



Market Basket Analysis Using Python 2

Analyzing Retail Transaction Data with Apriori
Algorithm to Generate Association Rules



PROFIL ANALYST

ROZALINDA TITALIA PUTRI



SURABAYA, JAWA TIMUR



LINDAROZA509@GMAIL.COM



ROZALINDA TITALIA PUTRI

SAYA ADALAH MAHASISWA SISTEM INFORMASI SEMESTER 4 YANG MEMILIKI SEMANGAT TINGGI DIBIDANG IT DAN DATA ANALYST. SEBAGAI CALON DATA ANALYST, SAYA SANGAT ANTUSIAS UNTUK MENERAPKAN KEMAMPUAN ANALISIS DAN PENGETAHUAN SAYA DALAM MENYELESAIKAN PERMASALAHAN TERUTAMA DIBIDANG DATA.



LATAR BELAKANG

Dalam data transaksi ritel, pelanggan sering membeli lebih dari satu produk dalam satu kali belanja, sehingga terbentuk pola pembelian yang tidak langsung terlihat dari data mentah. Oleh karena itu, proyek ini menggunakan Market Basket Analysis dengan Python untuk mengolah data transaksi agar dapat dianalisis dengan lebih mudah. Data diubah menggunakan one-hot encoding, lalu dianalisis dengan algoritma Apriori untuk menemukan produk-produk yang sering dibeli bersamaan. Hasil analisis ini membantu bisnis memahami hubungan antarproduk dan menyusun strategi penjualan yang lebih efektif, seperti cross-selling, bundling produk, dan penataan produk.

TUJUAN

- Menganalisis data transaksi ritel untuk mengetahui pola pembelian pelanggan.
- Mengidentifikasi produk-produk yang sering dibeli secara bersamaan menggunakan Market Basket Analysis.
- Menerapkan algoritma Apriori untuk menghasilkan frequent itemsets dan association rules.
- Memberikan insight yang dapat digunakan sebagai dasar strategi bisnis, seperti cross-selling, bundling produk, dan penataan produk yang lebih efektif.



TOOLS

Jupyter Notebook

1 Digunakan sebagai lingkungan utama untuk melakukan analisis data menggunakan Python, menulis kode secara terstruktur, serta menampilkan hasil analisis dan visualisasi dalam satu dokumen interaktif.

CSV File (Online Retail Dataset)

Berperan sebagai sumber data mentah yang berisi data transaksi pengguna, seperti ID pelanggan, tanggal transaksi, jumlah pembelian, dan nilai transaksi, yang menjadi dasar analisis user retention dan cohort.

TEKNOLOGI

1

Data Preparation & Cleaning

Melakukan pengolahan awal data transaksi menggunakan Python (pandas) dengan mengimpor data dari file CSV ke Jupyter Notebook, mengecek struktur data, membersihkan data kotor serta memastikan data transaksi siap digunakan untuk analisis

2

Transaction Encoding

Mengubah data transaksi ke dalam format basket menggunakan teknik one-hot encoding, sehingga setiap produk direpresentasikan sebagai nilai 1 (dibeli) atau 0 (tidak dibeli) dalam satu transaks



TEKNOLOGI

1

Market Basket Analysis & Association Rules

Menerapkan Market Basket Analysis menggunakan algoritma Apriori (mlxtend) untuk menemukan frequent itemsets, kemudian membentuk association rules berdasarkan nilai support, confidence, dan lift

2

Data Interpretation & Insight Generation

Menginterpretasikan hasil association rules untuk memahami pola pembelian pelanggan dan menghasilkan insight yang dapat dimanfaatkan dalam strategi cross-selling, upselling, bundling produk, serta mendukung pengambilan keputusan bisnis berbasis data





OVERVIEW

Data yang digunakan berasal dari Kaggle (Bukan data sesungguhnya). Dataset yang digunakan adalah Online Retail Dataset yang berisi data transaksi penjualan produk ritel. Dataset ini mencatat aktivitas pemesanan produk oleh pelanggan dalam periode waktu tertentu dan digunakan untuk keperluan analisis data.

