

PRAKTIKUM DATA WAREHOUSING DAN DATA MINING
MODUL 12
INDUKSI DAN ATURAN ASOSIASI



Disusun oleh:
Adinda Aulia Hapsari
L200220037

PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS KOMUNIKASI DAN INFORMATIKA
UNIVERSITAS MUHAMMADIYAH SURAKARTA
TAHUN 2024

Setelah kegiatan selesai, lembar kerja ini dicetak (di-print) dan dikumpulkan ke asisten.

NIM : L200220037
Nama : Adinda Aulia Hapsari
Nama Asisten : Diva Halimah
Tanggal Praktikum : 21 Desember 2024

(Diisi oleh Asisten)

Nilai Praktek :

Tanda Tangan :

KEGIATAN PRAKTIKUM

12.4.1 Mengimport Library

Meng-import library yang diperlukan, yaitu library pandas, numpy, dan apriori. Untuk meng-import library yang akan digunakan, kita meng import library pandas, numpy, dan apriori.

```
[ ] import numpy as np
import pandas as pd
from apyori import apriori
```

12.4.2 Membaca Dataset

Membaca dataset dari sebuah directory. Disini, kita mengambil dataset store_data.csv yang diambil dari platform gitea. Dataset tersebut akan tersimpan di variable store_data sebagai sebuah dataframe. Kemudian, kita akan mencetak tujuh data teratas pada dataset.

```
store_data = pd.read_csv("/content/drive/MyDrive/dwdm/store_data.csv")
store_data.head(7)
```

	shrimp	almonds	avocado	vegetables mix	green grapes	whole wheat flour	yams	cottage cheese	energy drink	tomato juice	low fat yogurt	green tea	honey	salad	mineral water	salmon	antioxydant juice	frozen smoothie	spinach	olive oil
0	burgers	meatballs	eggs	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
1	chutney	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
2	turkey	avocado	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
3	mineral water	milk	energy bar	whole wheat rice	green tea	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
4	low fat yogurt	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
5	whole wheat pasta	french fries	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
6	soup	light cream	shallot	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN

12.4.3 Mengkonversikan Dataframe ke dalam Array

Pada kode di bawah ini, kita akan mengkonversikan dataframe menjadi sebuah array bernama records. Disini, kita mengaplikasikan for loop dimulai dari baris pertama hingga terakhir. Kemudian di setiap iterasi tersebut, kita membuat sebuah list yang menyimpan nilai kolom pada setiap barisnya. Terakhir, kita cetak array records tersebut.

```
records = []
for i in range(0, store_data.shape[0]):
    records.append([str(store_data.values[i, j]) for j in range(0, store_data.shape[1])])

print(records)
```

[['burgers', 'meatballs', 'eggs', 'nan', 'nan', 'nan', 'nan', 'nan', 'nan', 'nan', 'nan', 'n

12.4.4 Membuat Model Aturan Asosiasi

Selanjutnya, kita jalankan kode untuk aturan asosiasi menggunakan algoritma apriori dengan cara memanggil fungsi apriori. Adapun fungsi apriori memiliki beberapa parameter sebagai berikut:

1. Parameter pertama adalah array yang akan digunakan yaitu records.
2. Parameter kedua adalah nilai minimum support (`min_support=0.0045`) yang berarti nilai minimum confidence yang diimplementasikan sebagai ambang batas confidence adalah 0.45%
3. Parameter ketiga adalah minimum confidence (`min_confidence=0.2`) yang berarti nilai minimum confidence yang diimplementasikan sebagai ambang batas confidence adalah 20%
4. Parameter keempat adalah minimum lift ratio (`min_lift = 3`) yang berarti nilai minimum lift ratio yang diimplementasikan sebagai ambang batas lift ratio adalah 3.
5. Parameter kelima adalah `min_length = 2` adalah jumlah minimum item yang kita inginkan dalam aturan asosiasi. Disini kita menggunakan `min_length = 2` berarti kita menginginkan setidaknya dua produk dalam aturan asosiasi yang dihasilkan.

