```
#pip install mlxtend
# pip install openpyxl
import pandas as pd
import numpy as np
import datetime

from mlxtend.frequent_patterns import apriori, association_rules
from mlxtend.preprocessing import TransactionEncoder
# dataset dapat diunduh di https://bit.ly/2USHhwA
# Muat data dari file excel
data = pd.read_excel('data_retail2.xlsx')

# Tampilkan informasi dasar dari dataset
data.info()
data.head(20) #Menampilkan 20 data teratas dari dataset
<b>Preprocessing & Cleansing </b>
# Konversi tanggal Invoice menjadi format yang dikenali komputer
data['InvoiceDate'] = pd.to_datetime(data['InvoiceDate'])
# Menghapus/menghilangkan spasi di awal dan akhi

data['PRODUCT'] = data['PRODUCT'].str.strip()
data['PRODUCT_CATEGORY'] = data['PRODUCT_CATEGORY'].str.strip()
# menghapus semua baris yang memiliki nilai NULL pada kolom ‘InvoiceNo’
data.dropna(axis=0, subset=['InvoiceNo'], inplace=True)
# Menghapus variabel dengan awalan huruf C
data['InvoiceNo'] =  data['InvoiceNo'].astype('str')
data = data[~(data['InvoiceNo'].str[0]== 'C')]
Data transformation
keranjang = (data[data['PROVINSI'] == 'JAWA TENGAH'].groupby(['InvoiceNo',
'PRODUCT_CATEGORY'])['Quantity'].count()\
   .unstack().reset_index().fillna(0)\
```

```
    .set_index('InvoiceNo') )
keranjang.head()
# Menampilkan subset dataset
keranjang.iloc[:, [0,1,2,3,4,5,6,7]].head()
#encoding -> mengubah data string menjadi angka,agar komputer dapat memahami informasi
def encode_data(x):
    if x <= 0:
        return 0
    if x >= 1:
        return 1

keranjang_sets = keranjang.applymap(encode_data)
keranjang_sets.head(5)
# membuat fungsi item yang paling sering dibeli, aturan dan model
# keranjang_sets: Gunakan subset data 'keranjang_sets' untuk menemukan barang yang paling sering dibeli
# min_support: Nilai ambang batas untuk menentukan apakah suatu itemset dianggap sering muncul.
# use_colnames=True: Menggunakan nama kolom (nama item) sebagai label itemsets yang paling sering dibeli.

frequent_itemsets = apriori(keranjang_sets, min_support=0.1, use_colnames=True)
frequent_itemsets
# gunakan algoritma model asosiasi berdasarkan subset data frequent itemsets
# dengan menggunakan metrik “lift”
# dan ambang batas minimum sebesar 1.
# Hasilnya akan disimpan dalam variabel rules1
#
rules1 = association_rules(frequent_itemsets, metric='lift', min_threshold=1 )
rules1.head()
result1 = rules1[(rules1['lift'] >= 1) & (rules1['confidence']) >= 0.8]
```

```
apriori_result = result1.sort_values(by='confidence', ascending=False)
apriori_result.head(20)
```