# rasassociac204167a204131o-resposta

April 12, 2024

#Tarefa 4 - Regras de associação

## 0.1 Nesta tarefa, você deve carregar um dataset e minerar regras de associação usando o algoritmo Apriori, visto em aula. As métricas de avaliação das regras mineradas devem observar os cuidados vistos em aula.

Dica: Para toda a tarefa, além da biblioteca pandas e matplotlib, você pode querer explorar funções da biblioteca mlxtend.frequent_patterns (em particular os pacotes apriori e association_rules). Além disso, você vai precisar usar uma função de pré-processamento que transforma a base de dados de transações em uma base de dados de registros adequada para a extração das regras. Busque por TransactionEncoder

###Importe os pacotes e carregue os arquivos com os dados

Os datasets a serem utilizados encontram-se nos arquivos `compras_cafeteria.csv` e `product_data.csv`, disponível no EAD.

**product_data**: dataset que relaciona um número identificador de um produto com o seu nome, sabor, preço e categoria.

**compras_cafeteria**: Dataset de registros de compras de uma cafeteria. Possui as colunas `id` como identificador do cliente e `product` como o número do produto comprado.

[36]:

```python
import pandas as pd
import sklearn
import matplotlib.pyplot as plt
from sklearn.preprocessing import MinMaxScaler
import seaborn as sns
import numpy as np
import mlxtend
```

```
/usr/local/lib/python3.10/dist-packages/ipykernel/ipkernel.py:283:
DeprecationWarning: `should_run_async` will not call `transform_cell`
automatically in the future. Please pass the result to `transformed_cell`
argument and any exception that happen during thetransform in
`preprocessing_exc_tuple` in IPython 7.17 and above.
  and should_run_async(code)
```

[37]:

```python
df = pd.read_csv('compras_cafeteria.csv',sep=',')
df.head()
```

```
/usr/local/lib/python3.10/dist-packages/ipykernel/ipkernel.py:283:
DeprecationWarning: `should_run_async` will not call `transform_cell`
automatically in the future. Please pass the result to `transformed_cell`
argument and any exception that happen during thetransform in
`preprocessing_exc_tuple` in IPython 7.17 and above.
  and should_run_async(code)
```

```
[37]:    id  product_1
      0   1          3
      1   1          4
      2   1          2
      3   1          5
      4   2          1
```

```
[38]: df2 = pd.read_csv('product_data.csv',sep=',')
      df2.head()
```

```
/usr/local/lib/python3.10/dist-packages/ipykernel/ipkernel.py:283:
DeprecationWarning: `should_run_async` will not call `transform_cell`
automatically in the future. Please pass the result to `transformed_cell`
argument and any exception that happen during thetransform in
`preprocessing_exc_tuple` in IPython 7.17 and above.
  and should_run_async(code)
```

```
[38]:    product_number      flavor product  price category
      0               0   Chocolate    Cake   8.95     Food
      1               1       Lemon    Cake   8.95     Food
      2               2      Casino    Cake  15.95     Food
      3               3       Opera    Cake  15.95     Food
      4               4  Strawberry    Cake  11.95     Food
```

### 0.1.1 Transforme as bases de dados

---

primeiro agrupe os itens comprados para cada cliente depois transforme os dados agrupados em uma lista de transações e então crie a base de dados (DataFrame), codificada para minerar as regras de associação (use o TransactionEncoder) após criar a base de dados, troque o nome das features pelos nomes armazenados no dataset product_data.csv (use a função rename) "Some" as colunas flavor e product para conseguir o nome inteiro do produto.

Exemplo:

flavor = "Chocolate"

product = "Cake"

precisamos do nome do produto como "Chocolate Cake"

[39]:
```python
dados_agrupados = df.groupby('id')['product_1'].apply(list).reset_index()
print(dados_agrupados)
```

```
       id                                   product_1
0       1                 [3, 4, 2, 5, 7, 15, 49, 44]
1       2                               [1, 2, 1, 19]
2       3                               [1, 1, 1, 19]
3       4    [1, 1, 5, 5, 1, 1, 18, 35, 3, 15, 44, 4]
4       5             [4, 4, 2, 5, 5, 4, 9, 23, 2, 7]
..    …                                         …
995   996  [3, 2, 3, 5, 3, 4, 46, 33, 31, 10, 2, 22]
996   997                  [3, 5, 4, 2, 22, 2, 32, 6]
997   998            [1, 3, 5, 4, 5, 4, 9, 0, 33, 30]
998   999                        [2, 5, 2, 18, 35, 3]
999  1000                       [4, 3, 3, 15, 47, 34]

[1000 rows x 2 columns]
```

```
/usr/local/lib/python3.10/dist-packages/ipykernel/ipkernel.py:283:
DeprecationWarning: `should_run_async` will not call `transform_cell`
automatically in the future. Please pass the result to `transformed_cell`
argument and any exception that happen during thetransform in
`preprocessing_exc_tuple` in IPython 7.17 and above.
  and should_run_async(code)
```

