DE_mini_project

January 23, 2023

Contents

Chapter 1

Import Packages

```
In [1]: import pandas as pd
        import numpy as np
        pd.options.display.max_rows = 200
        pd.options.display.max_columns = 500
        import sqlalchemy as sa
        import matplotlib.pyplot as plt
        import seaborn as sns
        import plotly.express as px
        from datetime import datetime, timedelta
```

Chapter 2

Import Data from DB and Export Data as CSV

2.1 Create connection to Database

```
In [2]: from sqlalchemy import create_engine
        from sqlalchemy.engine import URL

        from sqlalchemy.engine import URL
        connection_url = URL.create(
            "mssql+pyodbc",
            username="sa",
            password="Bii12345",
            host="localhost",
            port=1439,
            database="Northwind",
            query={
                "driver": "ODBC Driver 17 for SQL Server"
            },
        )
        engine = sa.create_engine(connection_url)
        cnxn = engine.connect()
```

2.2 Import OrderFact

```
In [3]: df_product = pd.read_sql('SELECT * FROM dbo.OrderFact', cnxn)

        df_product.head()
```

```
Out[3]:    OrderID CustomerID  EmployeeID  OrderDate RequiredDate ShippedDate  \
        0    10248      VINET           5 1996-07-04   1996-08-01  1996-07-16
        1    10248      VINET           5 1996-07-04   1996-08-01  1996-07-16
        2    10248      VINET           5 1996-07-04   1996-08-01  1996-07-16
        3    10249      TOMSP           6 1996-07-05   1996-08-16  1996-07-10
        4    10249      TOMSP           6 1996-07-05   1996-08-16  1996-07-10

           ShipVia  Freight                    ShipName         ShipAddress ShipCity  \
        0        3    32.38  Vins et alcools Chevalier  59 rue de l'Abbaye    Reims
        1        3    32.38  Vins et alcools Chevalier  59 rue de l'Abbaye    Reims
        2        3    32.38  Vins et alcools Chevalier  59 rue de l'Abbaye    Reims
        3        1    11.61         Toms Spezialitäten       Luisenstr. 48  Münster
        4        1    11.61         Toms Spezialitäten       Luisenstr. 48  Münster

          ShipRegion ShipPostalCode ShipCountry                  ProductName  \
        0       None          51100      France         Mozzarella di Giovanni
        1       None          51100      France                 Queso Cabrales
        2       None          51100      France  Singaporean Hokkien Fried Mee
        3       None          44087     Germany          Manjimup Dried Apples
        4       None          44087     Germany                           Tofu

                           CompanyName    CategoryName  UnitPrice  Quantity  Discount  \
        0                         None  Dairy Products       34.8         5       0.0
        1  Heli Süßwaren GmbH & Co. KG  Dairy Products       14.0        12       0.0
        2                         None  Grains/Cereals        9.8        10       0.0
        3                         None         Produce       42.4        40       0.0
        4      Formaggi Fortini s.r.l.         Produce       18.6         9       0.0

           TotalSales  FinalSales
        0       174.0       174.0
        1       168.0       168.0
        2        98.0        98.0
        3      1696.0      1696.0
        4       167.4       167.4
```

```
In [4]: df_product.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2155 entries, 0 to 2154
Data columns (total 22 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   OrderID         2155 non-null   int64
 1   CustomerID      2155 non-null   object
 2   EmployeeID      2155 non-null   int64
 3   OrderDate       2155 non-null   datetime64[ns]
 4   RequiredDate    2155 non-null   datetime64[ns]
 5   ShippedDate     2082 non-null   datetime64[ns]
 6   ShipVia         2155 non-null   int64
 7   Freight         2155 non-null   float64
 8   ShipName        2155 non-null   object
 9   ShipAddress     2155 non-null   object
 10  ShipCity        2155 non-null   object
 11  ShipRegion      856 non-null    object
 12  ShipPostalCode  2100 non-null   object
 13  ShipCountry     2155 non-null   object
 14  ProductName     2155 non-null   object
 15  CompanyName     744 non-null    object
 16  CategoryName    2155 non-null   object
 17  UnitPrice       2155 non-null   float64
 18  Quantity        2155 non-null   int64
 19  Discount        2155 non-null   float64
 20  TotalSales      2155 non-null   float64
 21  FinalSales      2155 non-null   float64
dtypes: datetime64[ns](3), float64(5), int64(4), object(10)
```

```
memory usage: 370.5+ KB
```

```
In [5]: # Export the data
        df_product.to_csv('../data/processed/OrderFact.csv',
                          sep = '|',
                          index=False)
```

