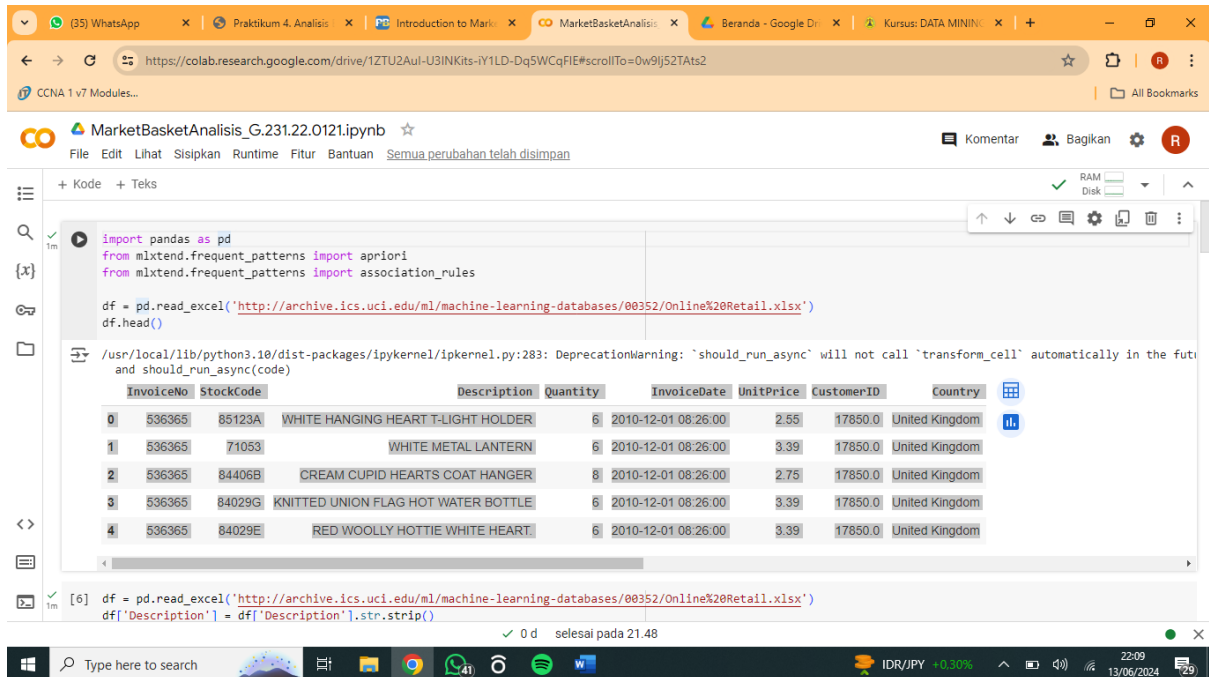


NAMA : RADHITA AMELIA PASHA
NIM : G.231.22.0121
MATKUL : DATA MINING (Praktikum 4)



The screenshot shows a Jupyter Notebook interface with the following code and output:

