@Odemir Depieri Jr

Guia de Machine Learning - Clusterização K-Means

Python

Case com os dados de Vinho

seaborn

matplotlib

NumPy

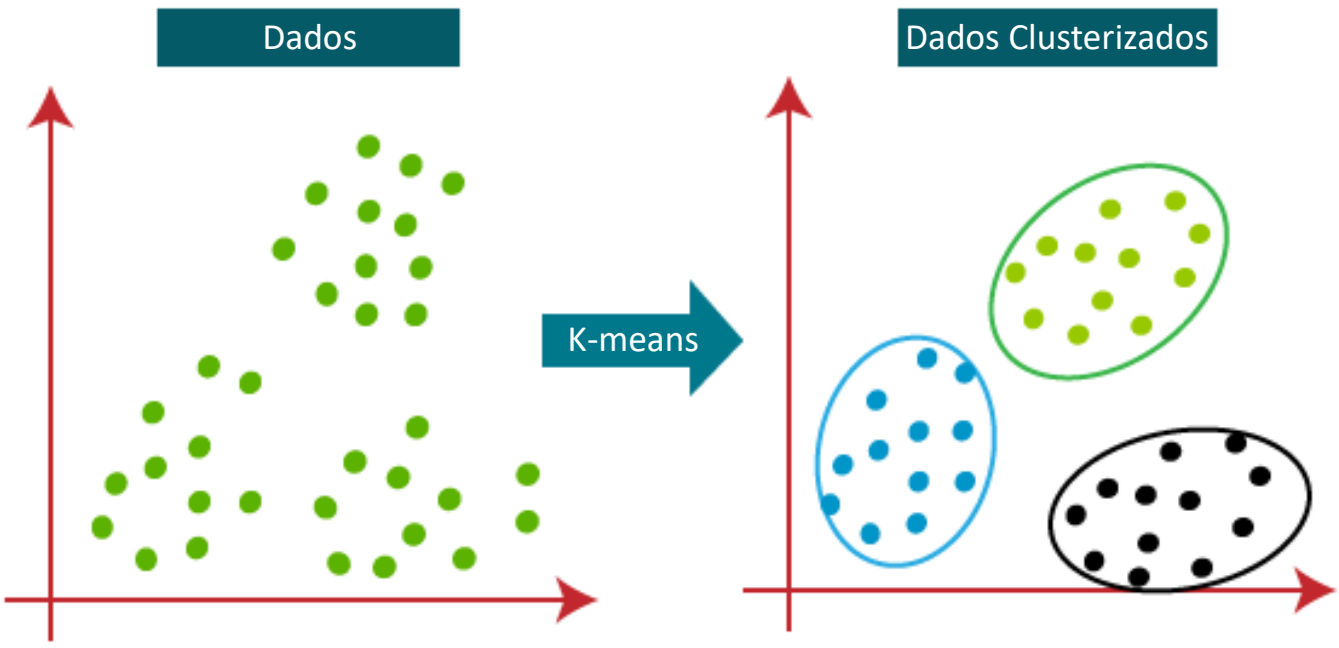
scikit learn

O que é Clusterização ?

Clusterização

Clusterização é a tarefa de dividir a população ou os pontos de dados em vários grupos, de modo que os pontos de dados nos mesmos grupos sejam mais semelhantes a outros pontos de dados no mesmo grupo do que os de outros grupos. Em palavras simples, o objetivo é segregar grupos com traços semelhantes e atribuí-los a clusters.

Vamos exemplificar:



Basicamente vamos criar grupos com as mesmas caracterizadas.

Conteúdo dos dados

Vamos explorar os dados

Descrição

Este conjunto de dados é adaptado do Wine Data Set de https://archive.ics.uci.edu/ml/datasets/wine , removendo as informações sobre os tipos de vinho para aprendizagem não supervisionada.

As seguintes descrições foram adaptadas da página da web da UCI:

Esses dados são resultados de uma análise química de vinhos cultivados na mesma região da Itália, mas derivados de três cultivares diferentes. A análise determinou as quantidades de 13 constituintes encontrados em cada um dos três tipos de vinhos.

Os atributos são:

- Álcool
- Ácido málico
- Cinzas
- Alcalinidade de cinzas
- Magnésio
- Fenóis totais
- Flavonóides
- Fenóis não flavanoides
- Proantocianinas
- Intensidade da cor
- Matiz
- OD280 / OD315 de vinhos diluídos
- Proline

^

Sobre este arquivo

"Esses dados são resultados de uma análise química de vinhos cultivados na mesma região da Itália, mas derivados de três cultivares diferentes. A análise determinou as quantidades de 13 constituintes encontrados em cada um dos três tipos de vinhos."

