Kelompok 6

1. Kyla Belva Queena (164221015)
2. Jihan Ashifa Hakim (164221016)
3. Moh. Okka Omarrosi (164221105)
4. M. Arief Mulyawan (164221074)
5. M. Fahd Ali Hillaby (164221067)

Dataset

Groceries Dataset https://www.kaggle.com/datasets/heeraldedhia/groceries-dataset

Penjelasan :

data tentang pembelian produk kebutuhan sehari-hari, seperti makanan ringan, minuman, dan barang-barang rumah tangga lainnya. Data ini terdiri dari 3 kolom dan memiliki total data 38.765.

Import Library

```
import numpy as np
import pandas as pd
import plotly.graph_objects as go
import plotly.express as px
```

```
try:
    import apyori
except:
    !pip install apyori

from apyori import apriori
```

```
Collecting apyori
  Downloading apyori-1.1.2.tar.gz (8.6 kB)
  Preparing metadata (setup.py) ... done
Building wheels for collected packages: apyori
  Building wheel for apyori (setup.py) ... done
  Created wheel for apyori: filename=apyori-1.1.2-py3-none-any.whl size=5955 sha256=e37790a6bd3e11a7dcba455f1d8b6fcfa472ed06c1de082
  Stored in directory: /root/.cache/pip/wheels/c4/1a/79/20f55c470a50bb3702a8cb7c94d8ada15573538c7f4baebe2d
Successfully built apyori
Installing collected packages: apyori
Successfully installed apyori-1.1.2
```

Penjelasan :

import numpy as np:

Mengimpor library NumPy dengan alias np, yang sering digunakan untuk komputasi numerik dalam Python. Alias np digunakan untuk memanggil fungsi dan objek dari NumPy.

import pandas as pd:

Mengimpor library Pandas dengan alias pd, yang digunakan untuk manipulasi dan analisis data dalam Python. Alias pd digunakan untuk memanggil fungsi dan objek dari Pandas.

import plotly.graph_objects as go:

Mengimpor modul graph_objects dari library Plotly dengan alias go. Plotly adalah library yang digunakan untuk membuat visualisasi data interaktif di Python.

import plotly.express as px:

Mengimpor modul express dari library Plotly dengan alias px. Plotly Express adalah antarmuka tingkat tinggi yang memudahkan pembuatan visualisasi data dengan Plotly.

try: import apyori except: !pip install apyori:

Mencoba mengimpor modul apyori dari library apyori. Jika modul tersebut tidak tersedia, maka perintah pip install apyori akan dijalankan untuk menginstalnya. Library apyori digunakan untuk menerapkan algoritma Apriori, yang digunakan dalam analisis asosiasi pada data.

from apyori import apriori:

Mengimpor fungsi apriori dari modul apyori. Fungsi ini digunakan untuk menjalankan algoritma Apriori pada data

Import Dataset

```
df = pd.read_csv('/content/Groceries_dataset.csv', parse_dates=['Date'])
df.head()
```

```
<ipython-input-2-6505c9727d62>:1: UserWarning: Parsing dates in %d-%m-%Y format when
  df = pd.read_csv('/content/Groceries_dataset.csv', parse_dates=['Date'])
```

	Member_number	Date	itemDescription
0	1808	2015-07-21	tropical fruit
1	2552	2015-01-05	whole milk
2	2300	2015-09-19	pip fruit
3	1187	2015-12-12	other vegetables
4	3037	2015-02-01	whole milk

Next steps: View recommended plots

Penjelasan :

Mengimpor dataset dan membaca file csv. parse_dates=['Date'] untuk memberitahu Pandas agar mengubah kolom 'Date' menjadi tipe data tanggal (datetime) saat membaca data. Ini memungkinkan kita untuk melakukan operasi-operasi terkait tanggal dengan mudah di DataFrame.

```
df.isnull().any()
```

```
Member_number      False
Date               False
itemDescription    False
dtype: bool
```

