intro

March 2, 2025

[1]:
```
import pandas as pd
import numpy as np
```

0.1 Oz data read

[3]:
```
def matriz(num_linhas, array):
    """
    Transforma um array unidimensional em uma matriz organizada por colunas.

    Args:
        array (list ou np.array): Lista de números a serem organizados.
        num_linhas (int): Número de linhas desejadas na matriz.

    Returns:
        np.array: Matriz ordenada.
    """

    # Reshape para matriz (por linha) e depois transpõe para organizar por␣
 ↪colunas
    matriz = np.array(array).reshape(-1, num_linhas).T

    return matriz # Retorna como lista para melhor legibilidade
```

[4]:
```
array1 = np.arange(600)
array1
```

[4]:
```
array([  0,   1,   2,   3,   4,   5,   6,   7,   8,   9,  10,  11,  12,
        13,  14,  15,  16,  17,  18,  19,  20,  21,  22,  23,  24,  25,
        26,  27,  28,  29,  30,  31,  32,  33,  34,  35,  36,  37,  38,
        39,  40,  41,  42,  43,  44,  45,  46,  47,  48,  49,  50,  51,
        52,  53,  54,  55,  56,  57,  58,  59,  60,  61,  62,  63,  64,
        65,  66,  67,  68,  69,  70,  71,  72,  73,  74,  75,  76,  77,
        78,  79,  80,  81,  82,  83,  84,  85,  86,  87,  88,  89,  90,
        91,  92,  93,  94,  95,  96,  97,  98,  99, 100, 101, 102, 103,
       104, 105, 106, 107, 108, 109, 110, 111, 112, 113, 114, 115, 116,
       117, 118, 119, 120, 121, 122, 123, 124, 125, 126, 127, 128, 129,
       130, 131, 132, 133, 134, 135, 136, 137, 138, 139, 140, 141, 142,
       143, 144, 145, 146, 147, 148, 149, 150, 151, 152, 153, 154, 155,
```

```
       156, 157, 158, 159, 160, 161, 162, 163, 164, 165, 166, 167, 168,
       169, 170, 171, 172, 173, 174, 175, 176, 177, 178, 179, 180, 181,
       182, 183, 184, 185, 186, 187, 188, 189, 190, 191, 192, 193, 194,
       195, 196, 197, 198, 199, 200, 201, 202, 203, 204, 205, 206, 207,
       208, 209, 210, 211, 212, 213, 214, 215, 216, 217, 218, 219, 220,
       221, 222, 223, 224, 225, 226, 227, 228, 229, 230, 231, 232, 233,
       234, 235, 236, 237, 238, 239, 240, 241, 242, 243, 244, 245, 246,
       247, 248, 249, 250, 251, 252, 253, 254, 255, 256, 257, 258, 259,
       260, 261, 262, 263, 264, 265, 266, 267, 268, 269, 270, 271, 272,
       273, 274, 275, 276, 277, 278, 279, 280, 281, 282, 283, 284, 285,
       286, 287, 288, 289, 290, 291, 292, 293, 294, 295, 296, 297, 298,
       299, 300, 301, 302, 303, 304, 305, 306, 307, 308, 309, 310, 311,
       312, 313, 314, 315, 316, 317, 318, 319, 320, 321, 322, 323, 324,
       325, 326, 327, 328, 329, 330, 331, 332, 333, 334, 335, 336, 337,
       338, 339, 340, 341, 342, 343, 344, 345, 346, 347, 348, 349, 350,
       351, 352, 353, 354, 355, 356, 357, 358, 359, 360, 361, 362, 363,
       364, 365, 366, 367, 368, 369, 370, 371, 372, 373, 374, 375, 376,
       377, 378, 379, 380, 381, 382, 383, 384, 385, 386, 387, 388, 389,
       390, 391, 392, 393, 394, 395, 396, 397, 398, 399, 400, 401, 402,
       403, 404, 405, 406, 407, 408, 409, 410, 411, 412, 413, 414, 415,
       416, 417, 418, 419, 420, 421, 422, 423, 424, 425, 426, 427, 428,
       429, 430, 431, 432, 433, 434, 435, 436, 437, 438, 439, 440, 441,
       442, 443, 444, 445, 446, 447, 448, 449, 450, 451, 452, 453, 454,
       455, 456, 457, 458, 459, 460, 461, 462, 463, 464, 465, 466, 467,
       468, 469, 470, 471, 472, 473, 474, 475, 476, 477, 478, 479, 480,
       481, 482, 483, 484, 485, 486, 487, 488, 489, 490, 491, 492, 493,
       494, 495, 496, 497, 498, 499, 500, 501, 502, 503, 504, 505, 506,
       507, 508, 509, 510, 511, 512, 513, 514, 515, 516, 517, 518, 519,
       520, 521, 522, 523, 524, 525, 526, 527, 528, 529, 530, 531, 532,
       533, 534, 535, 536, 537, 538, 539, 540, 541, 542, 543, 544, 545,
       546, 547, 548, 549, 550, 551, 552, 553, 554, 555, 556, 557, 558,
       559, 560, 561, 562, 563, 564, 565, 566, 567, 568, 569, 570, 571,
       572, 573, 574, 575, 576, 577, 578, 579, 580, 581, 582, 583, 584,
       585, 586, 587, 588, 589, 590, 591, 592, 593, 594, 595, 596, 597,
       598, 599])
```

[5]:
```
matriz1 = matriz(60, array1)
matriz1
```

[5]:
```
array([[  0,  60, 120, 180, 240, 300, 360, 420, 480, 540],
       [  1,  61, 121, 181, 241, 301, 361, 421, 481, 541],
       [  2,  62, 122, 182, 242, 302, 362, 422, 482, 542],
       [  3,  63, 123, 183, 243, 303, 363, 423, 483, 543],
       [  4,  64, 124, 184, 244, 304, 364, 424, 484, 544],
       [  5,  65, 125, 185, 245, 305, 365, 425, 485, 545],
       [  6,  66, 126, 186, 246, 306, 366, 426, 486, 546],
       [  7,  67, 127, 187, 247, 307, 367, 427, 487, 547],
```

```
[  8,  68, 128, 188, 248, 308, 368, 428, 488, 548],
[  9,  69, 129, 189, 249, 309, 369, 429, 489, 549],
[ 10,  70, 130, 190, 250, 310, 370, 430, 490, 550],
[ 11,  71, 131, 191, 251, 311, 371, 431, 491, 551],
[ 12,  72, 132, 192, 252, 312, 372, 432, 492, 552],
[ 13,  73, 133, 193, 253, 313, 373, 433, 493, 553],
[ 14,  74, 134, 194, 254, 314, 374, 434, 494, 554],
[ 15,  75, 135, 195, 255, 315, 375, 435, 495, 555],
[ 16,  76, 136, 196, 256, 316, 376, 436, 496, 556],
[ 17,  77, 137, 197, 257, 317, 377, 437, 497, 557],
[ 18,  78, 138, 198, 258, 318, 378, 438, 498, 558],
[ 19,  79, 139, 199, 259, 319, 379, 439, 499, 559],
[ 20,  80, 140, 200, 260, 320, 380, 440, 500, 560],
[ 21,  81, 141, 201, 261, 321, 381, 441, 501, 561],
[ 22,  82, 142, 202, 262, 322, 382, 442, 502, 562],
[ 23,  83, 143, 203, 263, 323, 383, 443, 503, 563],
[ 24,  84, 144, 204, 264, 324, 384, 444, 504, 564],
[ 25,  85, 145, 205, 265, 325, 385, 445, 505, 565],
[ 26,  86, 146, 206, 266, 326, 386, 446, 506, 566],
[ 27,  87, 147, 207, 267, 327, 387, 447, 507, 567],
[ 28,  88, 148, 208, 268, 328, 388, 448, 508, 568],
[ 29,  89, 149, 209, 269, 329, 389, 449, 509, 569],
[ 30,  90, 150, 210, 270, 330, 390, 450, 510, 570],
[ 31,  91, 151, 211, 271, 331, 391, 451, 511, 571],
[ 32,  92, 152, 212, 272, 332, 392, 452, 512, 572],
[ 33,  93, 153, 213, 273, 333, 393, 453, 513, 573],
[ 34,  94, 154, 214, 274, 334, 394, 454, 514, 574],
[ 35,  95, 155, 215, 275, 335, 395, 455, 515, 575],
[ 36,  96, 156, 216, 276, 336, 396, 456, 516, 576],
[ 37,  97, 157, 217, 277, 337, 397, 457, 517, 577],
[ 38,  98, 158, 218, 278, 338, 398, 458, 518, 578],
[ 39,  99, 159, 219, 279, 339, 399, 459, 519, 579],
[ 40, 100, 160, 220, 280, 340, 400, 460, 520, 580],
[ 41, 101, 161, 221, 281, 341, 401, 461, 521, 581],
[ 42, 102, 162, 222, 282, 342, 402, 462, 522, 582],
[ 43, 103, 163, 223, 283, 343, 403, 463, 523, 583],
[ 44, 104, 164, 224, 284, 344, 404, 464, 524, 584],
[ 45, 105, 165, 225, 285, 345, 405, 465, 525, 585],
[ 46, 106, 166, 226, 286, 346, 406, 466, 526, 586],
[ 47, 107, 167, 227, 287, 347, 407, 467, 527, 587],
[ 48, 108, 168, 228, 288, 348, 408, 468, 528, 588],
[ 49, 109, 169, 229, 289, 349, 409, 469, 529, 589],
[ 50, 110, 170, 230, 290, 350, 410, 470, 530, 590],
[ 51, 111, 171, 231, 291, 351, 411, 471, 531, 591],
[ 52, 112, 172, 232, 292, 352, 412, 472, 532, 592],
[ 53, 113, 173, 233, 293, 353, 413, 473, 533, 593],
[ 54, 114, 174, 234, 294, 354, 414, 474, 534, 594],
```

```
       [ 55, 115, 175, 235, 295, 355, 415, 475, 535, 595],
       [ 56, 116, 176, 236, 296, 356, 416, 476, 536, 596],
       [ 57, 117, 177, 237, 297, 357, 417, 477, 537, 597],
       [ 58, 118, 178, 238, 298, 358, 418, 478, 538, 598],
       [ 59, 119, 179, 239, 299, 359, 419, 479, 539, 599]])
```

[7]:
```
data1 = pd.read_csv('/home/darkcover/Documentos/Out/dados/Saidas/FUNCOES/DOUBLE␣
 ↪- 17_09_s1.csv')
data1.columns
```