Hasil dari fungsi apriori disimpan ke dalam variabel `association_rules`, kemudian `association_results` akan menyimpan `association_rules` yang telah diubah menjadi sebuah list. Terakhir, kita cetak nilai `association_results` dengan menggunakan iterasi.

```
[ ] association_rules = apriori(records, min_support=0.0045,
                                min_confidence = 0.2,
                                min_lift = 3,
                                min_length = 2)

association_results = list(association_rules)
for i in association_results:
    print(i, "\n")
```

```

RelationRecord(items=frozenset({'chicken', 'light cream'}), support=0.004533333333333334, ordered_statistics=[OrderedStatistic(items_base=frozenset({'light cream'}), items_add=frozenset({'chicken'})
RelationRecord(items=frozenset({'mushroom cream sauce', 'escalope'}), support=0.005733333333333333, ordered_statistics=[OrderedStatistic(items_base=frozenset({'mushroom cream sauce'}), items_add=fro
RelationRecord(items=frozenset({'pasta', 'escalope'}), support=0.005866666666666667, ordered_statistics=[OrderedStatistic(items_base=frozenset({'pasta'}), items_add=frozenset({'escalope'}), confide
RelationRecord(items=frozenset({'ground beef', 'herb & pepper'}), support=0.016, ordered_statistics=[OrderedStatistic(items_base=frozenset({'herb & pepper'}), items_add=frozenset({'ground beef'}),
RelationRecord(items=frozenset({'ground beef', 'tomato sauce'}), support=0.005333333333333333, ordered_statistics=[OrderedStatistic(items_base=frozenset({'tomato sauce'}), items_add=frozenset({'gro
RelationRecord(items=frozenset({'whole wheat pasta', 'olive oil'}), support=0.008, ordered_statistics=[OrderedStatistic(items_base=frozenset({'whole wheat pasta'}), items_add=frozenset({'olive oil'
RelationRecord(items=frozenset({'shrimp', 'pasta'}), support=0.005066666666666666, ordered_statistics=[OrderedStatistic(items_base=frozenset({'pasta'}), items_add=frozenset({'shrimp'}), confidence=
RelationRecord(items=frozenset({'nan', 'chicken', 'light cream'}), support=0.004533333333333334, ordered_statistics=[OrderedStatistic(items_base=frozenset({'light cream'}), items_add=frozenset({'ne
RelationRecord(items=frozenset({'shrimp', 'frozen vegetables', 'chocolate'}), support=0.005333333333333333, ordered_statistics=[OrderedStatistic(items_base=frozenset({'frozen vegetables', 'chocolat
RelationRecord(items=frozenset({'ground beef', 'cooking oil', 'spaghetti'}), support=0.0048, ordered_statistics=[OrderedStatistic(items_base=frozenset({'ground beef', 'cooking oil'}), items_add=fro
RelationRecord(items=frozenset({'nan', 'mushroom cream sauce', 'escalope'}), support=0.005733333333333333, ordered_statistics=[OrderedStatistic(items_base=frozenset({'mushroom cream sauce'}), items
RelationRecord(items=frozenset({'nan', 'pasta', 'escalope'}), support=0.005866666666666667, ordered_statistics=[OrderedStatistic(items_base=frozenset({'pasta'}), items_add=frozenset({'nan', 'escal
RelationRecord(items=frozenset({'ground beef', 'frozen vegetables', 'spaghetti'}), support=0.008666666666666666, ordered_statistics=[OrderedStatistic(items_base=frozenset({'frozen vegetables', 'spa
RelationRecord(items=frozenset({'milk', 'frozen vegetables', 'olive oil'}), support=0.0048, ordered_statistics=[OrderedStatistic(items_base=frozenset({'milk', 'frozen vegetables'}), items_add=froze
RelationRecord(items=frozenset({'mineral water', 'frozen vegetables', 'shrimp'}), support=0.0072, ordered_statistics=[OrderedStatistic(items_base=frozenset({'mineral water', 'shrimp'}), items_add=f
RelationRecord(items=frozenset({'frozen vegetables', 'olive oil', 'spaghetti'}), support=0.005733333333333333, ordered_statistics=[OrderedStatistic(items_base=frozenset({'frozen vegetables', 'spag
RelationRecord(items=frozenset({'shrimp', 'frozen vegetables', 'spaghetti'}), support=0.006, ordered_statistics=[OrderedStatistic(items_base=frozenset({'frozen vegetables', 'spaghetti'}), items_add=
RelationRecord(items=frozenset({'frozen vegetables', 'tomatoes', 'spaghetti'}), support=0.006666666666666667, ordered_statistics=[OrderedStatistic(items_base=frozenset({'frozen vegetables', 'spaghe
RelationRecord(items=frozenset({'ground beef', 'spaghetti', 'grated cheese'}), support=0.005333333333333333, ordered_statistics=[OrderedStatistic(items_base=frozenset({'spaghetti', 'grated cheese'
RelationRecord(items=frozenset({'ground beef', 'herb & pepper', 'mineral water'}), support=0.006666666666666667, ordered_statistics=[OrderedStatistic(items_base=frozenset({'mineral water', 'herb &
RelationRecord(items=frozenset({'ground beef', 'nan', 'herb & pepper'}), support=0.016, ordered_statistics=[OrderedStatistic(items_base=frozenset({'herb & pepper'}), items_add=frozenset({'ground be
RelationRecord(items=frozenset({'ground beef', 'spaghetti', 'herb & pepper'}), support=0.0064, ordered_statistics=[OrderedStatistic(items_base=frozenset({'spaghetti', 'herb & pepper'}), items_add=f
RelationRecord(items=frozenset({'ground beef', 'milk', 'olive oil'}), support=0.004933333333333333, ordered_statistics=[OrderedStatistic(items_base=frozenset({'ground beef', 'milk'}), items_add=fro
RelationRecord(items=frozenset({'ground beef', 'nan', 'tomato sauce'}), support=0.005333333333333333, ordered_statistics=[OrderedStatistic(items_base=frozenset({'tomato sauce'}), items_add=frozens