2.3 Import & Export CustomerFact

```
In [6]: df_cst = pd.read_sql('SELECT * FROM dbo.CustomerFact', cnxn)

        df_cst.head()
```

```
Out[6]:   CustomerID   ContactName        Address    City Region PostalCode  Country  \
        0      ALFKI  Maria Anders  Obere Str. 57  Berlin   None      12209  Germany
        1      ALFKI  Maria Anders  Obere Str. 57  Berlin   None      12209  Germany
        2      ALFKI  Maria Anders  Obere Str. 57  Berlin   None      12209  Germany
        3      ALFKI  Maria Anders  Obere Str. 57  Berlin   None      12209  Germany
        4      ALFKI  Maria Anders  Obere Str. 57  Berlin   None      12209  Germany

           OrderDate ShippedDate RequiredDate        ProductName  Quantity  UnitPrice  \
        0 1997-08-25  1997-09-02   1997-09-22   Chartreuse verte      21.0       18.0
        1 1997-08-25  1997-09-02   1997-09-22  Rössle Sauerkraut      15.0       45.6
        2 1997-08-25  1997-09-02   1997-09-22          Spegesild       2.0       12.0
        3 1997-10-03  1997-10-13   1997-10-31       Vegie-spread      20.0       43.9
        4 1997-10-13  1997-10-21   1997-11-24      Aniseed Syrup       6.0       10.0

           Discount  TotalSales  FinalSales
        0      0.25       378.0       283.5
        1      0.25       684.0       513.0
        2      0.25        24.0        18.0
        3      0.00       878.0       878.0
        4      0.00        60.0        60.0
```

```
In [7]: df_cst.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2157 entries, 0 to 2156
Data columns (total 16 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   CustomerID    2157 non-null   object
 1   ContactName   2157 non-null   object
 2   Address       2157 non-null   object
 3   City          2157 non-null   object
 4   Region        826 non-null    object
 5   PostalCode    2102 non-null   object
 6   Country       2157 non-null   object
 7   OrderDate     2155 non-null   datetime64[ns]
 8   ShippedDate   2082 non-null   datetime64[ns]
 9   RequiredDate  2155 non-null   datetime64[ns]
 10  ProductName   2155 non-null   object
 11  Quantity      2155 non-null   float64
 12  UnitPrice     2155 non-null   float64
 13  Discount      2155 non-null   float64
 14  TotalSales    2155 non-null   float64
 15  FinalSales    2155 non-null   float64
dtypes: datetime64[ns](3), float64(5), object(8)
memory usage: 269.8+ KB
```

```
In [8]: df_cst.to_csv('../data/processed/CustomerFact.csv',
                      sep='|',
                      index=False)
```

Chapter 3

Product Analysis

```
In [9]: for col in [i for i in df_product.columns if 'ID' in i]:
            df_product[col] = df_product[col].astype(str)
```

```
In [10]: # Generate Year, month, date as separate file tipe
         df_product['OrderYear'] = df_product['OrderDate'].astype(str).str[0:4]
         df_product['OrderMonth'] = df_product['OrderDate'].astype(str).str[5:7]
         df_product['OrderDt'] = df_product['OrderDate'].astype(str).str[8:10]
         df_product['OrderYM'] = df_product['OrderDate'].astype(str).str[0:7]
```

```
In [11]: df_product['OrderYM'].value_counts().sort_index()
```

```
Out[11]: 1996-07     59
         1996-08     69
         1996-09     57
         1996-10     73
         1996-11     66
         1996-12     81
         1997-01     85
         1997-02     79
         1997-03     77
         1997-04     81
         1997-05     96
         1997-06     76
         1997-07     77
         1997-08     84
         1997-09     95
         1997-10    106
         1997-11     89
         1997-12    114
         1998-01    152
         1998-02    122
         1998-03    178
         1998-04    180
         1998-05     59
         Name: OrderYM, dtype: int64
```

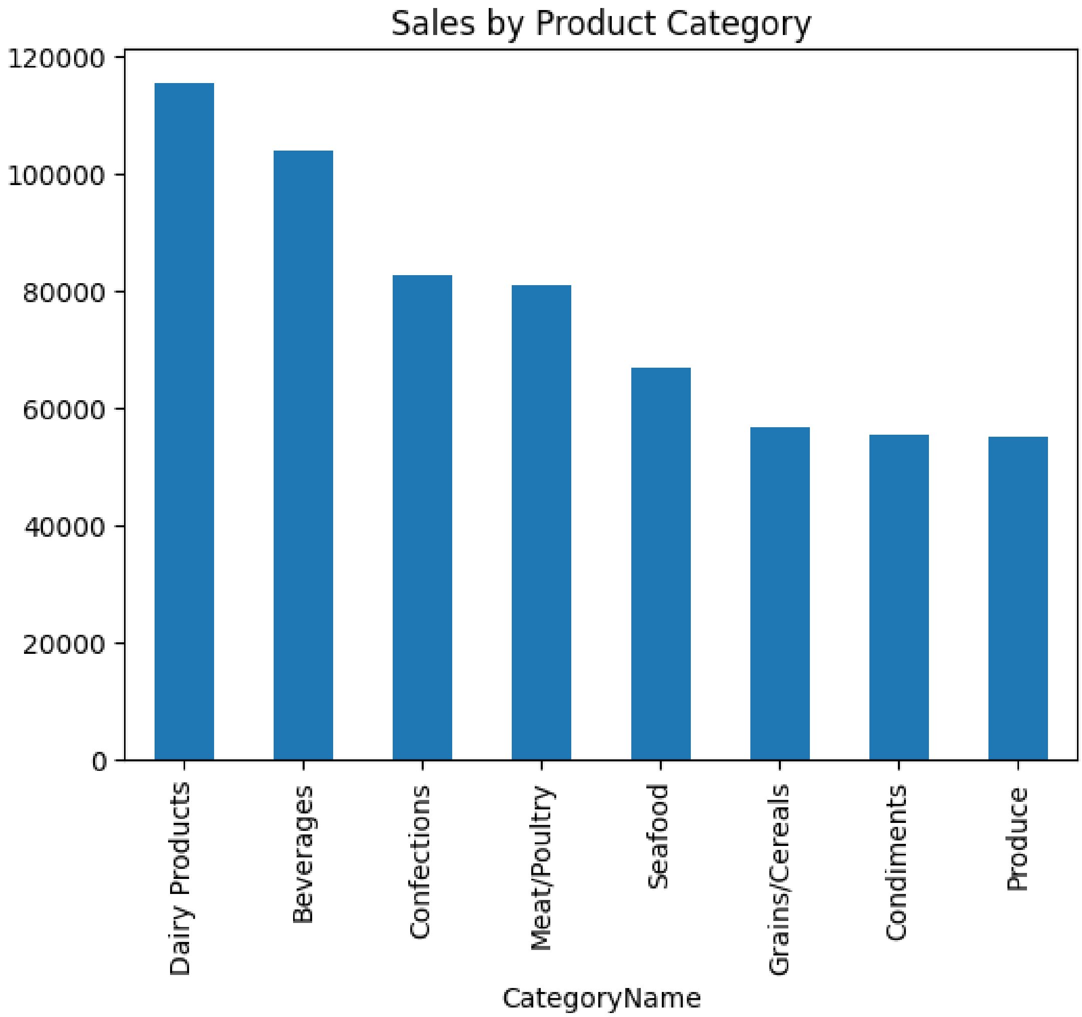
Findings:

