



UNIVERSITAS NEGERI SURABAYA

User Segmentation Analysis Using Python

RFM-BASED ONLINE RETAIL SEGMENTATION

SISTEM INFORMASI - 18 JANUARI 2026





PROFIL ANALYST



ROZALINDA TITALIA PUTRI



SURABAYA, JAWA TIMUR



LINDAROZA509@GMAIL.COM



ROZALINDA TITALIA PUTRI

SAYA ADALAH MAHASISWA SISTEM INFORMASI SEMESTER 3 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 bisnis ritel online, memahami perilaku pelanggan menjadi hal krusial untuk meningkatkan penjualan dan mempertahankan pelanggan bernilai tinggi. Dataset transaksi online retail menyimpan informasi penting terkait waktu pembelian, frekuensi transaksi, dan nilai belanja pelanggan yang dapat dimanfaatkan untuk analisis perilaku pelanggan. Salah satu pendekatan yang umum digunakan adalah metode RFM (Recency, Frequency, Monetary), yang mengelompokkan pelanggan berdasarkan seberapa baru mereka bertransaksi(recency), seberapa sering mereka berbelanja(frequency), dan seberapa besar nilai transaksi yang dihasilkan(monetary)



TUJUAN



- Melakukan segmentasi pelanggan berdasarkan metode RFM menggunakan data transaksi online retail.
- Mengidentifikasi karakteristik dan pola perilaku pelanggan pada setiap segmen.
- Mengukur nilai dan loyalitas pelanggan berdasarkan recency, frequency, dan monetary value.
- Memberikan insight berbasis data untuk mendukung strategi pemasaran, retensi pelanggan, dan pengambilan keputusan bisnis.



Tools & Teknologi

- Jupyter Notebook : 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.

DATA PREPARATION & CLEANING

Meliputi proses mengimpor dataset transaksi online retail ke Jupyter Notebook, penyesuaian tipe data, penanganan nilai kosong dan data duplikat, serta pemrosesan kolom tanggal untuk mendukung analisis berbasis RFM

RFM ANALYSIS & USER SEGMENTATION

Melakukan perhitungan metrik Recency, Frequency, dan Monetary untuk setiap pelanggan, pemberian skor RFM, serta pengelompokan pelanggan ke dalam segmen seperti Champion, Loyal Customers, dan At Risk berdasarkan karakteristik perilaku transaksi



DATA VISUALIZATION & INSIGHT

Menyajikan hasil segmentasi dalam bentuk tabel ringkasan dan visualisasi untuk memudahkan interpretasi karakteristik tiap segmen serta mendukung pengambilan keputusan 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 UNTUK MEMBUAT USER SEGMENTATION COHORT DI PYTHON

```
[8]: import pandas as pd
      import numpy as np
      import datetime as dt

[ ]: import os
      os.getcwd()
```

1. *Impor data transaksi ke DataFrame*

- 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 UNTUK MEMBUAT USER SEGMENTATION COHORT 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

1. *Impor data transaksi ke DataFrame*

- 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 UNTUK MEMBUAT USER SEGMENTATION COHORT 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
```

1. *Impor data transaksi ke DataFrame*

- 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 UNTUK MEMBUAT USER SEGMENTATION COHORT DI PYTHON

2. Lakukan data cleansing

```
df_clean = df.copy()
# membuat kolom date
df_clean['date'] = pd.to_datetime(df_clean['order_date']).astype('datetime64[ns]')
# menghapus semua baris tanpa customer_id
df_clean = df_clean[~df_clean['customer_id'].isna()]
# 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 '))]
# membuat kolom order_status dengan nilai 'cancelled' jika order_id diawali dengan huruf
# 'c' dan 'delivered' jika order_id tanpa awalan huruf 'c'
df_clean['order_status'] = np.where(df_clean['order_id'].str[:1]=='C', 'cancelled', 'delivered')
# 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']
```

