Nama : Iqbal Yusuf

NIM : G.231.22.0104

Mata Kuliah : Data Mining

Penjelasan Coding Pada Praktikum 4

1. Cell 1

```
import pandas as pd
from mlxtend.frequent_patterns import apriori
from mlxtend.frequent_patterns import association_rules

df = pd.read_excel('http://archive.ics.uci.edu/ml/machine-learning-databases/00352/Online%20Retail.xlsx')
df.head()
```

- Memasukkan library pandas (untuk menganalis dan manipulasi data) kemudian disingkat atau dinamakan ulang dengan “pd”.
- Memasukkan apriori (salah satu algoritma untuk menemukan frequent itemsets & association rules dalam dataset transaksi) dari modul frequent patterns dalam library mlxtend (salah satu ekstensi untuk mesin learning).
- Memasukkan association rules (aturan dari frequent itemsets yang menunjukkan hubungan antara item-tem) dari modul frequent patterns dalam library mlxtend (salah satu ekstensi untuk mesin learning)
- Membaca data excel dengan pandas (disingkat “pd”) dan memuatnya dalam dataframe yang didefinisikan / disingkat df. Jadi hanya dengan memanggil df, maka bisa membaca data excel tersebut.
- df.head() = menampilkan kepada / head pada dataframe yang telah dibuat yaitu df, dan menampilkan data awal sebanyak 5 buah data.

2. Cell 2

```
df['Description'] = df['Description'].str.strip()
df.dropna(axis=0, subset=['InvoiceNo'], inplace=True)
df['InvoiceNo'] = df['InvoiceNo'].astype('str')
df = df[~df['InvoiceNo'].str.contains('C')]
```

- df['Description'] = df['Description'].str.strip() : membuat definisi baru yaitu df[‘Description’] yang mana untuk mengakses kolom Deskripsi dalam df (DataFrame) yang memiliki data string, maka perlu .str , kemudian kita ingin menghilangkan spasi putih pada data string pada kolom deskripsi dengan strip(). Pada tahap ini kita melakukan cleaning data.
- df.dropna(axis=0, subset=['InvoiceNo'], inplace=True) : df.dropna digunakan untuk menghapus pada DataFrame (df) yang memiliki nilai “NaN” atau kosong dengan

beberapa parameter, yaitu axis = 0 yang berarti nilai kosong yang ingin dihapus adalah pada barisnya (jika axis = 1 maka yang dihapus pada kolomnya), lalu subset=['InvoiceNo'] berarti kolom yang ingin dihapus nilai kosongnya yaitu kolom "InvoiceNo", dan inplace=True berarti operasi penghapusan nilai kosong itu terjadi pada DataFrame asli (jika False maka akan membuat Salinan baru).

- df['InvoiceNo'] = df['InvoiceNo'].astype('str') : mengubah tipe data pada kolom "InvoiceNo" menjadi str atau string dan disimpan kembali pada definisi df['InvoiceNo'].
- df = df[~df['InvoiceNo'].str.contains('C')] : melakukan filtering pada DataFrame, dimana dalam kolom InvoiceNo memiliki karakter "C" akan dihilangkan. Contains sendiri berguna untuk memeriksa setiap elemen, dimana jika True maka elemen dalam kolom InvoiceNo mengandung 'C' dan begitu sebaliknya. Jika ingin membalikknya maka pengunaan ~ atau operator NOT dibutuhkan, untuk membalikannya penilaian pada contains.

3. Cell 3

```
basket = (df[df['Country'] =="France"]
          .groupby(['InvoiceNo', 'Description'])['Quantity']
          .sum().unstack().reset_index().fillna(0)
          .set_index('InvoiceNo'))
```

- Membuat DataFrame "basket" yang memuat beberapa hal yaitu
  - ❖ Memilih dari DataFrame utama (df) yang memiliki 'Country" = "France".
  - ❖ Membuat kelompok gabungan dari 'InvoiceNo' dan 'Description' menggunakan groupby() dan menggunakan .sum() untuk menambahkan pada tiap isi dari kolom 'Quantity' jika memiliki 'InvoiceNo' dan 'Description' yang sama.
  - ❖ unstack() digunakan untuk mengubah indeks menjadi kolom, dimana dari grup yang telah dibuat tadi 'Description' menjadi kolom dan 'InvoiceNo' menjadi baris.
  - ❖ Mengatur ulang indeks DataFrame dari multilevel menjadi kolom datar, menggunakan .reset_indeks().
  - ❖ fillna(0) digunakan untuk mengganti nilai 'NaN' menjadi 0 karena menggunakan operasi unstack(), karena operasi unstack() membuat nilai yang tidak ada akan diisi 'NaN'.
  - ❖ .set_index('InvoiceNo') digunakan untuk mengubah 'InvoiceNo' dari kolom menjadi indeks utama.