[40]:
```python
lista_transacoes = dados_agrupados['product_1'].tolist()
print(lista_transacoes)
```

```
[[3, 4, 2, 5, 7, 15, 49, 44], [1, 2, 1, 19], [1, 1, 1, 19], [1, 1, 5, 5, 1, 1,
18, 35, 3, 15, 44, 4], [4, 4, 2, 5, 5, 4, 9, 23, 2, 7], [2, 4, 3, 14, 44, 21],
[4, 3, 2, 1, 1, 1, 12, 31, 36, 48, 44, 4], [5, 1, 3, 28, 27, 15], [3, 1, 28, 2],
[5, 2, 1, 18, 35, 3], [4, 5, 1, 2, 5, 23, 24, 40, 41, 43], [1, 4, 3, 43, 48,
20], [4, 49], [2, 3, 2, 1, 19, 26], [5, 3, 4, 22, 5, 39], [2, 2, 1, 16, 32, 45],
[2, 1, 2, 5, 3, 3, 4, 9, 6, 16, 22, 10], [2, 5, 3, 1, 19, 23], [2, 4, 2, 5, 7,
11, 37, 45], [4, 4, 4, 3, 18, 35, 3, 32], [4, 3, 1, 5, 8, 47, 1, 19], [2, 1, 3,
44, 34, 39], [3, 5, 13, 19], [4, 4, 5, 4, 9, 38], [3, 3, 1, 48, 7, 22], [5, 4,
2, 14, 11, 7], [5, 2, 4, 2, 4, 23, 24, 40, 41, 43], [2, 4, 14, 9], [4, 1, 5, 0,
2, 42], [1, 5, 35, 13], [3, 23], [4, 4, 4, 1, 25, 21, 8, 38], [1, 4, 4, 46], [3,
5, 3, 2, 2, 23, 24, 40, 41, 43], [1, 4, 5, 3, 17, 47, 29, 4], [1, 1, 4, 12, 31,
36], [4, 5, 4, 4, 4, 14, 44, 26, 22, 37], [5, 5, 1, 4, 4, 5, 16, 32, 45, 47, 30,
0], [2, 2, 4, 2, 4, 1, 5, 1, 19, 27, 11, 25, 46, 29], [2, 5, 2, 2, 4, 18, 15,
21, 16, 26], [1, 5, 3, 14, 4, 10], [4, 2, 36, 3], [2, 5, 3, 28, 27, 23], [5, 2,
5, 21, 40, 15], [2, 1, 1, 2, 32, 10, 19, 25], [4, 1, 4, 44, 11, 22], [3, 8], [3,
4, 3, 0, 46, 2], [2, 5, 42, 33], [1, 2, 28, 39], [4, 3, 5, 28, 17, 7], [2, 1, 1,
19], [4, 2, 34, 32], [2, 5, 4, 0, 46, 2], [4, 3, 2, 30, 45, 15], [1, 1, 39, 49],
[1, 46], [5, 4, 1, 4, 9, 19], [3, 3, 2, 1, 3, 0, 46, 2, 16, 19], [4, 4, 1, 17,
40, 21], [2, 3, 5, 2, 1, 4, 9, 39, 2, 6], [5, 2, 5, 2, 1, 23, 24, 40, 41, 43],
[4, 2, 1, 28, 27, 13], [2, 40], [3, 1, 2, 14, 44, 12], [3, 3, 28, 27], [3, 3, 1,
36], [2, 4, 3, 1, 12, 31, 36, 48], [1, 5, 5, 18, 35, 3], [1, 4, 2, 12, 5, 16],
[5, 2, 49, 30], [3, 4, 2, 29, 7, 30], [3, 4, 5, 17, 19, 30], [4, 5, 4, 1, 36,
13], [4, 1, 5, 49, 19, 33], [1, 1, 4, 1, 14, 1, 38, 5], [5, 31], [4, 5, 5, 1, 2,
5, 22, 5, 26, 14, 1, 8], [5, 3, 1, 1, 1, 5, 43, 36, 30, 37, 7, 27], [3, 5, 2,
13, 35, 42], [5, 3, 1, 22, 5, 32], [3, 4, 5, 5, 3, 3, 7, 11, 37, 45, 28, 20],
[3, 5, 5, 1, 23, 24, 40, 41], [2, 4, 5, 4, 1, 22, 5, 36, 3, 14], [1, 4, 2, 2,
12, 31, 36, 48], [1, 45], [4, 1, 1, 5, 12, 31, 36, 48], [1, 2, 14, 44], [2, 4,
5, 2, 3, 22, 5, 34, 43, 19], [1, 3, 1, 1, 0, 46, 2, 45], [4, 5, 2, 4, 2, 16, 32,
45, 47, 48], [5, 1, 1, 48], [3, 5, 4, 1, 18, 35, 3, 1], [5, 3, 4, 5, 7, 11, 37,
45], [4, 5, 2, 3, 5, 16, 32, 45, 41, 26], [1, 3, 5, 29, 1, 30], [4, 5, 1, 19],
[1, 1], [1, 5, 1, 22, 37, 14], [4, 4, 4, 22, 34, 40], [5, 31], [4, 5, 2, 17, 16,
43], [5, 3, 1, 2, 20, 48, 19, 8], [4, 5, 4, 5, 2, 3, 3, 5, 12, 31, 36, 48, 27,
15, 1, 34], [4, 3, 1, 3, 40, 45], [2, 3, 3, 2, 4, 2, 47, 48, 41, 21, 18, 13],
[2, 1, 2, 1, 22, 18, 27, 3], [3, 1, 3, 2, 5, 1, 42, 33, 16, 31, 17, 6], [3, 5,
4, 1, 22, 5, 27, 35], [1, 2, 1, 7, 15, 49], [1, 4, 2, 1, 2, 2, 3, 3, 12, 31, 36,
48, 29, 26, 6, 32], [5, 4, 5, 4, 14, 44, 13, 11], [3, 1, 3, 23, 33, 35], [2, 5,
2, 3, 2, 46, 8, 5, 29, 18], [3, 41], [5, 4, 5, 1, 2, 28, 3, 8, 27, 1], [3, 4, 2,
5, 1, 19, 33, 11], [2, 4, 1, 1, 23, 24, 40, 41], [3, 3, 4, 2, 4, 3, 4, 3, 30,
23, 49, 16, 27, 40, 29, 18], [4, 4, 1, 14, 44, 18], [4, 4, 2, 0, 46, 2], [3, 3,
3, 18, 35, 3], [4, 4, 3, 1, 12, 31, 36, 48], [1, 1, 2, 4, 4, 4, 7, 11, 37, 45,
20, 29], [4, 2, 5, 24, 27, 2], [2, 1, 1, 22, 5, 13], [2, 1, 1, 2, 42, 47, 46,
22], [4, 1, 41, 39], [3, 5, 13, 23], [1, 2, 1, 2, 24, 42, 1, 40], [2, 3, 3, 4,
14, 44, 20, 26], [3, 1, 4, 3, 12, 31, 36, 48], [3, 2, 5, 1, 4, 3, 5, 28, 27, 47,
21, 7, 36, 33], [1, 5, 2, 17, 47, 29], [1, 2, 2, 12, 31, 36], [3, 5, 4, 4, 9,
11], [4, 3, 2, 1, 4, 3, 18, 35, 3, 11, 29, 14], [4, 4, 5, 42, 33, 20], [3, 2, 3,
22, 46, 44], [1, 1, 2, 3, 4, 5, 2, 22, 5, 38, 49, 6, 33, 16], [5, 2, 5, 3, 4, 5,
1, 0, 46, 2, 12, 31, 26, 24], [1, 4, 4, 0, 6, 25], [4, 2, 1, 22, 5, 19], [1, 3,
5, 3, 2, 14, 35, 48, 13, 11], [3, 2, 2, 1, 4, 17, 47, 29, 35, 44], [1, 2, 7,
15], [1, 4, 22, 5], [4, 5, 2, 2, 0, 46, 2, 1], [3, 1, 3, 1, 4, 9, 18, 41], [1,
3, 1, 5, 1, 1, 19, 39, 12, 18], [1, 1, 4, 4, 3, 0, 46, 2, 22, 19], [2, 5, 1, 42,
33, 37], [1, 3, 1, 2, 0, 46, 2, 37], [4, 4, 1, 4, 18, 35, 45, 11], [1, 4, 2, 4,
9, 15], [1, 4, 4, 1, 1, 28, 49, 24, 35, 19], [3, 2, 3, 22, 5, 37], [2, 5, 1, 2,
3, 3, 4, 1, 0, 46, 5, 33, 39, 35, 2, 19], [4, 1, 1, 4, 4, 3, 5, 48], [4, 3, 5,
1, 1, 18, 35, 4, 6, 28], [1, 1, 5, 3, 18, 35, 3, 31], [4, 3, 5, 3, 49, 6, 17,
25], [3, 17], [5, 2, 1, 4, 3, 3, 3, 8, 3, 20, 9, 23, 22, 42], [1, 1, 4, 1, 5, 4,
5, 15, 26, 17, 21, 4, 36, 48], [1, 1, 2, 14, 41, 44], [4, 1, 5, 42, 28, 19], [1,
1, 4, 9], [1, 1, 3, 3, 2, 42, 33, 41, 31, 13], [4, 4, 5, 22, 5, 28], [5, 4, 2,
16, 32, 0], [1, 3, 2, 5, 28, 43], [1, 4, 2, 5, 42, 37, 36, 24], [2, 31], [1, 2,
42, 40], [4, 5, 1, 34, 48, 11], [3, 1, 3, 1, 43, 40, 28, 14], [1, 3, 5, 0, 13,
26], [2, 5, 1, 16, 32, 45], [4, 4, 4, 4, 14, 44, 46, 10], [3, 3, 2, 3, 3, 4, 9,
7, 36, 19], [2, 3, 3, 4, 1, 23, 24, 40, 41, 43], [2, 3, 5, 4, 9, 37], [2, 4, 1,
22, 5, 31], [1, 2, 4, 29, 21, 24], [4, 2, 5, 2, 22, 46, 0, 6], [3, 3, 3, 3, 3,
4, 18, 35, 3, 21, 46, 39], [3, 4, 4, 4, 3, 12, 31, 36, 48, 38], [5, 28], [4, 3,
5, 2, 1, 23, 24, 40, 41, 43], [3, 3, 12, 24], [4, 4, 2, 28, 27, 6], [2, 5, 2, 1,
3, 23, 24, 40, 41, 43], [1, 1, 28, 27], [2, 3, 5, 2, 22, 5, 16, 14], [1, 1, 22,
5], [1, 3, 1, 4, 28, 27, 44, 47], [2, 1, 22, 5], [3, 4, 3, 0, 46, 2], [3, 3, 5,
30, 0, 10], [4, 5, 2, 42, 33, 15], [5, 1, 2, 4, 9, 40], [1, 1, 1, 36], [2, 4, 1,
5, 23, 24, 40, 41], [1, 1, 4, 2, 28, 27, 32, 25], [5, 4, 4, 5, 7, 11, 37, 45],
[5, 4, 48, 20], [4, 3, 4, 3, 7, 11, 37, 45], [5, 3, 3, 28, 27, 9], [3, 2, 5, 7,
11, 37], [1, 2, 4, 5, 1, 9, 2, 1, 48, 22], [1, 5, 1, 5, 39, 27, 7, 45], [3, 3,
4, 3, 2, 4, 3, 3, 1, 19, 18, 15, 6, 8, 21, 42], [3, 1, 2, 3, 23, 24, 40, 41],
[3, 4, 1, 2, 4, 1, 44, 14, 20, 32, 4, 36], [4, 3, 4, 16, 32, 45], [3, 2, 5, 5,
26, 18, 45, 42], [4, 4, 2, 11, 2, 23], [2, 3, 4, 11, 35, 15], [5, 4, 5, 4, 1,
43, 46, 31, 45, 39], [4, 3, 1, 5, 3, 0, 46, 2, 28, 40], [4, 4, 1, 2, 3, 2, 26,
35, 39, 13, 24, 47], [2, 3, 1, 3, 1, 36, 7, 30, 21, 23], [1, 1, 4, 40, 23, 47],
[1, 3, 22, 5], [1, 1, 5, 3, 1, 19, 30, 5], [3, 2, 1, 7, 15, 49], [1, 2, 4, 5, 7,
11, 37, 45], [5, 2, 3, 3, 23, 45, 31, 34], [5, 5, 1, 48, 29, 7], [1, 3, 28, 27],
[2, 1, 24, 44], [1, 3, 3, 4, 4, 9, 5, 39], [3, 4, 3, 5, 4, 5, 28, 22, 5, 25, 46,
34], [2, 2, 2, 48, 23, 32], [5, 5, 42, 16], [2, 1, 22, 5], [5, 4, 5, 3, 4, 9,
15, 25], [2, 2, 1, 1, 1, 3, 4, 1, 19, 16, 17, 48, 26, 12], [1, 1, 5, 5, 1, 1,
19, 16, 3, 35], [1, 1, 11, 12], [5, 2, 5, 4, 5, 4, 9, 12, 10, 45], [2, 2, 2, 5,
5, 12, 31, 36, 48, 46], [3, 5, 4, 17, 47, 28], [4, 4, 2, 18, 35, 3], [3, 1, 4,
9], [2, 2, 1, 1, 7, 11, 37, 45], [1, 2, 2, 22, 5, 12], [5, 4, 2, 2, 48, 38, 6,
14], [2, 2, 4, 3, 5, 36, 9, 42, 10, 34], [1, 3, 4, 18, 35, 3], [3, 3], [4, 5,
14, 27], [4, 4, 5, 1, 19, 0], [2, 1, 2, 1, 18, 35, 3, 24], [2, 4, 5, 1, 12, 31,
36, 48], [4, 2, 4, 41, 31, 19], [3, 2, 2, 3, 4, 5, 14, 35, 34, 32, 1, 11], [2,
3, 1, 2, 2, 5, 14, 39, 49, 28], [4, 1, 3, 4, 9, 21, 46, 14], [1, 6], [1, 4, 4,
22, 5, 31], [5, 2, 11, 40], [5, 5, 3, 0, 46, 2], [2, 2, 5, 5, 1, 23, 24, 40, 41,
43], [5, 1, 1, 1, 14, 44, 3, 23], [1, 1, 4, 4, 42, 18, 49, 22], [1, 4, 24, 3],
[3, 1, 2, 2, 12, 31, 36, 48], [5, 5, 3, 20, 25, 27], [4, 5, 4, 39, 15, 30], [2,
3, 2, 7, 15, 49], [1, 5, 4, 4, 12, 31, 36, 48], [1, 3, 1, 3, 16, 32, 42, 18],
[5, 2, 3, 3, 42, 33, 31, 27], [4, 2, 3, 3, 22, 5, 43, 9], [5, 4, 2, 4, 3, 1, 4,
40, 36, 41, 48, 2], [5, 3, 1, 2, 28, 30, 19, 32], [5, 4, 4, 4, 31, 2, 44, 39],
[3, 1, 4, 3, 1, 2, 4, 1, 19, 2, 28, 45, 21, 23], [3, 3, 5, 2, 3, 5, 1, 16, 32,
45, 14, 7, 23, 10], [4, 4, 2, 3, 5, 4, 9, 36, 14, 44], [3, 3, 21, 29], [1, 1, 5,
3, 17, 9, 13, 46], [1, 3, 38, 39], [5, 2, 5, 1, 19, 8], [2, 1, 1, 42, 33, 0],
[4, 5, 5, 4, 2, 36, 47, 33], [1, 3, 2, 12, 31, 36], [3, 28], [4, 4, 1, 28, 27,
16], [4, 4, 1, 7, 15, 49], [2, 2, 3, 2, 1, 4, 3, 4, 40, 33, 44, 42, 34, 8, 21,
24], [4, 2, 4, 1, 5, 1, 32, 14, 26, 41], [2, 4, 1, 42, 33, 23], [4, 3, 2, 3, 24,
21, 23, 14], [5, 4, 1, 0, 46, 2], [5, 2, 11, 29], [5, 4, 3, 1, 1, 42, 33, 1, 23,
47], [3, 2, 3, 16, 32, 45], [4, 4, 1, 1, 2, 3, 9, 37, 33, 43, 3, 4], [2, 4, 4,
3, 4, 30, 46, 25, 43, 19], [2, 4, 4, 0, 46, 2], [1, 3, 3, 34, 16, 17], [3, 5, 1,
2, 4, 3, 3, 22, 5, 25, 21, 31, 42, 27], [4, 5, 4, 3, 1, 12, 31, 36, 16, 5], [1,
3, 4, 22, 8, 42], [3, 2, 5, 3, 27, 36], [5, 2, 2, 16, 32, 45], [2, 5, 2, 5, 12,
31, 36, 48], [3, 2, 22, 5], [5, 1, 5, 4, 7, 11, 37, 45], [4, 2, 5, 4, 5, 5, 5,
3, 13, 41, 20, 39, 21, 9, 45, 24], [1, 3, 4, 2, 4, 1, 37, 15, 9, 24], [2, 4, 4,
9], [2, 5, 2, 18, 35, 45], [2, 2, 1, 0, 46, 2], [4, 1, 1, 2, 14, 5, 47, 10], [2,
2, 3, 5, 2, 2, 3, 1, 36, 10, 25, 4, 49, 14], [2, 1, 12, 34], [4, 5, 1, 7, 15,
49], [2, 4, 23, 28], [3, 2, 1, 19], [4, 2, 4, 0, 46, 2], [2, 4, 2, 1, 19, 2],
[1, 4, 1, 4, 7, 11, 37, 45], [5, 4, 5, 4, 5, 48, 38, 29, 45, 33], [3, 2, 3, 4,