[7]:
```
Index(['n', 'Entrada', 'Odd', 'P60', 'P120', 'P180', 'P240', 'P300', 'P360',
       'P420', 'P480', 'P540', 'P600', 'P660', 'P720', 'P780', 'P840', 'P900',
       'P960', 'P1020', 'P1080', 'P1140', 'P1200', 'P1260', 'P1320', 'P1380',
       'P1440', 'P1500', 'P1560', 'P1620', 'P1680', 'P1740', 'P1800', 'P1860',
       'P1920', 'P1980', 'P1200.1', 'Media Movel', 'Unnamed: 38',
       'Unnamed: 39', 'Unnamed: 40', 'Unnamed: 41', 'Unnamed: 42',
       'Unnamed: 43', 'Unnamed: 44', 'Unnamed: 45', 'Acertos 60',
       'Unnamed: 47', 'Unnamed: 48', 'Unnamed: 49', 'Unnamed: 50',
       'Unnamed: 51', 'Unnamed: 52', 'Unnamed: 53', 'Unnamed: 54',
       'Unnamed: 55', 'Unnamed: 56', 'Unnamed: 57', 'Unnamed: 58',
       'Unnamed: 59', 'Unnamed: 60', 'Unnamed: 61', 'Unnamed: 62',
       'Unnamed: 63', 'Unnamed: 64', 'Unnamed: 65', 'Unnamed: 66',
       'Unnamed: 67', 'Unnamed: 68', 'Acertos Geral', 'Média Global',
       'Unnamed: 71', 'Unnamed: 72', 'Unnamed: 73', 'Unnamed: 74',
       'Unnamed: 75', 'Unnamed: 76', 'Unnamed: 77', 'Unnamed: 78',
       'Unnamed: 79', 'Unnamed: 80', 'Unnamed: 81', 'Unnamed: 82',
       'Unnamed: 83', 'acertos_intervalos'],
      dtype='object')
```

[8]:
```
array2 = data1['Entrada']
array2
```

[8]:
```
0        11,6
1        2,02
2        2,02
3        1,54
4         2,2
        …
1667        1
1668     4,34
1669    19,98
1670     2,52
1671     10,2
Name: Entrada, Length: 1672, dtype: object
```

[18]:
```
array3 = []
array4 = []
```

```
for i in range(600):
    odd = array2[i].replace(',','.')
    if float(odd) >= 6:
        odd = 6
    array3.append(float(odd))
    if float(odd) >= 3:
        corte1 = 1
    else:
        corte1 = 0
    array4.append(corte1)

array3
```

[18]:
```
[6.0,
 2.02,
 2.02,
 1.54,
 2.2,
 1.51,
 6.0,
 3.89,
 1.02,
 1.48,
 1.25,
 1.07,
 1.93,
 4.33,
 3.59,
 3.26,
 1.83,
 1.6,
 2.07,
 6.0,
 1.21,
 5.46,
 3.81,
 1.3,
 1.95,
 1.04,
 1.28,
 1.03,
 1.05,
 3.39,
 1.05,
 2.59,
 2.53,
 6.0,
```

```
1.34,
1.56,
1.0,
1.22,
1.05,
6.0,
2.54,
2.44,
2.34,
6.0,
6.0,
1.44,
1.63,
2.57,
1.73,
1.22,
1.34,
1.33,
1.0,
2.87,
2.28,
6.0,
1.0,
1.73,
1.05,
1.52,
5.63,
4.44,
3.91,
6.0,
2.44,
1.43,
1.83,
1.31,
1.61,
1.73,
1.4,
2.12,
1.49,
5.97,
1.12,
1.55,
1.4,
2.62,
2.87,
1.77,
1.02,
```

```
1.28,
6.0,
2.18,
1.99,
4.71,
1.01,
1.15,
1.41,
6.0,
1.92,
1.2,
1.0,
2.35,
6.0,
1.24,
4.81,
1.0,
1.27,
1.49,
1.94,
1.98,
5.93,
1.76,
1.9,
1.71,
6.0,
6.0,
6.0,
2.03,
1.14,
1.29,
6.0,
1.0,
2.21,
3.09,
1.49,
3.34,
1.04,
2.26,
1.21,
1.26,
1.01,
1.21,
3.08,
2.7,
2.3,
2.51,
```

```
1.17,
1.02,
4.1,
1.17,
1.61,
6.0,
2.74,
1.67,
5.5,
1.41,
1.13,
1.0,
1.49,
2.3,
1.31,
1.18,
6.0,
6.0,
3.16,
4.08,
6.0,
6.0,
6.0,
2.16,
5.1,
6.0,
1.65,
1.0,
6.0,
2.12,
1.1,
6.0,
1.13,
1.64,
1.01,
2.11,
1.21,
1.71,
3.63,
6.0,
6.0,
3.42,
6.0,
5.45,
2.62,
3.2,
6.0,
```

```
3.1,
3.65,
2.88,
1.74,
6.0,
6.0,
2.47,
6.0,
1.52,
1.5,
2.36,
1.0,
6.0,
1.87,
2.11,
5.53,
1.46,
1.14,
6.0,
1.0,
2.3,
1.41,
3.8,
1.03,
3.41,
6.0,
1.0,
1.39,
2.36,
2.18,
1.3,
1.1,
1.87,
3.76,
1.52,
6.0,
2.08,
4.89,
5.59,
1.18,
6.0,
2.66,
1.65,
1.02,
1.59,
6.0,
1.28,
```

```
1.42,
1.08,
2.57,
1.02,
5.11,
1.96,
1.26,
1.17,
1.34,
1.05,
1.18,
2.99,
1.08,
2.19,
1.08,
1.14,
6.0,
2.29,
3.45,
1.07,
6.0,
1.19,
1.05,
1.67,
1.1,
6.0,
1.02,
1.53,
2.38,
1.38,
1.28,
5.21,
6.0,
1.17,
1.06,
3.69,
1.04,
1.54,
1.45,
1.47,
3.41,
6.0,
1.08,
2.15,
1.38,
6.0,
1.02,
```

```
4.11,
2.2,
4.17,
2.7,
1.04,
2.14,
1.37,
1.88,
2.85,
6.0,
1.09,
2.08,
4.32,
1.08,
5.52,
2.37,
2.47,
1.66,
1.3,
6.0,
2.6,
6.0,
1.65,
1.0,
1.88,
6.0,
1.51,
3.01,
2.11,
4.35,
1.14,
1.8,
5.07,
6.0,
2.43,
6.0,
1.97,
6.0,
6.0,
3.93,
1.57,
1.31,
5.01,
3.79,
1.34,
5.43,
1.48,
```

```
1.26,
1.02,
1.63,
3.74,
1.16,
6.0,
1.29,
2.93,
6.0,
1.01,
6.0,
1.02,
2.73,
2.0,
1.11,
1.49,
4.1,
1.0,
5.61,
1.19,
1.03,
1.22,
1.27,
2.54,
1.36,
1.16,
1.14,
2.85,
1.38,
1.36,
6.0,
1.28,
1.25,
2.47,
3.92,
2.87,
1.09,
1.02,
1.59,
1.72,
1.28,
1.19,
3.48,
1.79,
1.0,
1.08,
3.69,
```

```
1.32,
6.0,
1.42,
1.0,
4.26,
1.37,
4.83,
1.65,
1.36,
6.0,
6.0,
6.0,
2.95,
2.53,
6.0,
1.42,
1.65,
4.93,
1.3,
1.76,
1.21,
4.58,
1.84,
3.43,
6.0,
6.0,
1.61,
1.16,
4.51,
6.0,
2.75,
6.0,
1.65,
1.48,
1.53,
1.09,
1.12,
1.89,
3.44,
5.54,
1.37,
6.0,
3.48,
6.0,
1.07,
1.53,
1.14,
```

2.26,
1.01,
1.08,
1.01,
1.87,
1.04,
1.87,
3.55,
6.0,
1.05,
2.15,
1.54,
1.06,
1.35,
1.14,
2.66,
2.07,
1.37,
3.49,
1.09,
5.71,
1.85,
6.0,
1.5,
1.58,
2.32,
1.82,
3.7,
1.01,
6.0,
2.69,
3.86,
1.37,
1.74,
2.34,
1.75,
1.95,
1.25,
6.0,
6.0,
4.43,
2.03,
4.2,
6.0,
1.0,
1.08,
1.65,

1.09,
3.88,
3.35,
1.34,
1.64,
2.08,
2.59,
1.1,
1.08,
3.16,
6.0,
1.39,
6.0,
2.74,
3.18,
1.22,
1.12,
1.0,
1.5,
6.0,
1.73,
1.29,
6.0,
1.59,
1.55,
1.21,
1.7,
1.66,
4.99,
1.02,
1.73,
1.19,
2.93,
1.35,
1.07,
2.48,
1.12,
1.28,
1.96,
4.83,
1.2,
1.06,
1.09,
1.04,
2.14,
6.0,
2.36,

```
1.22,
6.0,
1.19,
1.17,
3.17,
2.85,
1.66,
2.03,
3.91,
4.78,
2.15,
1.23,
1.39,
2.1,
6.0,
2.43,
2.0,
1.12,
1.95,
1.0,
6.0,
1.0,
1.25,
1.58,
1.75,
1.81,
6.0,
3.14,
5.62,
1.37,
6.0,
1.0,
1.51,
1.43,
2.12,
1.33,
1.6,
1.06,
1.05,
1.01,
1.08,
1.67,
1.96,
5.43,
1.21,
1.11,
6.0,
```

2.13,
1.04,
2.98,
2.78,
1.87,
1.66,
1.09,
1.83,
1.7,
6.0,
1.35,
1.47,
1.01,
1.09,
1.21,
1.87,
1.51,
1.59,
2.17,
4.36,
3.61,
1.3,
1.06,
1.17,
1.11,
4.19,
1.78,
1.71,
1.12,
2.36,
1.71,
6.0,
1.2,
4.29,
1.23,
1.67,
1.02,
2.33,
1.0,
6.0,
3.95,
1.77,
1.33,
6.0,
2.49,
2.24,
6.0,

```
       3.22,
       1.49]
```