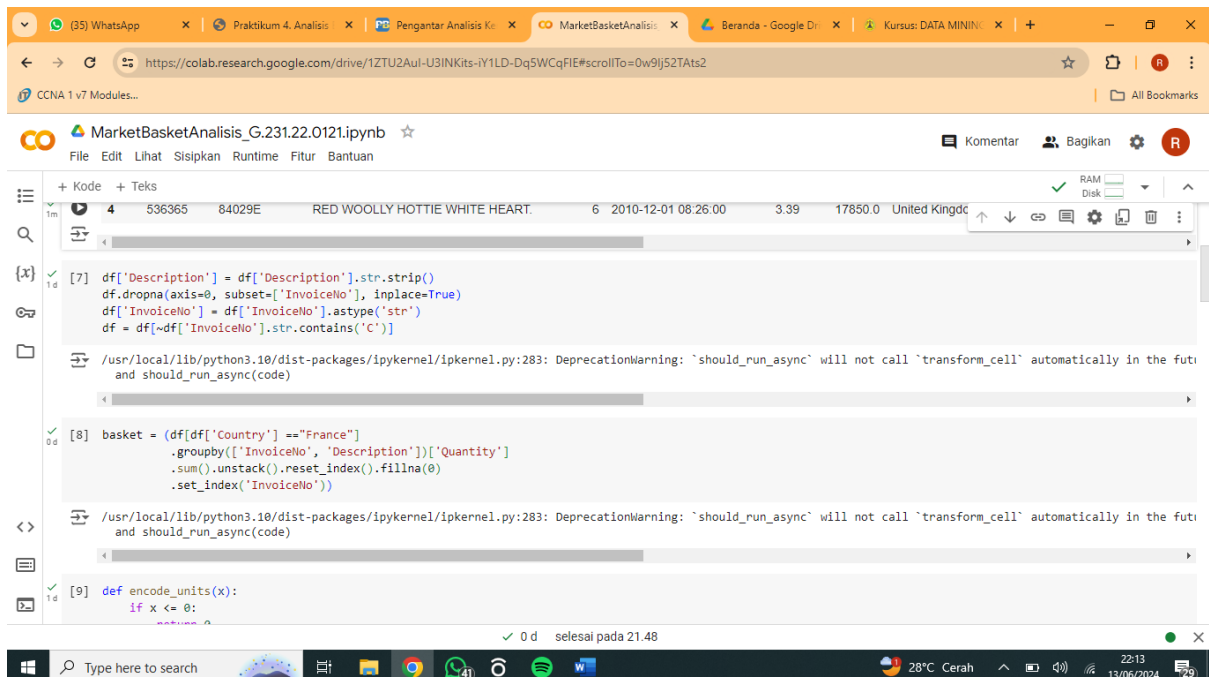
```
import pandas as pd
from mlxtend.frequent_patterns import apriori
from mlxtend.frequent_patterns import association_rules

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

The output displays the first five rows of the dataset:

	InvoiceNo	StockCode	Description	Quantity	InvoiceDate	UnitPrice	CustomerID	Country
0	536365	85123A	WHITE HANGING HEART T-LIGHT HOLDER	6	2010-12-01 08:26:00	2.55	17850.0	United Kingdom
1	536365	71053	WHITE METAL LANTERN	6	2010-12-01 08:26:00	3.39	17850.0	United Kingdom
2	536365	84406B	CREAM CUPID HEARTS COAT HANGER	8	2010-12-01 08:26:00	2.75	17850.0	United Kingdom
3	536365	84029G	KNITTED UNION FLAG HOT WATER BOTTLE	6	2010-12-01 08:26:00	3.39	17850.0	United Kingdom
4	536365	84029E	RED WOOLLY HOTTIE WHITE HEART.	6	2010-12-01 08:26:00	3.39	17850.0	United Kingdom

Memasukkan data dengan menggunakan link yang akan langsung terbaca tanpa memasukkan file terlebih dahulu, sehingga ada sedikit pembersihan, yang perlu kita lakukan. Pertama, beberapa deskripsi memiliki spasi yang perlu dihilangkan. Dan juga akan menghapus baris yang tidak memiliki nomor faktur dan menghapus transaksi kredit (yang memiliki nomor faktur berisi C).



The screenshot shows the following code and output for data cleaning:

```
[7] 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')]
```

The output shows the first row of the cleaned dataset:

	InvoiceNo	StockCode	Description	Quantity	InvoiceDate	UnitPrice	CustomerID	Country
4	536365	84029E	RED WOOLLY HOTTIE WHITE HEART.	6	2010-12-01 08:26:00	3.39	17850.0	United Kingdom

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

```
[9] def encode_units(x):
    if x <= 0:
        return 0
    else:
        return x
```

Setelah pembersihan, sehingga perlu menggabungkan item menjadi 1 transaksi per baris dengan setiap produk 1 hot encoded. Demi menjaga kumpulan data tetap kecil, cara yang

dilakukan hanya melihat penjualan untuk Perancis. Namun pada kode tambahan di bawah ini, saya akan membandingkan hasil tersebut dengan penjualan dari Jerman. Perbandingan negara lebih lanjut akan menarik untuk diselidiki.

