notebooks-04-segmentation

January 6, 2026

0.1 MERCHANT SEGMENTATION

0.1.1 RFM Metrics

[6]:
```
import pandas as pd
import numpy as np
from faker import Faker
from datetime import timedelta
```

[7]:
```
merchants = pd.read_csv(
    r"D:\Shopify-Revenue-Growth-Analytics\Data\merchants.csv",
    parse_dates=["signup_date", "churn_date"],
    dayfirst=True
)

orders = pd.read_csv(
    r"D:\Shopify-Revenue-Growth-Analytics\Data\orders.csv",
    parse_dates=["order_date"],
    dayfirst=True
)
```

[9]:
```
orders.dtypes
```

[9]:
```
order_id           object
merchant_id        object
order_date         object
order_value       float64
channel            object
payment_method     object
dtype: object
```

[10]:
```
orders["order_date"] = pd.to_datetime(orders["order_date"])
```

[11]:
```
orders.dtypes
```

[11]:
```
order_id                    object
merchant_id                 object
order_date          datetime64[ns]
```

```
order_value              float64
channel                   object
payment_method            object
dtype: object
```

[12]:
```
analysis_date = pd.Timestamp("2024-12-31")

rfm = orders.groupby("merchant_id").agg({
    "order_date": lambda x: (analysis_date - x.max()).days,
    "order_id": "count",
    "order_value": "sum"
}).reset_index()

rfm.columns = ["merchant_id", "recency", "frequency", "monetary"]
```

[13]:
```
rfm.head()
```

[13]:

	merchant_id	recency	frequency	monetary
0	M00001	568	60	3104.35
1	M00002	207	51	5336.55
2	M00003	2	53	2656.85
3	M00004	1	175	17345.86
4	M00005	2	89	4364.08

[14]:
```
rfm.describe()
```

[14]:

	recency	frequency	monetary
count	1198.000000	1198.000000	1198.000000
mean	59.230384	106.074290	10114.115017
std	152.444253	54.374182	9677.493828
min	-332.000000	10.000000	540.360000
25%	1.000000	61.000000	4062.075000
50%	4.000000	107.000000	7276.740000
75%	28.750000	153.000000	12745.550000
max	703.000000	199.000000	52861.490000

[15]:
```
from sklearn.preprocessing import StandardScaler
from sklearn.cluster import KMeans

scaler = StandardScaler()
rfm_scaled = scaler.fit_transform(rfm[["recency", "frequency", "monetary"]])

kmeans = KMeans(n_clusters=4, random_state=42)
rfm["segment"] = kmeans.fit_predict(rfm_scaled)
```

[16]:
```
rfm.head()
```

[16]:

	merchant_id	recency	frequency	monetary	segment
0	M00001	568	60	3104.35	2
1	M00002	207	51	5336.55	1
2	M00003	2	53	2656.85	1
3	M00004	1	175	17345.86	0
4	M00005	2	89	4364.08	1

[17]:
```
rfm["segment"].value_counts()
```

[17]:
```
segment
1    497
0    461
2    144
3     96
Name: count, dtype: int64
```

[18]:
```
rfm.groupby("segment")[["recency", "frequency", "monetary"]].mean().round(1)
```

[18]:

segment	recency	frequency	monetary
0	8.6	150.7	11078.4
1	11.9	55.3	4718.5
2	413.9	114.4	7685.3
3	15.5	142.6	37060.2

[19]:
```
rfm["recency_days"] = rfm["recency"]
```

[20]:
```
from sklearn.preprocessing import StandardScaler
from sklearn.cluster import KMeans

# 1. Scale RFM values
scaler = StandardScaler()
rfm_scaled = scaler.fit_transform(rfm[["recency", "frequency", "monetary"]])

# 2. Run KMeans
kmeans = KMeans(n_clusters=4, random_state=42)
rfm["segment"] = kmeans.fit_predict(rfm_scaled)

# 3. Preserve original recency value (Power BI safe)
rfm["recency_days"] = rfm["recency"]
```

[21]:
```
rfm.head()
```

[21]:

	merchant_id	recency	frequency	monetary	segment	recency_days
0	M00001	568	60	3104.35	2	568
1	M00002	207	51	5336.55	1	207
2	M00003	2	53	2656.85	1	2

```
3      M00004         1         175  17345.86         0              1
4      M00005         2          89   4364.08         1              2
```

[22]:
```
rfm.groupby("segment")[["recency", "frequency", "monetary"]].mean().round(1)
```