```
# 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')
# mengkonversi customer_id menjadi string
df_clean['customer_id'] = df_clean['customer_id'].astype(str)
# 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
```

TAHAPAN UNTUK MEMBUAT USER SEGMENTATION COHORT DI PYTHON

	order_id	product_code	product_name	quantity	order_date	price	customer_id	date	order_status	amount
0	C493411	21539	red retrospot butter dish	1	2010-01-04 09:43:00	4.25	14590.0	2010-01-04 09:43:00	cancelled	4.25
1	493414	21844	red retrospot mug	36	2010-01-04 10:28:00	2.55	14590.0	2010-01-04 10:28:00	delivered	91.80
2	493414	21533	retro spot large milk jug	12	2010-01-04 10:28:00	4.25	14590.0	2010-01-04 10:28:00	delivered	51.00
3	493414	37508	new england ceramic cake server	2	2010-01-04 10:28:00	2.55	14590.0	2010-01-04 10:28:00	delivered	5.10
4	493414	35001G	hand open shape gold	2	2010-01-04 10:28:00	4.25	14590.0	2010-01-04 10:28:00	delivered	8.50
...
358464	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	delivered	1.25
358465	539988	84849D	hot baths soap holder	1	2010-12-23 16:06:00	1.69	18116.0	2010-12-23 16:06:00	delivered	1.69
358466	539988	84849B	fairy soap soap holder	1	2010-12-23 16:06:00	1.69	18116.0	2010-12-23 16:06:00	delivered	1.69
358467	539988	22854	cream sweetheart egg holder	2	2010-12-23 16:06:00	4.95	18116.0	2010-12-23 16:06:00	delivered	9.90
358468	539988	475598	tea time oven glove	2	2010-12-23 16:06:00	1.25	18116.0	2010-12-23 16:06:00	delivered	2.50

2. Lakukan data cleansing

- Nilai quantity dan price sudah divalidasi (> 0), serta missing value dan data error telah dihapus.
- Kolom amount dihitung sebagai quantity \times price untuk menunjukkan nilai transaksi per produk.
- Nama produk telah dinormalisasi (lowercase) dan customer_id dikonversi ke string.
- Outlier pada quantity dan amount dihapus menggunakan Z-score ($|z| < 3$).
- Dataset akhir siap digunakan untuk analisis penjualan, pelanggan, dan performa produk.

TAHAPAN UNTUK MEMBUAT USER SEGMENTATION COHORT DI PYTHON

2. Lakukan *data cleansing*

- DataFrame hasil pembersihan data memiliki 358.469 baris dan 10 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, dan order_status bertipe object.
- Total penggunaan memori sebesar 27,3 MB, masih tergolong efisien untuk analisis lanjutan.

```
df_clean.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 358469 entries, 0 to 358468
Data columns (total 10 columns):
 #   Column      Non-Null Count  Dtype  
--- 
 0   order_id    358469 non-null   object 
 1   product_code 358469 non-null   object 
 2   product_name 358469 non-null   object 
 3   quantity     358469 non-null   int64  
 4   order_date   358469 non-null   object 
 5   price        358469 non-null   float64
 6   customer_id 358469 non-null   object 
 7   date         358469 non-null   datetime64[ns] 
 8   order_status 358469 non-null   object 
 9   amount        358469 non-null   float64 

dtypes: datetime64[ns](1), float64(2), int64(1), object(6)
memory usage: 27.3+ MB
```

TAHAPAN UNTUK MEMBUAT USER SEGMENTATION COHORT DI PYTHON

```
df_user = df_clean.groupby('customer_id', as_index=False).agg(order_cnt=('order_id','nunique'),
                                                               max_order_date=("date",'max'),
                                                               total_order_value=("amount",'sum'))
```

	customer_id	order_cnt	max_order_date	total_order_value
0	12346.0	5	2010-10-04 09:54:00	602.40
1	12608.0	1	2010-10-31 10:49:00	415.79
2	12745.0	2	2010-08-10 10:14:00	723.85
3	12746.0	2	2010-06-30 08:19:00	266.35
4	12747.0	19	2010-12-13 10:41:00	4094.79
...
3884	18283.0	6	2010-11-22 15:30:00	641.77
3885	18284.0	2	2010-10-06 12:31:00	486.68
3886	18285.0	1	2010-02-17 10:24:00	427.00
3887	18286.0	2	2010-08-20 11:57:00	941.48
3888	18287.0	4	2010-11-22 11:51:00	2345.71

3889 rows × 4 columns

3. *Agregat data transaksi ke bentuk summary total transaksi(order), total nilai order (order value), tanggal order terakhir dari setiap pengguna*

- Kode mengelompokkan data berdasarkan customer_id, sehingga setiap baris merepresentasikan satu pelanggan unik.
- Kolom order_cnt menunjukkan jumlah transaksi unik per pelanggan hasil agregasi nunique.
- Kolom max_order_date menunjukkan tanggal terakhir transaksi pelanggan dari agregasi max.
- Kolom total_order_value merupakan total nilai belanja pelanggan hasil agregasi sum pada amount.
- Output menghasilkan 3.889 pelanggan unik dengan 4 kolom, siap digunakan untuk analisis loyalitas dan segmentasi pelanggan.

TAHAPAN UNTUK MEMBUAT USER SEGMENTATION COHORT DI PYTHON