```

[8] /usr/local/lib/python3.10/dist-packages/ipykernel/ipkernel.py:283: DeprecationWarning: `should_run_async` will not call `transform_cell` automatically in the future and should_run_async(code)

[9] 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)

/usr/local/lib/python3.10/dist-packages/ipykernel/ipkernel.py:283: DeprecationWarning: `should_run_async` will not call `transform_cell` automatically in the future and should_run_async(code)

[10] frequent_itemsets = apriori(basket_sets, min_support=0.07, use_colnames=True)

/usr/local/lib/python3.10/dist-packages/ipykernel/ipkernel.py:283: DeprecationWarning: `should_run_async` will not call `transform_cell` automatically in the future and should_run_async(code)
/usr/local/lib/python3.10/dist-packages/mlxtend/frequent_patterns/fpcommon.py:110: DeprecationWarning: DataFrames with non-bool types result in worse computation: warnings.warn(
  
```

Ada banyak angka nol dalam data tetapi kita juga perlu memastikan setiap nilai positif diubah menjadi 1 dan nilai apa pun yang kurang dari 0 disetel ke 0. Langkah ini akan menyelesaikan pengkodean data yang paling menarik dan menghapus kolom ongkos kirim (karena biaya tersebut bukanlah biaya yang ingin kami jelajahi)

Dan rumus yang kedua yaitu setelah data terstruktur dengan benar, kita dapat menghasilkan kumpulan item frekuensi yang memiliki dukungan minimal 7% (angka ini dipilih sehingga saya bisa mendapatkan cukup banyak contoh berguna)

```

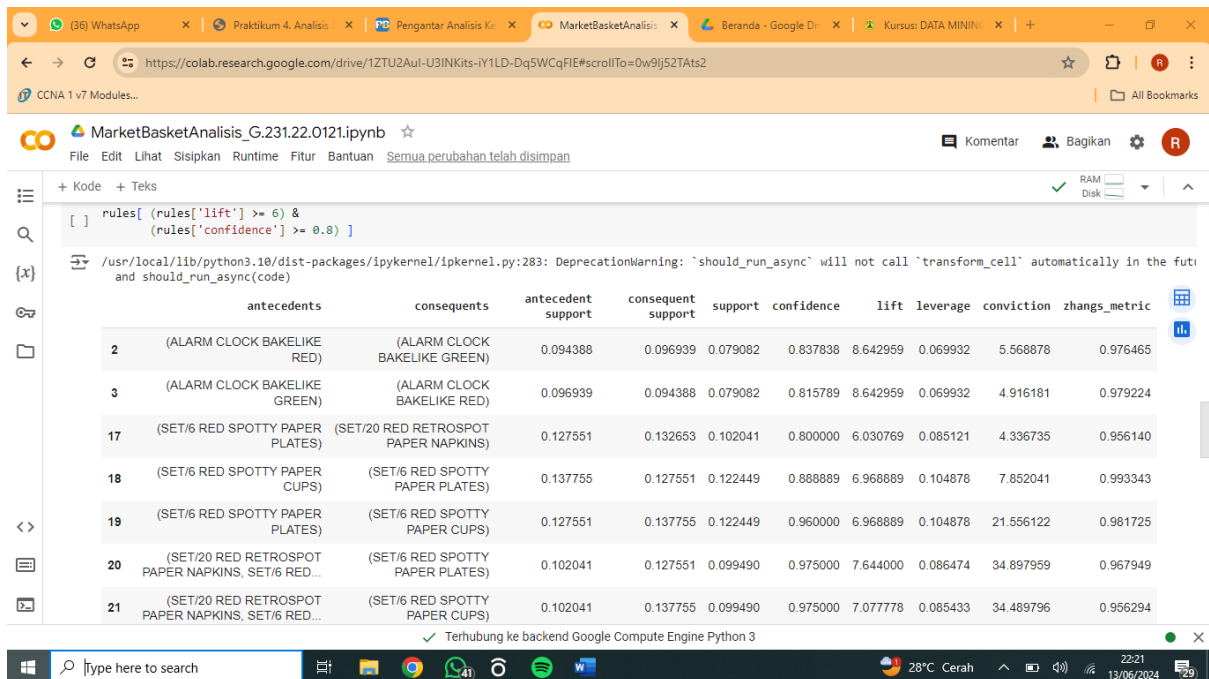
rules = association_rules(frequent_itemsets, metric="lift", min_threshold=1)
rules.head()