Mão na Massa !

Estou usando o Google Colab para compilar o script

```
[1]  # Biblioteca para modelagem de dados
     import pandas as pd

     # Biblioteca para recursos matemáticos
     import numpy as np

     # Bibliotecas de plotagem de dados
     import seaborn as sns
     import matplotlib.pyplot as plt
```

```
[2]  # Lendo a Base de Dados
     Base_Dados = pd.read_csv('wine-clustering.csv')
```

```
[3]  # Verificando as primeiras linhas
     Base_Dados.head()
```

	Alcohol	Malic_Acid	Ash	Ash_Alcanity	Magnesium	Total_Phenols	Flavanoids	Nonflavanoid_Phenols	Proanthocyanins	Color_Intensity	Hue	OD280	Proline
0	14.23	1.71	2.43	15.6	127	2.80	3.06	0.28	2.29	5.64	1.04	3.92	1065
1	13.20	1.78	2.14	11.2	100	2.65	2.76	0.26	1.28	4.38	1.05	3.40	1050
2	13.16	2.36	2.67	18.6	101	2.80	3.24	0.30	2.81	5.68	1.03	3.17	1185
3	14.37	1.95	2.50	16.8	113	3.85	3.49	0.24	2.18	7.80	0.86	3.45	1480
4	13.24	2.59	2.87	21.0	118	2.80	2.69	0.39	1.82	4.32	1.04	2.93	735

```
[4]  # Verificando as colunas
     for Coluna in Base_Dados.columns:
       print( Coluna )
```

```
Alcohol
Malic_Acid
Ash
Ash_Alcanity
Magnesium
Total_Phenols
Flavanoids
Nonflavanoid_Phenols
Proanthocyanins
Color_Intensity
Hue
OD280
Proline
```

```
[5]  # Verificando a dimensão da base de dados
     Base_Dados.shape
```

```
(178, 13)
```

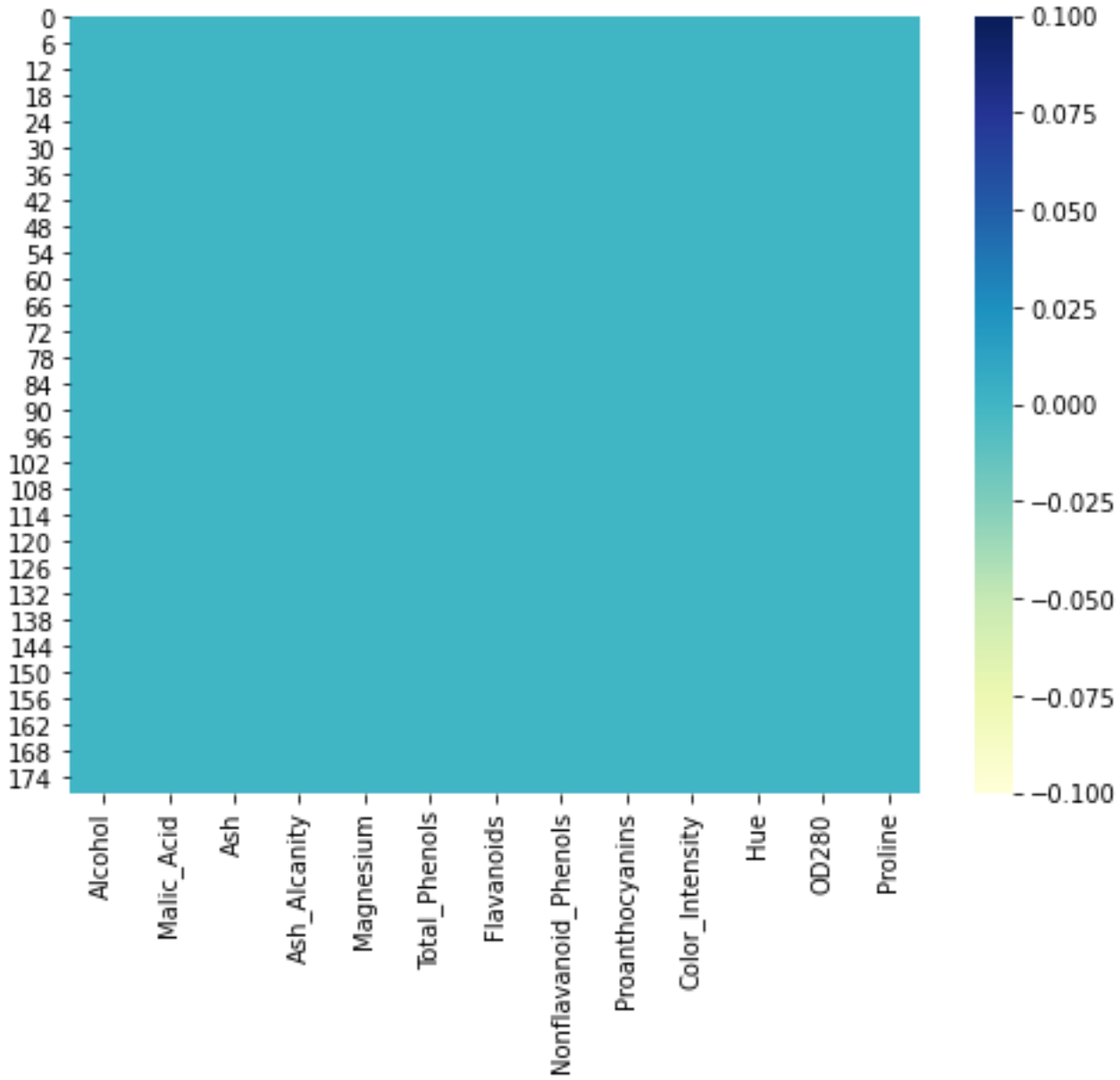
```
[6]  # Verificando o formato dos campos
     Base_Dados.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 178 entries, 0 to 177
Data columns (total 13 columns):
 #   Column                Non-Null Count  Dtype  
---  ------                --------------  -----  
 0   Alcohol               178 non-null    float64
 1   Malic_Acid            178 non-null    float64
 2   Ash                   178 non-null    float64
 3   Ash_Alcanity          178 non-null    float64
 4   Magnesium             178 non-null    int64  
 5   Total_Phenols         178 non-null    float64
 6   Flavanoids            178 non-null    float64
 7   Nonflavanoid_Phenols  178 non-null    float64
 8   Proanthocyanins       178 non-null    float64
 9   Color_Intensity       178 non-null    float64
 10  Hue                   178 non-null    float64
 11  OD280                 178 non-null    float64
 12  Proline               178 non-null    int64  
dtypes: float64(11), int64(2)
memory usage: 18.2 KB
```