```
today = df_clean['date'].max()
df_user['day_since_last_order'] = (today - df_user['max_order_date']).dt.days
df_user
```

	customer_id	order_cnt	max_order_date	total_order_value	day_since_last_order
0	12346.0	5	2010-10-04 09:54:00	602.40	80
1	12608.0	1	2010-10-31 10:49:00	415.79	53
2	12745.0	2	2010-08-10 10:14:00	723.85	135
3	12746.0	2	2010-06-30 08:19:00	266.35	176
4	12747.0	19	2010-12-13 10:41:00	4094.79	10
...
3884	18283.0	6	2010-11-22 15:30:00	641.77	31
3885	18284.0	2	2010-10-06 12:31:00	486.68	78
3886	18285.0	1	2010-02-17 10:24:00	427.00	309
3887	18286.0	2	2010-08-20 11:57:00	941.48	125
3888	18287.0	4	2010-11-22 11:51:00	2345.71	31

3889 rows × 5 columns

4. Buat kolom jumlah hari sejak order terakhir

- Kode mengambil tanggal transaksi terakhir dalam dataset sebagai acuan (today).
- Kolom day_since_last_order dihitung dari selisih hari antara tanggal terakhir dataset dan max_order_date tiap pelanggan.
- Nilai pada kolom ini menunjukkan tingkat recency pelanggan:
- Nilai kecil → pelanggan baru saja bertransaksi.
- Nilai besar → pelanggan sudah lama tidak bertransaksi.
- Output menghasilkan 3.889 pelanggan dengan 5 kolom, siap digunakan untuk analisis RFM dan churn

TAHAPAN UNTUK MEMBUAT USER SEGMENTATION COHORT DI PYTHON

df_user.describe()				
	order_cnt	max_order_date	total_order_value	day_since_last_order
count	3889.000000	3889	3889.000000	3889.000000
mean	5.128568	2010-09-23 18:15:51.267678208	1544.623084	90.651581
min	1.000000	2010-01-05 12:43:00	1.250000	0.000000
25%	1.000000	2010-08-19 12:30:00	296.360000	25.000000
50%	3.000000	2010-10-26 18:45:00	648.200000	57.000000
75%	6.000000	2010-11-28 14:54:00	1585.940000	126.000000
max	163.000000	2010-12-23 16:06:00	71970.390000	352.000000
std	8.499330	Nan	3434.816315	88.883201

4. Buat kolom jumlah hari sejak order terakhir

- Rata-rata pelanggan melakukan ± 5 transaksi, dengan median 3 transaksi.
- Nilai maksimum order_cnt mencapai 163 transaksi, menunjukkan adanya pelanggan sangat loyal.
- Rata-rata total nilai belanja pelanggan ≈ 1.545 , dengan variasi yang cukup besar (std tinggi).
- Median day_since_last_order sebesar 57 hari, menandakan sebagian besar pelanggan tidak bertransaksi dalam 1–2 bulan terakhir.
- Rentang day_since_last_order dari 0 hingga 352 hari menunjukkan perbedaan tingkat aktivitas pelanggan yang signifikan

TAHAPAN UNTUK MEMBUAT USER SEGMENTATION COHORT DI PYTHON