Penjelasan :

isnull() untuk mengecek keberadaan nilai-nilai null (kosong) di setiap kolom DataFrame. Kemudian, metode any() digunakan untuk mengecek apakah ada nilai null di setiap kolom. Jika nilai adalah True, artinya ada nilai null di kolom tersebut. Jika nilainya False, artinya tidak ada nilai null di kolom tersebut. Disini, hasilnya menunjukkan bahwa tidak ada nilai null dalam setiap kolom DataFrame df, karena semua nilai adalah False. Artinya, setiap kolom memiliki data yang lengkap tanpa nilai null.

```
all_products = df['itemDescription'].unique()
print("Total products: {}".format(len(all_products)))
```

```
Total products: 167
```

Penjelasan :

Mengambil kolom 'itemDescription' dari DataFrame (df) untuk menghitung daftar unik dari semua produk yang terdapat dalam kolom tersebut. Didapatkan jumlah total products unik sebanyak 167. Jadi, kode tersebut secara efektif menghitung total produk yang terdaftar dalam dataset.

Top 10 frequently sold products

```
def ditribution_plot(x,y,name=None,xaxis=None,yaxis=None):
    fig = go.Figure([
        go.Bar(x=x, y=y)
    ])

    fig.update_layout(
        title_text=name,
        xaxis_title=xaxis,
        yaxis_title=yaxis
    )
    fig.show()
```

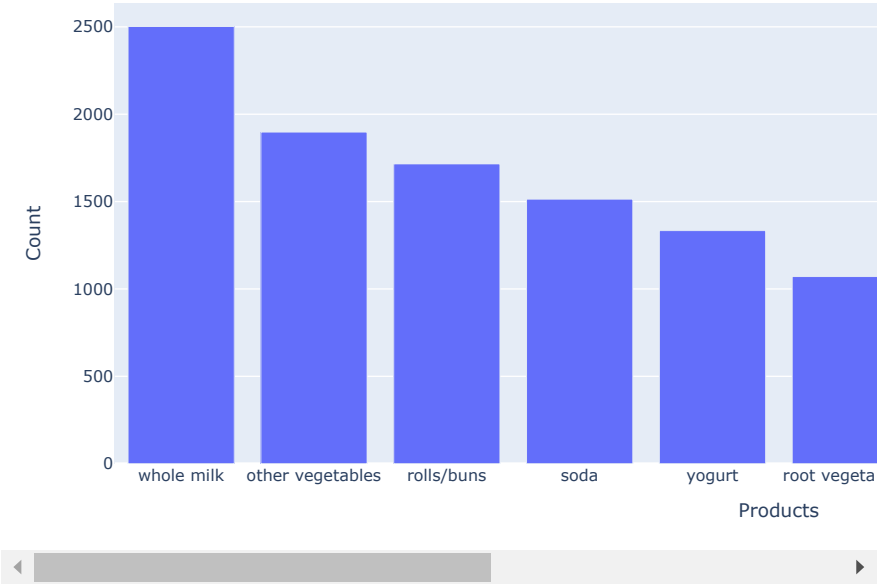
Penjelasan :

Membuat fungsi "distribution_plot" untuk membuat plot distribusi dengan menggunakan bar plot dari library plotly. Fungsi ini menerima empat parameter, yaitu x (data untuk sumbu-x), y (data untuk sumbu-y), name (judul plot), xaxis (judul sumbu-x), dan yaxis (judul sumbu-y). Fungsi ini

menghasilkan plot dengan bar plot berdasarkan data yang diberikan dan menambahkan judul untuk plot serta label sumbu-x dan sumbu-y.

```
x = df['itemDescription'].value_counts()
x = x.sort_values(ascending = False)
x = x[:10]

ditribution_plot(x=x.index, y=x.values, yaxis="Count", xaxis="Products")
```



Penjelasan :

Menghasilkan plot distribusi dengan menggunakan fungsi "distribution_plot" yang sebelumnya didefinisikan untuk menampilkan sepuluh produk teratas yang paling sering muncul dalam kolom 'itemDescription' dari DataFrame (df). Pertama, code menghitung jumlah kemunculan setiap produk dan mengurutkannya secara menurun. Kemudian, code membatasi data hanya untuk sepuluh produk teratas. Setelah itu, code menggunakan fungsi "distribution_plot" untuk membuat plot dengan sumbu-x berisi nama produk dan sumbu-y berisi jumlah kemunculan produk tersebut. Judul plot diberikan sebagai "Products", dan label sumbu-x dan sumbu-y ditambahkan sesuai dengan data yang diplot.

