sklearn.cluster import DBSCAN
from sklearn.datasets import load_iris
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt

#carregando a base de dados
data = load_iris()
print("Dataset shape:", data.feature_names)
#print(data.data)

x = data.data
plt.scatter(x[:, 0], x[:,1], cmap = "plasma")
plt.title("Before DBSCAN")
plt.xlabel("Sepal length (cm)")
plt.ylabel("Sepal width (cm)")
plt.show()

#Calculando epsilon (distancia maxima entre pontos para cluster)
#from sklearn.neighbors import NearestNeighbors
#neighb = NearestNeighbors(n_neighbors = 2)
#nbrs = neighb.fit(x)
#distances, indices=nbrs.kneighbors(x)

#imprimindo grafico para calculo do epsilon com base no vizinho mais proximo
#distances = np.sort(distances, axis = 0)
#distances = distances[:, 1]
#plt.rcParams['figure.figsize'] = (5,3)
#plt.plot(distances)
#plt.show()

#utilizando dbscan
#epsilon = 0.7, dado pela visualizacao do grafico comentado acima
```

```
#numero de amostras igual a 2 vezes a dimensao (4)
dbscan = DBSCAN(eps = 0.7, min_samples = 8).fit(x)
labels = dbscan.labels_

#imprimindo o grafico
plt.scatter(x[:, 0], x[:,1], c = labels, cmap = "plasma")
plt.title("After DBSCAN")
plt.xlabel("Sepal length (cm)")
plt.ylabel("Sepal width (cm)")
plt.show()
```

3. TESTE E RESULTADOS

Para o algoritmo PCA temos os dados antes da aplicação do algoritmo do datasheet wine.

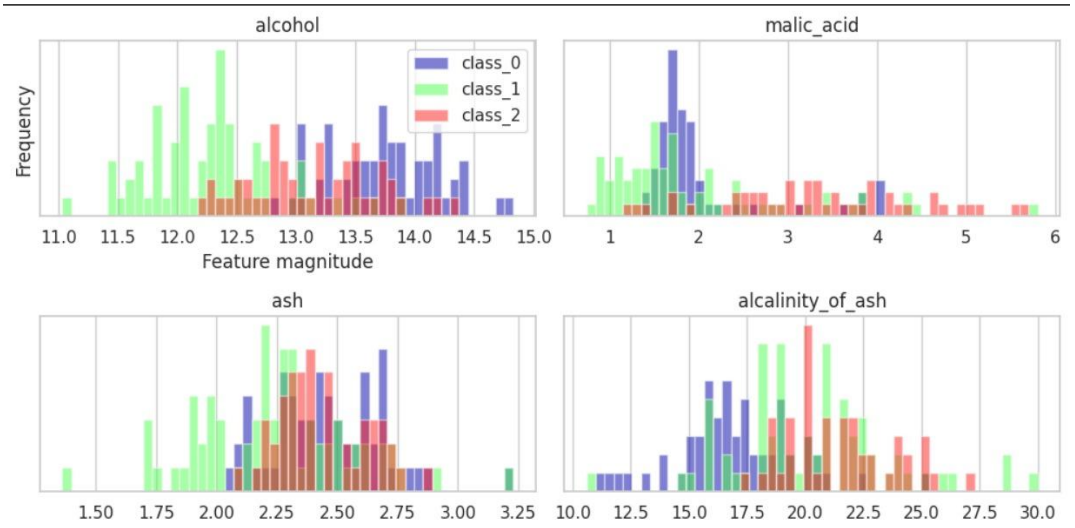

Figura 1 *- dados do datasheet wine*

Após ser usado o algoritmo, recebemos este gráfico.

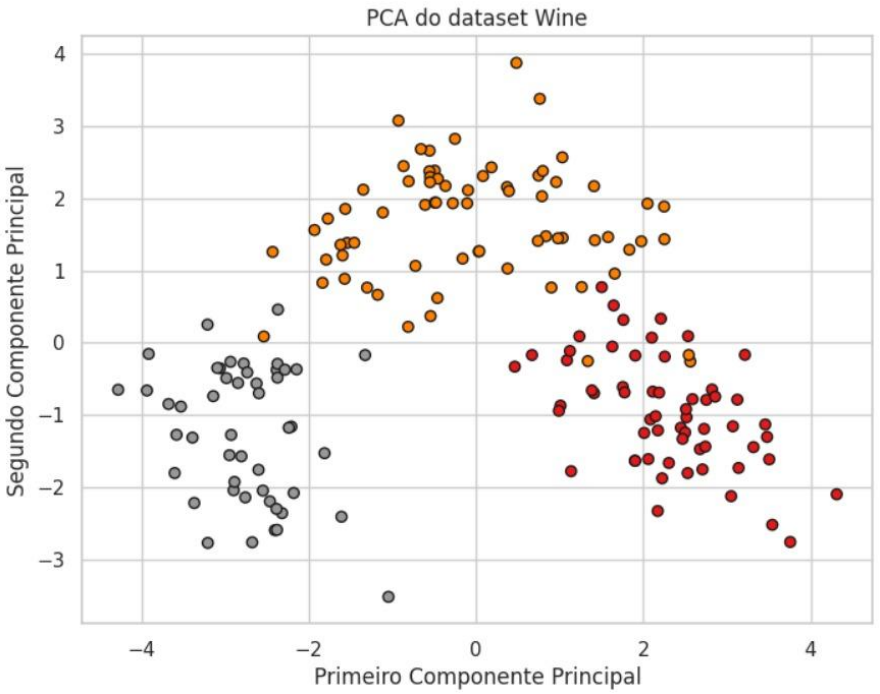

Figura 2 *- resultado do algoritmo PCA*

Para o algoritmo DBSCAN temos os dados antes da aplicação do algoritmo do datasheet iris.