```
df_user['recency_score'] = pd.cut(df_user['day_since_last_order'],
                                    bins=[df_user['day_since_last_order'].min(),
                                          np.percentile(df_user['day_since_last_order'], 20),
                                          np.percentile(df_user['day_since_last_order'], 40),
                                          np.percentile(df_user['day_since_last_order'], 60),
                                          np.percentile(df_user['day_since_last_order'], 80),
                                          df_user['day_since_last_order'].max()],
                                          labels=[5, 4, 3, 2, 1],
                                          include_lowest=True).astype(int)
```

	customer_id	order_cnt	max_order_date	total_order_value	day_since_last_order	recency_score
0	12346.0	5	2010-10-04 09:54:00	602.40	80	2
1	12608.0	1	2010-10-31 10:49:00	415.79	53	3
2	12745.0	2	2010-08-10 10:14:00	723.85	135	2
3	12746.0	2	2010-06-30 08:19:00	266.35	176	1
4	12747.0	19	2010-12-13 10:41:00	4094.79	10	5
...
3884	18283.0	6	2010-11-22 15:30:00	641.77	31	4
3885	18284.0	2	2010-10-06 12:31:00	486.68	78	2
3886	18285.0	1	2010-02-17 10:24:00	427.00	309	1
3887	18286.0	2	2010-08-20 11:57:00	941.48	125	2
3888	18287.0	4	2010-11-22 11:51:00	2345.71	31	4

5. Buat binning dari jumlah hari sejak order terakhir yang terdiri dari bins dengan batas-batasnya merupakan min.P20,P40,P60,P80,max dan beri label 1 sampai 5 dari bin tertinggi ke terendah sebagai skor recency

- Kode `pd.cut()` mengelompokkan `day_since_last_order` ke dalam 5 kategori berbasis persentil (20%–80%) agar distribusi data lebih merata.
- Proses ini menghasilkan kolom baru `recency_score` sebagai indikator kebaruan transaksi pelanggan.
- Skor recency bernilai 1–5, di mana semakin kecil jarak hari sejak transaksi terakhir, semakin tinggi skornya.
- Seluruh hasil pengelompokan dikonversi ke tipe data int sehingga mudah digunakan untuk analisis lanjutan.

TAHAPAN UNTUK MEMBUAT USER SEGMENTATION COHORT DI PYTHON

```
df_user['frequency_score'] = pd.cut(df_user['order_cnt'],
                                     bins=[0,
                                           np.percentile(df_user['order_cnt'], 20),
                                           np.percentile(df_user['order_cnt'], 40),
                                           np.percentile(df_user['order_cnt'], 60),
                                           np.percentile(df_user['order_cnt'], 80),
                                           df_user['order_cnt'].max()],
                                     labels=[1, 2, 3, 4, 5],
                                     include_lowest=True).astype(int)
```

	customer_id	order_cnt	max_order_date	total_order_value	day_since_last_order	recency_score	frequency_score
0	12346.0	5	2010-10-04 09:54:00	602.40	80	2	4
1	12608.0	1	2010-10-31 10:49:00	415.79	53	3	1
2	12745.0	2	2010-08-10 10:14:00	723.85	135	2	2
3	12746.0	2	2010-06-30 08:19:00	266.35	176	1	2
4	12747.0	19	2010-12-13 10:41:00	4094.79	10	5	5
...
3884	18283.0	6	2010-11-22 15:30:00	641.77	31	4	4
3885	18284.0	2	2010-10-06 12:31:00	486.68	78	2	2
3886	18285.0	1	2010-02-17 10:24:00	427.00	309	1	1
3887	18286.0	2	2010-08-20 11:57:00	941.48	125	2	2
3888	18287.0	4	2010-11-22 11:51:00	2345.71	31	4	3

3889 rows × 7 columns

6. Buat binning dari total transaksi (order) yang terdiri dari 5 bins dengan batas-batasnya merupakan min, P20, P40, P60, PBO, max dan beri label 1 sampai 5 dari bin terendah ke tertinggi sebagai skor frequency

- Kode pd.cut() digunakan untuk mengelompokkan jumlah transaksi (order_cnt) ke dalam 5 kategori berdasarkan persentil (20%–80%).
- Proses ini menghasilkan kolom baru frequency_score yang merepresentasikan frekuensi transaksi pelanggan.
- Skor frequency bernilai 1–5, di mana semakin sering pelanggan bertransaksi, semakin tinggi skornya.
- Hasil pengelompokan dikonversi ke tipe data int agar mudah digunakan dalam analisis.
- Output menunjukkan DataFrame memiliki 3.889 baris dan 7 kolom, dengan frequency_score berhasil ditambahkan untuk analisis RFM

TAHAPAN UNTUK MEMBUAT USER SEGMENTATION COHORT DI PYTHON