One-hot representation of products purchased

```
one_hot = pd.get_dummies(df['itemDescription'])
df.drop('itemDescription', inplace=True, axis=1)
df = df.join(one_hot)
df.head()
```

	Member_number	Date	Instant food products	UHT-milk	abrasive cleaner	artif. sweetener	baby cosmetics	bags	baking powder
0	1808	2015-07-21	False	False	False	False	False	False	False
1	2552	2015-01-05	False	False	False	False	False	False	False
2	2300	2015-09-19	False	False	False	False	False	False	False
3	1187	2015-12-12	False	False	False	False	False	False	False
4	3037	2015-02-01	False	False	False	False	False	False	False

5 rows × 169 columns

Penjelasan :

Melakukan proses one-hot encoding pada kolom 'itemDescription' dari DataFrame (df). Mengubah setiap nilai kategori menjadi vektor biner di mana setiap kategori direpresentasikan sebagai kolom baru dengan nilai biner. Setelah encoding, kolom 'itemDescription' dihapus dari

DataFrame asli, dan DataFrame diupdate dengan menambahkan kolom-kolom hasil encoding. Hasilnya adalah DataFrame baru dengan kolom-kolom baru yang mewakili setiap kategori produk sebagai vektor biner yang dapat digunakan untuk analisis lebih lanjut.

Transaction

```
records = df.groupby(["Member_number","Date"])[all_products[:]].apply(sum)
records = records.reset_index()[all_products]
```

```
records.head()
```

	tropical fruit	whole milk	pip fruit	other vegetables	rolls/buns	pot plants	citrus fruit	beef	frankfurter
0	0	1	0	0	0	0	0	0	0
1	0	1	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0

5 rows × 167 columns

Penjelasan :

Melakukan proses pengelompokan data berdasarkan nomor anggota dan tanggal pada DataFrame (df), kemudian menjumlahkan nilai-nilai produk untuk setiap kelompok tersebut.

```
## Replacing non-zero values with product names
def get_Pnames(x):
    for product in all_products:
        if x[product] > 0:
            x[product] = product
    return x

records = records.apply(get_Pnames, axis=1)
records.head()
```

	tropical fruit	whole milk	pip fruit	other vegetables	rolls/buns	pot plants	citrus fruit	beef	frankfurter
0	0	whole milk	0	0	0	0	0	0	0
1	0	whole milk	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0

5 rows × 167 columns

Penjelasan :

Membuat fungsi "get_Pnames" untuk menggantikan nilai non-nol dalam DataFrame "records" dengan nama produk yang sesuai. Fungsi ini melakukan iterasi melalui setiap produk dalam "all_products" dan jika nilai produk dalam baris tertentu lebih besar dari nol, maka nilai tersebut diganti dengan nama produk tersebut. Kemudian, fungsi diterapkan pada setiap baris DataFrame "records" menggunakan metode "apply" dengan sumbu 1 (baris). Output yang muncul adalah DataFrame baru yang telah dimodifikasi, dimana nilai non-nol telah digantikan dengan nama produk yang sesuai, sedangkan nilai-nilai nol tetap tidak berubah.

```
print("total transactions: {}".format(len(records)))
```

```
total transactions: 14963
```

Penjelasan :

Menampilkan jumlah transaksi yang tercatat dalam DataFrame "records" dengan menggunakan format string untuk menampilkan nilai yang sesuai. Didapatkan total transaksinya berjumlah 14963.

```
## Removing zeros
x = records.values
x = [sub[~(sub == 0)].tolist() for sub in x if sub[sub != 0].tolist()]
transactions = x
```

Penjelasan :