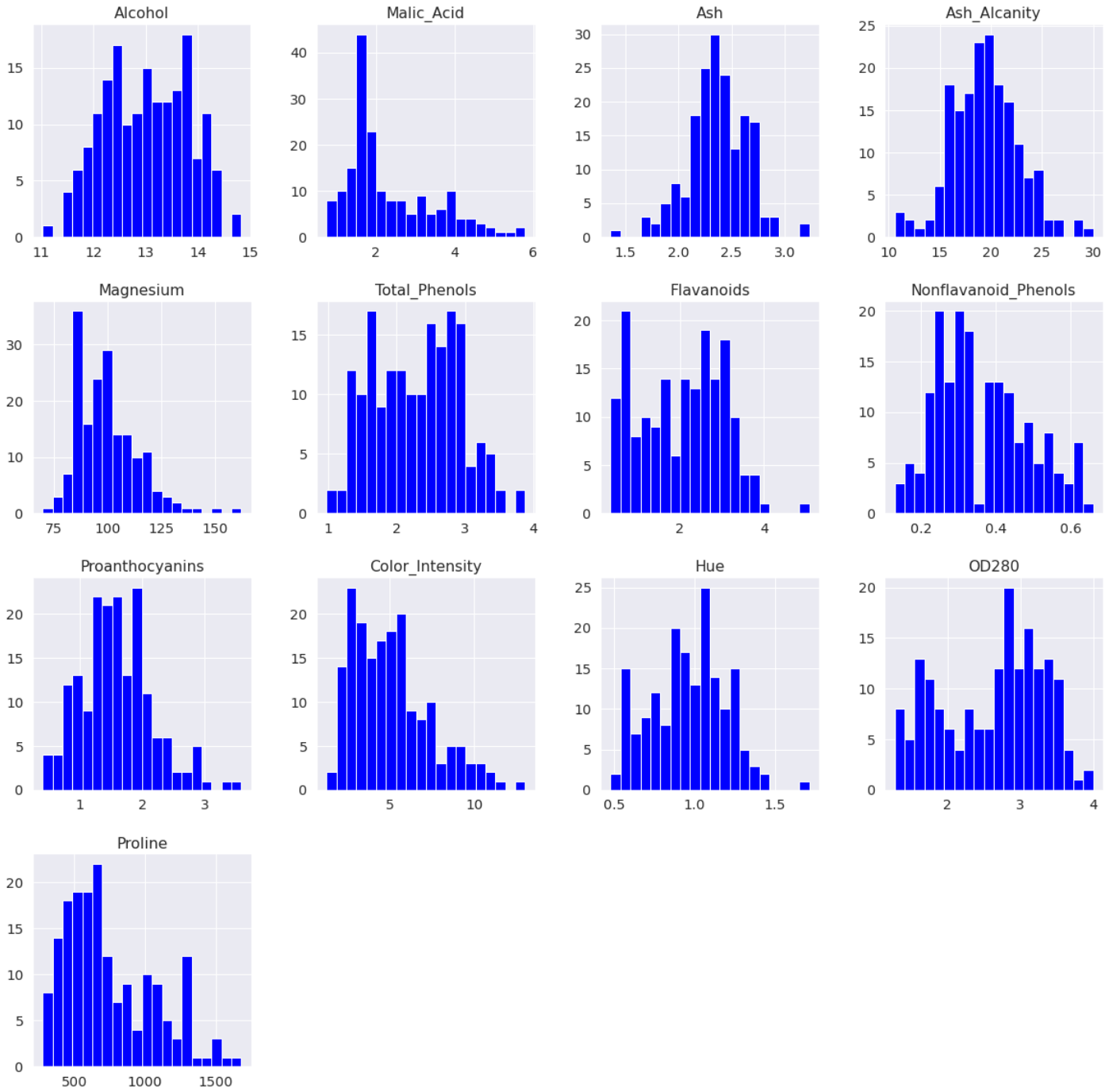
```
[7]  # Gerando algumas estatística para entender um pouco os dados

     # Dicionário para entender as estatísticas abaixo:
     # count --> Total de registros
     # mean --> Média
     # std --> Desvio Padrão
     # min --> Valor mínimo
     # 25% --> 1º Quartil
     # 50% --> Mediana
     # 75% --> 3º Quartil
     # max --> Valor Maior

     # Comando para gerar estatísticas sobre os dados
     Base_Dados.describe()
```