```
df_user['monetary_score'] = pd.cut(df_user['total_order_value'],
                                     bins=[df_user['total_order_value'].min(),
                                           np.percentile(df_user['total_order_value'], 20),
                                           np.percentile(df_user['total_order_value'], 40),
                                           np.percentile(df_user['total_order_value'], 60),
                                           np.percentile(df_user['total_order_value'], 80),
                                           df_user['total_order_value'].max()],
                                     labels=[1, 2, 3, 4, 5],
                                     include_lowest=True).astype(int)
```

	customer_id	order_cnt	max_order_date	total_order_value	day_since_last_order	recency_score	frequency_score	monetary_score
0	12346.0	5	2010-10-04 09:54:00	602.40	80	2	4	3
1	12608.0	1	2010-10-31 10:49:00	415.79	53	3	1	2
2	12745.0	2	2010-08-10 10:14:00	723.85	135	2	2	3
3	12746.0	2	2010-06-30 08:19:00	266.35	176	1	2	2
4	12747.0	19	2010-12-13 10:41:00	4094.79	10	5	5	5
...
3884	18283.0	6	2010-11-22 15:30:00	641.77	31	4	4	3
3885	18284.0	2	2010-10-06 12:31:00	486.68	78	2	2	3
3886	18285.0	1	2010-02-17 10:24:00	427.00	309	1	1	2
3887	18286.0	2	2010-08-20 11:57:00	941.48	125	2	2	4
3888	18287.0	4	2010-11-22 11:51:00	2345.71	31	4	3	5

3889 rows × 8 columns

7. Buat binning dari total nilai order (order value) yang terdiri dari 5 bins dengan batas-batasnya merupakan min, P20, P40, P60, P80, max dan beri label 1 sampai 5 dari bin terendah ke tertinggi sebagai skor monetary

- Kode pd.cut() digunakan untuk mengelompokkan nilai total_order_value ke dalam 5 kategori berbasis persentil.
- Proses ini menghasilkan kolom baru monetary_score yang mencerminkan total nilai belanja pelanggan.
- Skor monetary bernilai 1–5, di mana semakin besar total transaksi, semakin tinggi skornya.
- Parameter include_lowest=True memastikan seluruh data nilai minimum tetap terkласifikasi.
- Output menunjukkan DataFrame kini memiliki 3.889 baris dan 8 kolom, lengkap dengan skor RFM (Recency, Frequency, Monetary) siap untuk segmentasi pelanggan

TAHAPAN UNTUK MEMBUAT USER SEGMENTATION COHORT DI PYTHON

```
df_user['segment'] = np.select([
    (df_user['recency_score'] == 5) & (df_user['frequency_score'] >= 4),
    (df_user['recency_score'].between(3, 4)) & (df_user['frequency_score'] >= 4),
    (df_user['recency_score'] >= 4) & (df_user['frequency_score'].between(2, 3)),
    (df_user['recency_score'] <= 2) & (df_user['frequency_score'] == 5),
    (df_user['recency_score'] == 3) & (df_user['frequency_score'] == 3),
    (df_user['recency_score'] == 5) & (df_user['frequency_score'] == 1),
    (df_user['recency_score'] == 4) & (df_user['frequency_score'] == 1),
    (df_user['recency_score'] <= 2) & (df_user['frequency_score'].between(3, 4)),
    (df_user['recency_score'] == 3) & (df_user['frequency_score'] <= 2),
    (df_user['recency_score'] <= 2) & (df_user['frequency_score'] <= 2)
],
[
    '01-Champion',
    '02-Loyal Customers',
    '03-Potential Loyalists',
    "04-Can't Lose Them",
    '05-Need Attention',
    '06-New Customers',
    '07-Promising',
    '08-At Risk',
    '09-About to Sleep',
    '10-Hibernating'
],
default='Unclassified'
)