- order_id : ID unik untuk setiap transaksi atau pesanan yang dilakukan pelanggan.
- product_code : Kode unik produk sebagai identitas setiap barang yang dijual.
- product_name : Nama atau deskripsi produk yang dibeli oleh pelanggan.
- quantity : Jumlah unit produk yang dibeli dalam satu transaksi. Nilai negatif menunjukkan adanya retur atau pembatalan pesanan.
- order_date : Tanggal dan waktu terjadinya transaksi pemesanan produk.
- price : Harga satuan produk pada saat transaksi dilakukan
- customer_id : ID unik pelanggan yang melakukan transaksi. Digunakan untuk mengidentifikasi dan menganalisis perilaku pelanggan.

TAHAPAN APLIKASI APRIORI ALGORITHM DI PYTHON

```
[8]: import pandas as pd  
import numpy as np  
import datetime as dt  
  
[ ]: import os  
os.getcwd()
```

IMPOR DATA TRANSAKSI KE DATA FRAME

Interpretation

- Kode Python ini digunakan untuk mengimpor data transaksi Online Retail dari file CSV ke dalam DataFrame menggunakan library pandas.
- import pandas as pd, import numpy as np, dan import datetime as dt berfungsi untuk memanggil library yang dibutuhkan dalam pengolahan data numerik dan tanggal.
- import os dan os.getcwd() digunakan untuk mengetahui direktori kerja saat ini sehingga lokasi file CSV dapat dipastikan dengan benar.



TAHAPAN APLIKASI APRIORI ALGORITHM DI PYTHON

```
[10]: df = pd.read_csv('Salinan Online Retail Data.csv', header=0)
df
```

	order_id	product_code	product_name	quantity	order_date	price	customer_id
0	493410	TEST001	This is a test product.	5	2010-01-04 09:24:00	4.50	12346.0
1	C493411	21539	RETRO SPOTS BUTTER DISH	-1	2010-01-04 09:43:00	4.25	14590.0
2	493412	TEST001	This is a test product.	5	2010-01-04 09:53:00	4.50	12346.0
3	493413	21724	PANDA AND BUNNIES STICKER SHEET	1	2010-01-04 09:54:00	0.85	NaN
4	493413	84578	ELEPHANT TOY WITH BLUE T-SHIRT	1	2010-01-04 09:54:00	3.75	NaN
...
461768	539991	21618	4 WILDFLOWER BOTANICAL CANDLES	1	2010-12-23 16:49:00	1.25	NaN
461769	539991	72741	GRAND CHOCOLATECANDLE	4	2010-12-23 16:49:00	1.45	NaN
461770	539992	21470	FLOWER VINE RAFFIA FOOD COVER	1	2010-12-23 17:41:00	3.75	NaN
461771	539992	22258	FELT FARM ANIMAL RABBIT	1	2010-12-23 17:41:00	1.25	NaN
461772	539992	21155	RED RETROSPOT PEG BAG	1	2010-12-23 17:41:00	2.10	NaN

461773 rows × 7 columns

IMPOR DATA TRANSAKSI KE DATA FRAME

Interpretation

- pd.read_csv('Salinan Online Retail Data.csv', header=0) berfungsi untuk membaca file CSV dan menyimpannya ke dalam DataFrame df, dengan baris pertama sebagai header kolom.
- Perintah df digunakan untuk menampilkan isi DataFrame guna melihat struktur dan contoh data.
- Output menunjukkan dataset terdiri dari 461.773 baris dan 7 kolom, yaitu order_id, product_code, product_name, quantity, order_date, price, dan customer_id.



TAHAPAN APLIKASI APRIORI ALGORITHM DI PYTHON

```
[12]: df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 461773 entries, 0 to 461772
Data columns (total 7 columns):
 #   Column      Non-Null Count  Dtype  
--- 
 0   order_id    461773 non-null   object 
 1   product_code 461773 non-null   object 
 2   product_name 459055 non-null   object 
 3   quantity     461773 non-null   int64  
 4   order_date   461773 non-null   object 
 5   price        461773 non-null   float64
 6   customer_id  360853 non-null   float64
dtypes: float64(2), int64(1), object(4)
memory usage: 24.7+ MB
```