Berdasarkan ketersediaan data, data yang memiliki history 1 tahun lengkap hanya data transaksi pada tahun 2017, dimana semua data memiliki bulan yang lengkap (dari Januari s.d Desember) sehingga analisis akan difokuskan pada transaksi-transaksi pada tahun 1997

```
In [12]: df_product = df_product[(df_product['OrderDate'] >= '1997-01-01') &
                                 (df_product['OrderDate'] < '1998-01-01')]
```

3.0.1 Where does the main revenue of our company come from in 1997?

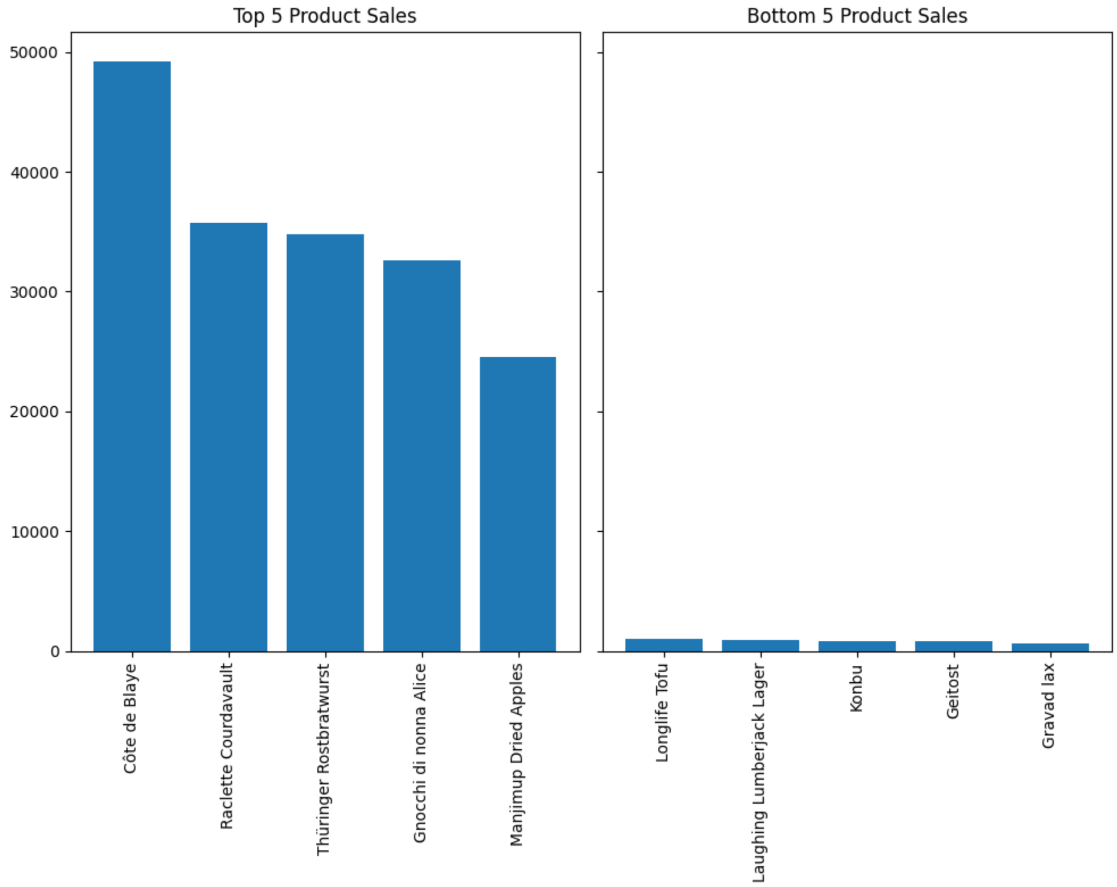
```
In [13]: df_product.groupby('CategoryName')['FinalSales'].sum().sort_values(ascending=False).plot(kind=
         plt.title("Sales by Product Category")

Out[13]: Text(0.5, 1.0, 'Sales by Product Category')
```



```
In [14]: fig, (ax1, ax2) = plt.subplots(1, 2, figsize=(10, 8), sharey = True)
         ax1.bar(x = df_product.groupby('ProductName')['FinalSales'].sum().sort_values(ascending=False)
                 height = df_product.groupby('ProductName')['FinalSales'].sum().sort_values(ascending=Fa
         ax1.set_title("Top 5 Product Sales")
         ax2.bar(x = df_product.groupby('ProductName')['FinalSales'].sum().sort_values(ascending=False)
                 height = df_product.groupby('ProductName')['FinalSales'].sum().sort_values(ascending=Fa
         ax2.set_title("Bottom 5 Product Sales")
         ax1.set_xticklabels(ax1.get_xticklabels(), rotation = 90)
```

```
        ax2.set_xticklabels(ax2.get_xticklabels(), rotation = 90)
        plt.tight_layout()
        plt.show()
```

```
C:\Users\asus\AppData\Local\Temp\ipykernel_73944\3737252104.py:8: UserWarning: FixedFormatter should onl
  ax1.set_xticklabels(ax1.get_xticklabels(), rotation = 90)
C:\Users\asus\AppData\Local\Temp\ipykernel_73944\3737252104.py:9: UserWarning: FixedFormatter should onl
  ax2.set_xticklabels(ax2.get_xticklabels(), rotation = 90)
```