```

```
[ ] print(association_results[0])
```

```
➦ RelationRecord(items=frozenset({'chicken', 'light cream'}), support=0.004533333333333334,
```

12.4.5 Mencetak Rules, Support, Confidence, dan Lift Ratio

Kode berikut menampilkan aturan asosiasi, support, confidence, dan lift ratio untuk setiap aturan asosiasi:

```

[ ] for item in association_results:
    pair = item[0]
    items = [x for x in pair]

    print("Rule: " + items[0] + " -> " + items[1])
    print("Support: " + str(item[1]))
    print("Confidence: " + str(item[2][0][2]))
    print("Lift: " + str(item[2][0][3]))
    print("=====")

```

```

Rule: chicken -> light cream
Support: 0.004533333333333334
Confidence: 0.2905982905982906
Lift: 4.843304843304844
=====
Rule: mushroom cream sauce -> escalope
Support: 0.005733333333333333
Confidence: 0.30069930069930073
Lift: 3.7903273197390845
=====
Rule: pasta -> escalope
Support: 0.005866666666666667
Confidence: 0.37288135593220345
Lift: 4.700185158809287
=====
Rule: ground beef -> herb & pepper
Support: 0.016
Confidence: 0.3234501347708895
Lift: 3.2915549671393096
=====
Rule: ground beef -> tomato sauce
Support: 0.005333333333333333
Confidence: 0.37735849056603776
Lift: 3.840147461662528
=====
Rule: whole wheat pasta -> olive oil
Support: 0.008
Confidence: 0.2714932126696833
Lift: 4.130221288078346
=====
Rule: shrimp -> pasta
Support: 0.005066666666666666
Confidence: 0.3220338983050848
Lift: 4.514493901473151
=====
Rule: nan -> chicken
Support: 0.004533333333333334
Confidence: 0.2905982905982906
Lift: 4.843304843304844
=====
Rule: shrimp -> frozen vegetables
Support: 0.005333333333333333
Confidence: 0.23255813953488372
Lift: 3.260160834601174
=====
Rule: ground beef -> cooking oil
Support: 0.0048
Confidence: 0.5714285714285714
Lift: 3.281557646029315

```

Nilai support untuk aturan pertama adalah 0.0045. Jumlah ini dihitung dengan membagi jumlah transaksi yang mengandung light cream dibagi dengan jumlah total transaksi. Tingkat confidence untuk aturan tersebut adalah 0.2905 yang menunjukkan bahwa dari semua transaksi yang mengandung light cream, 29.05% transaksi juga mengandung chicken. Terakhir, lift 4.84 berarti bahwa pembelian chicken 4.84 kali lebih mungkin dibeli oleh pelanggan yang membeli light cream dibandingkan dengan kemungkinan penjualan ayam secara umum.