	Alcohol	Malic_Acid	Ash	Ash_Alcanity	Magnesium	Total_Phenols	Flavanoids	Nonflavanoid_Phenols	Proanthocyanins	Color_Intensity	Hue	OD280	Proline
count	178.000000	178.000000	178.000000	178.000000	178.000000	178.000000	178.000000	178.000000	178.000000	178.000000	178.000000	178.000000	178.000000
mean	13.000618	2.336348	2.366517	19.494944	99.741573	2.295112	2.029270	0.361854	1.590899	5.058090	0.957449	2.611685	746.893258
std	0.811827	1.117146	0.274344	3.339564	14.282484	0.625851	0.998859	0.124453	0.572359	2.318286	0.228572	0.709990	314.907474
min	11.030000	0.740000	1.360000	10.600000	70.000000	0.980000	0.340000	0.130000	0.410000	1.280000	0.480000	1.270000	278.000000
25%	12.362500	1.602500	2.210000	17.200000	88.000000	1.742500	1.205000	0.270000	1.250000	3.220000	0.782500	1.937500	500.500000
50%	13.050000	1.865000	2.360000	19.500000	98.000000	2.355000	2.135000	0.340000	1.555000	4.690000	0.965000	2.780000	673.500000
75%	13.677500	3.082500	2.557500	21.500000	107.000000	2.800000	2.875000	0.437500	1.950000	6.200000	1.120000	3.170000	985.000000
max	14.830000	5.800000	3.230000	30.000000	162.000000	3.880000	5.080000	0.660000	3.580000	13.000000	1.710000	4.000000	1680.000000