df_user
```

8. Buat kolom nama segmen berdasarkan skor recency dan frequency

- np.select() digunakan untuk membuat kolom segment berdasarkan kombinasi recency_score dan frequency_score.
- '01-Champion' diberikan kepada pelanggan dengan recency_score = 5 dan frequency_score ≥ 4.
- '02-Loyal Customers' untuk pelanggan dengan recency_score 3–4 dan frequency_score ≥ 4.
- '10-Hibernating' untuk pelanggan dengan recency_score ≤ 2 dan frequency_score ≤ 2.
- Pelanggan yang tidak memenuhi seluruh kondisi diberi label 'Unclassified' sebagai nilai default.

TAHAPAN UNTUK MEMBUAT USER SEGMENTATION COHORT DI PYTHON

	customer_id	order_cnt	max_order_date	total_order_value	day_since_last_order	recency_score	frequency_score	monetary_score	segment
0	12346.0	5	2010-10-04 09:54:00	602.40	80	2	4	3	08-At Risk
1	12608.0	1	2010-10-31 10:49:00	415.79	53	3	1	2	09-About to Sleep
2	12745.0	2	2010-08-10 10:14:00	723.85	135	2	2	3	10-Hibernating
3	12746.0	2	2010-06-30 08:19:00	266.35	176	1	2	2	10-Hibernating
4	12747.0	19	2010-12-13 10:41:00	4094.79	10	5	5	5	01-Champion
...
3884	18283.0	6	2010-11-22 15:30:00	641.77	31	4	4	3	02-Loyal Customers
3885	18284.0	2	2010-10-06 12:31:00	486.68	78	2	2	3	10-Hibernating
3886	18285.0	1	2010-02-17 10:24:00	427.00	309	1	1	2	10-Hibernating
3887	18286.0	2	2010-08-20 11:57:00	941.48	125	2	2	4	10-Hibernating
3888	18287.0	4	2010-11-22 11:51:00	2345.71	31	4	3	5	03-Potential Loyalists

8. Buat kolom nama segmen berdasarkan skor recency dan frequency

- Output berisi ringkasan data per customer_id termasuk hasil skor RFM.
- day_since_last_order menunjukkan jarak hari sejak transaksi terakhir hingga tanggal analisis.
- recency_score, frequency_score, monetary_score bernilai 1–5 sebagai dasar segmentasi.
- Contoh ID 12747.0 masuk '01-Champion' karena skor RFM sempurna (5).
- Contoh ID 18285.0 masuk '10-Hibernating' karena 309 hari tidak bertransaksi dan skor rendah.
- Total terdapat 4.389 data pelanggan yang dianalisis

TAHAPAN UNTUK MEMBUAT USER SEGMENTATION COHORT DI PYTHON

```
summary = pd.pivot_table(df_user, index='segment',
                         values=['customer_id','day_since_last_order','order_cnt','total_order_value'],
                         aggfunc={'customer_id': pd.Series.nunique,
                                   'day_since_last_order': [np.mean, np.median],
                                   'order_cnt': [np.mean, np.median],
                                   'total_order_value': [np.mean, np.median]})

summary['pct_unique'] = (summary['customer_id'] / summary['customer_id'].sum() * 100).round(1)

C:\Users\Rozalinda\AppData\Local\Temp\ipykernel_6136\690746973.py:1: FutureWarning: The provided callable <function mean> currently uses SeriesGroupBy.mean. In a future version of pandas, the provided callable will be used directly. To keep current behavior, use SeriesGroupBy.mean instead.
summary = pd.pivot_table(df_user, index='segment',
C:\Users\Rozalinda\AppData\Local\Temp\ipykernel_6136\690746973.py:1: FutureWarning: The provided callable <function median> currently uses SeriesGroupBy.median. In a future version of pandas, the provided callable will be used directly. To keep current behavior, use SeriesGroupBy.median instead.
summary = pd.pivot_table(df_user, index='segment',
                         customer_id day_since_last_order          order_cnt      total_order_value  pct_unique
                           nunique        mean   median        mean   median        mean   median
                           segment
    01-Champion       553  10.533454      9.0  15.432188     10.0  4989.208761  2773.910    14.2
    02-Loyal Customers    549  41.200364     37.0  8.744991      7.0  2618.121117  1937.050    14.1
    03-Potential Loyalists    514  23.083658     24.0  2.830739      3.0   766.076265   621.005    13.2
    04-Can't Lose Them      62  123.274194    113.0  11.467742     10.0  2851.737258  2268.405     1.6
    05-Need Attention       184  58.505435     59.0  3.402174      3.0  1004.317071   826.370     4.7
    06-New Customers         50  14.000000     16.0  1.000000      1.0  244.689000   193.675     1.3
    07-Promising            133  31.954887    32.0  1.000000      1.0  288.694135   239.460     3.4
    08-At Risk              418 141.531100    120.0  4.126794      4.0  1141.224835  866.320    10.7
    09-About to Sleep        370  58.175676     58.0  1.416216      1.0  448.176081   336.735     9.5
    10-Hibernating          1056 197.151515   199.0  1.312500      1.0  342.618450   256.900    27.2
```

9. Tampilkan summary dari RFM segmentation (poin 8) berupa banyaknya pengguna,rata-rata dan median dari total order,total order value, dan jumlah hari sejak order terakhir

- Segmen terbesar adalah '10-Hibernating' dengan 1.056 pelanggan (27,2%) dan rata-rata 197 hari sejak transaksi terakhir.
- Segmen '01-Champion' mencakup 14,2% pelanggan dengan rata-rata nilai belanja tertinggi (4.989).
- '03-Potential Loyalists' mencakup 13,2% pelanggan, rata-rata terakhir transaksi 23 hari lalu, berpotensi menjadi Champion.
- Segmen '08-At Risk' (10,7%) perlu perhatian karena meski rata-rata 4 order, pelanggan sudah tidak bertransaksi selama 141 hari.

TAHAPAN UNTUK MEMBUAT USER SEGMENTATION COHORT DI PYTHON