[16]:
```
matriz2 = matriz(60, array3)
matriz2
```

[16]:
```
array([[6.  , 5.63, 1.21, 6.  , 3.45, 1.8 , 1.  , 2.15, 1.59, 1.6 ],
       [2.02, 4.44, 1.26, 2.47, 1.07, 5.07, 1.08, 1.54, 1.55, 1.06],
       [2.02, 3.91, 1.01, 6.  , 6.  , 6.  , 3.69, 1.06, 1.21, 1.05],
       [1.54, 6.  , 1.21, 1.52, 1.19, 2.43, 1.32, 1.35, 1.7 , 1.01],
       [2.2 , 2.44, 3.08, 1.5 , 1.05, 6.  , 6.  , 1.14, 1.66, 1.08],
       [1.51, 1.43, 2.7 , 2.36, 1.67, 1.97, 1.42, 2.66, 4.99, 1.67],
       [6.  , 1.83, 2.3 , 1.  , 1.1 , 6.  , 1.  , 2.07, 1.02, 1.96],
       [3.89, 1.31, 2.51, 6.  , 6.  , 6.  , 4.26, 1.37, 1.73, 5.43],
       [1.02, 1.61, 1.17, 1.87, 1.02, 3.93, 1.37, 3.49, 1.19, 1.21],
       [1.48, 1.73, 1.02, 2.11, 1.53, 1.57, 4.83, 1.09, 2.93, 1.11],
       [1.25, 1.4 , 4.1 , 5.53, 2.38, 1.31, 1.65, 5.71, 1.35, 6.  ],
       [1.07, 2.12, 1.17, 1.46, 1.38, 5.01, 1.36, 1.85, 1.07, 2.13],
       [1.93, 1.49, 1.61, 1.14, 1.28, 3.79, 6.  , 6.  , 2.48, 1.04],
       [4.33, 5.97, 6.  , 6.  , 5.21, 1.34, 6.  , 1.5 , 1.12, 2.98],
       [3.59, 1.12, 2.74, 1.  , 6.  , 5.43, 6.  , 1.58, 1.28, 2.78],
       [3.26, 1.55, 1.67, 2.3 , 1.17, 1.48, 2.95, 2.32, 1.96, 1.87],
       [1.83, 1.4 , 5.5 , 1.41, 1.06, 1.26, 2.53, 1.82, 4.83, 1.66],
       [1.6 , 2.62, 1.41, 3.8 , 3.69, 1.02, 6.  , 3.7 , 1.2 , 1.09],
       [2.07, 2.87, 1.13, 1.03, 1.04, 1.63, 1.42, 1.01, 1.06, 1.83],
       [6.  , 1.77, 1.  , 3.41, 1.54, 3.74, 1.65, 6.  , 1.09, 1.7 ],
       [1.21, 1.02, 1.49, 6.  , 1.45, 1.16, 4.93, 2.69, 1.04, 6.  ],
       [5.46, 1.28, 2.3 , 1.  , 1.47, 6.  , 1.3 , 3.86, 2.14, 1.35],
       [3.81, 6.  , 1.31, 1.39, 3.41, 1.29, 1.76, 1.37, 6.  , 1.47],
       [1.3 , 2.18, 1.18, 2.36, 6.  , 2.93, 1.21, 1.74, 2.36, 1.01],
       [1.95, 1.99, 6.  , 2.18, 1.08, 6.  , 4.58, 2.34, 1.22, 1.09],
       [1.04, 4.71, 6.  , 1.3 , 2.15, 1.01, 1.84, 1.75, 6.  , 1.21],
       [1.28, 1.01, 3.16, 1.1 , 1.38, 6.  , 3.43, 1.95, 1.19, 1.87],
       [1.03, 1.15, 4.08, 1.87, 6.  , 1.02, 6.  , 1.25, 1.17, 1.51],
       [1.05, 1.41, 6.  , 3.76, 1.02, 2.73, 6.  , 6.  , 3.17, 1.59],
       [3.39, 6.  , 6.  , 1.52, 4.11, 2.  , 1.61, 6.  , 2.85, 2.17],
       [1.05, 1.92, 6.  , 6.  , 2.2 , 1.11, 1.16, 4.43, 1.66, 4.36],
       [2.59, 1.2 , 2.16, 2.08, 4.17, 1.49, 4.51, 2.03, 2.03, 3.61],
       [2.53, 1.  , 5.1 , 4.89, 2.7 , 4.1 , 6.  , 4.2 , 3.91, 1.3 ],
       [6.  , 2.35, 6.  , 5.59, 1.04, 1.  , 2.75, 6.  , 4.78, 1.06],
       [1.34, 6.  , 1.65, 1.18, 2.14, 5.61, 6.  , 1.  , 2.15, 1.17],
       [1.56, 1.24, 1.  , 6.  , 1.37, 1.19, 1.65, 1.08, 1.23, 1.11],
       [1.  , 4.81, 6.  , 2.66, 1.88, 1.03, 1.48, 1.65, 1.39, 4.19],
       [1.22, 1.  , 2.12, 1.65, 2.85, 1.22, 1.53, 1.09, 2.1 , 1.78],
       [1.05, 1.27, 1.1 , 1.02, 6.  , 1.27, 1.09, 3.88, 6.  , 1.71],
       [6.  , 1.49, 6.  , 1.59, 1.09, 2.54, 1.12, 3.35, 2.43, 1.12],
       [2.54, 1.94, 1.13, 6.  , 2.08, 1.36, 1.89, 1.34, 2.  , 2.36],
```

```
       [2.44, 1.98, 1.64, 1.28, 4.32, 1.16, 3.44, 1.64, 1.12, 1.71],
       [2.34, 5.93, 1.01, 1.42, 1.08, 1.14, 5.54, 2.08, 1.95, 6.  ],
       [6.  , 1.76, 2.11, 1.08, 5.52, 2.85, 1.37, 2.59, 1.  , 1.2 ],
       [6.  , 1.9 , 1.21, 2.57, 2.37, 1.38, 6.  , 1.1 , 6.  , 4.29],
       [1.44, 1.71, 1.71, 1.02, 2.47, 1.36, 3.48, 1.08, 1.  , 1.23],
       [1.63, 6.  , 3.63, 5.11, 1.66, 6.  , 6.  , 3.16, 1.25, 1.67],
       [2.57, 6.  , 6.  , 1.96, 1.3 , 1.28, 1.07, 6.  , 1.58, 1.02],
       [1.73, 6.  , 6.  , 1.26, 6.  , 1.25, 1.53, 1.39, 1.75, 2.33],
       [1.22, 2.03, 3.42, 1.17, 2.6 , 2.47, 1.14, 6.  , 1.81, 1.  ],
       [1.34, 1.14, 6.  , 1.34, 6.  , 3.92, 2.26, 2.74, 6.  , 6.  ],
       [1.33, 1.29, 5.45, 1.05, 1.65, 2.87, 1.01, 3.18, 3.14, 3.95],
       [1.  , 6.  , 2.62, 1.18, 1.  , 1.09, 1.08, 1.22, 5.62, 1.77],
       [2.87, 1.  , 3.2 , 2.99, 1.88, 1.02, 1.01, 1.12, 1.37, 1.33],
       [2.28, 2.21, 6.  , 1.08, 6.  , 1.59, 1.87, 1.  , 6.  , 6.  ],
       [6.  , 3.09, 3.1 , 2.19, 1.51, 1.72, 1.04, 1.5 , 1.  , 2.49],
       [1.  , 1.49, 3.65, 1.08, 3.01, 1.28, 1.87, 6.  , 1.51, 2.24],
       [1.73, 3.34, 2.88, 1.14, 2.11, 1.19, 3.55, 1.73, 1.43, 6.  ],
       [1.05, 1.04, 1.74, 6.  , 4.35, 3.48, 6.  , 1.29, 2.12, 3.22],
       [1.52, 2.26, 6.  , 2.29, 1.14, 1.79, 1.05, 6.  , 1.33, 1.49]])
```

[20]:
```
matriz3 = matriz(60,array4)
matriz3
```

[20]:
```
array([[1, 1, 0, 1, 1, 0, 0, 0, 0, 0],
       [0, 1, 0, 0, 0, 1, 0, 0, 0, 0],
       [0, 1, 0, 1, 1, 1, 1, 0, 0, 0],
       [0, 1, 0, 0, 0, 0, 0, 0, 0, 0],
       [0, 0, 1, 0, 0, 1, 1, 0, 0, 0],
       [0, 0, 0, 0, 0, 0, 0, 0, 1, 0],
       [1, 0, 0, 0, 0, 1, 0, 0, 0, 0],
       [1, 0, 0, 1, 1, 1, 1, 0, 0, 1],
       [0, 0, 0, 0, 0, 1, 0, 1, 0, 0],
       [0, 0, 0, 0, 0, 0, 1, 0, 0, 0],
       [0, 0, 1, 1, 0, 0, 0, 1, 0, 1],
       [0, 0, 0, 0, 0, 1, 0, 0, 0, 0],
       [0, 0, 0, 0, 0, 1, 1, 1, 0, 0],
       [1, 1, 1, 1, 1, 0, 1, 0, 0, 0],
       [1, 0, 0, 0, 1, 1, 1, 0, 0, 0],
       [1, 0, 0, 0, 0, 0, 0, 0, 0, 0],
       [0, 0, 1, 0, 0, 0, 0, 0, 1, 0],
       [0, 0, 0, 1, 1, 0, 1, 1, 0, 0],
       [0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
       [1, 0, 0, 1, 0, 1, 0, 1, 0, 0],
       [0, 0, 0, 1, 0, 0, 1, 0, 0, 1],
       [1, 0, 0, 0, 0, 1, 0, 1, 0, 0],
       [1, 1, 0, 0, 1, 0, 0, 0, 1, 0],
       [0, 0, 0, 0, 1, 0, 0, 0, 0, 0],
```

```
        [0, 0, 1, 0, 0, 1, 1, 0, 0, 0],
        [0, 1, 1, 0, 0, 0, 0, 0, 1, 0],
        [0, 0, 1, 0, 0, 1, 1, 0, 0, 0],
        [0, 0, 1, 0, 1, 0, 1, 0, 0, 0],
        [0, 0, 1, 1, 0, 0, 1, 1, 1, 0],
        [1, 1, 1, 0, 1, 0, 0, 1, 0, 0],
        [0, 0, 1, 1, 0, 0, 0, 1, 0, 1],
        [0, 0, 0, 0, 1, 0, 1, 0, 0, 1],
        [0, 0, 1, 1, 0, 1, 1, 1, 1, 0],
        [1, 0, 1, 1, 0, 0, 0, 1, 1, 0],
        [0, 1, 0, 0, 0, 1, 1, 0, 0, 0],
        [0, 0, 0, 1, 0, 0, 0, 0, 0, 0],
        [0, 1, 1, 0, 0, 0, 0, 0, 0, 1],
        [0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
        [0, 0, 0, 0, 1, 0, 0, 1, 1, 0],
        [1, 0, 1, 0, 0, 0, 0, 1, 0, 0],
        [0, 0, 0, 1, 0, 0, 0, 0, 0, 0],
        [0, 0, 0, 0, 1, 0, 1, 0, 0, 0],
        [0, 1, 0, 0, 0, 0, 1, 0, 0, 1],
        [1, 0, 0, 0, 1, 0, 0, 0, 0, 0],
        [1, 0, 0, 0, 0, 0, 1, 0, 1, 1],
        [0, 0, 0, 0, 0, 0, 1, 0, 0, 0],
        [0, 1, 1, 1, 0, 1, 1, 1, 0, 0],
        [0, 1, 1, 0, 0, 0, 0, 1, 0, 0],
        [0, 1, 1, 0, 1, 0, 0, 0, 0, 0],
        [0, 0, 1, 0, 0, 0, 0, 1, 0, 0],
        [0, 0, 1, 0, 1, 1, 0, 0, 1, 1],
        [0, 0, 1, 0, 0, 0, 0, 1, 1, 1],
        [0, 1, 0, 0, 0, 0, 0, 0, 1, 0],
        [0, 0, 1, 0, 0, 0, 0, 0, 0, 0],
        [0, 0, 1, 0, 1, 0, 0, 0, 1, 1],
        [1, 1, 1, 0, 0, 0, 0, 0, 0, 0],
        [0, 0, 1, 0, 1, 0, 0, 1, 0, 0],
        [0, 1, 0, 0, 0, 0, 1, 0, 0, 1],
        [0, 0, 0, 1, 1, 1, 1, 0, 0, 1],
        [0, 0, 1, 0, 0, 0, 0, 1, 0, 0]])
```