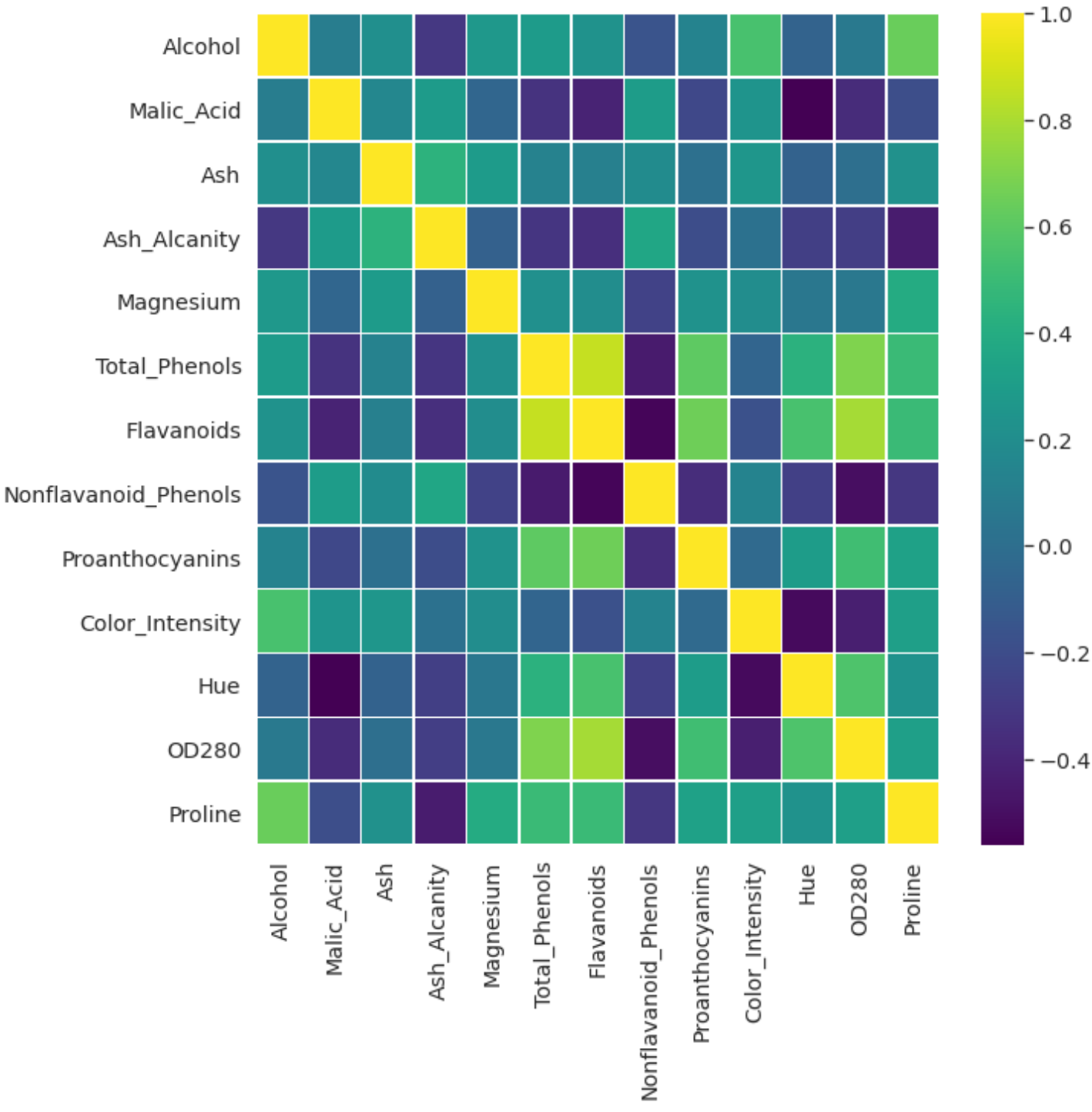
```
[8]  # Verificando se há um valor nulo na base de dados
     # Caso exista haverá linhas brancas no gráfico
     plt.figure( figsize=(8,6) )
     sns.heatmap( Base_Dados.isnull(), cmap="YlGnBu", cbar=True  );
```



```
[9]  # --- Plotando todos os dados
     sns.set( font_scale=1.3, rc={'figure.figsize':(20, 20)} )
     ax = Base_Dados.hist( bins=20, color='blue' )
```



```
[10] # --- Analisando as correlações nos dados
     # Função para analisar correlação
     corr = Base_Dados.corr()
     # Definindo Tamanho do Gráfico
     plt.figure(figsize=(10,10))
     # Fazendo o plot do gráfico
     sns.heatmap(corr, linewidths=.5, cmap='viridis');
```



```
[11] # Bilbioteca para fazer o escalonamento dos Dados
     from sklearn.preprocessing import StandardScaler

     # Fazendo uma copia da base de dados original
     Base_Dados_002 = Base_Dados.copy()

     # Fazendo o escalonamento dos Dados
     Funcao_Escalonamento = StandardScaler()
     Dados_Escalonados = Funcao_Escalonamento.fit_transform( Base_Dados_002 )
```

```
[12] # Verificando os primeiros registros
     Dados_Escalonados[0:4]
```

```
     array([[ 1.51861254, -0.5622498 ,  0.23205254, -1.16959318,  1.91390522,
              0.80899739,  1.03481896, -0.65956311,  1.22488398,  0.25171685,
              0.36217728,  1.84791957,  1.01300893],
            [ 0.24628963, -0.49941338, -0.82799632, -2.49084714,  0.01814502,
              0.56864766,  0.73362894, -0.82071924, -0.54472099, -0.29332133,
              0.40605066,  1.1134493 ,  0.96524152],
            [ 0.19687903,  0.02123125,  1.10933436, -0.2687382 ,  0.08835836,
              0.80899739,  1.21553297, -0.49840699,  2.13596773,  0.26901965,