4. Cell 4

```
def encode_units(x):
    if x <= 0:
        return 0
    if x >= 1:
        return 1

basket_sets = basket.applymap(encode_units)
basket_sets.drop('POSTAGE', inplace=True, axis=1)
```

- Membuat variabel baru dengan ‘def’ untuk membuat fungsi ‘encode_unit(x)’ dimana fungsi tersebut hanya menerima argument untuk x. Dalam fungsi tersebut menggunakan if yang memiliki 2 aturan. Jika argument x <= 0 maka hasilnya 0, jika x >= 1 maka hasilnya 1. Maksud pembuatan fungsi tersebut untuk mengelompokkan secara biner (0 / 1).
- Lalu membuat variabel baru “basket_sets” menggunakan .applymap untuk menerapkan pada setiap DataFrame ‘basket’ menggunakan metode fungsi ‘encode_units’.
- .drop digunakan untuk menghapus kolom/baris, dimana yang dihapus adalah kolom ‘POSTAGE’, hasil dari ubahan tersebut akan disimpan tanpa salinan dengan inplace=True, dan axis=1 untuk menunjukkan bahwa yang dihapus ialah kolom, yang mengacu pada kolom ‘POSTAGE’ tadi.

5. Cell 5

```
frequent_itemsets = apriori(basket_sets, min_support=0.07, use_colnames=True)
```

- Membuat variabel baru ‘frequent_itemsets” dengan fungsi apriori yang memiliki argument yaitu, basket_sets, mengacu DataFrame biner yang sudah dibuat sebelumnya, kemudian dipanggil kembali, lalu min_support untuk memiliki nilai minimum pada support harus sama dengan atau lebih besar 0.07 atau 7%. ‘use_colnames=True untuk memastikan penggunakan kolom nama asli dari DataFrame digunakan pada hasil itemsets.

6. Cell 6

```
rules = association_rules(frequent_itemsets, metric="lift", min_threshold=1)
rules.head()
```

- Membuat variabel baru ‘rules’ dengan fungsi ‘association_rules’ yang memiliki argument yakni, frequent_itemsets untuk mengakses variabel yang berisi itemsets apriori tadi, metric=’lift’ bermakna metode evaluasi yang digunakan ialah lift (mengukur kehadiran A meningkatkan kemungkinan B terjadi, lift bernilai lebih dari 1 menunjukkan adanya hubungan A dan B), serta min_threshold=1 bermakna batas minimum metrik lift adalah >= 1, dan akan diperlihatkan dalam hasil.
- rules.head() : menampilkan lima baris pertama dari DataFrame ‘rules’.

7. Cell 7

```
rules[ (rules['lift'] >= 6) &
       (rules['confidence'] >= 0.8) ]
```

- Mengakses variabel rules dengan filter dimana rules harus memiliki kondisi karena menggunakan operator & atau AND.
- rules[‘lift’] >=6 : aturan pertama yaitu nilai lift harus lebih besar atau sama dengan 6.

- rules[‘confidence’] >= 0.8 : aturan kedua yakni nilai confidence harus lebih besar atau sama dengan 0.8.
- Maka DataFrame ‘rules’ yang akan ditampilkan yang memenuhi syarat lift >= 6 dan confidence >= 0.8.

8. Cell 8

```
basket['ALARM CLOCK BAKELIKE GREEN'].sum()

340.0

basket['ALARM CLOCK BAKELIKE RED'].sum()

316.0
```

- Menghitung hasil total kuantitas pada produk ‘ALARM CLOCK BAKELIKE GREEN’ dalam DataFrame ‘basket’ yakni 340 unit dan itu berasal dari masing-masing transaksi pada tiap DataFrame, makanya .sum() diperlukan.
- Menghitung hasil total kuantitas pada produk ‘ALARM CLOCK BAKELIKE RED’ dalam DataFrame ‘basket’ yakni 316 unit dan itu berasal dari masing-masing transaksi pada tiap DataFrame, makanya .sum() diperlukan.