[27]:
```
array5 = []
for i in range(len(array4) - 1):
    if i >= 59:
        order = sum(array4[i - 59: i])
        array5.append(order)
array5
```

[27]:
```
[15,
 14,
 15,
```

16,
17,
18,
18,
17,
16,
16,
16,
16,
16,
16,
16,
15,
15,
14,
14,
14,
14,
13,
13,
12,
11,
12,
12,
12,
13,
13,
13,
12,
13,
13,
13,
12,
12,
13,
13,
14,
14,
13,
13,
13,
13,
13,
12,
12,
12,
13,
14,

```
15,
15,
15,
15,
16,
16,
15,
16,
16,
17,
17,
16,
15,
14,
13,
13,
14,
14,
14,
14,
14,
14,
15,
15,
14,
15,
15,
15,
16,
16,
16,
16,
16,
15,
15,
15,
15,
15,
16,
17,
18,
18,
19,
20,
20,
21,
21,
21,
```

```
20,
21,
21,
21,
22,
22,
21,
21,
21,
21,
20,
20,
20,
21,
22,
23,
23,
23,
24,
24,
25,
25,
25,
25,
26,
27,
27,
27,
28,
27,
27,
27,
27,
27,
28,
28,
28,
27,
28,
28,
28,
27,
28,
28,
28,
27,
27,
28,
28,
28,
29,
30,
30,
```

```
30,
29,
28,
27,
26,
25,
25,
24,
25,
24,
24,
25,
25,
25,
25,
25,
24,
24,
25,
25,
25,
25,
25,
24,
24,
23,
22,
21,
20,
20,
19,
18,
17,
16,
16,
16,
16,
15,
16,
15,
16,
16,
16,
16,
15,
16,
16,
```

15,
15,
15,
14,
15,
16,
16,
15,
16,
15,
14,
14,
14,
15,
16,
16,
16,
16,
16,
16,
16,
16,
16,
16,
15,
15,
14,
14,
14,
14,
15,
14,
14,
15,
15,
16,
16,
15,
15,
15,
16,
16,
17,
17,
17,
17,
18,
18,

```
19,
18,
19,
18,
18,
18,
19,
19,
20,
20,
20,
21,
22,
22,
22,
23,
23,
22,
23,
23,
22,
22,
22,
23,
23,
23,
22,
22,
23,
23,
23,
23,
22,
22,
21,
21,
22,
22,
23,
23,
23,
22,
22,
22,
21,
21,
20,
```

```
20,
20,
20,
21,
20,
20,
19,
20,
20,
20,
19,
19,
18,
18,
17,
18,
18,
17,
16,
17,
16,
17,
16,
15,
15,
15,
16,
15,
14,
15,
15,
16,
16,
16,
17,
16,
16,
16,
16,
16,
16,
15,
16,
15,
16,
17,
18,
18,
```

```
18,
18,
19,
18,
19,
19,
19,
19,
19,
19,
19,
20,
21,
21,
22,
22,
23,
23,
23,
22,
22,
22,
22,
22,
22,
22,
22,
22,
22,
23,
23,
23,
22,
22,
21,
21,
21,
20,
20,
20,
20,
21,
20,
20,
19,
19,
19,
18,
```

```
19,
19,
19,
19,
20,
20,
19,
19,
18,
17,
16,
17,
18,
18,
17,
18,
18,
18,
18,
18,
18,
19,
20,
19,
18,
18,
17,
16,
15,
16,
17,
17,
18,
18,
19,
19,
19,
19,
19,
19,
19,
18,
18,
19,
19,
19,
19,
19,
```

```
19,
20,
20,
19,
19,
18,
18,
17,
17,
17,
17,
17,
17,
17,
17,
16,
16,
15,
15,
16,
16,
16,
16,
17,
17,
16,
16,
15,
15,
14,
14,
15,
15,
15,
15,
14,
14,
14,
14,
14,
14,
14,
14,
14,
15,
14,
13,
13,
12,
12,
12,
```

13,
14,
14,
15,
14,
14,
14,
13,
13,
13,
13,
13,
13,
12,
12,
12,
13,
13,
13,
14,
14,
14,
14,
14,
13,
13,
13,
13,
13,
14,
13,
13,
13,
12,
12,
12,
11,
11,
11,
12,
12,
11,
11,
11,
11,
12,
11,

```
        11,
        11,
        11,
        11,
        12,
        11,
        12,
        12,
        12,
        12,
        12,
        11,
        11,
        11,
        11,
        10,
        11,
        11,
        11,
        12]
```

[30]:
```
matriz4 = matriz(60, array5)
matriz4
```

[30]:
```
array([[15, 17, 25, 16, 19, 18, 23, 18, 13],
       [14, 16, 26, 15, 18, 18, 23, 19, 13],
       [15, 15, 27, 16, 18, 17, 23, 19, 13],
       [16, 14, 27, 15, 18, 16, 22, 19, 13],
       [17, 13, 28, 16, 19, 17, 22, 19, 13],
       [18, 13, 27, 16, 19, 16, 21, 19, 13],
       [18, 14, 27, 16, 20, 17, 21, 19, 12],
       [17, 14, 27, 16, 20, 16, 21, 20, 12],
       [16, 14, 27, 15, 20, 15, 20, 20, 12],
       [16, 14, 28, 16, 21, 15, 20, 19, 13],
       [16, 14, 28, 16, 22, 15, 20, 19, 13],
       [16, 14, 27, 15, 22, 16, 20, 18, 13],
       [16, 15, 28, 15, 22, 15, 21, 18, 14],
       [16, 15, 28, 15, 23, 14, 20, 17, 14],
       [15, 14, 27, 14, 23, 15, 20, 17, 14],
       [15, 15, 28, 15, 22, 15, 19, 17, 14],
       [14, 15, 28, 16, 23, 16, 19, 17, 14],
       [14, 15, 27, 16, 23, 16, 19, 17, 13],
       [14, 16, 27, 15, 22, 16, 18, 17, 13],
       [14, 16, 28, 16, 22, 17, 19, 17, 13],
       [13, 16, 28, 15, 22, 16, 19, 16, 13],
       [13, 16, 29, 14, 23, 16, 19, 16, 13],
       [12, 16, 30, 14, 23, 16, 19, 15, 14],
```

```
       [11, 15, 30, 14, 23, 16, 20, 15, 13],
       [12, 15, 30, 15, 22, 16, 20, 16, 13],
       [12, 15, 29, 16, 22, 15, 19, 16, 13],
       [12, 15, 28, 16, 23, 16, 19, 16, 12],
       [13, 16, 27, 16, 23, 15, 18, 17, 12],
       [13, 17, 26, 16, 23, 16, 17, 17, 12],
       [13, 18, 25, 16, 23, 17, 16, 16, 11],
       [12, 18, 25, 16, 22, 18, 17, 16, 11],
       [13, 19, 24, 16, 22, 18, 18, 15, 11],
       [13, 20, 25, 16, 21, 18, 18, 15, 12],
       [13, 20, 24, 16, 21, 18, 17, 14, 12],
       [12, 21, 24, 15, 22, 19, 18, 14, 11],
       [12, 21, 25, 15, 22, 18, 18, 15, 11],
       [13, 21, 25, 14, 23, 19, 18, 15, 11],
       [13, 20, 25, 14, 23, 19, 18, 15, 11],
       [14, 21, 25, 14, 23, 19, 18, 15, 12],
       [14, 21, 25, 14, 22, 19, 18, 14, 11],
       [13, 21, 24, 15, 22, 19, 19, 14, 11],
       [13, 22, 24, 14, 22, 19, 20, 14, 11],
       [13, 22, 25, 14, 21, 19, 19, 14, 11],
       [13, 21, 25, 15, 21, 20, 18, 14, 11],
       [13, 21, 25, 15, 20, 21, 18, 14, 12],
       [12, 21, 25, 16, 20, 21, 17, 14, 11],
       [12, 21, 25, 16, 20, 22, 16, 15, 12],
       [12, 20, 24, 15, 20, 22, 15, 14, 12],
       [13, 20, 24, 15, 21, 23, 16, 13, 12],
       [14, 20, 23, 15, 20, 23, 17, 13, 12],
       [15, 21, 22, 16, 20, 23, 17, 12, 12],
       [15, 22, 21, 16, 19, 22, 18, 12, 11],
       [15, 23, 20, 17, 20, 22, 18, 12, 11],
       [15, 23, 20, 17, 20, 22, 19, 13, 11],
       [16, 23, 19, 17, 20, 22, 19, 14, 11],
       [16, 24, 18, 17, 19, 22, 19, 14, 10],
       [15, 24, 17, 18, 19, 22, 19, 15, 11],
       [16, 25, 16, 18, 18, 22, 19, 14, 11],
       [16, 25, 16, 19, 18, 22, 19, 14, 11],
       [17, 25, 16, 18, 17, 22, 18, 14, 12]])
```

[34]:
```
array6 = []
array7 = []
for i in range(len(array4) - 1):
    array6.append(array4[i])
    if i >= 59:
        order = float(np.mean(array6))
        array7.append(order)

array7
```

```
[34]: [0.25,
       0.26229508196721313,
       0.27419354838709675,
       0.2857142857142857,
       0.296875,
       0.2923076923076923,
       0.2878787878787879,
       0.2835820895522388,
       0.27941176470588236,
       0.2753623188405797,
       0.2714285714285714,
       0.2676056338028169,
       0.2638888888888889,
       0.2602739726027397,
       0.2702702702702703,
       0.26666666666666666,
       0.2631578947368421,
       0.2597402597402597,
       0.2564102564102564,
       0.25316455696202533,
       0.25,
       0.24691358024691357,
       0.24390243902439024,
       0.25301204819277107,
       0.25,
       0.24705882352941178,
       0.2558139534883721,
       0.25287356321839083,
       0.25,
       0.24719101123595505,
       0.25555555555555554,
       0.25274725274725274,
       0.25,
       0.24731182795698925,
       0.24468085106382978,
       0.25263157894736843,
       0.25,
       0.25773195876288657,
       0.25510204081632654,
       0.25252525252525254,
       0.25,
       0.24752475247524752,
       0.24509803921568626,
       0.2524271844660194,
       0.25,
       0.24761904761904763,
       0.24528301886792453,
```