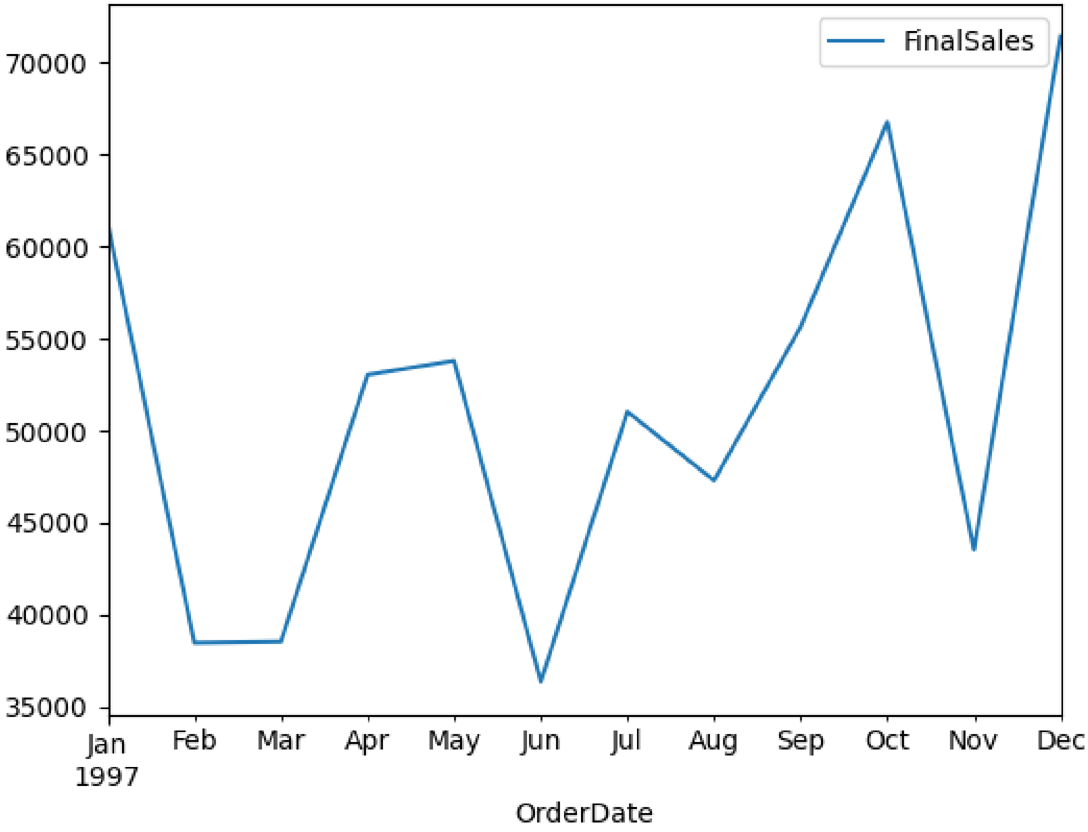
Findings :

- Top 3 penghasil revenue bagi perusahaan kita adalah beverages, dairy products, serta confections
- secara product revenue kita lebih banyak di generate oleh Caembert Pierrot, Alice Mutton, Carnarvon Tigers, Boston Crab Meat, dan Aniseed Syrup.
- Sementara itu product-product yang kurang terbeli adalah Uncle Bob's Organic Dried Pears, Wimmers gute Semmelknodel, Vegie-Spread, Tunnbrod, dan Valkoinen Suklaa
- Untuk meningkatkan penjualan produk-produk yang ada di top 5 terbawah penghasil revenue bisa dilakukan bundling ataupun promo agar penjualannya meningkat, pun jika penjualan produk tersebut tidak dapat ditingkatkan

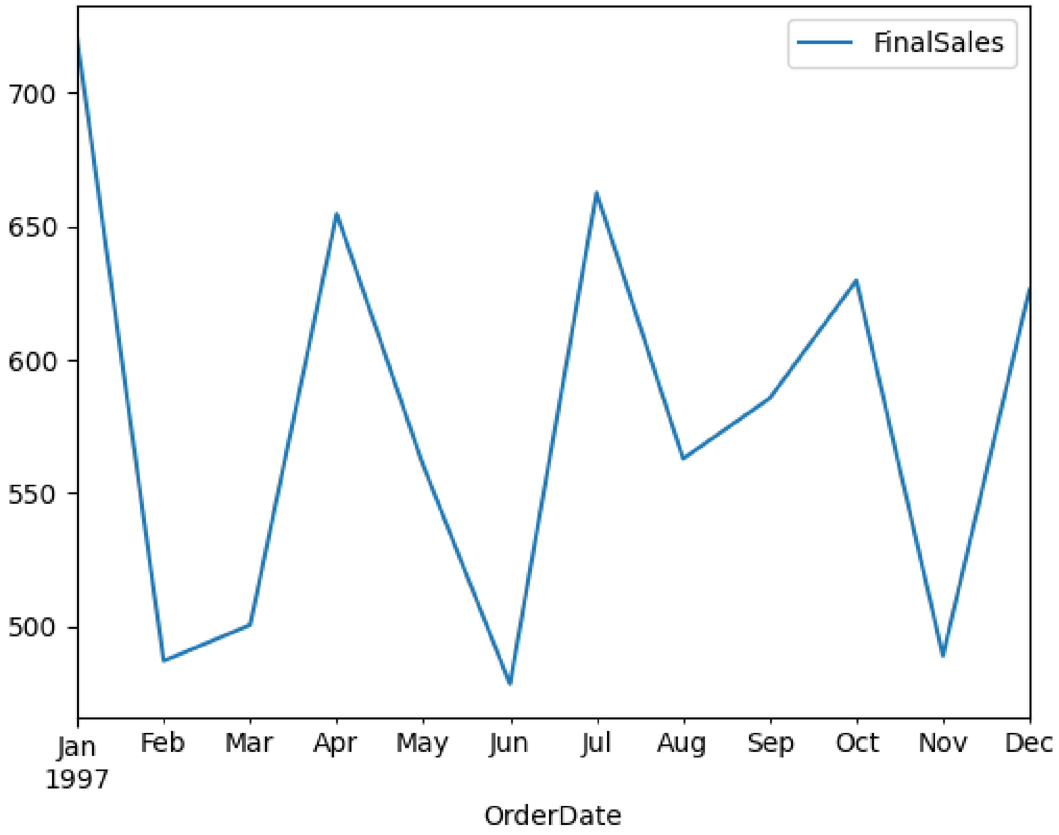
3.0.2 How about our monthly performance in 1997?

```
In [15]: df_product[['OrderDate', 'FinalSales']].set_index('OrderDate').resample('1M').sum().plot()
```

```
Out[15]: <AxesSubplot: xlabel='OrderDate'>
```