```
import matplotlib.pyplot as plt

counts = summary[['customer_id', 'nunique']]
pct = summary['pct_unique']

labels = [
    f'{seg}\n{n:int(c)} ({p}%)'
    for seg, c, p in zip(counts.index, counts.values, pct.values)
]

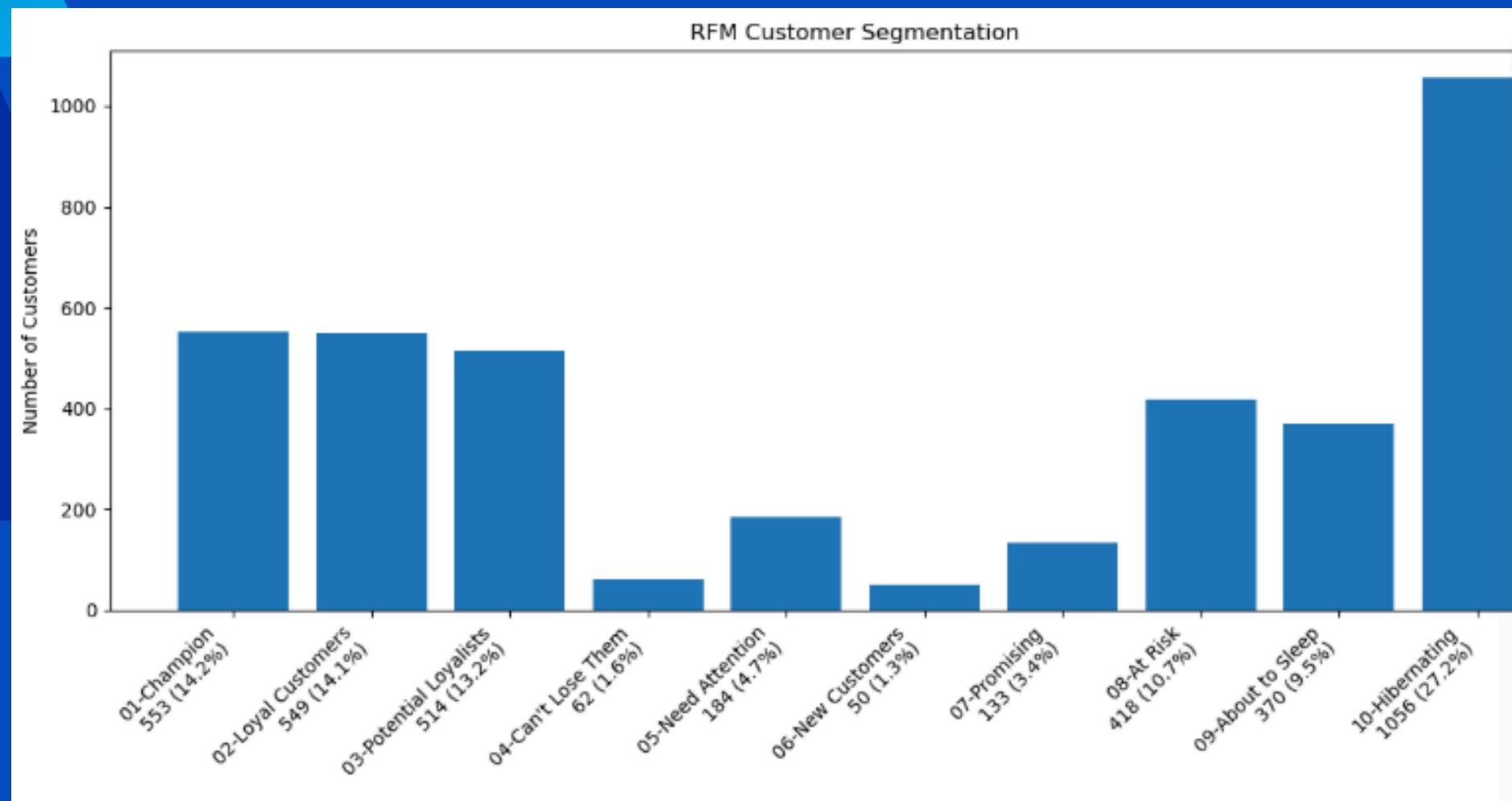
plt.figure(figsize=(12,6))
plt.bar(labels, counts.values)
plt.xticks(rotation=45, ha='right')
plt.title('RFM Customer Segmentation')
plt.ylabel('Number of Customers')
plt.tight_layout()
plt.show()
```

10. Visualisasi RFM

- Kode membuat grafik batang segmentasi pelanggan RFM menggunakan matplotlib.
- counts berisi jumlah pelanggan unik per segmen dan pct berisi persentasenya.
- Label grafik menampilkan nama segmen, jumlah pelanggan, dan persentase.
- Grafik diatur dengan ukuran 12×6, rotasi label 45°, judul grafik, label sumbu Y, dan tight_layout() agar rapi.

INSIGHT

.....
.....
.....



- Segmen Hibernating (27,2%), At Risk (10,7%), dan About to Sleep (9,5%) jika digabung mencapai 47,4% pelanggan, menunjukkan masalah retensi yang serius karena hampir setengah pelanggan sudah menjauh atau tidak aktif
- Segmen Champion dan Loyal Customers mencakup sekitar 28,3% pelanggan, dengan Champion sebagai aset utama karena nilai transaksi tertinggi (4.989) dan aktivitas tinggi (rata-rata 15 order)
- Segmen Potential Loyalists (13,2%) masih aktif (terakhir transaksi 23 hari lalu), sehingga perlu difokuskan untuk dikonversi menjadi Loyal atau Champion agar tidak turun menjadi At Risk

RECOMMENDATION



Prioritaskan Reaktivasi Pelanggan Kritis