[22]:
```
          recency  frequency  monetary
segment
0             8.6      150.7   11078.4
1            11.9       55.3    4718.5
2           413.9      114.4    7685.3
3            15.5      142.6   37060.2
```

[23]:
```
segment_profile = (
    rfm
    .groupby("segment")[["recency", "frequency", "monetary"]]
    .mean()
    .round(1)
)

segment_profile
```

[23]:
```
          recency  frequency  monetary
segment
0             8.6      150.7   11078.4
1            11.9       55.3    4718.5
2           413.9      114.4    7685.3
3            15.5      142.6   37060.2
```

[24]:
```
segment_map = {
    0: "High-GMV Loyal Merchants",
    1: "New / Fast-Growing Merchants",
    2: "At-Risk Merchants",
    3: "Low-Value / Dormant Merchants"
}
```

[25]:
```
rfm["segment_name"] = rfm["segment"].map(segment_map)
```

[26]:
```
rfm[["segment", "segment_name"]].drop_duplicates()
```

[26]:
```
    segment                    segment_name
0         2              At-Risk Merchants
1         1   New / Fast-Growing Merchants
3         0       High-GMV Loyal Merchants
11        3  Low-Value / Dormant Merchants
```

[27]:
```
rfm.groupby("segment_name")[["recency", "frequency", "monetary"]].mean().
↪round(1)
```

[27]:
```
                              recency  frequency  monetary
segment_name
At-Risk Merchants               413.9      114.4    7685.3
High-GMV Loyal Merchants          8.6      150.7   11078.4
Low-Value / Dormant Merchants    15.5      142.6   37060.2
New / Fast-Growing Merchants     11.9       55.3    4718.5
```

[28]:
```
merchants.head()
rfm.head()
```

[28]:
```
  merchant_id  recency  frequency  monetary  segment  recency_days  \
0      M00001      568         60   3104.35        2           568
1      M00002      207         51   5336.55        1           207
2      M00003        2         53   2656.85        1             2
3      M00004        1        175  17345.86        0             1
4      M00005        2         89   4364.08        1             2

                  segment_name
0             At-Risk Merchants
1  New / Fast-Growing Merchants
2  New / Fast-Growing Merchants
3      High-GMV Loyal Merchants
4  New / Fast-Growing Merchants
```

[29]:
```
rfm_features = rfm[[
    "merchant_id",
    "recency",
    "frequency",
    "monetary",
    "recency_days",
    "segment",
    "segment_name"
]]
```

[30]:
```
merchant_master = merchants.merge(
    rfm_features,
    on="merchant_id",
    how="left"
)
```

[31]:
```
merchant_master[["recency", "frequency", "monetary"]] = (
    merchant_master[["recency", "frequency", "monetary"]]
    .fillna(0)
)

merchant_master["segment_name"] = merchant_master["segment_name"].fillna(
    "No Orders / Not Activated"
```

```
)
```

[32]:
```
merchant_master.shape[0] == merchants.shape[0]
```

[32]:
```
True
```

[33]:
```
merchant_master["segment_name"].value_counts()
```

[33]:
```
segment_name
New / Fast-Growing Merchants     497
High-GMV Loyal Merchants         461
At-Risk Merchants                144
Low-Value / Dormant Merchants     96
No Orders / Not Activated          2
Name: count, dtype: int64
```

[34]:
```
merchant_master.head()
```

[34]:
```
  merchant_id signup_date    country plan_type industry churned churn_date  \
0      M00001  2023-04-13  Australia     Basic     Food     Yes 2023-06-14
1      M00002  2024-03-11      India   Shopify     Food     Yes 2024-06-11
2      M00003  2023-09-28         US     Basic   Beauty      No        NaT
3      M00004  2023-04-17      India   Shopify     Food      No        NaT
4      M00005  2023-03-13     Canada     Basic   Beauty      No        NaT

   recency  frequency  monetary  recency_days  segment  \
0    568.0       60.0   3104.35         568.0      2.0
1    207.0       51.0   5336.55         207.0      1.0
2      2.0       53.0   2656.85           2.0      1.0
3      1.0      175.0  17345.86           1.0      0.0
4      2.0       89.0   4364.08           2.0      1.0

                  segment_name
0            At-Risk Merchants
1  New / Fast-Growing Merchants
2  New / Fast-Growing Merchants
3     High-GMV Loyal Merchants
4  New / Fast-Growing Merchants
```

[35]:
```
merchant_master.to_csv("D:/Shopify-Revenue-Growth-Analytics/Data/
↪merchant_master.csv", index=False)
```