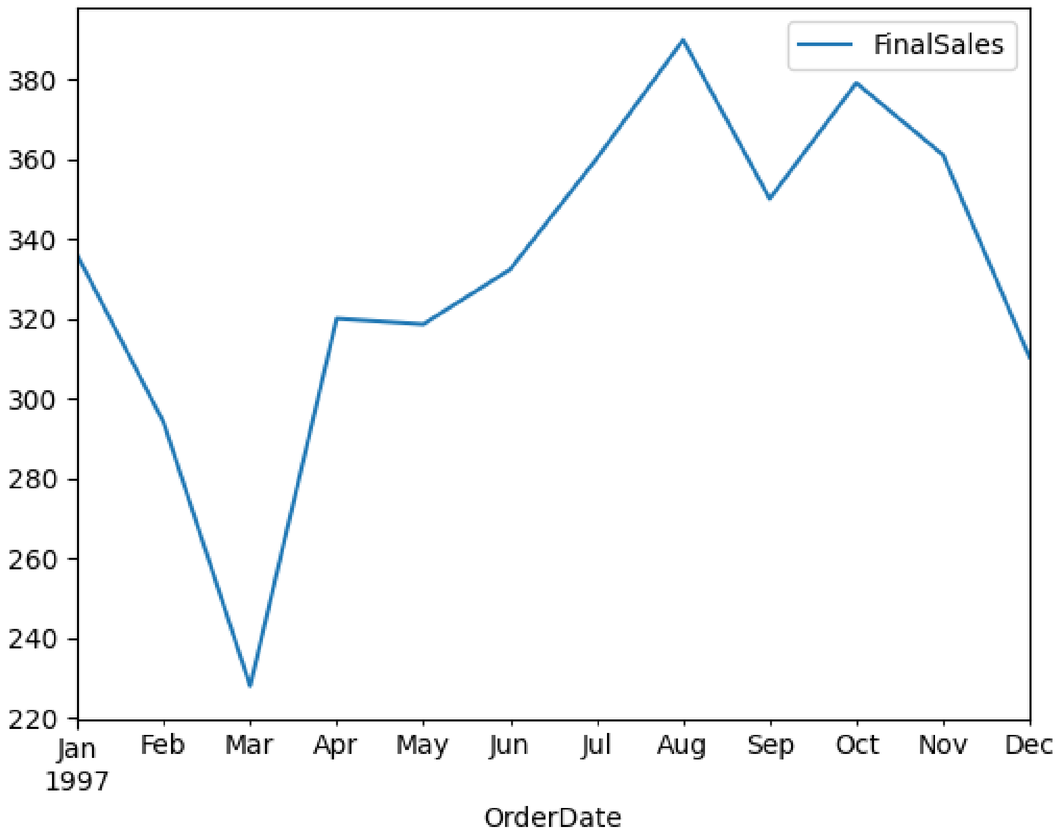


```
In [16]: df_product[['OrderDate', 'FinalSales']].set_index('OrderDate').resample('1M').mean().plot()

Out[16]: <AxesSubplot: xlabel='OrderDate'>
```



```
In [17]: df_product[['OrderDate', 'FinalSales']].set_index('OrderDate').resample('1M').median().plot()
```

```
Out[17]: <AxesSubplot: xlabel='OrderDate'>
```