0.2523364485981308,
0.25925925925925924,
0.26605504587155965,
0.2636363636363636,
0.26126126126126126,
0.25892857142857145,
0.26548672566371684,
0.2631578947368421,
0.2608695652173913,
0.2672413793103448,
0.26495726495726496,
0.2711864406779661,
0.2689075630252101,
0.26666666666666666,
0.2644628099173554,
0.26229508196721313,
0.2601626016260163,
0.25806451612903225,
0.264,
0.2619047619047619,
0.25984251968503935,
0.2578125,
0.2558139534883721,
0.25384615384615383,
0.2595419847328244,
0.25757575757575757,
0.2556390977443609,
0.26119402985074625,
0.25925925925925924,
0.25735294117647056,
0.26277372262773724,
0.2608695652173913,
0.2589928057553957,
0.2571428571428571,
0.2553191489361702,
0.2535211267605634,
0.2517482517482518,
0.25,
0.25517241379310346,
0.2602739726027397,
0.2653061224489796,
0.2702702702702703,
0.2751677852348993,
0.28,
0.2847682119205298,
0.28289473684210525,
0.2875816993464052,

```
0.2922077922077922,
0.2903225806451613,
0.28846153846153844,
0.2929936305732484,
0.2911392405063291,
0.2893081761006289,
0.29375,
0.2919254658385093,
0.29012345679012347,
0.2883435582822086,
0.2865853658536585,
0.28484848484848485,
0.28313253012048195,
0.2874251497005988,
0.2916666666666667,
0.2958579881656805,
0.3,
0.30409356725146197,
0.3081395348837209,
0.3063583815028902,
0.3103448275862069,
0.3142857142857143,
0.3181818181818182,
0.3220338983050847,
0.3202247191011236,
0.31843575418994413,
0.32222222222222224,
0.3259668508287293,
0.3241758241758242,
0.32786885245901637,
0.32608695652173914,
0.32432432432432434,
0.3225806451612903,
0.32085561497326204,
0.324468085106383,
0.32275132275132273,
0.32105263157894737,
0.32460732984293195,
0.3229166666666667,
0.32124352331606215,
0.3247422680412371,
0.3230769230769231,
0.32142857142857145,
0.3197969543147208,
0.32323232323232326,
0.32160804020100503,
0.325,
```

0.3283582089552239,
0.32673267326732675,
0.3251231527093596,
0.3235294117647059,
0.32195121951219513,
0.32038834951456313,
0.3188405797101449,
0.3173076923076923,
0.32057416267942584,
0.319047619047619,
0.3222748815165877,
0.32075471698113206,
0.323943661971831,
0.32710280373831774,
0.32558139534883723,
0.3287037037037037,
0.3271889400921659,
0.3256880733944954,
0.3242009132420091,
0.32272727272727275,
0.3257918552036199,
0.32432432432432434,
0.32286995515695066,
0.32142857142857145,
0.32,
0.3185840707964602,
0.32158590308370044,
0.3201754385964912,
0.31877729257641924,
0.3173913043478261,
0.31601731601731603,
0.3146551724137931,
0.3133047210300429,
0.31196581196581197,
0.31063829787234043,
0.3093220338983051,
0.3080168776371308,
0.3067226890756303,
0.30962343096234307,
0.30833333333333335,
0.3112033195020747,
0.30991735537190085,
0.31275720164609055,
0.3114754098360656,
0.31020408163265306,
0.3089430894308943,
0.3076923076923077,

```
0.31048387096774194,
0.3092369477911647,
0.308,
0.30677290836653387,
0.3055555555555556,
0.30434782608695654,
0.30708661417322836,
0.30980392156862746,
0.30859375,
0.30739299610894943,
0.31007751937984496,
0.3088803088803089,
0.3076923076923077,
0.3065134099616858,
0.3053435114503817,
0.30798479087452474,
0.3106060606060606,
0.30943396226415093,
0.3082706766917293,
0.30711610486891383,
0.30970149253731344,
0.30855018587360594,
0.3111111111111111,
0.30996309963099633,
0.3125,
0.31135531135531136,
0.3102189781021898,
0.3090909090909091,
0.3079710144927536,
0.30685920577617326,
0.3057553956834532,
0.30824372759856633,
0.30714285714285716,
0.30604982206405695,
0.30851063829787234,
0.30742049469964666,
0.30985915492957744,
0.3087719298245614,
0.3076923076923077,
0.30662020905923343,
0.3055555555555556,
0.3079584775086505,
0.30689655172413793,
0.30927835051546393,
0.3082191780821918,
0.30716723549488056,
0.30612244897959184,
```

0.30847457627118646,
0.30743243243243246,
0.30976430976430974,
0.3087248322147651,
0.3110367892976589,
0.31,
0.3089700996677741,
0.31125827814569534,
0.31353135313531355,
0.3125,
0.31475409836065577,
0.3137254901960784,
0.31596091205211724,
0.3181818181818182,
0.32038834951456313,
0.3193548387096774,
0.3183279742765273,
0.32051282051282054,
0.3226837060702875,
0.321656050955414,
0.3238095238095238,
0.3227848101265823,
0.3217665615141956,
0.32075471698113206,
0.31974921630094044,
0.321875,
0.32087227414330216,
0.32298136645962733,
0.3219814241486068,
0.32098765432098764,
0.3230769230769231,
0.3220858895705521,
0.3241590214067278,
0.3231707317073171,
0.3221884498480243,
0.3212121212121212,
0.3202416918429003,
0.3192771084337349,
0.3213213213213213,
0.3203592814371258,
0.32238805970149254,
0.32142857142857145,
0.32047477744807124,
0.31952662721893493,
0.3185840707964602,
0.3176470588235294,
0.31671554252199413,

```
0.3157894736842105,
0.31486880466472306,
0.313953488372093,
0.3130434782608696,
0.31213872832369943,
0.31412103746397696,
0.3132183908045977,
0.3123209169054441,
0.31142857142857144,
0.31339031339031337,
0.3125,
0.311614730878187,
0.3107344632768362,
0.30985915492957744,
0.3089887640449438,
0.3081232492997199,
0.30726256983240224,
0.30919220055710306,
0.30833333333333335,
0.3074792243767313,
0.30662983425414364,
0.3085399449035813,
0.3076923076923077,
0.3095890410958904,
0.3087431693989071,
0.3079019073569482,
0.30978260869565216,
0.3089430894308943,
0.3108108108108108,
0.30997304582210244,
0.30913978494623656,
0.3109919571045576,
0.31283422459893045,
0.31466666666666665,
0.31382978723404253,
0.3129973474801061,
0.3148148148148148,
0.31398416886543534,
0.3131578947368421,
0.31496062992125984,
0.31413612565445026,
0.3133159268929504,
0.3125,
0.3142857142857143,
0.3134715025906736,
0.3152454780361757,
0.3170103092783505,
```

0.31876606683804626,
0.31794871794871793,
0.3171355498721228,
0.31887755102040816,
0.32061068702290074,
0.3197969543147208,
0.32151898734177214,
0.3207070707070707,
0.3198992443324937,
0.31909547738693467,
0.3182957393483709,
0.3175,
0.3167082294264339,
0.31840796019900497,
0.3200992555831266,
0.3193069306930693,
0.32098765432098764,
0.3226600985221675,
0.32432432432432434,
0.3235294117647059,
0.32273838630806845,
0.32195121951219513,
0.32116788321167883,
0.32038834951456313,
0.3196125907990315,
0.3188405797101449,
0.3180722891566265,
0.3173076923076923,
0.31654676258992803,
0.3181818181818182,
0.3198090692124105,
0.319047619047619,
0.3182897862232779,
0.3175355450236967,
0.31678486997635935,
0.3160377358490566,
0.31529411764705884,
0.3145539906103286,
0.31381733021077285,
0.3130841121495327,
0.3146853146853147,
0.313953488372093,
0.31554524361948955,
0.3148148148148148,
0.3163972286374134,
0.315668202764977,
0.31494252873563217,

```
0.31422018348623854,
0.3135011441647597,
0.3150684931506849,
0.3143507972665148,
0.3159090909090909,
0.31519274376417233,
0.3167420814479638,
0.3160270880361174,
0.3153153153153153,
0.3146067415730337,
0.31390134529147984,
0.3131991051454139,
0.3125,
0.31403118040089084,
0.31555555555555553,
0.3170731707317073,
0.3163716814159292,
0.31788079470198677,
0.31938325991189426,
0.31868131868131866,
0.31798245614035087,
0.3172866520787746,
0.3165938864628821,
0.31808278867102396,
0.31956521739130433,
0.3188720173535792,
0.3181818181818182,
0.3174946004319654,
0.3168103448275862,
0.3161290322580645,
0.315450643776824,
0.3169164882226981,
0.31837606837606836,
0.31769722814498935,
0.3191489361702128,
0.3184713375796178,
0.3199152542372881,
0.3192389006342495,
0.31856540084388185,
0.3178947368421053,
0.3172268907563025,
0.31865828092243187,
0.3179916317991632,
0.3173277661795407,
0.31875,
0.3180873180873181,
0.31742738589211617,
```

0.3167701863354037,
0.31611570247933884,
0.3154639175257732,
0.3168724279835391,
0.3162217659137577,
0.3155737704918033,
0.3149284253578732,
0.3142857142857143,
0.3136456211812627,
0.3130081300813008,
0.31237322515212984,
0.3117408906882591,
0.3111111111111111,
0.31048387096774194,
0.3118712273641851,
0.3112449799196787,
0.3106212424849699,
0.31,
0.3093812375249501,
0.30876494023904383,
0.3101391650099404,
0.30952380952380953,
0.3089108910891089,
0.3102766798418972,
0.3096646942800789,
0.3090551181102362,
0.3104125736738703,
0.30980392156862746,
0.30919765166340507,
0.30859375,
0.30994152046783624,
0.311284046692607,
0.3106796116504854,
0.31007751937984496,
0.30947775628626695,
0.3088803088803089,
0.31021194605009633,
0.3096153846153846,
0.30902111324376197,
0.30842911877394635,
0.3078393881453155,
0.30725190839694655,
0.30857142857142855,
0.30798479087452474,
0.30740037950664134,
0.3068181818181818,
0.30623818525519847,

0.30566037735849055,
0.3069679849340866,
0.3082706766917293,
0.30956848030018763,
0.3089887640449438,
0.3102803738317757,
0.30970149253731344,
0.3091247672253259,
0.30855018587360594,
0.3079777365491651,
0.3074074074074074,
0.3068391866913124,
0.3062730627306273,
0.30570902394106814,
0.30514705882352944,
0.30458715596330277,
0.304029304029304,
0.30347349177330896,
0.30474452554744524,
0.30418943533697634,
0.30363636363636365,
0.30490018148820325,
0.30434782608695654,
0.3037974683544304,
0.30324909747292417,
0.3027027027027027,
0.302158273381295,
0.3016157989228007,
0.3010752688172043,
0.3005366726296959,
0.3,
0.30124777183600715,
0.30071174377224197,
0.30017761989342806,
0.299645390070922,
0.2991150442477876,
0.29858657243816256,
0.2980599647266314,
0.2975352112676056,
0.29701230228471004,
0.2964912280701754,
0.29772329246935203,
0.29895104895104896,
0.29842931937172773,
0.2979094076655052,
0.29739130434782607,
0.296875,

```
 0.29809358752166376,
 0.2975778546712803,
 0.2970639032815199,
 0.296551724137931,
 0.29604130808950085,
 0.29553264604810997,
 0.2967409948542024,
 0.2962328767123288,
 0.29743589743589743,
 0.29692832764505117,
 0.29642248722316866,
 0.29591836734693877,
 0.29541595925297115,
 0.29491525423728815,
 0.2961082910321489,
 0.2972972972972973,
 0.29679595278246207,
 0.2962962962962963,
 0.29747899159663865,
 0.29697986577181207,
 0.2964824120603015,
 0.2976588628762542,
 0.2988313856427379]
```