TUGAS

Terdapat dataset pada Grocery Store Dataset yang dapat diunduh pada link berikut:
https://gitea.ums.ac.id/yusufsn/Praktikum_DWDM/src/branch/master/Data/Bab12/GroceryStoreDataSet.csv

Pada dataset tersebut terdapat 19 data transaksi dengan daftar item sebagai berikut:

	product
0	MILK,BREAD,BISCUIT
1	BREAD,MILK,BISCUIT,CORNFLAKES
2	BREAD,TEA,BOURNVITA
3	JAM,MAGGI,BREAD,MILK
4	MAGGI,TEA,BISCUIT
5	BREAD,TEA,BOURNVITA
6	MAGGI,TEA,CORNFLAKES
7	MAGGI,BREAD,TEA,BISCUIT
8	JAM,MAGGI,BREAD,TEA
9	BREAD,MILK
10	COFFEE,COCK,BISCUIT,CORNFLAKES
11	COFFEE,COCK,BISCUIT,CORNFLAKES
12	COFFEE,SUGER,BOURNVITA
13	BREAD,COFFEE,COCK
14	BREAD,SUGER,BISCUIT
15	COFFEE,SUGER,CORNFLAKES
16	BREAD,SUGER,BOURNVITA
17	BREAD,COFFEE,SUGER
18	BREAD,COFFEE,SUGER
19	TEA,MILK,COFFEE,CORNFLAKES

Kemudian kerjakanlah soal-soal berikut ini:

```
#1
GroceryStore = pd.read_csv("/content/drive/MyDrive/dwdm/GroceryStoreDataSet.csv", names = ["product"], header=None)
GroceryStore

[ ] records = list(GroceryStore["product"].apply(lambda x: x.split(",")))
print(records)

[[ 'MILK', 'BREAD', 'BISCUIT'], [ 'BREAD', 'MILK', 'BISCUIT', 'CORNFLAKES'],
```

1. Dengan ketentuan jumlah minimum support = 0.3% dan mininum confidence = 20%. Tuliskan hasil aturan asosiasi yang dihasilkan?

```
association_results = list(association_rules)
for i in association_results:
    print(i, "\n")
```

```
[ ] print(association_results[0])
```



```

▶ for item in association_results:
    pair = item[0]
    items = [x for x in pair]

    print("Rule: " + items[0] + " -> " + items[1])
    print("Support: " + str(item[1]))
    print("Confidence: " + str(item[2][0][2]))
    print("Lift: " + str(item[2][0][3]))
    print("=====")

```

```
Confidence: 1.0
Lift: 3.3333333333333335
=====
Rule: TEA -> COFFEE
Support: 0.05
Confidence: 1.0
Lift: 3.3333333333333335
=====
Rule: TEA -> MILK
Support: 0.05
Confidence: 0.2
Lift: 4.0
=====
Rule: TEA -> MILK
Support: 0.05
Confidence: 1.0
Lift: 3.3333333333333335
=====
Rule: MAGGI -> MILK
Support: 0.05
Confidence: 0.5
Lift: 10.0
=====
Rule: MAGGI -> TEA
Support: 0.05
Confidence: 0.2
Lift: 4.0
=====
Rule: BISCUIT -> BREAD
Support: 0.05
Confidence: 0.2
Lift: 4.0
=====
Rule: BISCUIT -> TEA
Support: 0.05
Confidence: 0.2
Lift: 4.0
=====
Rule: BISCUIT -> COFFEE
Support: 0.1
Confidence: 0.6666666666666667
Lift: 6.666666666666667
=====
Rule: MAGGI -> MILK
Support: 0.05
Confidence: 0.5
Lift: 10.0
=====
Rule: MAGGI -> TEA
Support: 0.05
```