- Hampir 47,4% pelanggan berada di segmen Hibernating, At Risk, dan About to Sleep, sehingga risiko kehilangan pelanggan sangat tinggi.
- Fokus pada kampanye reaktivasi terarah dengan diskon personal, voucher, dan gratis ongkir.
- Terapkan pendekatan berbeda tiap segmen untuk meningkatkan peluang pelanggan kembali aktif.



Jaga dan Maksimalkan Pelanggan Terbaik

- Segmen Champion dan Loyal hanya 28,3% pelanggan, tetapi menjadi penyumbang pendapatan terbesar.
- Bangun program loyalitas eksklusif dan layanan personal agar mereka tetap setia dan meningkatkan nilai belanja.
- Manfaatkan mereka sebagai brand ambassador melalui program referra



Kembangkan Pelanggan Potensial Menjadi Loyal

- Segmen Potential Loyalists (13,2%) masih aktif dan berpotensi naik menjadi Loyal atau Champion.
- Berikan promo lanjutan dan reminder repeat order untuk membentuk kebiasaan belanja rutin.
- Pantau jarak transaksi agar mereka tidak turun menjadi At Risk

Conclusion

Segmentasi RFM menunjukkan bahwa meskipun bisnis memiliki basis pelanggan bernilai tinggi pada segmen Champion dan Loyal sekitar 28%, hampir setengah pelanggan berada pada segmen kritis seperti Hibernating, At Risk, dan About to Sleep. Dominasi segmen tidak aktif ini menandakan masalah retensi yang serius, di mana banyak pelanggan mulai menjauh setelah beberapa transaksi awal. Kondisi ini menunjukkan perlunya fokus pada strategi reaktivasi pelanggan lama dan penguatan program loyalitas untuk menjaga pelanggan terbaik agar tidak berpindah ke segmen berisiko.





THANK YOU!

TERBUKA UNTUK KOLABORASI
PROYEK LEBIH LANJUT