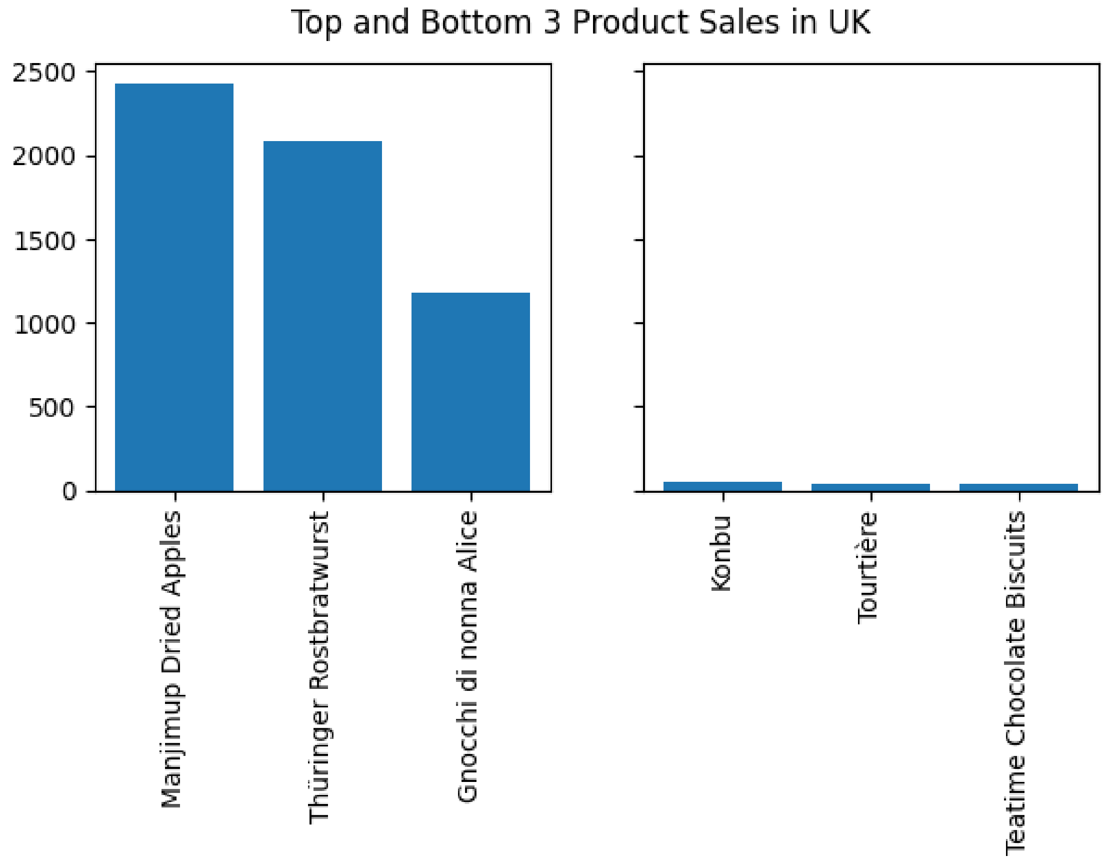
Findings :

Secara sales revenue kita cenderung mengalami sedikit peningkatan, namun basket size atau pembelian rata-rata setiap orang sepertinya stagnan dari 500 s.d 600 USD per customer

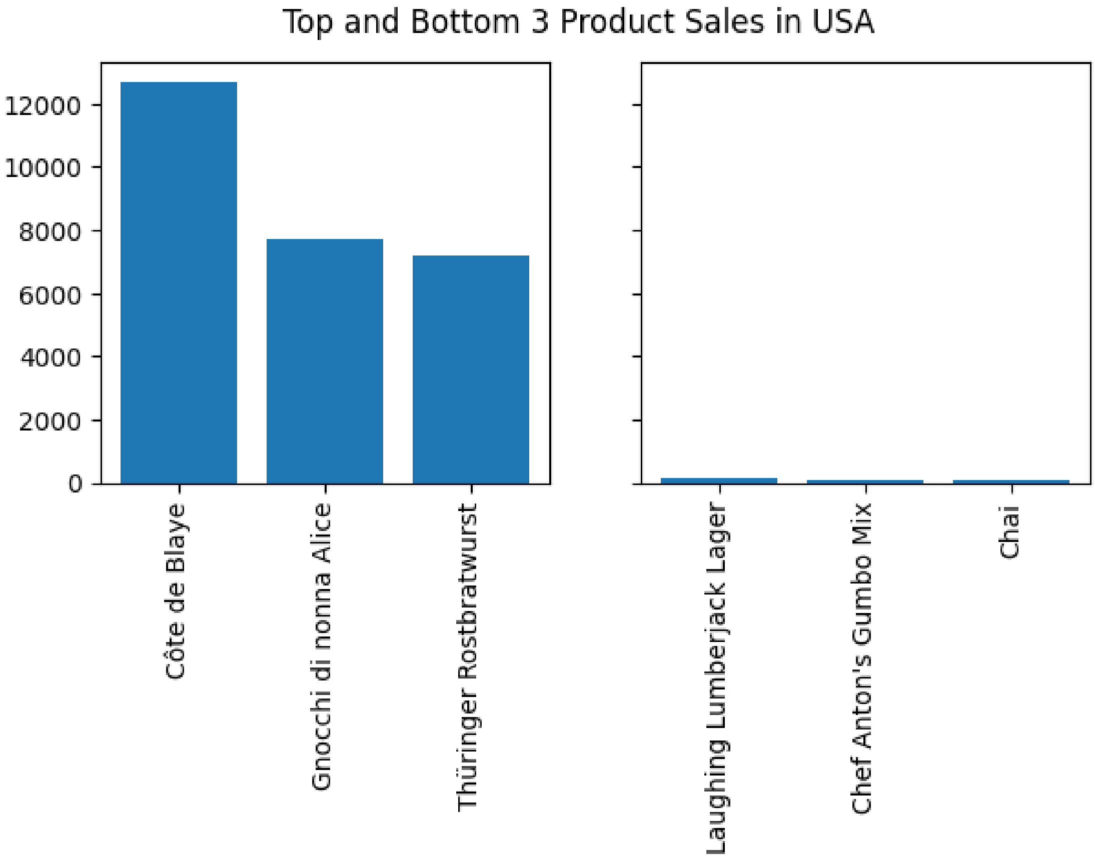
3.0.3 Apakah setiap negara memiliki preferensi yang sama terhadap produk?

```
In [18]: for i in df_product.ShipCountry.unique():
               fig, (ax1, ax2) = plt.subplots(1, 2, sharey=True, figsize = (7,3))
               ax1.bar(x = df_product[df_product.ShipCountry == i].groupby('ProductName')['FinalSales']
                       height = df_product[df_product.ShipCountry == i].groupby('ProductName')['FinalSal
               ax1.set_xticklabels(ax1.get_xticklabels(), rotation = 90)
               ax2.bar(x = df_product[df_product.ShipCountry == i].groupby('ProductName')['FinalSales']
                       height = df_product[df_product.ShipCountry == i].groupby('ProductName')['FinalSal
               ax2.set_xticklabels(ax2.get_xticklabels(), rotation = 90)
               plt.suptitle(f'Top and Bottom 3 Product Sales in {i}')
               plt.show()
               plt.clf()
```

```
C:\Users\asus\AppData\Local\Temp\ipykernel_73944\1843367482.py:5: UserWarning: FixedFormatter should onl
  ax1.set_xticklabels(ax1.get_xticklabels(), rotation = 90)
C:\Users\asus\AppData\Local\Temp\ipykernel_73944\1843367482.py:8: UserWarning: FixedFormatter should onl
  ax2.set_xticklabels(ax2.get_xticklabels(), rotation = 90)
```