[37]:
```
matriz5 = matriz(60, array7)
matriz5
```

[37]:
```
array([[0.25      , 0.26666667, 0.32222222, 0.30833333, 0.31      ,
        0.30833333, 0.31904762, 0.31875   , 0.30740741],
       [0.26229508, 0.26446281, 0.32596685, 0.31120332, 0.3089701 ,
        0.30747922, 0.31828979, 0.31808732, 0.30683919],
       [0.27419355, 0.26229508, 0.32417582, 0.30991736, 0.31125828,
        0.30662983, 0.31753555, 0.31742739, 0.30627306],
       [0.28571429, 0.2601626 , 0.32786885, 0.3127572 , 0.31353135,
        0.30853994, 0.31678487, 0.31677019, 0.30570902],
       [0.296875  , 0.25806452, 0.32608696, 0.31147541, 0.3125    ,
        0.30769231, 0.31603774, 0.3161157 , 0.30514706],
       [0.29230769, 0.264     , 0.32432432, 0.31020408, 0.3147541 ,
        0.30958904, 0.31529412, 0.31546392, 0.30458716],
       [0.28787879, 0.26190476, 0.32258065, 0.30894309, 0.31372549,
        0.30874317, 0.31455399, 0.31687243, 0.3040293 ],
       [0.28358209, 0.25984252, 0.32085561, 0.30769231, 0.31596091,
        0.30790191, 0.31381733, 0.31622177, 0.30347349],
       [0.27941176, 0.2578125 , 0.32446809, 0.31048387, 0.31818182,
        0.30978261, 0.31308411, 0.31557377, 0.30474453],
       [0.27536232, 0.25581395, 0.32275132, 0.30923695, 0.32038835,
        0.30894309, 0.31468531, 0.31492843, 0.30418944],
```

```
[0.27142857, 0.25384615, 0.32105263, 0.308     , 0.31935484,
 0.31081081, 0.31395349, 0.31428571, 0.30363636],
[0.26760563, 0.25954198, 0.32460733, 0.30677291, 0.31832797,
 0.30997305, 0.31554524, 0.31364562, 0.30490018],
[0.26388889, 0.25757576, 0.32291667, 0.30555556, 0.32051282,
 0.30913978, 0.31481481, 0.31300813, 0.30434783],
[0.26027397, 0.2556391 , 0.32124352, 0.30434783, 0.32268371,
 0.31099196, 0.31639723, 0.31237323, 0.30379747],
[0.27027027, 0.26119403, 0.32474227, 0.30708661, 0.32165605,
 0.31283422, 0.3156682 , 0.31174089, 0.3032491 ],
[0.26666667, 0.25925926, 0.32307692, 0.30980392, 0.32380952,
 0.31466667, 0.31494253, 0.31111111, 0.3027027 ],
[0.26315789, 0.25735294, 0.32142857, 0.30859375, 0.32278481,
 0.31382979, 0.31422018, 0.31048387, 0.30215827],
[0.25974026, 0.26277372, 0.31979695, 0.307393  , 0.32176656,
 0.31299735, 0.31350114, 0.31187123, 0.3016158 ],
[0.25641026, 0.26086957, 0.32323232, 0.31007752, 0.32075472,
 0.31481481, 0.31506849, 0.31124498, 0.30107527],
[0.25316456, 0.25899281, 0.32160804, 0.30888031, 0.31974922,
 0.31398417, 0.3143508 , 0.31062124, 0.30053667],
[0.25      , 0.25714286, 0.325     , 0.30769231, 0.321875  ,
 0.31315789, 0.31590909, 0.31      , 0.3       ],
[0.24691358, 0.25531915, 0.32835821, 0.30651341, 0.32087227,
 0.31496063, 0.31519274, 0.30938124, 0.30124777],
[0.24390244, 0.25352113, 0.32673267, 0.30534351, 0.32298137,
 0.31413613, 0.31674208, 0.30876494, 0.30071174],
[0.25301205, 0.25174825, 0.32512315, 0.30798479, 0.32198142,
 0.31331593, 0.31602709, 0.31013917, 0.30017762],
[0.25      , 0.25      , 0.32352941, 0.31060606, 0.32098765,
 0.3125    , 0.31531532, 0.30952381, 0.29964539],
[0.24705882, 0.25517241, 0.32195122, 0.30943396, 0.32307692,
 0.31428571, 0.31460674, 0.30891089, 0.29911504],
[0.25581395, 0.26027397, 0.32038835, 0.30827068, 0.32208589,
 0.3134715 , 0.31390135, 0.31027668, 0.29858657],
[0.25287356, 0.26530612, 0.31884058, 0.3071161 , 0.32415902,
 0.31524548, 0.31319911, 0.30966469, 0.29805996],
[0.25      , 0.27027027, 0.31730769, 0.30970149, 0.32317073,
 0.31701031, 0.3125    , 0.30905512, 0.29753521],
[0.24719101, 0.27516779, 0.32057416, 0.30855019, 0.32218845,
 0.31876607, 0.31403118, 0.31041257, 0.2970123 ],
[0.25555556, 0.28      , 0.31904762, 0.31111111, 0.32121212,
 0.31794872, 0.31555556, 0.30980392, 0.29649123],
[0.25274725, 0.28476821, 0.32227488, 0.3099631 , 0.32024169,
 0.31713555, 0.31707317, 0.30919765, 0.29772329],
[0.25      , 0.28289474, 0.32075472, 0.3125    , 0.31927711,
 0.31887755, 0.31637168, 0.30859375, 0.29895105],
[0.24731183, 0.2875817 , 0.32394366, 0.31135531, 0.32132132,
```

```
 0.32061069, 0.31788079, 0.30994152, 0.29842932],
[0.24468085, 0.29220779, 0.3271028 , 0.31021898, 0.32035928,
 0.31979695, 0.31938326, 0.31128405, 0.29790941],
[0.25263158, 0.29032258, 0.3255814 , 0.30909091, 0.32238806,
 0.32151899, 0.31868132, 0.31067961, 0.2973913 ],
[0.25      , 0.28846154, 0.3287037 , 0.30797101, 0.32142857,
 0.32070707, 0.31798246, 0.31007752, 0.296875  ],
[0.25773196, 0.29299363, 0.32718894, 0.30685921, 0.32047478,
 0.31989924, 0.31728665, 0.30947776, 0.29809359],
[0.25510204, 0.29113924, 0.32568807, 0.3057554 , 0.31952663,
 0.31909548, 0.31659389, 0.30888031, 0.29757785],
[0.25252525, 0.28930818, 0.32420091, 0.30824373, 0.31858407,
 0.31829574, 0.31808279, 0.31021195, 0.2970639 ],
[0.25      , 0.29375   , 0.32272727, 0.30714286, 0.31764706,
 0.3175    , 0.31956522, 0.30961538, 0.29655172],
[0.24752475, 0.29192547, 0.32579186, 0.30604982, 0.31671554,
 0.31670823, 0.31887202, 0.30902111, 0.29604131],
[0.24509804, 0.29012346, 0.32432432, 0.30851064, 0.31578947,
 0.31840796, 0.31818182, 0.30842912, 0.29553265],
[0.25242718, 0.28834356, 0.32286996, 0.30742049, 0.3148688 ,
 0.32009926, 0.3174946 , 0.30783939, 0.29674099],
[0.25      , 0.28658537, 0.32142857, 0.30985915, 0.31395349,
 0.31930693, 0.31681034, 0.30725191, 0.29623288],
[0.24761905, 0.28484848, 0.32      , 0.30877193, 0.31304348,
 0.32098765, 0.31612903, 0.30857143, 0.2974359 ],
[0.24528302, 0.28313253, 0.31858407, 0.30769231, 0.31213873,
 0.3226601 , 0.31545064, 0.30798479, 0.29692833],
[0.25233645, 0.28742515, 0.3215859 , 0.30662021, 0.31412104,
 0.32432432, 0.31691649, 0.30740038, 0.29642249],
[0.25925926, 0.29166667, 0.32017544, 0.30555556, 0.31321839,
 0.32352941, 0.31837607, 0.30681818, 0.29591837],
[0.26605505, 0.29585799, 0.31877729, 0.30795848, 0.31232092,
 0.32273839, 0.31769723, 0.30623819, 0.29541596],
[0.26363636, 0.3       , 0.3173913 , 0.30689655, 0.31142857,
 0.32195122, 0.31914894, 0.30566038, 0.29491525],
[0.26126126, 0.30409357, 0.31601732, 0.30927835, 0.31339031,
 0.32116788, 0.31847134, 0.30696798, 0.29610829],
[0.25892857, 0.30813953, 0.31465517, 0.30821918, 0.3125    ,
 0.32038835, 0.31991525, 0.30827068, 0.2972973 ],
[0.26548673, 0.30635838, 0.31330472, 0.30716724, 0.31161473,
 0.31961259, 0.3192389 , 0.30956848, 0.29679595],
[0.26315789, 0.31034483, 0.31196581, 0.30612245, 0.31073446,
 0.31884058, 0.3185654 , 0.30898876, 0.2962963 ],
[0.26086957, 0.31428571, 0.3106383 , 0.30847458, 0.30985915,
 0.31807229, 0.31789474, 0.31028037, 0.29747899],
[0.26724138, 0.31818182, 0.30932203, 0.30743243, 0.30898876,
 0.31730769, 0.31722689, 0.30970149, 0.29697987],
```

```
       [0.26495726, 0.3220339 , 0.30801688, 0.30976431, 0.30812325,
        0.31654676, 0.31865828, 0.30912477, 0.29648241],
       [0.27118644, 0.32022472, 0.30672269, 0.30872483, 0.30726257,
        0.31818182, 0.31799163, 0.30855019, 0.29765886],
       [0.26890756, 0.31843575, 0.30962343, 0.31103679, 0.3091922 ,
        0.31980907, 0.31732777, 0.30797774, 0.29883139]])
```

[38]:
```
matriz2.shape, matriz3.shape, matriz4.shape, matriz5.shape
```

[38]:
```
((60, 10), (60, 10), (60, 9), (60, 9))
```

[40]:
```
matrizfloat = matriz2[:,1:]
matrizint = matriz3[:,1:]
```

[43]:
```
matrizfloat.shape, matrizint.shape, matriz4.shape, matriz5.shape
```

[43]:
```
((60, 9), (60, 9), (60, 9), (60, 9))
```

[47]:
```
len(array2), len(array3), len(array4), len(array5), len(array7)
```

[47]:
```
(1672, 600, 600, 540, 540)
```

[59]:
```
x1 = matrizfloat[:,:(matrizfloat.shape[1] - 1)]
x2 = matriz4[:,:(matriz4.shape[1] - 1)]
x3 = matriz5[:,:(matriz5.shape[1] - 1)]

y = matrizint[:,-1]
```