IMPOR DATA TRANSAKSI KE DATA FRAME

Interpretation

- Kode df.info() digunakan untuk menampilkan ringkasan struktur DataFrame. Output menunjukkan bahwa DataFrame memiliki 461.773 baris dan 7 kolom.
- Kolom order_id, product_code, quantity, order_date, dan price terisi lengkap tanpa nilai kosong. Kolom product_name memiliki beberapa data kosong, sedangkan customer_id memiliki cukup banyak missing value.
- Dari sisi tipe data, terdapat 4 kolom bertipe object, 1 kolom int64, dan 2 kolom float64. Total penggunaan memori DataFrame sekitar 24,7 MB, yang masih efisien untuk proses analisis data.

TAHAPAN APLIKASI APRIORI ALGORITHM DI PYTHON

```
df_clean = df.copy()
# membuat kolom date
df_clean['date'] = pd.to_datetime(
    df_clean['order_date'], errors='coerce')
    .astype('datetime64[ns]')
# menghapus semua baris tanpa customer_id
df_clean = df_clean[~df_clean['customer_id'].isna()]
# mengonversi customer_id menjadi string
df_clean['customer_id'] = df_clean['customer_id'].astype(str)
# menghapus semua baris tanpa product_name
df_clean = df_clean[~df_clean['product_name'].isna()]
# membuat semua product_name berhuruf kecil
df_clean['product_name'] = df_clean['product_name'].str.lower()
```

```
# menghapus semua baris dengan product_code atau product_name test
df_clean = df_clean[(~df_clean['product_code'].str.lower().str.contains('test')) |
                     (~df_clean['product_name'].str.contains('test '))]
# menghapus baris dengan status cancelled, yaitu yang order_id-nya diawali 'C'
df_clean = df_clean[df_clean['order_id'].str[:1]!='C']
# mengubah nilai quantity yang negatif menjadi positif
# karena nilai negatif tersebut hanya menandakan order tersebut cancelled
df_clean['quantity'] = df_clean['quantity'].abs()
# menghapus baris dengan price bernilai negatif
df_clean = df_clean[df_clean['price']>0]
# membuat nilai amount, yaitu perkalian antara quantity dan price
df_clean['amount'] = df_clean['quantity'] * df_clean['price']
```

TAHAPAN APLIKASI APRIORI ALGORITHM DI PYTHON

```
# mengganti product_name dari product_code yang memiliki  
#beberapa product_name dengan salah satu product_name-nya yang paling sering muncul  
most_freq_product_name = df_clean.groupby(['product_code','product_name'],  
as_index=False).agg(order_cnt=('order_id','nunique')).sort_values(['product_code','order_cnt'],  
ascending=[True,False])most_freq_product_name['rank'] = most_freq_product_name.groupby('product_code')  
['order_cnt'].rank(method='first', ascending=False)  
most_freq_product_name = most_freq_product_name  
[most_freq_product_name['rank']==1].drop(columns=['order_cnt','rank'])  
df_clean = df_clean.merge(most_freq_product_name.rename(columns=  
{'product_name':'most_freq_product_name'}), how='left', on='product_code')  
df_clean['product_name'] = df_clean['most_freq_product_name']  
df_clean = df_clean.drop(columns='most_freq_product_name')  
# menghapus outlier  
from scipy import stats  
df_clean = df_clean[(np.abs(stats.zscore(df_clean[['quantity','amount']]))<3).all(axis=1)]  
df_clean = df_clean.reset_index(drop=True)  
df_clean
```