9, 24], [5, 1, 3, 4, 4, 1, 19, 23, 44, 30], [3, 5, 5, 13], [4, 1, 1, 2, 12, 31,
36, 48], [3, 1, 3, 0, 46, 2], [4, 5, 4, 3, 3, 4, 4, 3, 28, 14, 46, 24, 31, 20,
5, 39], [2, 4, 1, 5, 5, 5, 4, 9, 33, 13, 43, 46], [4, 5, 3, 2, 22, 5, 14, 29],
[5, 3, 3, 1, 1, 1, 19, 8, 45, 10, 35, 2], [3, 4, 3, 1, 14, 44, 24, 34], [3, 5,
2, 2, 4, 22, 5, 28, 30, 47], [2, 2, 2, 28, 47, 0], [4, 4, 3, 7, 15, 49], [3, 3,
2, 5, 28, 41, 5, 9], [5, 5, 5, 4, 7, 11, 37, 1], [5, 26], [2, 44], [1, 5, 3, 0,
46, 2], [1, 5, 2, 2, 26, 5, 10, 30], [4, 2, 3, 5, 1, 1, 12, 14, 47, 33, 44, 8],
[1, 4, 2, 5, 27, 3, 4, 42], [2, 2, 2, 1, 45, 11, 9, 40], [4, 2, 4, 5, 4, 3, 4,
9, 11, 3, 12, 47], [2, 4, 5, 22, 5, 35], [5, 5, 5, 45, 29, 26], [3, 1, 5, 4, 9,
31], [2, 2, 5, 5, 5, 2, 28, 27, 36, 0, 13, 31], [1, 3, 3, 17, 47, 32], [4, 3, 4,
41, 31, 1], [3, 2, 4, 14, 44, 43], [5, 3, 18, 35], [2, 1, 4, 10, 16, 5], [1, 1,
33, 37], [3, 2, 1, 4, 30, 31], [3, 1, 5, 25, 9, 14], [2, 4, 14, 44], [4, 4, 4,
28, 27, 23], [2, 5, 2, 22, 18, 9], [4, 5, 1, 22, 5, 8], [1, 2, 10, 29], [1, 1,
4, 4, 1, 5, 20, 15, 45, 16, 33, 3], [3, 1, 2, 3, 5, 34, 25, 12, 4, 8], [4, 4, 2,
28, 9, 26], [3, 4, 4, 4, 7, 15, 49, 9], [2, 2, 5, 48, 49, 3], [4, 2, 4, 1, 2,
31, 43, 21, 5, 30], [3, 4, 4, 9], [3, 3, 2, 1, 19, 28], [3, 1, 5, 4, 5, 4, 9,
33, 26, 47], [5, 3, 3, 2, 19, 13, 1, 41], [2, 3, 3, 18, 41, 15], [5, 1, 5, 14,
28, 48], [2, 4, 2, 22, 5, 47], [4, 49], [5, 3, 1, 7, 15, 49], [3, 3, 2, 28, 8,
47], [2, 2, 5, 25, 8, 29], [3, 3, 2, 1, 4, 10, 0, 21], [4, 41], [2, 3, 2, 3, 44,
5, 23, 26], [5, 25], [2, 2, 4, 9], [3, 5, 4, 3, 5, 5, 17, 47, 29, 31, 26, 12],
[5, 2, 3, 18, 35, 3], [5, 5, 2, 5, 2, 7, 11, 37, 45, 34], [4, 2, 4, 2, 4, 1, 2,
40, 18, 23, 9, 33, 13, 25], [3, 15], [2, 5, 4, 4, 3, 22, 5, 40, 17, 48], [2, 5,
4, 33, 46, 23], [3, 3, 1, 7, 15, 49], [3, 2, 4, 16, 32, 45], [3, 2, 3, 7, 15,
49], [5, 3, 48, 1], [4, 2], [4, 3, 5, 49, 39, 36], [2, 1, 3, 4, 10, 3], [4, 5,
36, 34], [5, 4, 2, 2, 4, 17, 34, 35, 46, 7], [4, 1, 22, 5], [5, 4, 2, 2, 36, 32,
42, 34], [5, 1, 14, 46], [1, 2, 2, 2, 4, 2, 18, 35, 3, 48, 37, 29], [1, 4, 5,
42, 33, 27], [3, 2, 28, 27], [5, 4, 28, 27], [1, 1, 3, 3, 4, 9, 13, 21], [1, 1,
1, 5, 1, 2, 1, 1, 22, 5, 21, 10, 31, 1, 30, 7], [5, 4, 4, 7, 15, 49], [3, 4, 4,
3, 2, 5, 4, 23, 42, 14], [1, 4, 1, 17, 47, 29], [2, 30], [5, 2, 2, 42, 33, 5],
[4, 4, 5, 28, 27, 31], [5, 5, 4, 7, 15, 49], [5, 5, 3, 16, 32, 45], [4, 4, 5, 4,
2, 25, 32, 1, 8, 2], [3, 2, 13, 12], [3, 4, 16, 44], [5, 1, 3, 24, 0, 17], [3,
26], [1, 4, 4, 4, 5, 28, 27, 43, 4, 10], [5, 14], [2, 1, 4, 47, 4, 19], [4, 5,
42, 33], [3, 47], [4, 1, 5, 1, 2, 3, 17, 47, 29, 23, 8, 43], [5, 5, 3, 3, 0, 46,
5, 33], [4, 5, 2, 3, 3, 3, 9, 47, 35, 18, 38, 40], [5, 4, 4, 5, 4, 11, 2, 39],
[3, 5, 2, 4, 4, 4, 9, 5, 24, 23], [1, 3, 4, 24, 4, 32], [4, 3, 47, 8], [3, 4, 2,
19], [5, 5, 5, 2, 4, 9, 2, 40], [2, 4, 2, 0, 46, 2], [2, 3, 2, 1, 42, 33, 19,
11], [3, 3, 1, 16, 32, 45], [2, 2, 2, 2, 4, 22, 5, 7, 42, 32], [2, 4, 1, 4, 47,
33], [1, 3, 3, 27, 36, 19], [2, 3, 5, 1, 40, 28], [2, 23], [1, 2, 3, 5, 9, 33,
4, 31], [1, 5, 5, 28, 4, 47], [3, 2, 1, 1, 25, 27, 49, 34], [2, 4, 1, 18, 35,
3], [5, 1, 2, 28, 27, 9], [1, 1, 1, 4, 36, 14, 35, 15], [1, 2, 2, 28, 27, 14],
[5, 2, 4, 16, 12, 34], [1, 4, 2, 2, 5, 5, 3, 5, 2, 24, 0, 13, 18, 36, 47, 46],
[1, 4, 5, 14, 44, 32], [5, 3, 4, 16, 32, 45], [4, 3, 4, 8, 16, 2], [4, 3, 5, 7,
15, 49], [4, 3, 4, 4, 1, 14, 44, 30, 39, 26], [5, 1, 5, 32, 1, 23], [3, 5, 2, 2,
2, 12, 0, 20, 22, 49], [2, 1, 5, 3, 16, 32, 20, 49], [1, 5, 2, 4, 23, 24, 40,
41], [1, 4, 1, 2, 5, 3, 3, 42, 29, 3, 1, 43, 46, 41], [2, 5, 2, 1, 25, 48, 16,
5], [3, 1, 2, 17, 47, 29], [2, 4, 2, 3, 16, 32, 45, 11], [2, 1, 1, 4, 9, 13],
[3, 4, 17, 47], [1, 5, 32, 11], [5, 1, 4, 4, 42, 33, 12, 46], [1, 4, 5, 1, 23,
24, 40, 41], [4, 2, 2, 4, 3, 4, 9, 13, 33, 38, 14, 49], [5, 5, 2, 3, 15, 26, 14,
47], [4, 4, 1, 18, 35, 3], [1, 5, 2, 3, 44, 21], [4, 5, 46, 3], [1, 9], [1, 5,
5, 3, 3, 24, 31, 48, 30, 16], [3, 19], [4, 5, 42, 33], [4, 5, 4, 34], [2, 5, 1,
4, 4, 23, 24, 40, 41, 43], [1, 1, 22, 5], [2, 2, 10, 31], [2, 4, 4, 3, 17, 47,
40, 4], [5, 3, 47, 39], [2, 3, 4, 14, 15, 31], [5, 2, 5, 4, 2, 42, 33, 5, 26,
44], [4, 3, 2, 13, 38, 30], [4, 5, 2, 1, 3, 18, 35, 3, 2, 47], [1, 4, 2, 5, 12,
31, 36, 48], [4, 3, 2, 28, 27, 17], [3, 4, 5, 17, 47, 29], [5, 4, 14, 44], [1,
4, 1, 0, 46, 2], [4, 1, 3, 3, 39, 33, 0, 21], [3, 3, 3, 14, 44, 27], [4, 4, 4,
12, 31, 36], [3, 3, 5, 18], [5, 2, 3, 2, 7, 11, 37, 45], [4, 1, 0, 20], [3, 3,
23, 30], [3, 5, 5, 1, 16, 32, 14, 0], [5, 8], [4, 3, 1, 4, 4, 28, 27, 46, 13,
26], [3, 5, 1, 3, 2, 4, 9, 12, 37, 17], [1, 5, 2, 10, 37, 20], [4, 1, 2, 3, 1,
21, 11, 3, 23, 7], [1, 3, 5, 21, 31, 32], [2, 3, 5, 2, 4, 1, 5, 32, 44, 1, 0,
42, 23, 5], [4, 2, 1, 0, 46, 2], [4, 3, 5, 34, 37, 47], [5, 4, 2, 3, 16, 32, 45,
28], [5, 4, 3, 18, 35, 3], [2, 1, 4, 3, 5, 1, 5, 0, 46, 48, 5, 33, 35, 23], [5,
3, 5, 1, 14, 44, 28, 29], [5, 3, 1, 13, 15, 24], [3, 4, 5, 22, 8, 23], [2, 2, 4,
4, 5, 3, 2, 14, 43, 41, 11, 45, 28, 26], [1, 1, 39, 22], [5, 4, 1, 4, 7, 11, 37,
45], [2, 5, 4, 4, 3, 2, 3, 0, 46, 2, 16, 15, 22, 35], [4, 4, 41, 42], [4, 3, 1,
9, 30, 6], [3, 2, 5, 1, 3, 37, 13, 45, 4, 49], [4, 4, 0, 16], [3, 5, 2, 1, 4,
28, 27, 48, 14, 37], [3, 4, 28, 13], [3, 4, 18, 13], [5, 3, 2, 1, 2, 4, 31, 29,
41, 35, 14, 49], [4, 2, 5, 3, 34, 48, 46, 15], [2, 5, 15, 28], [3, 4, 5, 18, 35,
3], [5, 4, 5, 5, 12, 31, 36, 48], [2, 1, 2, 42, 33, 35], [5, 1, 4, 16, 32, 45],
[1, 2, 4, 2, 7, 15, 43, 9], [1, 1, 5, 3, 7, 11, 37, 45], [3, 1, 3, 2, 12, 31,
36, 48], [5, 14], [2, 1, 4, 1, 1, 1, 33, 21, 38, 14, 8, 5], [4, 4, 4, 5, 2, 3,
10, 0, 40, 38], [1, 3, 2, 4, 14, 47, 10, 18], [2, 2, 4, 11, 37, 38], [1, 1, 3,
2, 4, 18, 35, 3, 1, 19], [3, 4, 1, 22, 5, 3], [5, 4, 1, 4, 9, 24, 34, 27], [1,
1, 5, 4, 9, 7], [4, 3, 1, 0, 35, 12], [4, 3, 1, 37, 28, 29], [5, 3, 3, 1, 2, 41,
33, 11, 9, 28], [1, 5, 14, 44], [1, 3, 2, 2, 4, 41, 43, 11], [2, 5, 17, 47], [1,
1, 3, 5, 2, 16, 32, 45, 27, 20], [3, 5, 1, 42, 2, 14], [3, 3, 3, 3, 4, 9, 43,
21], [5, 5, 1, 17, 47, 29], [1, 4, 3, 4, 42, 33, 28, 23], [5, 4, 6, 46], [3, 3,
3, 18, 35, 3], [3, 1, 4, 2, 1, 5, 14, 18, 9, 40], [3, 3, 4, 4, 9, 34], [3, 5, 4,
4, 14, 44, 13, 43], [1, 1, 28, 27], [4, 3, 16, 32], [2, 5, 2, 22, 6, 14], [1, 1,
1, 0, 41, 10], [1, 4, 4, 2, 5, 4, 3, 1, 31, 47, 39, 37, 27, 34, 18, 0], [5, 4,
1, 22, 5, 7], [3, 1, 4, 7], [1, 5, 1, 7, 15, 49], [5, 2, 15, 4], [2, 4, 1, 5, 2,
2, 1, 2, 16, 46, 1, 18, 36, 25, 20, 29], [4, 4, 5, 42, 30, 13], [1, 4, 2, 31,
24, 25], [4, 2, 1, 4, 7, 15, 4, 18], [2, 1, 22, 5], [1, 28], [4, 5, 2, 42, 33,
23], [2, 44], [3, 1, 2, 3, 28, 15, 27, 19], [5, 4, 2, 5, 5, 23, 11, 0, 18, 47],
[1, 5, 22, 5], [4, 1, 2, 7, 11, 37], [4, 3, 4, 5, 7, 11, 37, 45], [1, 1, 2, 5,
5, 42, 33, 6, 16, 39], [3, 3, 4, 3, 23, 24, 40, 41], [2, 3, 2, 2, 30, 31, 0, 9],
[4, 2, 3, 1, 32, 48], [3, 3, 3, 3, 28, 27, 43, 2], [1, 2, 1, 19], [1, 35], [1,
1, 2, 5, 18, 35, 3, 10], [2, 2, 5, 2, 4, 9, 12, 48], [2, 3, 2, 1, 19, 6], [2, 4,
1, 7, 15, 49], [2, 3, 5, 27, 20, 22], [3, 13], [3, 4, 4, 26, 16, 15], [2, 1, 4,
18, 35, 3], [5, 19], [5, 3, 22, 5], [3, 3, 1, 19], [2, 1, 1, 3, 39, 36, 12, 37],
[2, 4, 4, 1, 34, 10, 2, 3], [3, 1, 30, 44], [3, 3, 2, 2, 13, 20, 47, 22], [4, 2,
4, 1, 12, 4, 26, 45], [4, 1, 5, 1, 36, 27, 35, 13], [2, 3, 3, 0, 46, 2], [2, 4,
10, 7], [4, 4, 2, 1, 39, 2, 34, 1], [2, 3, 2, 2, 7, 11, 37, 45], [2, 2, 1, 39,
43, 44], [4, 3, 4, 3, 14, 30], [2, 3, 1, 4, 3, 12, 49, 32, 30, 2], [2, 4, 2, 1,
19, 32], [2, 3, 1, 2, 31, 28, 21, 10], [1, 3, 3, 4, 4, 28, 27, 45, 17, 9], [5,
4, 1, 5, 16, 32, 10, 11], [2, 5, 3, 1, 12, 31, 36, 48], [1, 4, 3, 4, 5, 18, 35,
3, 29, 40], [3, 4, 1, 4, 1, 23, 24, 40, 41, 43], [5, 5, 2, 4, 5, 17, 47, 29, 14,
44], [2, 38], [5, 5, 2, 0, 46, 2], [4, 1, 3, 1, 18, 35, 3, 20], [2, 4, 3, 3, 17,
47, 29, 43], [4, 4, 3, 1, 3, 19, 6, 0, 26, 43], [3, 2, 3, 18, 35, 3], [5, 4, 1,
5, 3, 5, 3, 4, 28, 27, 21, 42, 39, 26, 5, 49], [1, 2, 5, 1, 12, 31, 36, 48], [4,
1, 5, 3, 1, 0, 46, 2, 30, 47], [3, 5, 2, 22, 5, 33], [1, 5, 3, 42, 33, 14], [5,
2, 4, 5, 0, 46, 43, 30], [5, 2, 4, 2, 17, 47, 11, 35], [3, 3, 3, 0, 46, 2], [1,
3, 3, 16, 32, 45], [3, 4, 1, 4, 37, 20, 42, 39], [1, 2, 2, 14, 44, 30], [4, 3,
39, 36], [5, 5, 3, 3, 3, 1, 37, 41, 17, 1, 49, 25], [5, 2, 1, 44, 30, 36], [4,
4, 5, 5, 12, 31, 36, 48], [4, 1, 5, 5, 5, 3, 5, 28, 27, 48, 24, 34, 39, 45], [2,
3, 5, 0, 46, 37], [1, 1, 4, 18, 25, 44], [1, 3, 4, 9], [3, 4, 22, 5], [4, 3, 42,
33], [2, 2, 2, 4, 28, 27, 0, 26], [3, 5], [3, 3, 1, 4, 42, 33, 2, 22], [2, 1, 2,
7, 15, 49], [4, 2, 4, 4, 3, 4, 9, 37, 13, 17], [4, 3, 5, 1, 3, 22, 5, 38, 47,
35], [1, 2, 4, 0, 46, 2], [3, 1, 5, 1, 28, 27, 39, 42], [5, 2, 4, 4, 12, 31, 36,
48], [3, 2, 28, 27], [5, 2, 5, 4, 16, 32, 45, 49], [1, 2, 21, 40], [4, 4, 3, 22,
5, 48], [5, 4, 5, 1, 5, 42, 33, 45, 23, 39], [1, 1, 5, 1, 7, 25], [3, 1, 42,