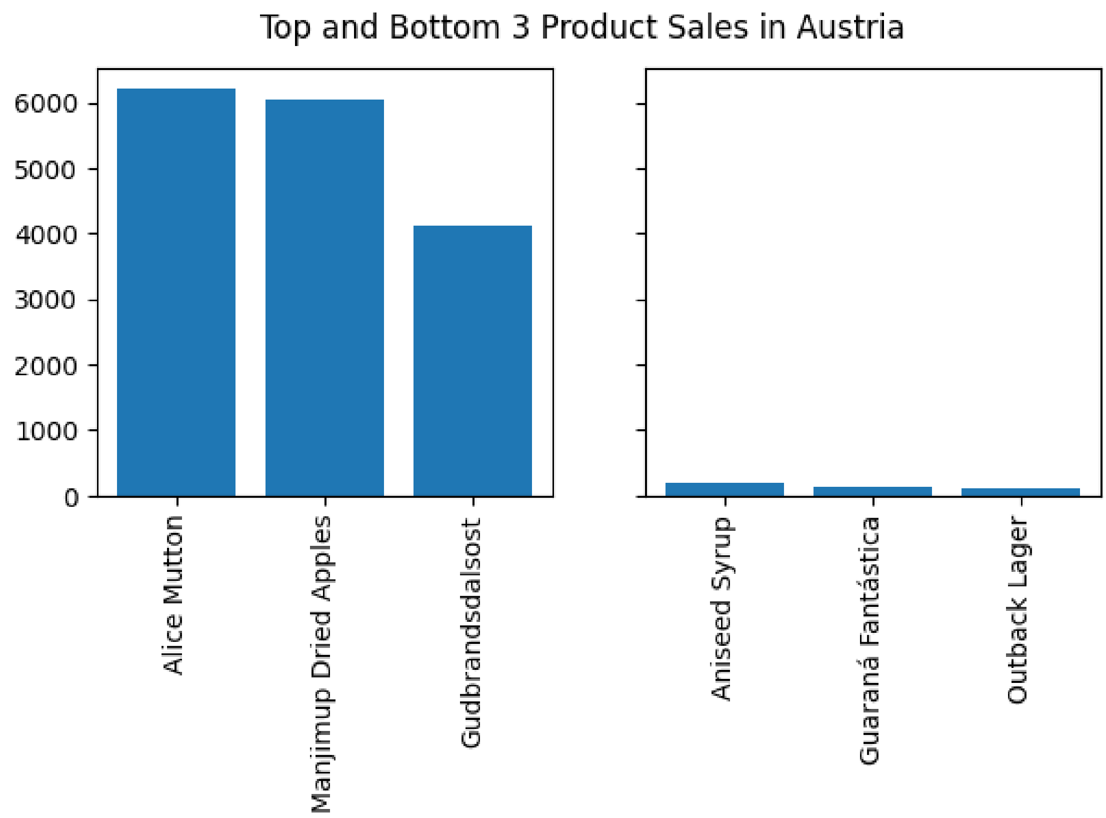


```
C:\Users\asus\AppData\Local\Temp\ipykernel_73944\1843367482.py:5: UserWarning: FixedFormatter should onl
  ax1.set_xticklabels(ax1.get_xticklabels(), rotation = 90)
C:\Users\asus\AppData\Local\Temp\ipykernel_73944\1843367482.py:8: UserWarning: FixedFormatter should onl
  ax2.set_xticklabels(ax2.get_xticklabels(), rotation = 90)


<Figure size 640x480 with 0 Axes>
```