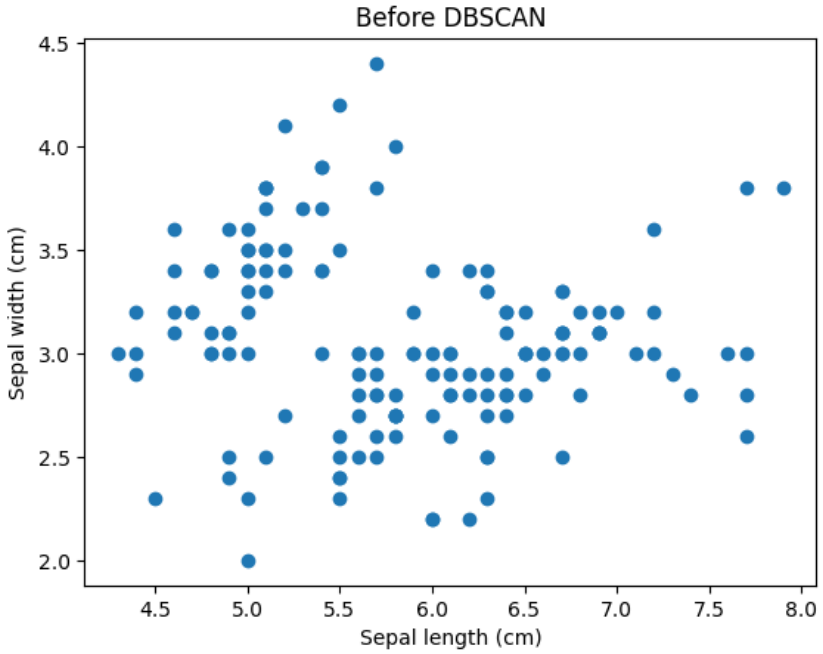

Figura 3 *- dados do datasheet iris*

Após ser usado o algoritmo, recebemos este gráfico.

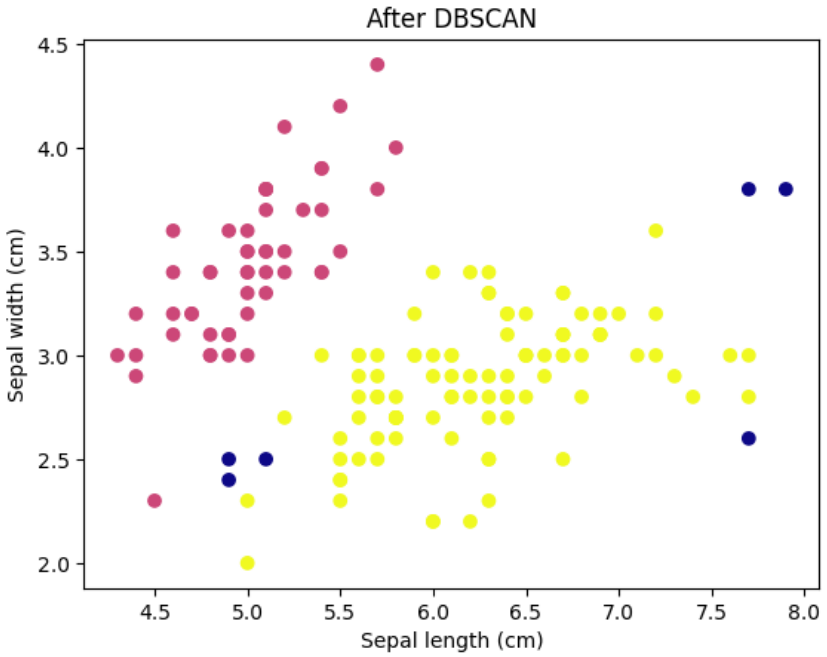

Figura 4 *- resultado do algoritmo DBSCAN*

4. DISCUSSÃO

Começando com o algoritmo PCA, foi selecionada a base de dados "load_wine" que é um conjunto de dados que contém informações sobre vinhos de três classes diferentes. Cada classe representa um tipo de vinho, e as características incluem várias medidas químicas desses vinhos, como teor alcoólico, acidez, etc. A tarefa comum associada a este conjunto de dados é a classificação dos vinhos em suas respectivas classes.

O PCA é uma técnica de redução de dimensionalidade amplamente utilizada que tem o objetivo de capturar a estrutura mais importante dos dados, reduzindo a dimensionalidade do conjunto de dados original. Isso é feito projetando os dados em um novo espaço, onde os eixos são chamados de "componentes principais". Isso permite representar os dados com menos dimensões, preservando o máximo de informação possível.

A visualização final permite observar como as amostras de vinhos se distribuem em um espaço bidimensional após a redução de dimensionalidade, o que pode ser útil para tarefas de classificação ou análise exploratória de dados.

Cada cor de pontos corresponde a uma das classes de vinho, sendo os pontos vermelho a classe 0, os pontos laranja a classe 1, e os pontos cinza a classe 2, podendo ver como eles ficam agrupados quando é feita a redução de dimensionalidade.