[60]:
```
# Empilhar as matrizes para ter um eixo extra (60, 8, 3)
X_stack = np.stack([x1, x2, x3], axis=2)  # Formato (60, 8, 3)

# Reorganizar para intercalar coluna por coluna
X_intercalado = X_stack.reshape(60, -1)  # Agora está no formato (60, 24)

print(X_intercalado.shape)  # Saída: (60, 24)
```

```
(60, 24)
```

[53]:
```
import numpy as np
import pandas as pd
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split
import xgboost as xgb
from sklearn.metrics import accuracy_score, f1_score

X = X_intercalado

# Dividir os dados em treino e teste
```

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,␣
 ↪random_state=42)

# Criar e treinar o modelo XGBoost
model = xgb.XGBClassifier(
    objective='multi:softmax',  # Classificação multiclasse
    num_class=2,  # Número de categorias na saída
    eval_metric='mlogloss',
    learning_rate=0.01,
    n_estimators=100,
    max_depth=6,
    subsample=0.8,
    colsample_bytree=0.8,
    random_state=42
)

model.fit(X_train, y_train)

# Fazer previsões
y_pred = model.predict(X_test)

# Avaliar o modelo
accuracy = accuracy_score(y_test, y_pred)
f1 = f1_score(y_test, y_pred, average='weighted')

print(f'Acurácia do modelo: {accuracy:.4f}')
print(f'F1-Score do modelo: {f1:.4f}')
```

```
Acurácia do modelo: 0.5000
F1-Score do modelo: 0.4444
```

[54]:
```
from sklearn.model_selection import train_test_split
from tensorflow.keras.metrics import Precision, Recall
from tensorflow import keras
from tensorflow.keras import layers
import tensorflow as tf
import tensorflow_addons as tfa
#activation = tf.keras.activations.elu
from tensorflow.keras.optimizers import Nadam

import skfuzzy as fuzz

# Libs
import time
import warnings
```

```
2025-03-02 12:38:14.693946: I external/local_xla/xla/tsl/cuda/cudart_stub.cc:32]
Could not find cuda drivers on your machine, GPU will not be used.
```

```
2025-03-02 12:38:14.877703: I external/local_xla/xla/tsl/cuda/cudart_stub.cc:32]
Could not find cuda drivers on your machine, GPU will not be used.
2025-03-02 12:38:14.991632: E
external/local_xla/xla/stream_executor/cuda/cuda_fft.cc:477] Unable to register
cuFFT factory: Attempting to register factory for plugin cuFFT when one has
already been registered
WARNING: All log messages before absl::InitializeLog() is called are written to
STDERR
E0000 00:00:1740933495.144493   24559 cuda_dnn.cc:8310] Unable to register cuDNN
factory: Attempting to register factory for plugin cuDNN when one has already
been registered
E0000 00:00:1740933495.182876   24559 cuda_blas.cc:1418] Unable to register
cuBLAS factory: Attempting to register factory for plugin cuBLAS when one has
already been registered
2025-03-02 12:38:15.557656: I tensorflow/core/platform/cpu_feature_guard.cc:210]
This TensorFlow binary is optimized to use available CPU instructions in
performance-critical operations.
To enable the following instructions: AVX2 FMA, in other operations, rebuild
TensorFlow with the appropriate compiler flags.
/home/darkcover/Documentos/Out/venv/lib/python3.12/site-
packages/tensorflow_addons/utils/tfa_eol_msg.py:23: UserWarning:

TensorFlow Addons (TFA) has ended development and introduction of new features.
TFA has entered a minimal maintenance and release mode until a planned end of
life in May 2024.
Please modify downstream libraries to take dependencies from other repositories
in our TensorFlow community (e.g. Keras, Keras-CV, and Keras-NLP).

For more information see: https://github.com/tensorflow/addons/issues/2807

  warnings.warn(
/home/darkcover/Documentos/Out/venv/lib/python3.12/site-
packages/tensorflow_addons/utils/ensure_tf_install.py:53: UserWarning:
Tensorflow Addons supports using Python ops for all Tensorflow versions above or
equal to 2.13.0 and strictly below 2.16.0 (nightly versions are not supported).
 The versions of TensorFlow you are currently using is 2.18.0 and is not
supported.
Some things might work, some things might not.
If you were to encounter a bug, do not file an issue.
If you want to make sure you're using a tested and supported configuration,
either change the TensorFlow version or the TensorFlow Addons's version.
You can find the compatibility matrix in TensorFlow Addon's readme:
https://github.com/tensorflow/addons
  warnings.warn(
```

[84]:
```python
def reden(array1, array3, m, n):
    print(m, n)
```

```
   # Dividindo os dados em treino e teste
   X = np.array(array3).astype("float32")  # Certificar que X está no formato␣
↪correto
   y = np.array(array1)

   x_train, x_test, y_train, y_test = train_test_split(X, y, test_size=0.2,␣
↪random_state=42)

   # Ajustando dimensões corretamente
   print("Shape inicial de x_train:", x_train.shape)  # Deve ser (48, 24, 1)

   input_shape = (n, 1)  # Usando n diretamente

   # Verificar os valores de y_train antes da conversão
   print("Valores únicos em y_train antes da conversão:", np.unique(y_train))

   # Se necessário, garantir que y_train só tenha 0 e 1
   y_train = np.where(y_train > 0, 1, 0)
   y_test = np.where(y_test > 0, 1, 0)

   # Converter para categórico
   num_classes = 2
   y_train = keras.utils.to_categorical(y_train, num_classes)
   y_test = keras.utils.to_categorical(y_test, num_classes)

   print("Shape de y_train após conversão:", y_train.shape)  # Deve ser (48, 2)
   print("Shape de x_train após conversão:", x_train.shape)  # Deve ser (48,␣
↪24, 1)

   # Definição do modelo
   model = keras.Sequential([
       keras.Input(shape=input_shape),
       layers.Flatten(),
       layers.Dense(128, activation="relu", use_bias=True),
       layers.Dropout(0.5),
       layers.Dense(64, activation="relu", use_bias=True),
       layers.Dense(32, activation=tf.keras.activations.swish, use_bias=True),
       layers.Dense(num_classes, activation="softmax"),
   ])

   model.compile(
       loss=tfa.losses.SigmoidFocalCrossEntropy(alpha=0.25, gamma=2.0),
       optimizer=tf.keras.optimizers.AdamW(learning_rate=0.001,␣
↪weight_decay=1e-4),
       metrics=['accuracy', Precision(name="precision"), Recall(name="recall")]
   )
```

```
    # Treinamento
    batch_size = 2**10
    epochs = 50
    model.fit(
        x_train, y_train,
        batch_size=batch_size,
        epochs=epochs,
        validation_split=0.2
    )

    # Avaliação
    score = model.evaluate(x_test, y_test, verbose=0)
    print(f"Test loss: {score[0]:.4f}")
    print(f"Test accuracy: {score[1]:.4f}")
    print(f"Precision: {score[2]:.4f}")
    print(f"Recall: {score[3]:.4f}")

    return model
```

[62]:
```
y
```

[62]:
```
array([0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0,
       1, 0, 0, 0, 0, 0, 1, 1, 0, 0, 1, 0, 0, 1, 1, 0])
```

[65]:
```
X.shape
```

[65]:
```
(60, 24)
```

[70]:
```
np.unique(y)
```

[70]:
```
array([0, 1])
```

[85]:
```
X = X_intercalado
model = reden(y, X, X.shape[0], X.shape[1])
```

```
60 24
Shape inicial de x_train: (48, 24)
Valores únicos em y_train antes da conversão: [0 1]
Shape de y_train após conversão: (48, 2)
Shape de x_train após conversão: (48, 24)
Epoch 1/50
1/1             4s 4s/step -
accuracy: 0.8684 - loss: 0.9758 - precision: 0.8684 - recall: 0.8684 -
val_accuracy: 0.6000 - val_loss: 1.5366 - val_precision: 0.6000 - val_recall:
0.6000
Epoch 2/50
```

```
1/1             0s 87ms/step -
accuracy: 0.8421 - loss: 1.1782 - precision: 0.8421 - recall: 0.8421 -
val_accuracy: 0.6000 - val_loss: 0.6279 - val_precision: 0.6000 - val_recall:
0.6000
Epoch 3/50
1/1             0s 91ms/step -
accuracy: 0.7895 - loss: 1.1674 - precision: 0.7895 - recall: 0.7895 -
val_accuracy: 0.7000 - val_loss: 0.1801 - val_precision: 0.7000 - val_recall:
0.7000
Epoch 4/50
1/1             0s 93ms/step -
accuracy: 0.6316 - loss: 1.0721 - precision: 0.6316 - recall: 0.6316 -
val_accuracy: 0.3000 - val_loss: 0.2713 - val_precision: 0.3000 - val_recall:
0.3000
Epoch 5/50
1/1             0s 86ms/step -
accuracy: 0.5789 - loss: 0.6631 - precision: 0.5789 - recall: 0.5789 -
val_accuracy: 0.4000 - val_loss: 0.3196 - val_precision: 0.4000 - val_recall:
0.4000
Epoch 6/50
1/1             0s 85ms/step -
accuracy: 0.6316 - loss: 0.9773 - precision: 0.6316 - recall: 0.6316 -
val_accuracy: 0.5000 - val_loss: 0.2963 - val_precision: 0.5000 - val_recall:
0.5000
Epoch 7/50
1/1             0s 92ms/step -
accuracy: 0.5789 - loss: 0.6510 - precision: 0.5789 - recall: 0.5789 -
val_accuracy: 0.6000 - val_loss: 0.2150 - val_precision: 0.6000 - val_recall:
0.6000
Epoch 8/50
1/1             0s 88ms/step -
accuracy: 0.7632 - loss: 0.5641 - precision: 0.7632 - recall: 0.7632 -
val_accuracy: 0.6000 - val_loss: 0.2119 - val_precision: 0.6000 - val_recall:
0.6000
Epoch 9/50
1/1             0s 84ms/step -
accuracy: 0.8158 - loss: 0.8105 - precision: 0.8158 - recall: 0.8158 -
val_accuracy: 0.5000 - val_loss: 0.2618 - val_precision: 0.5000 - val_recall:
0.5000
Epoch 10/50
1/1             3s 3s/step -
accuracy: 0.8158 - loss: 0.3802 - precision: 0.8158 - recall: 0.8158 -
val_accuracy: 0.5000 - val_loss: 0.3237 - val_precision: 0.5000 - val_recall:
0.5000
Epoch 11/50
1/1             0s 91ms/step -
accuracy: 0.8947 - loss: 0.3701 - precision: 0.8947 - recall: 0.8947 -
val_accuracy: 0.6000 - val_loss: 0.3599 - val_precision: 0.6000 - val_recall:
```