33], [5, 2, 4, 2, 3, 16, 32, 45, 38, 19], [2, 2, 5, 1, 7, 23, 30, 46], [4, 42],
[1, 4, 2, 2, 23, 2, 10, 42], [4, 4, 1, 26, 30, 48], [1, 5, 4, 7, 15, 49], [4, 3,
3, 3, 48, 25, 39, 12], [2, 4, 2, 3, 4, 17, 47, 16, 49, 33], [3, 49], [3, 3, 27,
45], [4, 2, 5, 4, 5, 43, 21, 1, 11, 42], [5, 3, 5, 22, 9, 8], [3, 3, 28, 27],
[3, 4, 1, 2, 3, 0, 46, 2, 26, 43], [5, 4, 2, 16, 32, 45], [1, 2, 1, 39, 22, 10],
[1, 4, 4, 1, 5, 14, 18, 11, 7, 23], [2, 4, 1, 22, 5, 43], [5, 3, 4, 2, 17, 47,
29, 23], [1, 5, 3, 5, 1, 19, 2, 15], [4, 1, 4, 5, 7, 11, 37, 45], [1, 3, 2, 4,
22, 5, 44, 12], [2, 5, 4, 2, 11, 22, 8, 37], [5, 5, 29, 20], [1, 1, 3, 2, 4, 3,
2, 7, 11, 37, 45, 39, 9, 13], [1, 5, 4, 3, 7, 11, 37, 45], [1, 4, 2, 4, 42, 36,
3, 49], [5, 4, 1, 16, 32, 45], [1, 2, 21, 35], [3, 4, 1, 0, 46, 2], [4, 1, 4,
18, 10, 44], [2, 5, 2, 4, 5, 4, 1, 4, 28, 27, 21, 6, 15, 46, 34, 41], [2, 2, 1,
4, 7, 11, 37, 45], [5, 3, 40, 16], [4, 4, 43, 37], [5, 1, 3, 3, 7, 15, 49, 29],
[5, 4, 3, 1, 12, 31, 36, 48], [2, 1, 2, 22, 5, 24], [2, 4, 2, 2, 14, 12, 13,
32], [4, 3, 4, 1, 4, 9, 32, 5], [5, 4, 5, 10, 44, 40], [5, 1, 5, 18, 35, 3], [2,
4, 5, 4, 2, 20, 1, 13, 31, 45], [2, 1, 4, 48, 47, 37], [3, 1, 4, 9], [3, 5, 1,
19], [5, 5, 1, 4, 12, 31], [5, 3, 10, 33], [5, 1, 14, 44], [1, 4, 1, 5, 34, 15],
[1, 4, 36, 23], [1, 4, 2, 5, 1, 42, 35, 5, 32, 29], [5, 4, 1, 3, 11, 29], [4, 3,
3, 2, 1, 2, 1, 2, 37, 6, 10, 22, 8, 35, 18, 4], [2, 5, 2, 24, 27, 21], [5, 2,
39, 33], [1, 5, 28, 27], [1, 1, 3, 44, 6, 10], [1, 22], [1, 4, 0, 46], [1, 2, 2,
16, 32, 45], [5, 3, 2, 5, 5, 3, 9, 37, 42, 44], [5, 3, 5, 4, 4, 19, 27, 23, 21,
30], [1, 4, 4, 30, 5, 19], [3, 4, 2, 2, 3, 22, 5, 19, 15, 23], [4, 5, 22, 5],
[4, 1, 4, 18, 35, 3], [1, 3, 4, 17, 47, 29], [4, 1, 5, 2, 4, 5, 12, 31, 36, 48,
26, 41], [4, 1, 5, 18, 35, 3], [4, 10], [1, 5, 7, 15], [5, 1, 4, 28, 43, 49],
[3, 3, 2, 28, 27, 12], [3, 5, 1, 19], [4, 4, 21, 17], [3, 3, 5, 2, 2, 43, 0, 24,
18, 48], [4, 4, 4, 16, 32, 45], [5, 1, 5, 33, 0, 11], [2, 3], [1, 4, 1, 15, 9,
40], [3, 4, 4, 12, 49, 30], [1, 45], [4, 3, 4, 1, 4, 23, 24, 40, 41, 43], [4, 2,
5, 14, 44, 40], [1, 5, 4, 41, 47, 15], [2, 5, 3, 28, 27, 29], [3, 2, 3, 4, 0,
46, 2, 23], [2, 3, 2, 4, 20, 11, 8, 7], [2, 3, 4, 2, 22, 32, 6, 15], [4, 3, 5,
26, 5, 13], [4, 5, 2, 20], [1, 3, 1, 3, 7, 15, 1, 48], [5, 4, 45, 34], [2, 4, 4,
1, 31, 46, 12, 24], [1, 3, 4, 28, 27, 26], [1, 1, 3, 3, 23, 24, 40, 41], [2, 2,
1, 27, 29, 20], [5, 26], [4, 4, 28, 27], [4, 2, 4, 1, 29, 13], [5, 4, 2, 5, 3,
5, 23, 11, 45, 25, 36, 38], [4, 4, 1, 17, 47, 29], [1, 5, 5, 39, 22, 30], [1, 2,
2, 3, 42, 33, 45, 47], [2, 2, 2, 5, 4, 1, 3, 0, 46, 2, 39, 49, 12, 41], [3, 3,
3, 4, 12, 31, 36, 48], [4, 4, 1, 2, 1, 27, 12, 45, 17, 43], [1, 5, 17, 47], [2,
1, 5, 2, 12, 31, 36, 48], [4, 5, 5, 4, 3, 29, 8, 20, 46, 32], [4, 3, 2, 4, 3,
36, 10, 22, 23, 26], [4, 4, 1, 2, 4, 47, 7, 20, 26, 44], [5, 3, 3, 1, 2, 28, 27,
33, 18, 2], [4, 3, 5, 5, 1, 1, 2, 46, 24, 23, 21, 42, 14, 30], [3, 3, 18, 35],
[4, 1], [4, 3, 5, 1, 1, 46, 44, 0, 14, 20], [4, 5, 5, 4, 5, 11, 7, 5, 27, 24],
[2, 2, 2, 2, 18, 39, 0, 25], [3, 4, 4, 1, 19, 27], [2, 2, 3, 5, 2, 23, 24, 40,
41, 43], [4, 16], [1, 3, 2, 2, 1, 2, 1, 48, 35, 28, 46, 0, 6, 33], [4, 4, 5, 1,
19, 44], [1, 1, 3, 17, 47, 29], [1, 3, 6, 31], [5, 1, 1, 3, 4, 23, 24, 40, 41,
43], [5, 1, 42, 33], [3, 4, 1, 5, 2, 1, 1, 19, 29, 3, 44, 31], [4, 4, 3, 17, 47,
29], [5, 3, 4, 4, 2, 36, 27, 31, 46, 32], [5, 1, 1, 16, 32, 45], [4, 4, 3, 4, 1,
19, 44, 34], [2, 5, 5, 2, 1, 48, 2, 18, 7, 0], [5, 3, 19, 14], [5, 3, 2, 7, 15,
49], [2, 2, 5, 3, 22, 5, 13, 30], [5, 3, 1, 17, 47, 29], [4, 5, 5, 4, 5, 28, 27,
48, 13, 42], [3, 2, 4, 22, 5, 1], [5, 1, 4, 9], [2, 4, 3, 4, 7, 11, 37, 45], [1,
5, 5, 4, 3, 2, 5, 40, 37, 7, 0, 3, 22, 39], [2, 1, 1, 19], [3, 5, 22, 26], [4,
3, 3, 42, 33, 37], [5, 2, 1, 3, 5, 4, 28, 27, 4, 8, 46, 13], [4, 5, 3, 28, 27,
40], [5, 3, 28, 27], [2, 1, 42, 33], [4, 1, 22, 5], [1, 5, 5, 14, 44, 22], [2,
2, 3, 16, 32, 45], [4, 3, 3, 4, 16, 3, 28, 48], [3, 3, 2, 3, 22, 23, 24, 39],
[4, 5, 1, 3, 1, 28, 26, 33, 15, 36], [5, 5, 1, 7, 15, 49], [3, 3, 5, 3, 3, 1,
22, 27, 40, 15, 31, 33], [1, 5, 5, 5, 4, 41, 49, 18, 43, 35], [2, 4, 4, 9], [5,
3, 1, 12, 31, 36], [5, 3, 2, 2, 1, 5, 1, 48, 31, 28, 19, 38, 1, 44], [1, 2, 1,
4, 9, 18], [2, 2, 1, 6, 34, 32], [1, 2, 42, 15], [3, 3, 1, 1, 28, 27, 13, 36],
[3, 2, 1, 3, 2, 4, 7, 15, 49, 43, 22, 37], [1, 3, 3, 30, 43, 7], [2, 2, 4, 25,
15, 49], [5, 2, 3, 33, 41, 22], [5, 34], [2, 1, 2, 4, 2, 14, 25, 41, 21, 4], [4,
1, 4, 3, 42, 33, 23, 20], [5, 5, 2, 3, 2, 13, 9, 42, 45, 22], [4, 3, 2, 5, 3,
29, 46, 0, 14, 22], [1, 3, 1, 19], [3, 4, 1, 3, 16, 32, 45, 3], [1, 3, 4, 9],
[2, 2, 5, 3, 4, 9, 21, 39], [3, 1, 5, 18, 39, 3], [2, 3, 1, 4, 32, 13], [5, 3,
5, 17, 47, 29], [4, 3, 4, 4, 12, 31, 36, 48], [3, 3, 14, 15], [4, 4, 23, 46],
[3, 2, 22, 5], [3, 3, 5, 18, 35, 3], [3, 4, 4, 1, 47, 22, 48, 12], [5, 4, 3, 2,
18, 35, 3, 5], [2, 3, 5, 3, 23, 24, 40, 41], [5, 22], [3, 5, 5, 4, 1, 4, 5, 12,
34, 8, 49, 29, 16, 18], [1, 1, 2, 1, 42, 33, 34, 44], [5, 5, 1, 5, 41, 13, 24,
23], [1, 4, 1, 4, 5, 23, 24, 40, 41, 43], [1, 1, 13, 25], [4, 2, 2, 5, 5, 41,
36, 42], [4, 5, 5, 1, 5, 31, 47, 37, 22, 5], [4, 5, 4, 5, 20, 7], [5, 2, 4, 5,
4, 14, 44, 37, 16, 4], [3, 1, 2, 18, 35, 3], [5, 3, 2, 4, 1, 42, 33, 21, 29, 8],
[2, 1, 4, 5, 14, 18], [2, 27], [1, 3, 4, 1, 14, 22, 20, 24], [1, 2, 5, 3, 5, 2,
3, 31, 45, 1, 44, 14, 3, 47], [1, 3, 2, 5, 36, 29, 46, 41], [5, 3, 1, 16, 32,
45], [1, 4, 1, 1, 4, 23, 24, 40, 41, 43], [1, 2, 3, 3, 5, 3, 2, 19, 11, 7, 47,
38, 43, 42], [5, 1, 2, 3, 28, 27, 32, 0], [4, 1, 5, 2, 2, 0, 46, 2, 41, 28], [2,
5, 4, 2, 3, 23, 24, 40, 41, 43], [4, 2, 3, 18, 35, 3], [3, 4, 14, 31], [4, 5, 2,
14, 44, 8], [4, 4, 1, 1, 3, 18, 35, 3, 22, 14], [4, 2, 4, 3, 1, 30], [4, 1, 5,
1, 6, 12, 44, 41], [1, 2, 2, 30, 43, 11], [5, 5, 1, 7, 15, 49], [1, 2, 1, 0, 46,
2], [3, 2, 1, 1, 43, 38, 17, 1], [1, 4, 36, 25], [5, 5, 5, 27, 9, 7], [5, 5, 4,
22, 5, 13], [4, 3, 2, 3, 12, 31, 36, 48], [4, 5, 4, 26, 39, 43], [3, 3, 1, 12,
26, 38], [2, 1, 2, 3, 16, 1, 40, 20], [5, 4, 3, 1, 5, 23, 24, 40, 41, 43], [2,
3, 3, 3, 4, 49, 6, 38, 28, 37], [3, 5, 4, 2, 3, 1, 14, 31, 18, 17, 16, 26], [3,
1, 37, 44], [4, 2, 30, 11], [1, 2, 4, 2, 41, 22, 28, 48], [5, 3, 4, 1, 12, 31,
36, 48], [4, 4, 4, 5, 7, 5, 21, 4], [1, 3, 4, 2, 2, 18, 35, 3, 9, 41], [1, 2, 3,
1, 28, 27, 5, 42], [3, 1, 4, 22, 5, 31], [4, 5, 42, 33], [1, 5, 1, 4, 9, 15],
[5, 4, 4, 3, 7, 11, 37, 45], [3, 2, 1, 2, 3, 23, 24, 40, 41, 43], [1, 40], [3,
2, 5, 5, 1, 19, 8, 14], [4, 2, 3, 14, 44, 31], [5, 1, 1, 1, 6, 28, 39, 33], [2,
4, 5, 2, 1, 5, 39, 0, 47, 6, 30, 16], [3, 1, 3, 12, 31, 36], [4, 1, 1, 1, 46,
36], [5, 1, 1, 27, 44, 42], [1, 3, 4, 9], [5, 20], [1, 1, 22, 5], [1, 2, 1, 16,
26, 10], [2, 1, 4, 42, 13, 36], [3, 1, 2, 2, 42, 33, 47, 12], [4, 5, 3, 28, 27,
4], [2, 1, 2, 1, 19, 39], [1, 5, 2, 16, 32, 1], [3, 1, 3, 12, 31, 36], [4, 5, 2,
4, 16, 32, 45, 41], [4, 1, 2, 42, 33, 16], [2, 1, 2, 4, 5, 3, 4, 12, 31, 36, 35,
13, 44, 41], [1, 3, 1, 1, 5, 19, 44, 6, 24, 9], [1, 4, 2, 1, 5, 23, 24, 40, 41,
43], [5, 2, 5, 1, 5, 12, 31, 36, 48, 5], [4, 4, 4, 5, 4, 0, 46, 2, 44, 20], [4,
4, 2, 18, 35, 3], [2, 1, 1, 4, 5, 4, 32, 19, 14, 48, 12, 4], [2, 41], [4, 3, 1,
14, 44, 0], [2, 4, 4, 3, 5, 4, 3, 5, 1, 19, 39, 33, 26, 10, 25, 13], [1, 5, 4,
7, 15, 16], [1, 1, 1, 4, 29, 46], [3, 2, 5, 5, 14, 43, 20, 40], [3, 2, 38, 7],
[1, 3, 4, 1, 45, 3, 47, 18], [5, 3, 1, 43, 29, 19], [5, 5, 9, 16], [3, 3, 2, 20,
42, 0], [2, 5, 3, 1, 1, 5, 44, 33, 10, 5, 39, 32], [5, 2, 28, 27], [4, 5, 3, 1,
4, 2, 22, 29, 25, 43, 37, 5], [5, 19], [5, 5, 5, 16, 34, 4], [1, 5, 3, 4, 9, 45,
44, 17], [5, 4, 2, 7, 15, 29], [3, 1, 5, 5, 1, 5, 4, 7, 11, 37, 45, 27, 49, 39],
[5, 4, 4, 4, 12, 31, 36, 48], [3, 3, 49, 34], [1, 31], [4, 2, 4, 1, 42, 33, 45,
25], [2, 1, 1, 3, 16, 32, 45, 27], [1, 5, 25, 41], [5, 1, 2, 4, 3, 3, 4, 45, 15,
40, 25, 6, 9, 24], [2, 1, 4, 1, 3, 5, 5, 7, 11, 37, 45, 9, 29, 8], [2, 1, 21,
36], [3, 1, 1, 3, 1, 22, 5, 14, 7, 17], [4, 4, 40, 38], [4, 3, 5, 3, 3, 1, 4,
41, 33, 22, 25, 7, 15, 2], [2, 5, 28, 27], [4, 3, 3, 16, 32, 45], [2, 4, 4, 18,
28, 8], [4, 13], [3, 41], [2, 2, 1, 2, 28, 27, 38, 48], [5, 3, 3, 18, 35, 3],
[4, 1, 3, 4, 16, 32, 20, 36], [2, 3, 3, 16, 32, 45], [3, 2, 4, 9], [5, 5, 4, 39,
22, 31], [3, 2, 2, 0, 46, 2], [5, 34], [5, 5, 3, 0, 46, 2], [1, 0], [2, 1, 4,
13, 3, 6], [5, 5, 4, 29, 3, 44], [3, 3, 3, 3, 28, 27, 45, 9], [3, 2, 3, 5, 3, 4,
46, 33, 31, 10, 2, 22], [3, 5, 4, 2, 22, 2, 32, 6], [1, 3, 5, 4, 5, 4, 9, 0, 33,
30], [2, 5, 2, 18, 35, 3], [4, 3, 3, 15, 47, 34]]
```

```
/usr/local/lib/python3.10/dist-packages/ipykernel/ipkernel.py:283:
DeprecationWarning: `should_run_async` will not call `transform_cell`
automatically in the future. Please pass the result to `transformed_cell`
argument and any exception that happen during thetransform in
`preprocessing_exc_tuple` in IPython 7.17 and above.
  and should_run_async(code)
```