To exit full screen

2. Dengan ketentuan jumlah minimum support = 0.7% dan minimum confidence = 60%. Tuliskan hasil aturan asosiasi yang dihasilkan?

```
[ ] association_rules = apriori(records, min_support=0.007,
                                min_confidence = 0.6,
                                min_lift = 3,
                                min_length = 2)

association_results = list(association_rules)
for i in association_results:
    print(i, "\n")
```



```

RelationRecord(items=frozenset({'MAGGI', 'JAM'}), support=0.1, ordered_statistics=[OrderedStatistic(items_base=frozenset({'JAM'}), items_add=frozenset({'MAGGI'}), confidence=1.0, lift=4.0)])
RelationRecord(items=frozenset({'BISCUIT', 'COFFEE', 'COCK'}), support=0.1, ordered_statistics=[OrderedStatistic(items_base=frozenset({'COCK'}), items_add=frozenset({'BISCUIT', 'COFFEE'}), confidence=1.0, lift=4.0)])
RelationRecord(items=frozenset({'BISCUIT', 'COCK', 'CORNFLAKES'}), support=0.1, ordered_statistics=[OrderedStatistic(items_base=frozenset({'COCK'}), items_add=frozenset({'BISCUIT', 'CORNFLAKES'}), confidence=1.0, lift=4.0)])
RelationRecord(items=frozenset({'BISCUIT', 'COFFEE', 'CORNFLAKES'}), support=0.1, ordered_statistics=[OrderedStatistic(items_base=frozenset({'BISCUIT', 'COFFEE'}), items_add=frozenset({'CORNFLAKES'}), confidence=1.0, lift=4.0)])
RelationRecord(items=frozenset({'BISCUIT', 'TEA', 'MAGGI'}), support=0.1, ordered_statistics=[OrderedStatistic(items_base=frozenset({'BISCUIT', 'TEA'}), items_add=frozenset({'MAGGI'}), confidence=1.0, lift=4.0)])
RelationRecord(items=frozenset({'SUGER', 'COFFEE', 'BOURNVITA'}), support=0.05, ordered_statistics=[OrderedStatistic(items_base=frozenset({'COFFEE', 'BOURNVITA'}), items_add=frozenset({'SUGER'}), confidence=1.0, lift=4.0)])
RelationRecord(items=frozenset({'BREAD', 'CORNFLAKES', 'MILK'}), support=0.05, ordered_statistics=[OrderedStatistic(items_base=frozenset({'BREAD', 'CORNFLAKES'}), items_add=frozenset({'MILK'}), confidence=1.0, lift=4.0)])
RelationRecord(items=frozenset({'MAGGI', 'BREAD', 'JAM'}), support=0.1, ordered_statistics=[OrderedStatistic(items_base=frozenset({'JAM'}), items_add=frozenset({'MAGGI', 'BREAD'}), confidence=1.0, lift=4.0)])
RelationRecord(items=frozenset({'CORNFLAKES', 'COFFEE', 'COCK'}), support=0.1, ordered_statistics=[OrderedStatistic(items_base=frozenset({'COCK'}), items_add=frozenset({'COFFEE', 'CORNFLAKES'}), confidence=1.0, lift=4.0)])
RelationRecord(items=frozenset({'MILK', 'COFFEE', 'CORNFLAKES'}), support=0.05, ordered_statistics=[OrderedStatistic(items_base=frozenset({'MILK', 'COFFEE'}), items_add=frozenset({'CORNFLAKES'}), confidence=1.0, lift=4.0)])
RelationRecord(items=frozenset({'TEA', 'COFFEE', 'CORNFLAKES'}), support=0.05, ordered_statistics=[OrderedStatistic(items_base=frozenset({'TEA', 'COFFEE'}), items_add=frozenset({'CORNFLAKES'}), confidence=1.0, lift=4.0)])
RelationRecord(items=frozenset({'TEA', 'MILK', 'COFFEE'}), support=0.05, ordered_statistics=[OrderedStatistic(items_base=frozenset({'TEA', 'COFFEE'}), items_add=frozenset({'MILK'}), confidence=1.0, lift=4.0)])
RelationRecord(items=frozenset({'TEA', 'MILK', 'CORNFLAKES'}), support=0.05, ordered_statistics=[OrderedStatistic(items_base=frozenset({'TEA', 'MILK'}), items_add=frozenset({'CORNFLAKES'}), confidence=1.0, lift=4.0)])
RelationRecord(items=frozenset({'MAGGI', 'MILK', 'JAM'}), support=0.05, ordered_statistics=[OrderedStatistic(items_base=frozenset({'MILK', 'JAM'}), items_add=frozenset({'MAGGI'}), confidence=1.0, lift=4.0)])
RelationRecord(items=frozenset({'MAGGI', 'TEA', 'JAM'}), support=0.05, ordered_statistics=[OrderedStatistic(items_base=frozenset({'TEA', 'JAM'}), items_add=frozenset({'MAGGI'}), confidence=1.0, lift=4.0)])
RelationRecord(items=frozenset({'BISCUIT', 'BREAD', 'CORNFLAKES', 'MILK'}), support=0.05, ordered_statistics=[OrderedStatistic(items_base=frozenset({'BREAD', 'CORNFLAKES'}), items_add=frozenset({'BISCUIT', 'MILK'}), confidence=1.0, lift=4.0)])
RelationRecord(items=frozenset({'BISCUIT', 'TEA', 'BREAD', 'MAGGI'}), support=0.05, ordered_statistics=[OrderedStatistic(items_base=frozenset({'BISCUIT', 'TEA', 'BREAD'}), items_add=frozenset({'MAGGI'}), confidence=1.0, lift=4.0)])
RelationRecord(items=frozenset({'BISCUIT', 'COFFEE', 'COCK', 'CORNFLAKES'}), support=0.1, ordered_statistics=[OrderedStatistic(items_base=frozenset({'COCK'}), items_add=frozenset({'BISCUIT', 'COFFEE', 'CORNFLAKES'}), confidence=1.0, lift=4.0)])
RelationRecord(items=frozenset({'MAGGI', 'MILK', 'BREAD', 'JAM'}), support=0.05, ordered_statistics=[OrderedStatistic(items_base=frozenset({'MILK', 'JAM'}), items_add=frozenset({'MAGGI', 'BREAD'}), confidence=1.0, lift=4.0)])
RelationRecord(items=frozenset({'MAGGI', 'TEA', 'BREAD', 'JAM'}), support=0.05, ordered_statistics=[OrderedStatistic(items_base=frozenset({'TEA', 'JAM'}), items_add=frozenset({'MAGGI', 'BREAD'}), confidence=1.0, lift=4.0)])
RelationRecord(items=frozenset({'TEA', 'MILK', 'COFFEE', 'CORNFLAKES'}), support=0.05, ordered_statistics=[OrderedStatistic(items_base=frozenset({'MILK', 'COFFEE'}), items_add=frozenset({'TEA', 'CORNFLAKES'}), confidence=1.0, lift=4.0)])