```
0.6000
Epoch 12/50
1/1             0s 98ms/step -
accuracy: 0.7632 - loss: 0.3748 - precision: 0.7632 - recall: 0.7632 -
val_accuracy: 0.6000 - val_loss: 0.3824 - val_precision: 0.6000 - val_recall:
0.6000
Epoch 13/50
1/1             0s 116ms/step -
accuracy: 0.7632 - loss: 0.4366 - precision: 0.7632 - recall: 0.7632 -
val_accuracy: 0.6000 - val_loss: 0.3838 - val_precision: 0.6000 - val_recall:
0.6000
Epoch 14/50
1/1             0s 89ms/step -
accuracy: 0.8947 - loss: 0.2992 - precision: 0.8947 - recall: 0.8947 -
val_accuracy: 0.5000 - val_loss: 0.3503 - val_precision: 0.5000 - val_recall:
0.5000
Epoch 15/50
1/1             0s 87ms/step -
accuracy: 0.8421 - loss: 0.1535 - precision: 0.8421 - recall: 0.8421 -
val_accuracy: 0.5000 - val_loss: 0.3124 - val_precision: 0.5000 - val_recall:
0.5000
Epoch 16/50
1/1             0s 93ms/step -
accuracy: 0.7632 - loss: 0.4323 - precision: 0.7632 - recall: 0.7632 -
val_accuracy: 0.5000 - val_loss: 0.2681 - val_precision: 0.5000 - val_recall:
0.5000
Epoch 17/50
1/1             0s 92ms/step -
accuracy: 0.7105 - loss: 0.4474 - precision: 0.7105 - recall: 0.7105 -
val_accuracy: 0.4000 - val_loss: 0.2340 - val_precision: 0.4000 - val_recall:
0.4000
Epoch 18/50
1/1             0s 88ms/step -
accuracy: 0.8684 - loss: 0.2197 - precision: 0.8684 - recall: 0.8684 -
val_accuracy: 0.4000 - val_loss: 0.2203 - val_precision: 0.4000 - val_recall:
0.4000
Epoch 19/50
1/1             0s 92ms/step -
accuracy: 0.8421 - loss: 0.2717 - precision: 0.8421 - recall: 0.8421 -
val_accuracy: 0.5000 - val_loss: 0.2212 - val_precision: 0.5000 - val_recall:
0.5000
Epoch 20/50
1/1             0s 96ms/step -
accuracy: 0.8421 - loss: 0.1402 - precision: 0.8421 - recall: 0.8421 -
val_accuracy: 0.5000 - val_loss: 0.2308 - val_precision: 0.5000 - val_recall:
0.5000
Epoch 21/50
1/1             0s 127ms/step -
```

```
accuracy: 0.6842 - loss: 0.2427 - precision: 0.6842 - recall: 0.6842 -
val_accuracy: 0.6000 - val_loss: 0.2421 - val_precision: 0.6000 - val_recall:
0.6000
Epoch 22/50
1/1             0s 229ms/step -
accuracy: 0.7895 - loss: 0.3221 - precision: 0.7895 - recall: 0.7895 -
val_accuracy: 0.6000 - val_loss: 0.2475 - val_precision: 0.6000 - val_recall:
0.6000
Epoch 23/50
1/1             0s 93ms/step -
accuracy: 0.8158 - loss: 0.1843 - precision: 0.8158 - recall: 0.8158 -
val_accuracy: 0.6000 - val_loss: 0.2425 - val_precision: 0.6000 - val_recall:
0.6000
Epoch 24/50
1/1             0s 85ms/step -
accuracy: 0.6842 - loss: 0.3558 - precision: 0.6842 - recall: 0.6842 -
val_accuracy: 0.6000 - val_loss: 0.2239 - val_precision: 0.6000 - val_recall:
0.6000
Epoch 25/50
1/1             0s 92ms/step -
accuracy: 0.6316 - loss: 0.2586 - precision: 0.6316 - recall: 0.6316 -
val_accuracy: 0.6000 - val_loss: 0.1988 - val_precision: 0.6000 - val_recall:
0.6000
Epoch 26/50
1/1             0s 103ms/step -
accuracy: 0.7895 - loss: 0.1816 - precision: 0.7895 - recall: 0.7895 -
val_accuracy: 0.5000 - val_loss: 0.1853 - val_precision: 0.5000 - val_recall:
0.5000
Epoch 27/50
1/1             0s 105ms/step -
accuracy: 0.8158 - loss: 0.1668 - precision: 0.8158 - recall: 0.8158 -
val_accuracy: 0.5000 - val_loss: 0.1865 - val_precision: 0.5000 - val_recall:
0.5000
Epoch 28/50
1/1             0s 90ms/step -
accuracy: 0.7895 - loss: 0.1876 - precision: 0.7895 - recall: 0.7895 -
val_accuracy: 0.4000 - val_loss: 0.2067 - val_precision: 0.4000 - val_recall:
0.4000
Epoch 29/50
1/1             0s 98ms/step -
accuracy: 0.8684 - loss: 0.3042 - precision: 0.8684 - recall: 0.8684 -
val_accuracy: 0.5000 - val_loss: 0.2275 - val_precision: 0.5000 - val_recall:
0.5000
Epoch 30/50
1/1             0s 87ms/step -
accuracy: 0.8684 - loss: 0.3129 - precision: 0.8684 - recall: 0.8684 -
val_accuracy: 0.5000 - val_loss: 0.2368 - val_precision: 0.5000 - val_recall:
0.5000
```

```
Epoch 31/50
1/1             0s 96ms/step -
accuracy: 0.8684 - loss: 0.2290 - precision: 0.8684 - recall: 0.8684 -
val_accuracy: 0.5000 - val_loss: 0.2270 - val_precision: 0.5000 - val_recall:
0.5000
Epoch 32/50
1/1             0s 89ms/step -
accuracy: 0.8421 - loss: 0.3663 - precision: 0.8421 - recall: 0.8421 -
val_accuracy: 0.5000 - val_loss: 0.2159 - val_precision: 0.5000 - val_recall:
0.5000
Epoch 33/50
1/1             0s 89ms/step -
accuracy: 0.8684 - loss: 0.3159 - precision: 0.8684 - recall: 0.8684 -
val_accuracy: 0.4000 - val_loss: 0.1957 - val_precision: 0.4000 - val_recall:
0.4000
Epoch 34/50
1/1             0s 88ms/step -
accuracy: 0.7895 - loss: 0.2135 - precision: 0.7895 - recall: 0.7895 -
val_accuracy: 0.4000 - val_loss: 0.1840 - val_precision: 0.4000 - val_recall:
0.4000
Epoch 35/50
1/1             0s 87ms/step -
accuracy: 0.8684 - loss: 0.1710 - precision: 0.8684 - recall: 0.8684 -
val_accuracy: 0.4000 - val_loss: 0.1837 - val_precision: 0.4000 - val_recall:
0.4000
Epoch 36/50
1/1             0s 90ms/step -
accuracy: 0.8158 - loss: 0.1571 - precision: 0.8158 - recall: 0.8158 -
val_accuracy: 0.8000 - val_loss: 0.1921 - val_precision: 0.8000 - val_recall:
0.8000
Epoch 37/50
1/1             0s 91ms/step -
accuracy: 0.8684 - loss: 0.2534 - precision: 0.8684 - recall: 0.8684 -
val_accuracy: 0.7000 - val_loss: 0.1973 - val_precision: 0.7000 - val_recall:
0.7000
Epoch 38/50
1/1             0s 103ms/step -
accuracy: 0.6316 - loss: 0.2976 - precision: 0.6316 - recall: 0.6316 -
val_accuracy: 0.7000 - val_loss: 0.1969 - val_precision: 0.7000 - val_recall:
0.7000
Epoch 39/50
1/1             0s 89ms/step -
accuracy: 0.8158 - loss: 0.1997 - precision: 0.8158 - recall: 0.8158 -
val_accuracy: 0.6000 - val_loss: 0.1927 - val_precision: 0.6000 - val_recall:
0.6000
Epoch 40/50
1/1             0s 181ms/step -
accuracy: 0.8158 - loss: 0.1635 - precision: 0.8158 - recall: 0.8158 -
```

```
val_accuracy: 0.7000 - val_loss: 0.1866 - val_precision: 0.7000 - val_recall:
0.7000
Epoch 41/50
1/1             0s 118ms/step -
accuracy: 0.8684 - loss: 0.1144 - precision: 0.8684 - recall: 0.8684 -
val_accuracy: 0.5000 - val_loss: 0.1822 - val_precision: 0.5000 - val_recall:
0.5000
Epoch 42/50
1/1             0s 91ms/step -
accuracy: 0.7368 - loss: 0.3052 - precision: 0.7368 - recall: 0.7368 -
val_accuracy: 0.5000 - val_loss: 0.1807 - val_precision: 0.5000 - val_recall:
0.5000
Epoch 43/50
1/1             0s 86ms/step -
accuracy: 0.8158 - loss: 0.1804 - precision: 0.8158 - recall: 0.8158 -
val_accuracy: 0.4000 - val_loss: 0.1858 - val_precision: 0.4000 - val_recall:
0.4000
Epoch 44/50
1/1             0s 169ms/step -
accuracy: 0.7368 - loss: 0.1784 - precision: 0.7368 - recall: 0.7368 -
val_accuracy: 0.5000 - val_loss: 0.2061 - val_precision: 0.5000 - val_recall:
0.5000
Epoch 45/50
1/1             0s 133ms/step -
accuracy: 0.8158 - loss: 0.0791 - precision: 0.8158 - recall: 0.8158 -
val_accuracy: 0.6000 - val_loss: 0.2500 - val_precision: 0.6000 - val_recall:
0.6000
Epoch 46/50
1/1             0s 88ms/step -
accuracy: 0.8684 - loss: 0.1770 - precision: 0.8684 - recall: 0.8684 -
val_accuracy: 0.6000 - val_loss: 0.2919 - val_precision: 0.6000 - val_recall:
0.6000
Epoch 47/50
1/1             0s 90ms/step -
accuracy: 0.8684 - loss: 0.1069 - precision: 0.8684 - recall: 0.8684 -
val_accuracy: 0.6000 - val_loss: 0.3275 - val_precision: 0.6000 - val_recall:
0.6000
Epoch 48/50
1/1             0s 86ms/step -
accuracy: 0.7632 - loss: 0.3018 - precision: 0.7632 - recall: 0.7632 -
val_accuracy: 0.6000 - val_loss: 0.3491 - val_precision: 0.6000 - val_recall:
0.6000
Epoch 49/50
1/1             0s 88ms/step -
accuracy: 0.8421 - loss: 0.1619 - precision: 0.8421 - recall: 0.8421 -
val_accuracy: 0.6000 - val_loss: 0.3509 - val_precision: 0.6000 - val_recall:
0.6000
Epoch 50/50
```

```
1/1             0s 91ms/step -
accuracy: 0.8421 - loss: 0.2703 - precision: 0.8421 - recall: 0.8421 -
val_accuracy: 0.6000 - val_loss: 0.3143 - val_precision: 0.6000 - val_recall:
0.6000
Test loss: 0.1995
Test accuracy: 0.6667
Precision: 0.6667
Recall: 0.6667
```