[41]:
```python
from mlxtend.preprocessing import TransactionEncoder
encoder = TransactionEncoder()
dados_encoded = encoder.fit_transform(lista_transacoes)
df_encoded = pd.DataFrame(dados_encoded, columns=encoder.columns_)
print(df_encoded)
```

```
         0      1      2      3      4      5      6      7      8      9  \
0    False  False   True   True   True   True  False   True  False  False
1    False   True   True  False  False  False  False  False  False  False
2    False   True  False  False  False  False  False  False  False  False
3    False   True  False   True   True   True  False  False  False  False
4    False  False   True  False   True   True  False   True  False   True
..     …    …    …    …    …    …    …    …    …    …
995  False  False   True   True   True   True  False  False  False  False
996  False  False   True   True   True   True   True  False  False  False
997   True   True  False   True   True   True  False  False  False   True
998  False  False   True   True  False   True  False  False  False  False
999  False  False  False   True   True  False  False  False  False  False

     …     40     41     42     43     44     45     46     47     48     49
0    …  False  False  False  False   True  False  False  False  False   True
```

```
1    …  False  False  False  False  False  False  False  False  False  False
2    …  False  False  False  False  False  False  False  False  False  False
3    …  False  False  False  False   True  False  False  False  False  False
4    …  False  False  False  False  False  False  False  False  False  False
..   …    …      …      …      …      …      …      …      …      …      …
995  …  False  False  False  False  False  False   True  False  False  False
996  …  False  False  False  False  False  False  False  False  False  False
997  …  False  False  False  False  False  False  False  False  False  False
998  …  False  False  False  False  False  False  False  False  False  False
999  …  False  False  False  False  False  False  False   True  False  False

[1000 rows x 50 columns]
```

```
/usr/local/lib/python3.10/dist-packages/ipykernel/ipkernel.py:283:
DeprecationWarning: `should_run_async` will not call `transform_cell`
automatically in the future. Please pass the result to `transformed_cell`
argument and any exception that happen during thetransform in
`preprocessing_exc_tuple` in IPython 7.17 and above.
  and should_run_async(code)
```