```

```

▶ print(association_results[0])
➡ RelationRecord(items=frozenset({'MAGGI', 'JAM'}), support=0.1, ordered_statistic

```

```

[ ] for item in association_results:
    pair = item[0]
    items = [x for x in pair]

    print("Rule: " + items[0] + " -> " + items[1])
    print("Support: " + str(item[1]))
    print("Confidence: " + str(item[2][0][2]))
    print("Lift: " + str(item[2][0][3]))
    print("=====")

```

```
Rule: MAGGI -> JAM
Support: 0.1
Confidence: 1.0
Lift: 4.0
=====
Rule: BISCUIT -> COFFEE
Support: 0.1
Confidence: 0.6666666666666667
Lift: 6.666666666666667
=====
Rule: BISCUIT -> COCK
Support: 0.1
Confidence: 0.6666666666666667
Lift: 4.444444444444445
=====
Rule: BISCUIT -> COFFEE
Support: 0.1
Confidence: 1.0
Lift: 3.3333333333333335
=====
Rule: BISCUIT -> TEA
Support: 0.1
Confidence: 1.0
Lift: 4.0
=====
Rule: SUGER -> COFFEE
Support: 0.05
Confidence: 1.0
Lift: 3.3333333333333335
=====
Rule: BREAD -> CORNFLAKES
Support: 0.05
Confidence: 1.0
Lift: 4.0
=====
Rule: MAGGI -> BREAD
Support: 0.1
Confidence: 1.0
Lift: 6.666666666666667
=====
Rule: CORNFLAKES -> COFFEE
Support: 0.1
Confidence: 0.6666666666666667
Lift: 3.3333333333333335
=====
Rule: MILK -> COFFEE
Support: 0.05
Confidence: 1.0
Lift: 3.3333333333333335
```

To exit full screen, press Esc

3. Jelaskan bagaimana nilai minimum support dan minimum confidence memengaruhi aturan asosiasi dan nilai lift ratio yang dihasilkan.
- Penetapan nilai minimum support dan minimum confidence harus seimbang agar aturan yang dihasilkan tidak terlalu banyak (*overfitting*) atau terlalu sedikit (*underfitting*).