Menghapus nilai nol dari data transaksi. Pertama, data transaksi diubah menjadi format array menggunakan .values. Kemudian, dilakukan iterasi untuk setiap array transaksi. Pada setiap iterasi, nilai nol dihapus dari array transaksi menggunakan operator bitwise ~ untuk mendapatkan indeks di mana nilai tidak sama dengan nol, kemudian hasilnya diubah menjadi daftar menggunakan .tolist(). Langkah ini dilakukan untuk setiap subarray yang tidak sama dengan nol. Terakhir, data transaksi yang telah dimodifikasi tersebut disimpan dalam variabel transactions.

Example Transactions

```
transactions[0:10]
```

```
[['whole milk', 'pastry', 'salty snack'],
 ['whole milk', 'yogurt', 'sausage', 'semi-finished bread'],
 ['soda', 'pickled vegetables'],
 ['canned beer', 'misc. beverages'],
 ['sausage', 'hygiene articles'],
 ['whole milk', 'rolls/buns', 'sausage'],
 ['whole milk', 'soda'],
 ['frankfurter', 'soda', 'whipped/sour cream'],
 ['beef', 'white bread'],
 ['frankfurter', 'curd']]
```

Penjelasan :

"transactions[0:10]" digunakan untuk menampilkan sepuluh transaksi pertama dalam dataset. Setiap transaksi direpresentasikan sebagai daftar barang yang dibeli. Output yang muncul adalah daftar dari sepuluh transaksi pertama yang mencakup berbagai barang belanjaan seperti susu, roti, daging, minuman ringan, dan barang-barang lainnya.

Association Rules

Algoritma Apriori

```
rules = apriori(transactions,min_support=0.00030,min_confidance=0.05,min_lift=3,min_length=2,target="rules")
association_results = list(rules)
```

Penjelasan :

Menggunakan algoritma Apriori untuk mengekstraksi aturan asosiasi dari data transaksi yang diberikan. Disini kami menentukan parameter-parameter seperti support minimum, confidence minimum, lift minimum, dan panjang minimum, setelah itu algoritma akan mencari aturan asosiasi yang signifikan. Setelah prosesnya selesai, hasilnya disimpan dalam variabel "association_results" sebagai list 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: margarine -> root vegetables
Support: 0.0003341575887188398
Confidence: 0.01037344398340249
Lift: 3.1043568464730287
====================================
Rule: margarine -> yogurt
Support: 0.0004009891064626077 5
Confidence: 0.0066445182724252485
Lift: 3.106935215946843
====================================
Rule: pastry -> newspapers
Support: 0.0003341575887188398
Confidence: 0.006459948320413437
Lift: 3.580007656235047
====================================
Rule: tropical fruit -> yogurt
Support: 0.0003341575887188398
Confidence: 0.016501650165016504
Lift: 3.1655665566556657
====================================
Rule: other vegetables -> waffles
Support: 0.0003341575887188398
Confidence: 0.002736726874657909
Lift: 3.4124703521255246
====================================
Rule: yogurt -> other vegetables
Support: 0.0003341575887188398
Confidence: 0.03333333333333333
Lift: 4.122038567493113
====================================
Rule: yogurt -> sausage
Support: 0.0004009891064626077 5
Confidence: 0.004669260700389105
Lift: 3.037658602605312
====================================
Rule: pastry -> whole milk
Support: 0.0003341575887188398
Confidence: 0.006459948320413437
Lift: 5.685894512843897
====================================
Rule: yogurt -> whole milk
Support: 0.0003341575887188398
Confidence: 0.005537098560354374
Lift: 4.142580287929125
====================================
```

Penjelasan :

Melakukan iterasi melalui hasil aturan asosiasi yang diperoleh dari proses menggunakan algoritma Apriori. Setiap aturan asosiasi direpresentasikan sebagai item dalam variabel "association_results". Dalam setiap iterasi, code mengekstraksi item yang berpartisipasi dalam aturan, serta nilai support, confidence, dan liftnya. Kemudian informasi tersebut ditampilkan dengan format yang sesuai. Proses ini terus berulang untuk setiap aturan asosiasi yang ditemukan sehingga pengguna dapat melihat secara langsung detail-detail penting dari setiap aturan asosiasi yang dihasilkan.

FP-Growth