              0.31830389,  0.78858745,  1.39514818],
            [ 1.69154964, -0.34681064,  0.4879264 , -0.80925118,  0.93091845,
              2.49144552,  1.46652465, -0.98187536,  1.03215473,  1.18606801,
             -0.42754369,  1.18407144,  2.33457383]])
```

```
[13] # Aplicando a redução de dimensionalidade
     # Reduzindo de 10 variaveis para 2
     from sklearn.decomposition import PCA
     Funcao_PCA = PCA( n_components=2 )
     Dados_PCA = Funcao_PCA.fit_transform( Dados_Escalonados )
```

```
[14] # Verificando os 1º primeiros registros no numpy
     Dados_PCA[0:5]
```

```
     array([[ 3.31675081, -1.44346263],
            [ 2.20946492,  0.33339289],
            [ 2.51674015, -1.0311513 ],
            [ 3.75706561, -2.75637191],
            [ 1.00890849, -0.86983082]])
```

```
[15] # Importando o modelo Kmeans
     from sklearn.cluster import KMeans

     # Lista para gravar a Distancia do cluster
     Distancia_Cluster = []
     Repeticoes = range(1,20)

     # Loop para analisar a distancia do cluster
     for x in Repeticoes:
       # Parametro para trocar o numero de cluster
       Modelo_Kmeans = KMeans( n_clusters=x )
       # Aplicando o modelo do kmeans
       Modelo_Kmeans.fit( Dados_Escalonados )
       # Salvando a distancia
       Distancia_Cluster.append(Modelo_Kmeans.inertia_ )
```

```
[16] # Verificando os valores
     print( Distancia_Cluster )
```

```
[2314.0, 1658.7588524290954, 1277.9284888446423, 1180.6960041746115, 1
```

```
[17] # Plotando a curva do joelho
     # Verifanco o melhor numero de cluster
     plt.figure( figsize=(10,8) )
     plt.title('Curva de Cluster')
     plt.plot( Distancia_Cluster, 'bx-' )
     plt.xlabel('Cluster')
     plt.ylabel('Distancia do Cluster');
```