Agora, ao aplicar o algoritmo DBSCAN na base de dados iris (no exemplo, as duas primeiras colunas), observou-se a divisão dos pontos em dois grupos principais e um terceiro com número menor.

Assim, observou-se a formação de dois clusters: o primeiro com cor amarela e o segundo com cor rosada. Isso demonstra a função do método utilizado: classificação. Baseado nos parâmetros que estabelecem o número mínimo de pontos para formar um cluster e o raio máximo de um cluster, o DBSCAN foi capaz de realizar a separação dos dados em dois grupos e ruídos (na cor mais escura e azulada), que não pertencem a nenhum grupo.

5. CONCLUSÃO

Neste trabalho foram analisados os algoritmos PCA e DBSCAN. O primeiro realiza uma redução de dimensionalidade para capturar a estrutura mais importante dos dados, reduzindo a dimensionalidade do conjunto de dados original para representar os dados com menos dimensões, preservando o máximo de informação possível como foi possível observar no resultado da execução do algoritmo em que na datasheet wine em que os vinhos foram organizados em três tipos após a sua execução. Assim facilitando a análise dos dados.

O algoritmo DBSCAN realiza um ajuntamento de pontos em clusters para a facilitação da interpretação de dados como pode ser observado no resultado da execução na datasheet iris em que os pontos foram organizados em dois clusters com ruídos que não pertencem a nenhum dos dois.

Portanto, observa-se que esses algoritmos realizaram suas funções com os datasheets utilizados. Sendo o primeiro útil para reduzir a quantidade de dimensões para a análise dos dados sem perder a resolução e o segundo para organizar vários dados em clusters para facilitar a interpretação dos dados do datasheet.