[42]:

```
product_columns = [f'product_{num}' for num in df2['product_number']]
df_encoded.columns = product_columns

for coluna in df_encoded.columns:
    product_info = df2.loc[df2['product_number'] == int(coluna.split('_')[1])]
    df_encoded.rename(columns={coluna: f"{product_info['flavor'].values[0]}␣
  ↪{product_info['product'].values[0]}"}, inplace=True)

print(df_encoded)
```

```
/usr/local/lib/python3.10/dist-packages/ipykernel/ipkernel.py:283:
DeprecationWarning: `should_run_async` will not call `transform_cell`
automatically in the future. Please pass the result to `transformed_cell`
argument and any exception that happen during thetransform in
`preprocessing_exc_tuple` in IPython 7.17 and above.
  and should_run_async(code)
```

```
     Chocolate Cake  Lemon Cake  Casino Cake  Opera Cake  Strawberry Cake  \
0             False       False         True        True             True
1             False        True         True       False            False
2             False        True        False       False            False
3             False        True        False        True             True
4             False       False         True       False             True
..              …           …          …           …               …
995           False       False         True        True             True
996           False       False         True        True             True
997            True        True        False        True             True
998           False       False         True        True            False
999           False       False        False        True             True
```

```
     Truffle Cake  Chocolate Eclair  Coffee Eclair  Vanilla Eclair  \
0            True             False           True           False
1           False             False          False           False
2           False             False          False           False
3            True             False          False           False
4            True             False           True           False
..            …               …            …             …
995          True             False          False           False
996          True              True          False           False
997          True             False          False           False
998          True             False          False           False
999         False             False          False           False

     Napoleon Cake  …  Lemon Lemonade  Raspberry Lemonade  Orange Juice  \
0            False  …           False               False         False
1            False  …           False               False         False
2            False  …           False               False         False
3            False  …           False               False         False
4             True  …           False               False         False
..             …  …             …                 …           …
995          False  …           False               False         False
996          False  …           False               False         False
997           True  …           False               False         False
998          False  …           False               False         False
999          False  …           False               False         False

     Green Tea  Bottled Water  Hot Coffee  Chocolate Coffee  \
0        False           True       False             False
1        False          False       False             False
2        False          False       False             False
3        False           True       False             False
4        False          False       False             False
..         …            …         …               …
995      False          False       False              True
996      False          False       False             False
997      False          False       False             False
998      False          False       False             False
999      False          False       False             False

     Vanilla Frappuccino  Cherry Soda  Single Espresso
0                   False        False             True
1                   False        False            False
2                   False        False            False
3                   False        False            False
4                   False        False            False
..                   …          …              …
995                 False        False            False
```

```
996                 False        False            False
997                 False        False            False
998                 False        False            False
999                  True        False            False

[1000 rows x 50 columns]
```

