

ANALISIS CODING

Nama : Nabela Yulian Anggraini

Nim : G.231.22.0026

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()
```

- Import panda ini untuk membaca ,memanipulasi dan menganalisis data
- Mlxtend.frequent_patterns untuk menemukan pola pembelian yang sering terjadi dan menghasilkan aturan asosiasi.
- df = pd.read_excel dalam coding ini memuat data ritel online dari url yang diberikan kedalam objek df menggunakan library pandas.
- df.head coding yang menampilkan baris awal dan untuk melihat sekilas struktur data.

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')]
```

- baris awal untuk menghapus karakter spasi diawal dan diakhir dari setiap nilai dalam kolom 'Description'.yang berguna untuk memastikan konsistensi dalam data,karena spasi ekstra dapat menyebabkan masalah selama analisis.
- Baris dua menghapus semua baris dari dataframe dimana kolom 'invoiceNo' memiliki nilai yang hilang(NaN). 'axis=0' menunjukan bahwa hanya kolom 'invoiceNo' yang diperiksa untuk nilai NaN. 'inplace=True' memodifikasi dataframe secara langsung tanpa perlu menetapkan kembali ke 'df'.
- Baris tiga mengubah tip data dari kolom 'invoiceNo' menjadi string.
- Baris ke 4 memfilter baris-baris dimana kolom 'invoiceNo' mengandung karakter 'C'. Operator tilde '~' digunakan untuk membalikkan mask Boolean yang dihasilkan oleh 'df['InvoiceNo'].str.contains('C')', yang berarti hanya baris dimana 'invoiceNo' tidak mengandung 'C' yang dipertahankan.

3.

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

- a) Memfilter data untuk prancis
- b) Mengelompokkan data berdasarkan nomer faktur dan diskripsi produk
- c) Mengubah bentuk data(unstack)
- d) Mereset indeks
- e) Mengisi nilai kosong dengan nol
- f) Menetapkan 'invoiceNo'

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)
```

- a) Fungsi 'encode_units' mengonversi nilai kuantitas produk menjadi nilai biner dimana mengubah data transaksi menjadi format yang cocok untuk analisis asosiasi.
 - Jika nilai x kurang dari atau sama dengan 0, maka dikonversikan menjadi 0.
 - Jika nilai x lebih dari atau sama dengan 1, maka dikonversikan menjadi 1.
- b) Baris kedua (`basket_set = basket.applymap(econde_units)`). Fungsi 'encode_units' menerapkan fungsi kesetiap elemen dalam dataframe 'basket' menggunakan 'applymap'.
- c) Baris ini menghapus kolom 'POSTAGE' dari datafraame 'basket_sets'
 - 'Inplace=True' memastikan bahwa perubahan dilakukan langsung pada 'basket_sets' tanpa perlu menetapkannya kembali.
 - 'axis=1' enunjukkan bahwa kolom yang akan dihapus.

5.

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

Baris ini menerapkan alogirtma apriori pada dataframe 'basket_sets' untuk menemukan itemset yang sering muncul. Algoritma apriori adalah salah satu metode yang ppaling umum digunakan untuk menemukan itemset yang sering dalam data transaksi, yang kemudian dapat digunakan untuk membangun aturan asosiasi.

- 'basket_sets' binr Dimana tiap klom mewakili sebuah priодук dan setiap baris mewakili sebuah trasaksi.
- 'min_support=0,07' paramenter ini menetapkan batas minimum dukungan untuk itemset yag sering muncul yang mencari itemset dalam setidaknya yang muncul 7% dari semua transaksi.

- 'use_colnames=True' parameter ini memastikan bahwa nama kolom asli digunakan dalam hasil itemset yang sering, bukan indeks kolom.

6.

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

coding ini menggunakan fungsi 'association_rules' dari pustaka 'm1xtend' untuk menghasilkan aturan asosiasi dari itemset yang sering muncul ('frequent_itemset').

- Metric="lift" metode evaluasi yang digunakan untuk menyaring aturan asosiasi. lift mengukur sejauh mana kehadiran satu item meningkatkan kemungkinan kehadiran item lain dalam satu transaksi.
- 'min_threshold=1' nilai minimum lift yang perlu untuk menyaring aturan dengan mempertahankan aturan $\text{lift} \geq 1$. $\text{lift} \geq 1$ menunjukkan bahwa item lebih sering muncul bersama dibandingkan secara kebetulan.
- Rules.head coding yang menampilkan hasil lima baris pertama dari dataframe 'rules' yang berisi aturan asosiasi.

7.

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

- Barisan pertama untuk penyaringan aturan asosiasi
'lift' >=6 hanya aturan dengan lift yang lebih besar atau sama dengan 6 yang disertakan.
'confidence >=0.8' hanya aturan dengan confidence yang lebih besar atau sama dengan 0.8 yang disertakan.

8.

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

340.0

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

316.0
```

Jumlah penjualan alarm warna hijau (340) unit sedikit lebih tinggi daripada jumlah penjualan alarm warna merah (316).

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)]
```

- a) Membuat dataframe 'basket2'
- b) Mengonversi kuantitas menjadi format biner.
- c) Menghasilkan itemset yang sering muncul.
- d) Membuat aturan asosiasi.