LAKUKAN DATA CLEANSING, HAPUS BARIS DGN STATUS CANCELLED

	order_id	product_code	product_name	quantity	order_date	price	customer_id	date	amount
0	493414	21844	red retrospot mug	36	2010-01-04 10:28:00	2.55	14590.0	2010-01-04 10:28:00	91.80
1	493414	21533	retro spot large milk jug	12	2010-01-04 10:28:00	4.25	14590.0	2010-01-04 10:28:00	51.00
2	493414	37508	new england ceramic cake server	2	2010-01-04 10:28:00	2.55	14590.0	2010-01-04 10:28:00	5.10
3	493414	35001G	hand open shape gold	2	2010-01-04 10:28:00	4.25	14590.0	2010-01-04 10:28:00	8.50
4	493414	21527	red retrospot traditional teapot	12	2010-01-04 10:28:00	6.95	14590.0	2010-01-04 10:28:00	83.40
...
350087	539988	84380	set of 3 butterfly cookie cutters	1	2010-12-23 16:06:00	1.25	18116.0	2010-12-23 16:06:00	1.25
350088	539988	84849D	hot baths soap holder	1	2010-12-23 16:06:00	1.69	18116.0	2010-12-23 16:06:00	1.69
350089	539988	84849B	fairy soap soap holder	1	2010-12-23 16:06:00	1.69	18116.0	2010-12-23 16:06:00	1.69
350090	539988	22854	cream sweetheart egg holder	2	2010-12-23 16:06:00	4.95	18116.0	2010-12-23 16:06:00	9.90
350091	539988	47559B	tea time oven glove	2	2010-12-23 16:06:00	1.25	18116.0	2010-12-23 16:06:00	2.50



TAHAPAN APLIKASI APRIORI ALGORITHM DI PYTHON

```
df_clean.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 350092 entries, 0 to 350091
Data columns (total 9 columns):
 #   Column      Non-Null Count  Dtype  
---  --  
 0   order_id    350092 non-null   object 
 1   product_code 350092 non-null   object 
 2   product_name 350092 non-null   object 
 3   quantity     350092 non-null   int64  
 4   order_date   350092 non-null   object 
 5   price        350092 non-null   float64
 6   customer_id 350092 non-null   object 
 7   date         350092 non-null   datetime64[ns]
 8   amount       350092 non-null   float64
dtypes: datetime64[ns](1), float64(2), int64(1), object(5)
memory usage: 24.0+ MB
```

LAKUKAN DATA CLEANSING, HAPUS BARIS DGN
STATUS CANCELLED

Interpretation

- DataFrame hasil pembersihan data memiliki 358.469 baris dan 9 kolom, seluruhnya tanpa missing value.
- Struktur data telah diperbaiki dengan kolom date bertipe datetime64 untuk analisis berbasis waktu.
- Kolom numerik terdiri dari quantity (int64) serta price dan amount (float64) untuk analisis kuantitatif.
- Kolom kategorikal seperti order_id, product_code, product_name, customer_id, bertipe object.
- Total penggunaan memori sebesar 24.0+ MB, masih tergolong efisien untuk analisis lanjutan.

TAHAPAN APLIKASI APRIORI ALGORITHM DI PYTHON

```
basket = pd.pivot_table(df_clean, index='order_id', columns='product_name', values='product_code',
aggfunc='nunique', fill_value=0)
basket
```

product_name	10 colour spaceboy pen	12 ass zinc christmas decorations	12 coloured party balloons	12 daisy pegs in wood box	12 egg house painted wood	12 ivory rose peg place settings	12 message cards with envelopes	12 mini toadstool pegs	12 pencil small tube woodland	12 pencils small tube posy	... zinc heart lattice ...	1 ... charger ch
order_id												
493414	0	0	0	0	0	0	0	0	0	0	...	0
493427	0	0	0	0	0	0	0	0	0	0	...	0
493428	0	0	0	0	0	0	0	0	0	0	...	0
493432	0	0	0	0	0	0	0	0	0	0	...	0
493433	0	0	0	0	0	0	0	0	0	0	...	0
...
539981	0	0	0	0	0	0	0	0	0	0	...	0
539982	0	0	0	0	0	0	0	0	0	0	...	0
539985	0	0	0	0	0	0	0	0	0	0	...	0
539987	0	0	0	0	0	0	0	0	0	0	...	0
539988	0	0	0	0	0	0	0	0	0	0	...	0

16272 rows × 3842 columns

BUAT DATAFRAME BASKET

Interpretation

- Data transaksi diubah ke bentuk pivot table dengan order_id sebagai baris dan product_name sebagai kolom.
- Setiap baris merepresentasikan satu transaksi, sedangkan kolom menunjukkan produk yang dibeli atau tidak dibeli (1 = dibeli, 0 = tidak).
- Penggunaan nunique memastikan setiap produk hanya dihitung satu kali dalam setiap transaksi, dan nilai kosong diisi dengan 0.
- Hasilnya adalah basket matrix berukuran 16.272 transaksi × 3.842 produk yang siap digunakan untuk analisis Market Basket