### 0.1.2 Minere as regras de associação

Obs: Para começar, você pode definir um suporte mínimo de 6% (se quiser, varie esse valor para visualizar os efeitos)

[43]:
```
from mlxtend.frequent_patterns import apriori, association_rules
frequent_itemsets = apriori(df_encoded, min_support=0.06, use_colnames=True)

rules = association_rules(frequent_itemsets, metric="confidence",␣
  ↪min_threshold=0.5)

print("regras de associação:")
print(rules)
```

```
/usr/local/lib/python3.10/dist-packages/ipykernel/ipkernel.py:283:
DeprecationWarning: `should_run_async` will not call `transform_cell`
automatically in the future. Please pass the result to `transformed_cell`
argument and any exception that happen during thetransform in
`preprocessing_exc_tuple` in IPython 7.17 and above.
  and should_run_async(code)
```

```
regras de associação:
                                   antecedents        consequents  \
0                             (Chocolate Cake)      (Casino Cake)
1                                (Casino Cake)       (Lemon Cake)
2                                 (Lemon Cake)      (Casino Cake)
3                                 (Opera Cake)       (Lemon Cake)
4                                 (Lemon Cake)       (Opera Cake)
..                                         …                  …
72  (Strawberry Cake, Truffle Cake, Opera Cake)       (Lemon Cake)
73       (Truffle Cake, Opera Cake, Lemon Cake)  (Strawberry Cake)
74   (Strawberry Cake, Casino Cake, Opera Cake)     (Truffle Cake)
75  (Strawberry Cake, Truffle Cake, Opera Cake)      (Casino Cake)
76      (Casino Cake, Truffle Cake, Opera Cake)  (Strawberry Cake)

    antecedent support  consequent support  support  confidence      lift  \
0                0.084               0.528    0.064    0.761905  1.443001
1                0.528               0.581    0.295    0.558712  0.961639
2                0.581               0.528    0.295    0.507745  0.961639
3                0.543               0.581    0.302    0.556169  0.957262
4                0.581               0.543    0.302    0.519793  0.957262
..                 …                   …      …         …         …
```

```
72               0.147               0.581    0.084    0.571429  0.983526
73               0.154               0.584    0.084    0.545455  0.933998
74               0.158               0.544    0.080    0.506329  0.930752
75               0.147               0.528    0.080    0.544218  1.030715
76               0.147               0.584    0.080    0.544218  0.931880

    leverage  conviction  zhangs_metric
0   0.019648    1.982400       0.335153
1  -0.011768    0.949494      -0.077930
2  -0.011768    0.958853      -0.086930
3  -0.013483    0.944054      -0.088998
4  -0.013483    0.951674      -0.096293
..       …         …            …
72 -0.001407    0.977667      -0.019258
73 -0.005936    0.915200      -0.077091
74 -0.005952    0.923692      -0.081187
75  0.002384    1.035582       0.034936
76 -0.005848    0.912716      -0.078933

[77 rows x 10 columns]
```

### 0.1.3 Gere e visualize as regras de associação

Obs: * Para começar, você pode definir uma confiança mínima de 50% (se quiser, varie esse valor para visualizar os efeitos) * para visualizar as regras geradas, ordene-as descrescentemente com relação à confiança

[44]:

```
rules = association_rules(frequent_itemsets, metric="confidence", min_threshold=0.5)

rules = rules.sort_values(by='confidence', ascending=False)

print("regras de associação:")
print(rules)
```

```
regras de associação:

/usr/local/lib/python3.10/dist-packages/ipykernel/ipkernel.py:283:
DeprecationWarning: `should_run_async` will not call `transform_cell`
automatically in the future. Please pass the result to `transformed_cell`
argument and any exception that happen during thetransform in
`preprocessing_exc_tuple` in IPython 7.17 and above.
  and should_run_async(code)

                                   antecedents        consequents  \
9                                 (Lemon Tart)       (Lemon Cake)
27                            (Apricot Danish)       (Opera Cake)
30                             (Napoleon Cake)  (Strawberry Cake)
35                          (Gongolais Cookie)     (Truffle Cake)
```

```
24                              (Cherry Tart)       (Opera Cake)
..                                        …                …
17                             (Truffle Cake)      (Casino Cake)
38                   (Opera Cake, Lemon Cake)      (Casino Cake)
23                               (Opera Cake)     (Truffle Cake)
74  (Strawberry Cake, Casino Cake, Opera Cake)     (Truffle Cake)
22                             (Truffle Cake)       (Opera Cake)

    antecedent support  consequent support  support  confidence      lift  \
9                0.076               0.581    0.062    0.815789  1.404113
27               0.075               0.543    0.061    0.813333  1.497851
30               0.090               0.584    0.073    0.811111  1.388889
35               0.108               0.544    0.085    0.787037  1.446759
24               0.084               0.543    0.065    0.773810  1.425064
..                 …                   …       …          …         …
17               0.544               0.528    0.276    0.507353  0.960896
38               0.302               0.528    0.153    0.506623  0.959512
23               0.543               0.544    0.275    0.506446  0.930966
74               0.158               0.544    0.080    0.506329  0.930752
22               0.544               0.543    0.275    0.505515  0.930966

    leverage  conviction  zhangs_metric
9   0.017844    2.274571       0.311479
27  0.020275    2.448214       0.359327
30  0.020440    2.202353       0.307692
35  0.026248    2.141217       0.346188
24  0.019388    2.020421       0.325630
..        …           …             …
17 -0.011232    0.958090      -0.081933
38 -0.006456    0.956671      -0.057007
23 -0.020392    0.923910      -0.139607
74 -0.005952    0.923692      -0.081187
22 -0.020392    0.924193      -0.139871

[77 rows x 10 columns]
```

### 0.1.4 Usando linguagem natural (Português), descreva a regra gerada que tem a maior confiança:

A regra geral que gera maior confiança é que cerca de 81,579% das vezes que há a compra de Lemon Tart, há a compra em seguida do Lemon Cake. Logo, quem compra um Lemon Tart provavemente também irá adquirir o Lemon Cake.

### 0.1.5 Qual o suporte e a confiança desta regra?

- Suporte: aproximadamente 6,20%.
- Confiança:cerca de 81,58%.