9. Cell 9

```
basket2 = (df[df['Country'] =="Germany"]
          .groupby(['InvoiceNo', 'Description'])['Quantity']
          .sum().unstack().reset_index().fillna(0)
          .set_index('InvoiceNo'))

basket_sets2 = basket2.applymap(encode_units)
basket_sets2.drop('POSTAGE', inplace=True, axis=1)
frequent_itemsets2 = apriori(basket_sets2, min_support=0.05, use_colnames=True)
rules2 = association_rules(frequent_itemsets2, metric="lift", min_threshold=1)

rules2[ (rules2['lift'] >= 4) &
        (rules2['confidence'] >= 0.5)]
```

- Membuat variabel baru ‘basket2’ dengan beberapa syarat yaitu, ‘Country’ dalam df atau DataFrame adalah ‘Germany’. Kemudian membuat kelompok dengan .groupby() untuk ‘InvoiceNo’ dan ‘Description’ serta menambahkan data dalam kolom ‘Quantity’ dengan .sum(). Lalu .unstack() digunakan untuk mengubah kelompok ‘InvoiceNo’ dan ‘Description’ dari indeks menjadi kolom, dimana nantinya ‘InvoiceNo’ menjadi barisnya sementara ‘Description’ menjadi kolom. Setelah itu .reset_index() digunakan untuk mengatur ulang indeks DataFrame dari multilevel menjadi kolom datar. Kemudian fillna(0) untuk mengubah nilai ‘NaN’ atau kosong karena fungsi unstack() menjadi nilai 0. Terakhir set_index(‘InvoiceNo’) untuk mengatur atau menetapkan indeksnya menjadi kolom ‘InvoiceNo’.

- Membuat variabel basket_set2 menggunakan .applymap untuk menerapkan pada setiap DataFrame ‘basket’ menggunakan metode fungsi ‘encode_units’. Fungsi tersebut sudah dibuat di Cell 4.
- .drop digunakan untuk menghapus kolom/baris pada DataFrame basket_sets2, dimana yang dihapus adalah kolom ‘POSTAGE’, hasil dari ubahan tersebut akan disimpan tanpa salinan dengan inplace=True, dan axis=1 untuk menunjukkan bahwa yang dihapus ialah kolom, yang mengacu pada kolom ‘POSTAGE’ tadi.
- Membuat variabel baru ‘frequent_itemsets2” dengan fungsi apriori yang memiliki argument yaitu, basket_sets2, mengacu DataFrame biner yang sudah dibuat sebelumnya, kemudian dipanggil kembali, lalu min_support untuk memiliki nilai minimum pada support harus sama dengan atau lebih besar 0.05 atau 5%. ‘use_colnames=True untuk memastikan penggunakan kolom nama asli dari DataFrame digunakan pada hasil itemsets.
- Membuat variabel baru ‘rules2’ dengan fungsi ‘association_rules’ yang memiliki argument yakni, frequent_itemsets2 untuk mengakses variabel yang berisi itemsets apriori tadi, metric=’lift’ bermakna metode evaluasi yang digunakan ialah lift (mengukur kehadiran A meningkatkan kemungkinan B terjadi, lift bernilai lebih dari 1 menunjukkan adanya hubungan A dan B), serta min_threshold=1 bermakna batas minimum metrik lift adalah >= 1, dan akan diperlihatkan dalam hasil.
- Mengakses variabel ‘rules2’ dengan filter dimana ‘rules2’ itu harus memiliki kondisi karena menggunakan operator & atau AND. rules[‘lift’] >= 4 : aturan pertama yaitu nilai lift harus lebih besar atau sama dengan 4. rules[‘confidence’] >= 0.5 : aturan kedua yakni nilai confidence harus lebih besar atau sama dengan 0.5.
- Maka DataFrame ‘rules’ yang akan ditampilkan yang memenuhi syarat lift >= 4 dan confidence >= 0.5.