TAHAPAN APLIKASI APRIORI ALGORITHM DI PYTHON

```
def encode(x):
    if x==0:
        return False
    if x>0:
        return True

basket_encode = basket.applymap(encode)
basket_encode
```

ENCODE DATAFRAME BASKET DENGAN NILAI TRUE UNTUK SEMUA NILAI DI ATAS 0 DAN FALSE UNTUK SEMUA NILAI 0

Interpretation

- Fungsi encode() digunakan untuk mengubah nilai numerik menjadi nilai boolean.
- Nilai 0 diubah menjadi False (produk tidak dibeli), sedangkan nilai > 0 diubah menjadi True (produk dibeli).
- Fungsi applymap() menerapkan proses encoding ke seluruh elemen pada tabel basket.
- Hasilnya adalah basket matrix berbentuk boolean yang siap digunakan untuk analisis pola pembelian produk

TAHAPAN APLIKASI APRIORI ALGORITHM DI PYTHON

product_name	10 colour spaceboy pen	12 ass zinc christmas decorations	12 coloured party balloons	12 daisy pegs in wood box	12 egg house painted wood	12 ivory rose peg place settings	12 message cards with envelopes	12 mini toadstool pegs	12 pencil small tube woodland
order_id									
493414	False	False	False	False	False	False	False	False	False
493427	False	False	False	False	False	False	False	False	False
493428	False	False	False	False	False	False	False	False	False
493432	False	False	False	False	False	False	False	False	False
493433	False	False	False	False	False	False	False	False	False
...
539981	False	False	False	False	False	False	False	False	False
539982	False	False	False	False	False	False	False	False	False
539985	False	False	False	False	False	False	False	False	False
539987	False	False	False	False	False	False	False	False	False
539988	False	False	False	False	False	False	False	False	False

ENCODE DATAFRAME BASKET DENGAN NILAI TRUE UNTUK SEMUA NILAI DI ATAS 0 DAN FALSE UNTUK SEMUA NILAI 0

Interpretation

- Setiap baris merepresentasikan satu order_id, sedangkan setiap kolom menunjukkan produk yang tersedia.
- Nilai True menandakan produk tersebut dibeli dalam transaksi, sedangkan False menandakan tidak dibeli.
- Seluruh nilai telah berhasil dikonversi ke format boolean, sehingga memudahkan proses analisis pola pembelian.
- Tabel ini merupakan hasil akhir basket encoding yang siap digunakan untuk mengidentifikasi hubungan antarproduk dalam Market Basket Analysis.

TAHAPAN APLIKASI APRIORI ALGORITHM DI PYTHON

```
basket_filter = basket_encode[(basket_encode>0).sum(axis=1)>1]
basket_filter
```

product_name	10 colour spaceboy pen	12 ass zinc christmas decorations	12 coloured party balloons	12 daisy pegs in wood box	12 egg house painted wood	12 ivory rose peg place settings	12 message cards with envelopes	12 mini toadstool pegs	12 pencil small tube woodland	12 pencils small tube posy
order_id										
493414	False	False	False	False	False	False	False	False	False	False
493427	False	False	False	False	False	False	False	False	False	False
493428	False	False	False	False	False	False	False	False	False	False
493432	False	False	False	False	False	False	False	False	False	False
493433	False	False	False	False	False	False	False	False	False	False
...
539978	False	False	False	False	False	False	False	False	False	False
539981	False	False	False	False	False	False	False	False	False	False
539982	False	False	False	False	False	False	False	False	False	False
539985	False	False	False	False	False	False	False	False	False	False
539988	False	False	False	False	False	False	False	False	False	False

AMBIL TRANSAKSI DENGAN BANYAKNYA PRODUK UNIK
LEBIH DARI 1 SAJA

Interpretation

- Data transaksi diubah menjadi tabel “ya/tidak” untuk setiap produk di tiap order (True = produk dibeli, False = tidak dibeli).
- Kode menghitung berapa banyak jenis produk yang dibeli dalam setiap order.
- Hanya order yang membeli lebih dari 1 produk yang dipilih (order dengan 1 produk saja dibuang).
- Hasilnya adalah daftar order yang berisi kombinasi beberapa produk, siap dipakai untuk analisis pola belanja (misalnya produk apa yang sering dibeli bersamaan).