/usr/local/lib/python3.10/dist-packages/ipykernel/ipkernel.py:283: DeprecationWarning: `should_run_async` will not call `transform_cell` automatically in the future and should_run_async(code)
  
```

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction	zhangs_metric
0	(ALARM CLOCK BAKELIKE PINK)	(ALARM CLOCK BAKELIKE GREEN)	0.102041	0.096939	0.073980	0.725000	7.478947	0.064088	3.283859	0.964734
1	(ALARM CLOCK BAKELIKE GREEN)	(ALARM CLOCK BAKELIKE PINK)	0.096939	0.102041	0.073980	0.763158	7.478947	0.064088	3.791383	0.959283
2	(ALARM CLOCK BAKELIKE RED)	(ALARM CLOCK BAKELIKE GREEN)	0.094388	0.096939	0.079082	0.837838	8.642959	0.069932	5.568878	0.976465
3	(ALARM CLOCK BAKELIKE GREEN)	(ALARM CLOCK BAKELIKE RED)	0.096939	0.094388	0.079082	0.815789	8.642959	0.069932	4.916181	0.979224
4	(ALARM CLOCK BAKELIKE RED)	(ALARM CLOCK BAKELIKE PINK)	0.094388	0.102041	0.073980	0.783784	7.681081	0.064348	4.153061	0.960466

Langkah berikutnya: [Lihat plot yang direkomendasikan](#)

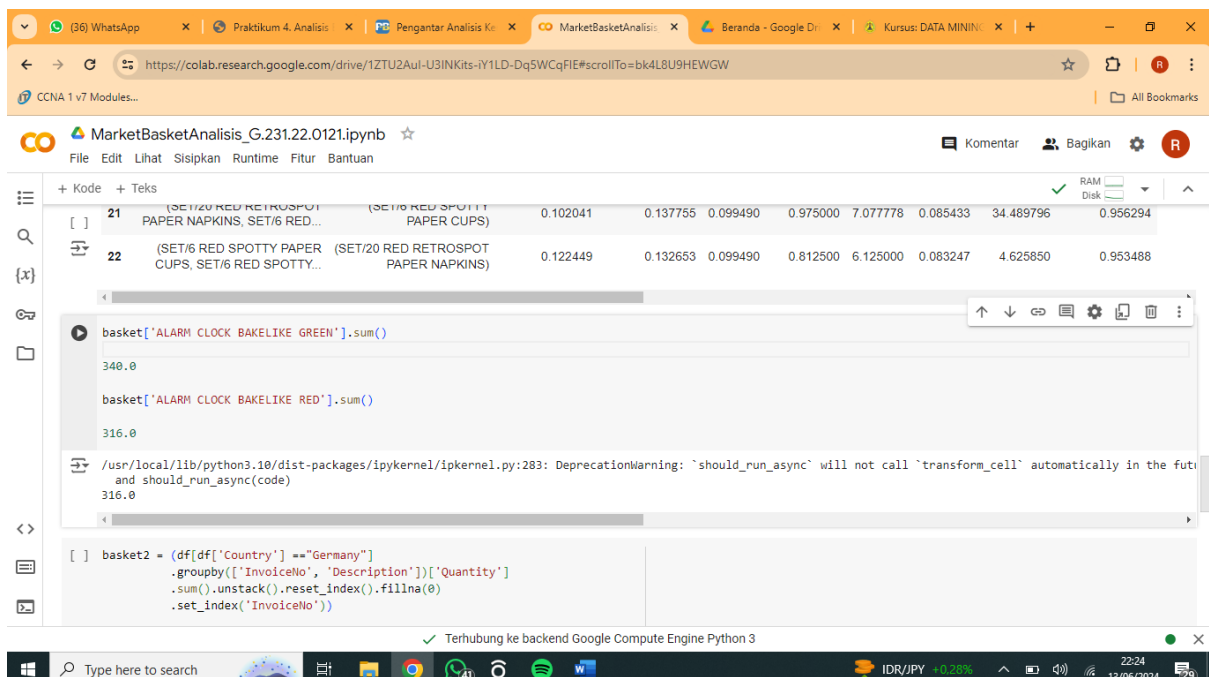
Berikut kode diatas langkah terakhir adalah menghasilkan aturan dengan dukungan, keyakinan, dan dorongan yang sesuai. Bangun item yang sering digunakan **apriori** lalu buat aturannya dengan **association_rules**.



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

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction	zhangs_metric
2	(ALARM CLOCK BAKELIKE RED)	(ALARM CLOCK BAKELIKE GREEN)	0.094388	0.096939	0.079082	0.837838	8.642959	0.069932	5.568878	0.976465
3	(ALARM CLOCK BAKELIKE GREEN)	(ALARM CLOCK BAKELIKE RED)	0.096939	0.094388	0.079082	0.815789	8.642959	0.069932	4.916181	0.979224
17	(SET/6 RED SPOTTY PAPER PLATES)	(SET/20 RED RETROSPOT PAPER NAPKINS)	0.127551	0.132653	0.102041	0.800000	6.030769	0.085121	4.336735	0.956140
18	(SET/6 RED SPOTTY PAPER CUPS)	(SET/6 RED SPOTTY PAPER PLATES)	0.137755	0.127551	0.122449	0.888889	6.968889	0.104878	7.852041	0.993343
19	(SET/6 RED SPOTTY PAPER PLATES)	(SET/6 RED SPOTTY PAPER CUPS)	0.127551	0.137755	0.122449	0.960000	6.968889	0.104878	21.556122	0.981725
20	(SET/20 RED RETROSPOT PAPER NAPKINS, SET/6 RED SPOTTY PAPER CUPS)	(SET/6 RED SPOTTY PAPER PLATES)	0.102041	0.127551	0.099490	0.975000	7.644000	0.086474	34.897959	0.967949
21	(SET/20 RED RETROSPOT PAPER NAPKINS, SET/6 RED SPOTTY PAPER CUPS)	(SET/6 RED SPOTTY PAPER CUPS)	0.102041	0.137755	0.099490	0.975000	7.077778	0.085433	34.489796	0.956294

Dengan menggunakan kode diatas kita dapat memfilter kerangka data menggunakan kode pandas standar. Dalam hal ini, carilah peningkatan yang besar (6) dan keyakinan yang tinggi (0,8)