### 0.1.6 Filtre as regras com confiança maior do que 55%

[45]:

```python
rules_filtered = rules[rules['confidence'] > 0.55]

rules_filtered = rules_filtered.sort_values(by='confidence', ascending=False)

print("Regras de Associação com confiança > 55%:")
print(rules_filtered)
```

```
/usr/local/lib/python3.10/dist-packages/ipykernel/ipkernel.py:283:
DeprecationWarning: `should_run_async` will not call `transform_cell`
automatically in the future. Please pass the result to `transformed_cell`
argument and any exception that happen during thetransform in
`preprocessing_exc_tuple` in IPython 7.17 and above.
  and should_run_async(code)
```

```
Regras de Associação com confiança > 55%:
                                    antecedents        consequents  \
9                                  (Lemon Tart)       (Lemon Cake)
27                             (Apricot Danish)       (Opera Cake)
30                              (Napoleon Cake)  (Strawberry Cake)
35                           (Gongolais Cookie)     (Truffle Cake)
24                                (Cherry Tart)       (Opera Cake)
0                              (Chocolate Cake)      (Casino Cake)
19                           (Chocolate Coffee)      (Casino Cake)
33                           (Chocolate Coffee)  (Strawberry Cake)
11                            (Apple Croissant)       (Lemon Cake)
31                                 (Berry Tart)  (Strawberry Cake)
34                                 (Berry Tart)     (Truffle Cake)
26                               (Tuile Cookie)       (Opera Cake)
71      (Casino Cake, Truffle Cake, Lemon Cake)  (Strawberry Cake)
32                           (Gongolais Cookie)  (Strawberry Cake)
59                  (Casino Cake, Truffle Cake)  (Strawberry Cake)
29                               (Truffle Cake)  (Strawberry Cake)
50                   (Truffle Cake, Lemon Cake)  (Strawberry Cake)
64        (Casino Cake, Opera Cake, Lemon Cake)  (Strawberry Cake)
18                           (Gongolais Cookie)      (Casino Cake)
72  (Strawberry Cake, Truffle Cake, Opera Cake)       (Lemon Cake)
15                                (Casino Cake)  (Strawberry Cake)
25                           (Gongolais Cookie)       (Opera Cake)
10                           (Gongolais Cookie)       (Lemon Cake)
6                                  (Lemon Cake)  (Strawberry Cake)
66      (Casino Cake, Truffle Cake, Lemon Cake)       (Opera Cake)
44                     (Opera Cake, Lemon Cake)  (Strawberry Cake)
5                             (Strawberry Cake)       (Lemon Cake)
42                (Strawberry Cake, Opera Cake)       (Lemon Cake)
45                   (Truffle Cake, Opera Cake)       (Lemon Cake)
1                                 (Casino Cake)       (Lemon Cake)
```

```
53                    (Casino Cake, Opera Cake)  (Strawberry Cake)
21                                 (Opera Cake)  (Strawberry Cake)
61   (Strawberry Cake, Casino Cake, Opera Cake)       (Lemon Cake)
3                                  (Opera Cake)       (Lemon Cake)
40                    (Casino Cake, Lemon Cake)  (Strawberry Cake)
7                                (Truffle Cake)       (Lemon Cake)

    antecedent support  consequent support  support  confidence      lift  \
9                0.076               0.581    0.062    0.815789  1.404113
27               0.075               0.543    0.061    0.813333  1.497851
30               0.090               0.584    0.073    0.811111  1.388889
35               0.108               0.544    0.085    0.787037  1.446759
24               0.084               0.543    0.065    0.773810  1.425064
0                0.084               0.528    0.064    0.761905  1.443001
19               0.085               0.528    0.064    0.752941  1.426025
33               0.085               0.584    0.060    0.705882  1.208703
11               0.091               0.581    0.061    0.670330  1.153752
31               0.095               0.584    0.062    0.652632  1.117520
34               0.095               0.544    0.062    0.652632  1.199690
26               0.102               0.543    0.064    0.627451  1.155527
71               0.142               0.584    0.086    0.605634  1.037044
32               0.108               0.584    0.065    0.601852  1.030568
59               0.276               0.584    0.162    0.586957  1.005063
29               0.544               0.584    0.317    0.582721  0.997809
50               0.300               0.584    0.173    0.576667  0.987443
64               0.153               0.584    0.088    0.575163  0.984869
18               0.108               0.528    0.062    0.574074  1.087262
72               0.147               0.581    0.084    0.571429  0.983526
15               0.528               0.584    0.301    0.570076  0.976157
25               0.108               0.543    0.061    0.564815  1.040175
10               0.108               0.581    0.061    0.564815  0.972143
6                0.581               0.584    0.328    0.564544  0.966685
66               0.142               0.543    0.080    0.563380  1.037533
44               0.302               0.584    0.170    0.562914  0.963894
5                0.584               0.581    0.328    0.561644  0.966685
42               0.303               0.581    0.170    0.561056  0.965673
45               0.275               0.581    0.154    0.560000  0.963855
1                0.528               0.581    0.295    0.558712  0.961639
53               0.283               0.584    0.158    0.558304  0.956000
21               0.543               0.584    0.303    0.558011  0.955498
61               0.158               0.581    0.088    0.556962  0.958627
3                0.543               0.581    0.302    0.556169  0.957262
40               0.295               0.584    0.163    0.552542  0.946134
7                0.544               0.581    0.300    0.551471  0.949175

    leverage  conviction  zhangs_metric
9   0.017844    2.274571       0.311479
27  0.020275    2.448214       0.359327
```

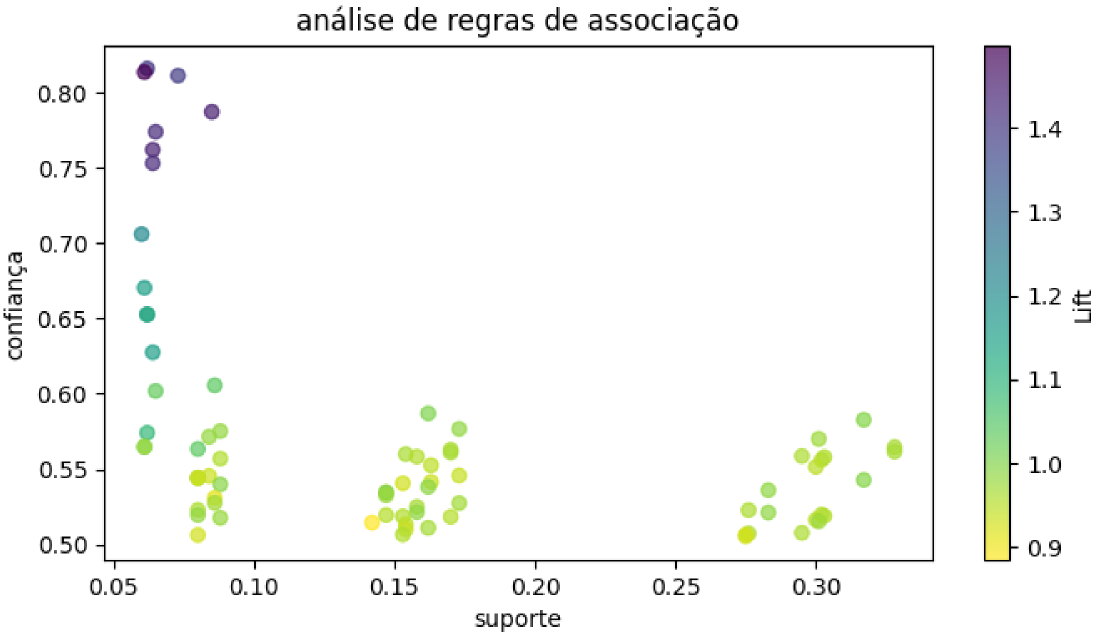
```
30  0.020440    2.202353      0.307692
35  0.026248    2.141217      0.346188
24  0.019388    2.020421      0.325630
0   0.019648    1.982400      0.335153
19  0.019120    1.910476      0.326503
33  0.010360    1.414400      0.188707
11  0.008129    1.270967      0.146603
31  0.006520    1.197576      0.116200
34  0.010320    1.312727      0.183924
26  0.008614    1.226684      0.149882
71  0.003072    1.054857      0.041633
32  0.001928    1.044837      0.033253
59  0.000816    1.007158      0.006957
29 -0.000696    0.996934     -0.004792
50 -0.002200    0.982677     -0.017843
64 -0.001352    0.979200     -0.017816
18  0.004976    1.108174      0.089975
72 -0.001407    0.977667     -0.019258
15 -0.007352    0.967612     -0.049202
25  0.002356    1.050128      0.043299
10 -0.001748    0.962809     -0.031125
6  -0.011304    0.955320     -0.076000
66  0.002894    1.046677      0.042162
44 -0.006368    0.951758     -0.050933
5  -0.011304    0.955844     -0.076507
42 -0.006043    0.954564     -0.048525
45 -0.005775    0.952273     -0.049180
1  -0.011768    0.949494     -0.077930
53 -0.007272    0.941824     -0.060320
21 -0.014112    0.941200     -0.092487
61 -0.003798    0.945743     -0.048759
3  -0.013483    0.944054     -0.088998
40 -0.009280    0.929697     -0.074721
7  -0.016064    0.934164     -0.105087
```

### 0.1.7 Plote um gráfico para analisar as regras de associação em base na confiança, suporte e elevação (lift) das regras de associação

Dica: use um gráfico scatter usando duas das variáveis como os eixos x e y e adicione a restante como shading.

[46]:
```
plt.figure(figsize=(8, 4))
plt.scatter(rules['support'], rules['confidence'], c=rules['lift'],␣
 ↪cmap='viridis_r', alpha=0.7)
plt.xlabel('suporte')
plt.ylabel('confiança')
plt.colorbar(label='Lift')
plt.title('análise de regras de associação')
plt.show()
```

```
/usr/local/lib/python3.10/dist-packages/ipykernel/ipkernel.py:283:
DeprecationWarning: `should_run_async` will not call `transform_cell`
automatically in the future. Please pass the result to `transformed_cell`
argument and any exception that happen during thetransform in
`preprocessing_exc_tuple` in IPython 7.17 and above.
  and should_run_async(code)
```



### 0.1.8 Imagine o seguinte contexto: O dono da cafeteria deseja aumentar as vendas do produto "Lemon Cake" e, para isso, seu analista de vendas sugere criar uma promoção que inclua o produto "Lemon Cake" e outros produtos que são frequentemente consumidos junto com "Lemon Cake". Quais produtos você indicaria para serem incluidos na promoção?

Dica: Explore a relação entre antecedentes e consequentes de uma regra.

```
[47]: lemon_cake_rules =rules[rules['antecedents'].apply(lambda x: 'Lemon Cake' in x)]

produtos_frquentes= lemon_cake_rules['consequents'].explode().value_counts().
  ↪index.tolist()

print("produtos indicados pra serem incluídos:")
for produto in produtos_frquentes:
    print( produto)
```

```
produtos indicados pra serem incluídos:
Strawberry Cake
Opera Cake
Truffle Cake
Casino Cake
```

```
/usr/local/lib/python3.10/dist-packages/ipykernel/ipkernel.py:283:
DeprecationWarning: `should_run_async` will not call `transform_cell`
automatically in the future. Please pass the result to `transformed_cell`
argument and any exception that happen during thetransform in
`preprocessing_exc_tuple` in IPython 7.17 and above.
  and should_run_async(code)
```