```
## Replacing non-zero values with one
def get_Pnames(x):
    for product in all_products:
        if x[product] != 0:
            x[product] = 1
    return x

records2 = records.apply(get_Pnames, axis=1)
records2.head()
```

	tropical fruit	whole milk	pip fruit	other vegetables	rolls/buns	pot plants	citrus fruit	beef	frankfurter
0	0	1	0	0	0	0	0	0	0
1	0	1	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0

5 rows × 167 columns

Penjelasan :

Menggunakan fungsi get_Pnames untuk mengubah setiap baris DataFrame yang menunjukkan transaksi sehingga nilai non-nol diganti dengan 1 dan menghasilkan representasi biner dari keberadaan produk dalam transaksi. Proses ini untuk membangun fondasi yang tepat untuk analisis berikutnya menggunakan algoritma FP-Growth dalam mengidentifikasi pola-pola frekuensi tinggi.

```
from mlxtend.frequent_patterns import fpgrowth

fpgrowth(records2, min_support=0.01)
```

```
/usr/local/lib/python3.10/dist-packages/mlxtend/frequent_patterns/fpcommon.py:110: De

DataFrames with non-bool types result in worse computationalperformance and their sup
```

	support	itemsets
0	0.157923	(1)
1	0.051728	(23)
2	0.018780	(79)
3	0.085879	(17)
4	0.060349	(18)
...	...	...
64	0.011161	(17, 1)
65	0.011629	(1, 37)
66	0.013968	(1, 4)
67	0.010559	(3, 4)
68	0.014837	(1, 3)

69 rows × 2 columns

Penjelasan :

Menggunakan fungsi "fpgrowth" (records2, min_support=0.01), dengan parameter min_support yang ditetapkan senilai 0.01 untuk menemukan pola frekuensi tinggi dalam dataset transaksional. Nilai ini menunjukkan jumlah support minimum yang diperlukan agar sebuah set item dianggap signifikan. Output yang keluar adalah kumpulan pola frekuensi tinggi yang ditemukan dalam data transaksional, seperti item transaksi yang menggambarkan hubungan antar produk atau kebiasaan belanja pelanggan.

```
fpgrowth(records2, min_support=0.01, 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. Please pass the result to `transformed_cell` argument

/usr/local/lib/python3.10/dist-packages/mlxtend/frequent_patterns/fpcommon.py:110: DeprecationWarning:

DataFrames with non-bool types result in worse computationalperformance and their support might be discontinued in the future.Please
```

	support	itemsets
0	0.157923	(whole milk)
1	0.051728	(pastry)
2	0.018780	(salty snack)
3	0.085879	(yogurt)
4	0.060349	(sausage)
...	...	...
64	0.011161	(whole milk, yogurt)
65	0.011629	(whole milk, soda)
66	0.013968	(whole milk, rolls/buns)
67	0.010559	(other vegetables, rolls/buns)
68	0.014837	(whole milk, other vegetables)

69 rows × 2 columns

Penjelasan :

Menjalankan algoritma FP-Growth untuk menemukan pola itemset yang sering muncul dalam dataset transaksi. Parameter min_support disini, kami buat sebesar 0.01, yang berarti itemset dengan dukungan (support) lebih rendah dari 0.01 akan diabaikan. Opsi use_colnames diatur ke True agar nama itemset ditampilkan dalam output. Hasilnya adalah tabel yang menampilkan itemset beserta dukungannya (support) dalam dataset. Contohnya, (whole milk) memiliki dukungan sebesar 0.157923, yang berarti sekitar 15.79% transaksi mengandung item "whole milk". Selain itu, itemset yang terdiri dari beberapa item, seperti (yogurt, whole milk) dengan dukungan sebesar 0.011161, yang berarti sekitar 1.12% transaksi mengandung kedua item tersebut bersama-sama.