```
basket['ALARM CLOCK BAKELIKE GREEN'].sum()
340.0

basket['ALARM CLOCK BAKELIKE RED'].sum()
316.0
```

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

Pada kode diatas, dapat melihat seberapa besar peluang yang ada untuk menggunakan popularitas satu produk untuk mendorong penjualan produk lain. Misalnya, kita dapat melihat bahwa kita menjual 340 Jam Alarm Hijau tetapi hanya 316 Jam Alarm Merah, jadi mungkin kita dapat mendorong lebih banyak penjualan Jam Alarm Merah melalui rekomendasi?

The screenshot shows a Google Colab notebook interface. The top bar includes the Google Colab logo and the notebook title "MarketBasketAnalysis_G.231.22.0121.ipynb". The left sidebar contains icons for code, text, and other notebook elements. The main area displays the following Python code:

```

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)]

```

The output of the code is a table of association rules. The table has the following columns: antecedents, consequents, antecedent support, consequent support, support, confidence, lift, leverage, conviction, and zhangs_metric. The table lists two rules:

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction	zhangs_metric
1	(PLASTERS IN TIN CIRCUS PARADE)	(PLASTERS IN TIN WOODLAND ANIMALS)	0.115974	0.137856	0.067834	0.584906	4.242887	0.051846	2.076984	0.864580
7	(PLASTERS IN TIN SPACEBOY)	(PLASTERS IN TIN WOODLAND ANIMALS)	0.107221	0.137856	0.061269	0.571429	4.145125	0.046488	2.011670	0.849877

The bottom of the notebook shows the status bar with the text "Terhubung ke backend Google Compute Engine Python 3".

Berikut dapat dilihat hasil dari MLxtend yang menggunakan analisis dasar excel sehingga tidak perlu repot memasukkan file ke dalam colab. Jika Anda tidak memiliki akses ke MLxtend dan analisis asosiasi ini, akan sangat sulit menemukan pola ini menggunakan analisis dasar Excel. Dengan python dan MLxtend, proses analisisnya relatif mudah dan karena Anda menggunakan python, Anda memiliki akses ke semua teknik visualisasi tambahan dan alat analisis data di ekosistem python.