Caso queira entender melhor sobre essa técnica, sugiro essa leitura

https://medium.com/pizzadedados/kmeans-e-metodo-do-cotovelo-94ded9fdf3a9

$$\text{distance}(P_0, P_1, (x, y)) = \frac{|(y_1 - y_0)x - (x_1 - x_0)y + x_1y_0 - y_1x_0|}{\sqrt{(y_1 - y_0)^2 + (x_1 - x_0)^2}}$$

Fórmula para o cálculo entre um ponto e uma reta que passa por P0 e P1

```
[18] # --- Aplicando os cluster nos dados
     # Definindo o numero de 3 cluster
     Modelo_Kmeans = KMeans( n_clusters=3 )
     # Aplicando a funcao nos dados
     Modelo_Kmeans.fit( Dados_Escalonados )
     # Identificando os centroides
     Centroides = Modelo_Kmeans.cluster_centers_
     # Identificando o numero do cluster
     Rotulos = Modelo_Kmeans.labels_
```

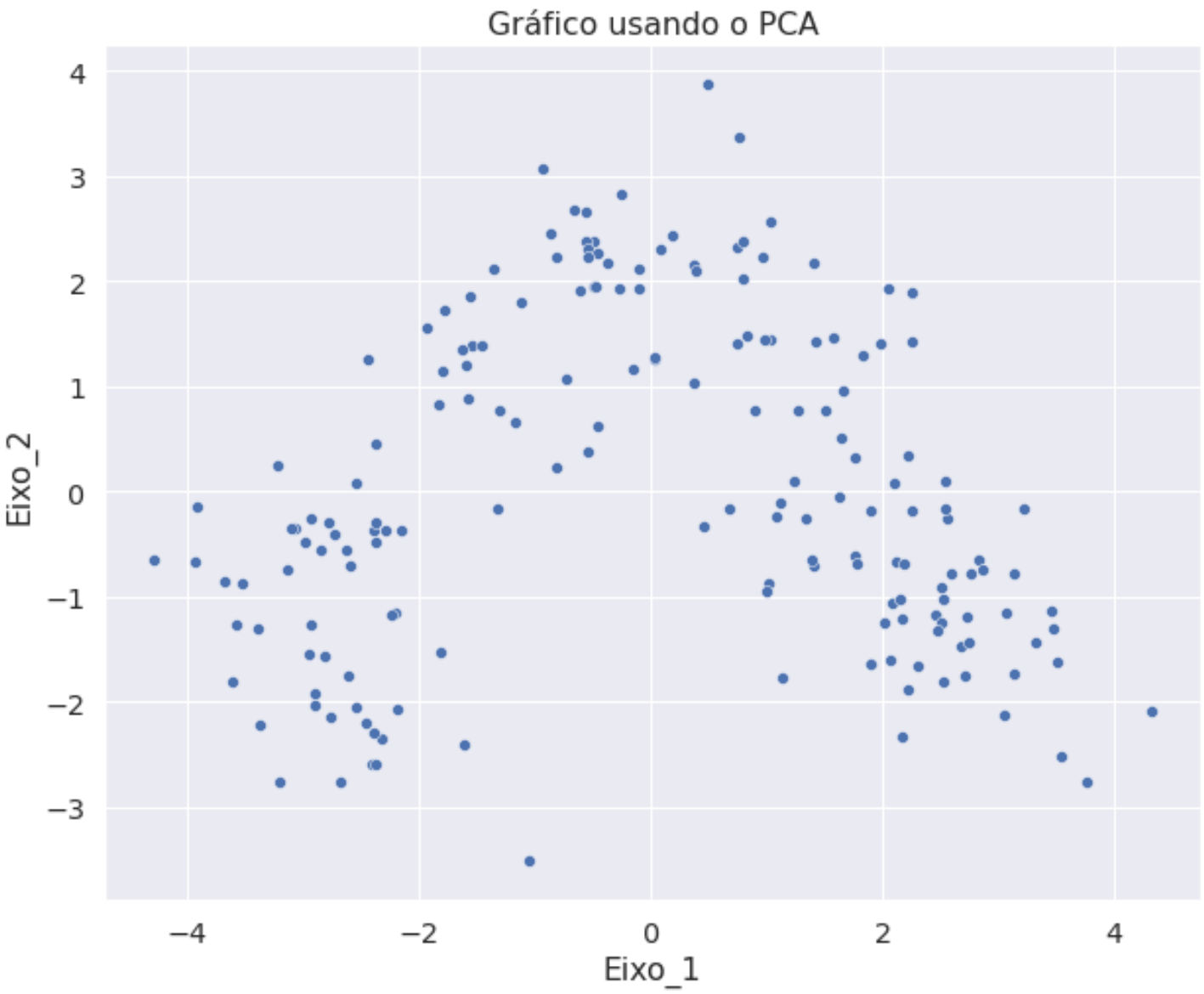
```
[19] # Verificando divisão dos cluster
     Rotulos
```

```
array([1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 2, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 2, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 2, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 2, 2,
       2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
       2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
       2, 2], dtype=int32)
```

```
[20] # Ajustando os dados para uma tabela
     Base_PCA = pd.DataFrame( data=Dados_PCA, columns=['Eixo_1', 'Eixo_2'] )
     Base_PCA.head()
```

	Eixo_1	Eixo_2
0	3.316751	-1.443463
1	2.209465	0.333393
2	2.516740	-1.031151
3	3.757066	-2.756372
4	1.008908	-0.869831