TAHAPAN APLIKASI APRIORI ALGORITHM DI PYTHON

```
%pip install mlxtend
from mlxtend.frequent_patterns import fpgrowth
frequent_itemset = (
    fpgrowth(basket_filter, min_support=0.01, use_colnames=True)
    .sort_values('support', ascending=False)
    .reset_index(drop=True)
)
frequent_itemset['product_cnt'] = frequent_itemset['itemsets'].apply(len)
frequent_itemset
```

BUAT LIST FREQUENT ITEMSET (KUMPULAN PRODUK YANG SERING DIBELI)

Interpretation

- Kode menggunakan library mlxtend untuk melakukan analisis Market Basket Analysis.
- Algoritma FP-Growth diterapkan pada data transaksi yang sudah berbentuk basket (basket_filter).
- Parameter min_support = 0.01 berarti hanya kombinasi produk yang muncul minimal di 1% transaksi yang akan ditampilkan.
- Hasil frequent itemset diurutkan berdasarkan nilai support dari yang paling tinggi ke paling rendah.
- Kolom product_cnt ditambahkan untuk menunjukkan jumlah produk dalam setiap kombinasi itemset.

TAHAPAN APLIKASI APRIORI ALGORITHM DI PYTHON

	support	itemsets	product_cnt
0	0.177953	(white hanging heart t-light holder)	1
1	0.099440	(regency cakestand 3 tier)	1
2	0.096841	(jumbo bag red retrospot)	1
3	0.079912	(pack of 72 retro spot cake cases)	1
4	0.078512	(assorted colour bird ornament)	1
...
1189	0.010064	(lunch bag spaceboy design, childrens apron sp...	2
1190	0.010064	(jumbo storage bag suki, lunch bag red spotty,...	3
1191	0.010064	(lunch bag black skull, lunch bag red spotty...	4
1192	0.010064	(jumbo bag red retrospot, lunch bag suki desi...	3
1193	0.010064	(banquet birthday card)	1

1194 rows × 3 columns

BUAT LIST FREQUENT ITEMSET (KUMPULAN PRODUK YANG SERING DIBELI)

Interpretation

- Setiap baris merepresentasikan satu frequent itemset (produk tunggal atau kombinasi produk).
- Kolom support menunjukkan seberapa sering produk atau kombinasi produk tersebut muncul dalam seluruh transaksi.
- Kolom itemsets berisi nama produk atau kumpulan produk yang sering dibeli bersamaan.
- Kolom product_cnt menunjukkan jumlah produk dalam satu itemset (1 = produk tunggal, >1 = kombinasi produk).
- Itemset dengan nilai support lebih tinggi menandakan produk tersebut lebih populer atau lebih sering dibeli.

TAHAPAN APLIKASI APRIORI ALGORITHM DI PYTHON

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	representativity	leverage	conviction
0	frozenset({red hanging heart t-light holder})	frozenset({white hanging heart t-light holder})	0.058784	0.177953	0.042455	0.722222	4.058510	1.0	0.031995	2.959371
1	frozenset({sweetheart ceramic trinket box})	frozenset({strawberry ceramic trinket box})	0.049254	0.074980	0.037457	0.760487	10.142533	1.0	0.033764	3.862089
2	frozenset({toilet metal sign})	frozenset({bathroom metal sign})	0.026993	0.040589	0.021728	0.804938	19.831353	1.0	0.020632	4.918499
3	frozenset({red retrospot sugar jam bowl})	frozenset({red retrospot small milk jug})	0.023660	0.037123	0.016796	0.709859	19.121592	1.0	0.015917	3.318652
4	frozenset({painted metal pears assorted})	frozenset({assorted colour bird ornament})	0.021927	0.078512	0.016596	0.756839	9.639738	1.0	0.014874	3.789618
...
81	frozenset({green 3 piece mini dots cutlery set...})	frozenset({pink 3 piece mini dots cutlery set})	0.011464	0.029992	0.010064	0.877907	29.271370	1.0	0.009720	7.944827

HITUNG NILAI SUPPORT, CONFIDENCE, DAN LIFT DARI SETIAP PASANGAN PRODUK YANG MUNGKIN

Interpretation

- Kode menggunakan fungsi association_rules untuk mencari hubungan antarproduk dari hasil frequent itemset.
- Aturan dibentuk berdasarkan metrik confidence dengan ambang minimum 0,7, artinya hubungan produk harus cukup kuat.
- Setiap baris menunjukkan aturan jika membeli produk A (antecedent), maka cenderung membeli produk B (consequent).
- Kolom support menunjukkan seberapa sering kombinasi produk muncul, sedangkan confidence menunjukkan tingkat kepercayaannya.
- Nilai lift > 1 menandakan hubungan antarproduk kuat dan layak digunakan untuk strategi cross-selling atau bundling

INSIGHT

- Produk Red Hanging Heart T-light Holder dibeli dalam 5.9% dari total transaksi.
- Saat pembeli sudah membeli Red Hanging Heart T-light Holder, 72% dari mereka akan membeli White Hanging Heart T-light Holder juga.
- Terdapat asosiasi yang kuat antara Red Hanging Heart T-light Holder dengan White Hanging Heart T-light Holder dengan peluang kedua produk tersebut dibeli bersamaan dalam 1 transaksi 4 kali lebih besar dari peluang kedua produk tersebut dibeli secara masing-masing.
- Berdasarkan insight tersebut, produk White Hanging Heart T-light Holder dapat ditempatkan di dekat/sebelah Red Hanging Heart T-light Holder atau kedua produk tersebut dapat dijual dalam bundel.



RECOMMENDATION

Buat paket produk (bundling)

1 Karena produk Red Hanging Heart T-light Holder sering dibeli bersamaan dengan White Hanging Heart T-light Holder (confidence 72% & lift > 4), maka kedua produk ini bisa dijual dalam satu paket.

Contoh: Paket Romantic Decoration Set → “RED Hanging Heart T-light Holder + WHITE Hanging Heart T-light Holder” dengan harga lebih hemat.

2 Kasih promo beli lebih dari 1 produk

Dorong pembeli yang awalnya hanya beli 1 produk supaya membeli 2 produk sekaligus dengan memberikan diskon.

Contoh: Promo: “Beli Red + White Hanging Heart T-light Holder → Diskon 10%”



RECOMMENDATION

Tampilkan rekomendasi “sering dibeli bersama”

3

Saat pelanggan melihat atau memasukkan Red Hanging Heart T-light Holder ke keranjang, tampilkan rekomendasi produk White Hanging Heart T-light Holder.

Contoh: Saat pelanggan lihat: “RED Hanging Heart T-light Holder” muncul rekomendasi: → “WHITE Hanging Heart T-light Holder”

4

Susun produk yang saling melengkapi berdekatan

Produk yang sering dibeli bersama sebaiknya diletakkan bersebelahan di etalase online maupun rak toko fisik, supaya lebih mudah terlihat dan mendorong pembelian ganda.

Contoh: Di etalase online / rak toko, taruh: “RED Hanging Heart T-light Holder” tepat di sebelah “WHITE Hanging Heart T-light Holder”



Conclusion

Berdasarkan hasil Market Basket Analysis, algoritma Apriori berhasil mengidentifikasi pola pembelian pelanggan yang tidak terlihat pada data mentah. Transformasi data ke dalam bentuk basket matrix memungkinkan analisis hubungan antarproduk menggunakan metrik support, confidence, dan lift. Hasil association rules menunjukkan keterkaitan produk yang kuat dan dapat dimanfaatkan untuk strategi cross-selling, bundling, sistem rekomendasi, serta peningkatan nilai transaksi dan efektivitas pemasaran.





Thank You

TERBUKA UNTUK KOLABORASI
PROYEK LEBIH LANJUT