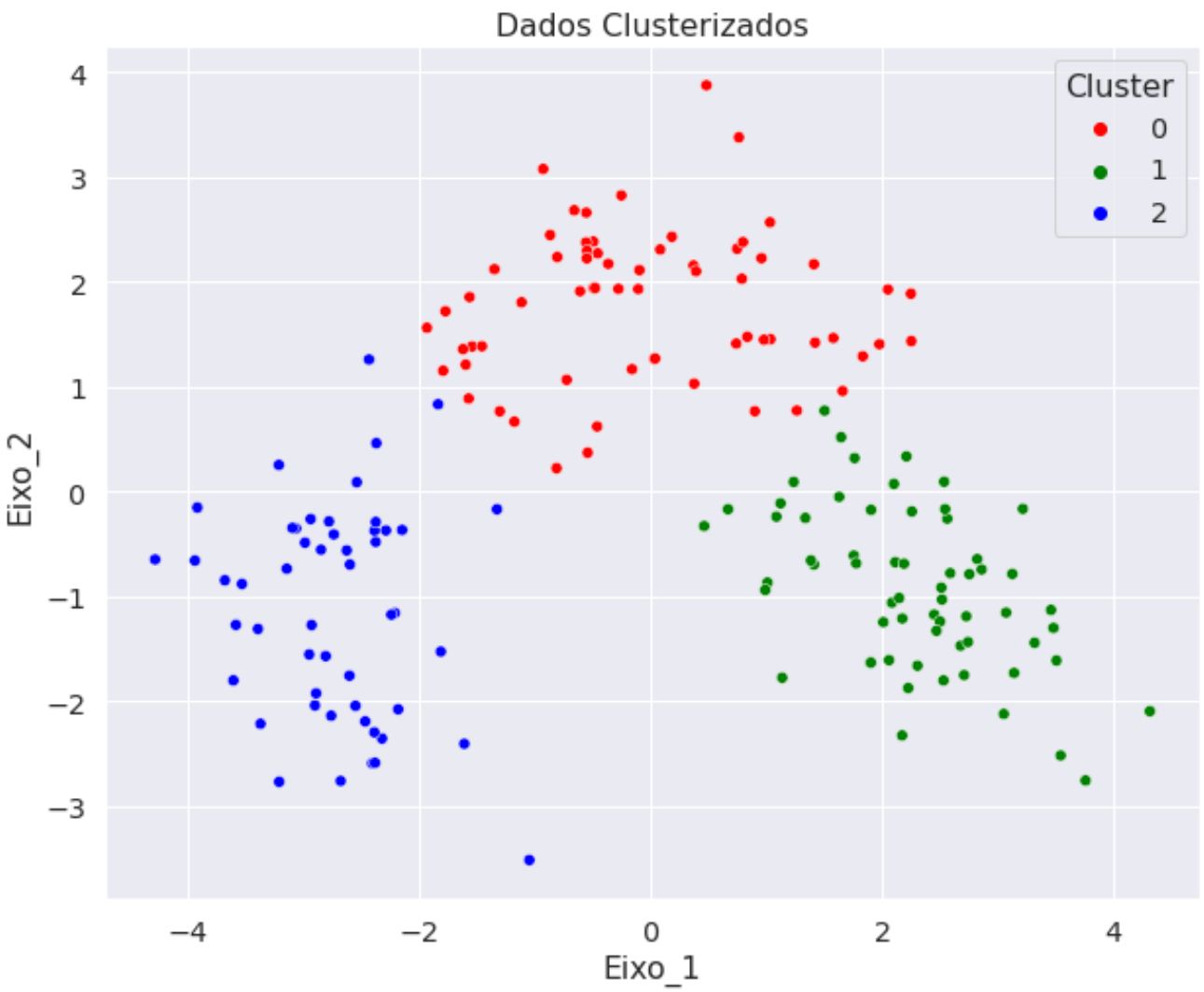
```
[21] # Plotando o gráfico com os dados 2 eixos
     plt.figure( figsize=(10,8) )
     plt.title('Gráfico usando o PCA')
     sns.scatterplot( x='Eixo_1', y='Eixo_2', data=Base_PCA );
```




```
[22] # Jutando os dados do PCA com o Cluster
     A = pd.concat( [ Base_PCA, pd.DataFrame( {'Cluster' : Rotulos} ) ], axis=1 )
     A.head()
```

	Eixo_1	Eixo_2	Cluster
0	3.316751	-1.443463	1
1	2.209465	0.333393	1
2	2.516740	-1.031151	1
3	3.757066	-2.756372	1
4	1.008908	-0.869831	1

```
[23] # Plotando o gráfico com os dados 2 eixos

     Paletas_Cores = ['red', 'green', 'blue', 'yellow', 'pink', 'gray',
                      'purple', 'black']

     plt.figure( figsize=(10,8) )
     plt.title('Dados Clusterizados')
     sns.scatterplot( x='Eixo_1', y='Eixo_2', hue='Cluster', data=A,
                     palette=Paletas_Cores[0:3] );
```



Testei o modelo com 3 a 6 clusters.

A melhor divisão ficou entre 3 e 4, porem mantive 3 grupos. Devido contem “poucos” registros e a divisão ficou bem homogenia e também melhor ilustra esse tutorial.

Final

Esse guia é um exemplo de uma modelo Clusterização usando o K-Means.

Artigo sobre K-means
https://www.analyticsvidhya.com/blog/2021/01/in-depth-intuition-of-k-means-clustering-algorithm-in-machine-learning/#:~:text=WCSS%20is%20the%20sum%20of,value%20will%20start%20to%20decrease.

Informações sobre clusterização
https://portaldatascience.com/introducao-a-clusterizacao-e-os-diferentes-metodos/#:~:text=K%2DMeans%20Clusters,dados%20no%20espa%C3%A7o%202%2DD.

Guia da documentação caso queira mais detalhes
https://scikit-learn.org/stable/modules/generated/sklearn.cluster.KMeans.html

Odemir Depieri Jr

Software Engineer Sr
Tech Lead
Specialization AI