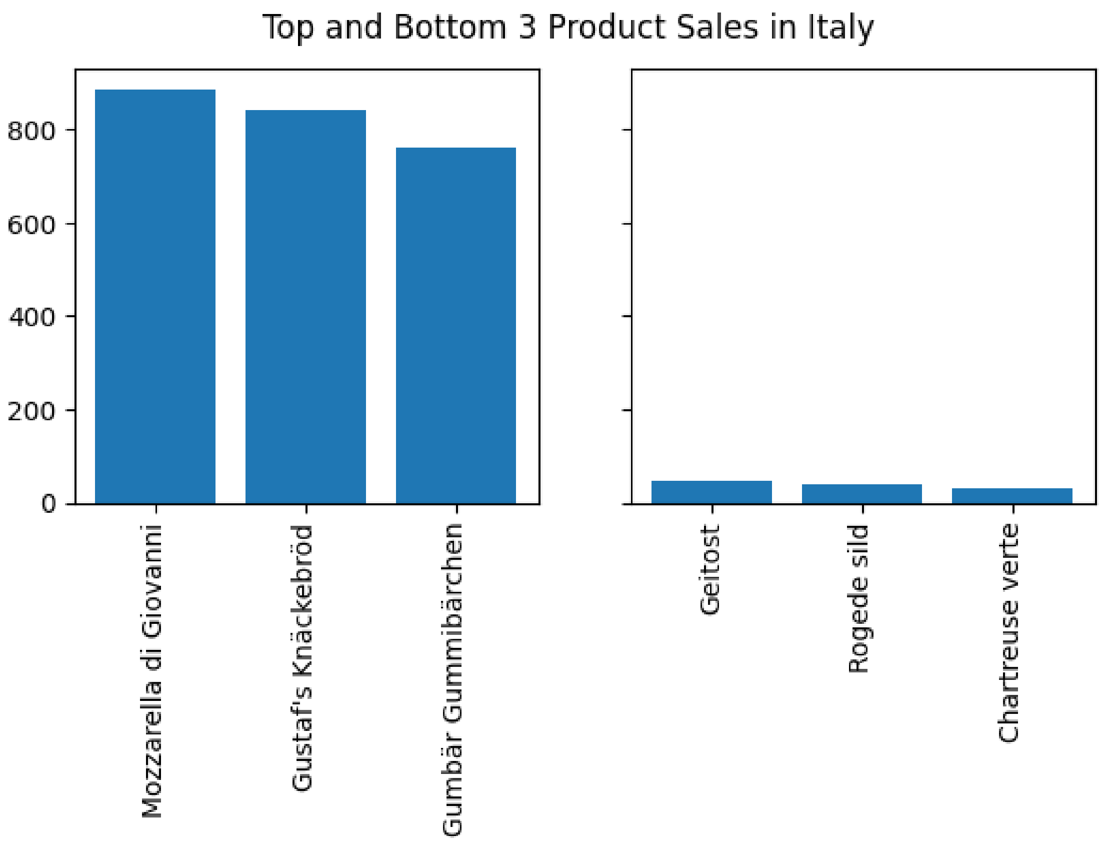
```
C:\Users\asus\AppData\Local\Temp\ipykernel_73944\1843367482.py:5: UserWarning: FixedFormatter should onl
  ax1.set_xticklabels(ax1.get_xticklabels(), rotation = 90)
C:\Users\asus\AppData\Local\Temp\ipykernel_73944\1843367482.py:8: UserWarning: FixedFormatter should onl
  ax2.set_xticklabels(ax2.get_xticklabels(), rotation = 90)
```

```
<Figure size 640x480 with 0 Axes>
```



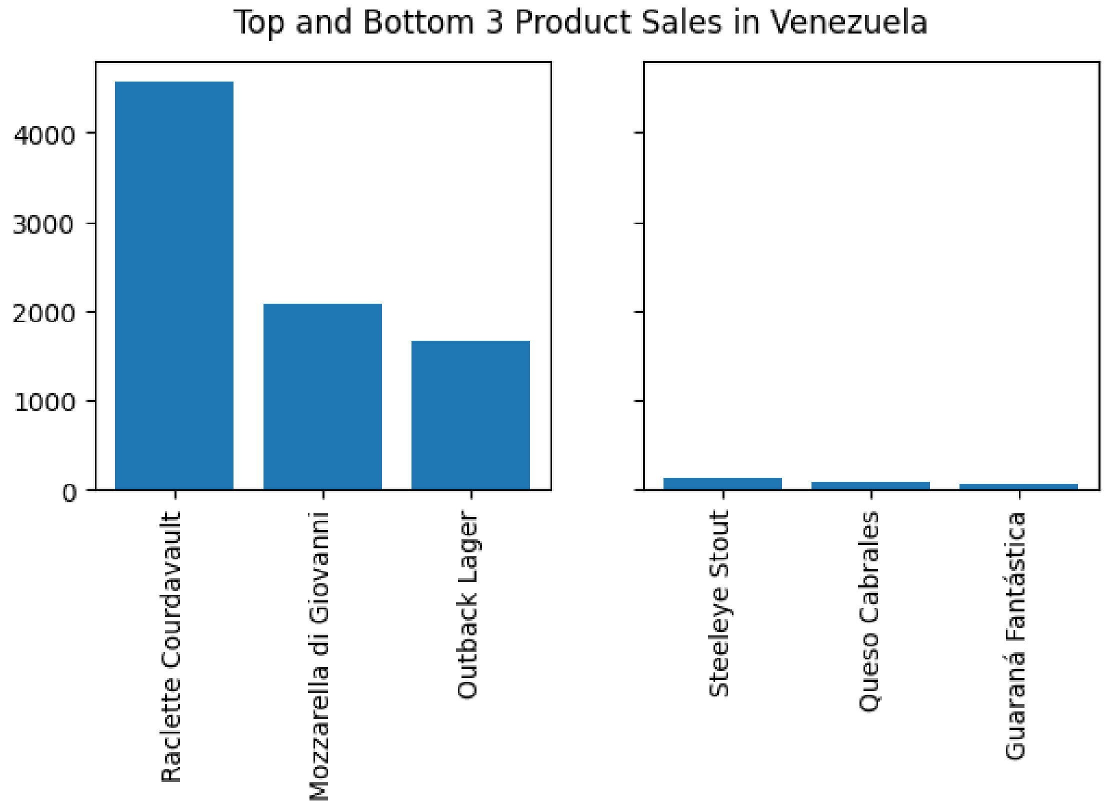
```
C:\Users\asus\AppData\Local\Temp\ipykernel_73944\1843367482.py:5: UserWarning: FixedFormatter should onl
  ax1.set_xticklabels(ax1.get_xticklabels(), rotation = 90)
C:\Users\asus\AppData\Local\Temp\ipykernel_73944\1843367482.py:8: UserWarning: FixedFormatter should onl
  ax2.set_xticklabels(ax2.get_xticklabels(), rotation = 90)
```

```
<Figure size 640x480 with 0 Axes>
```



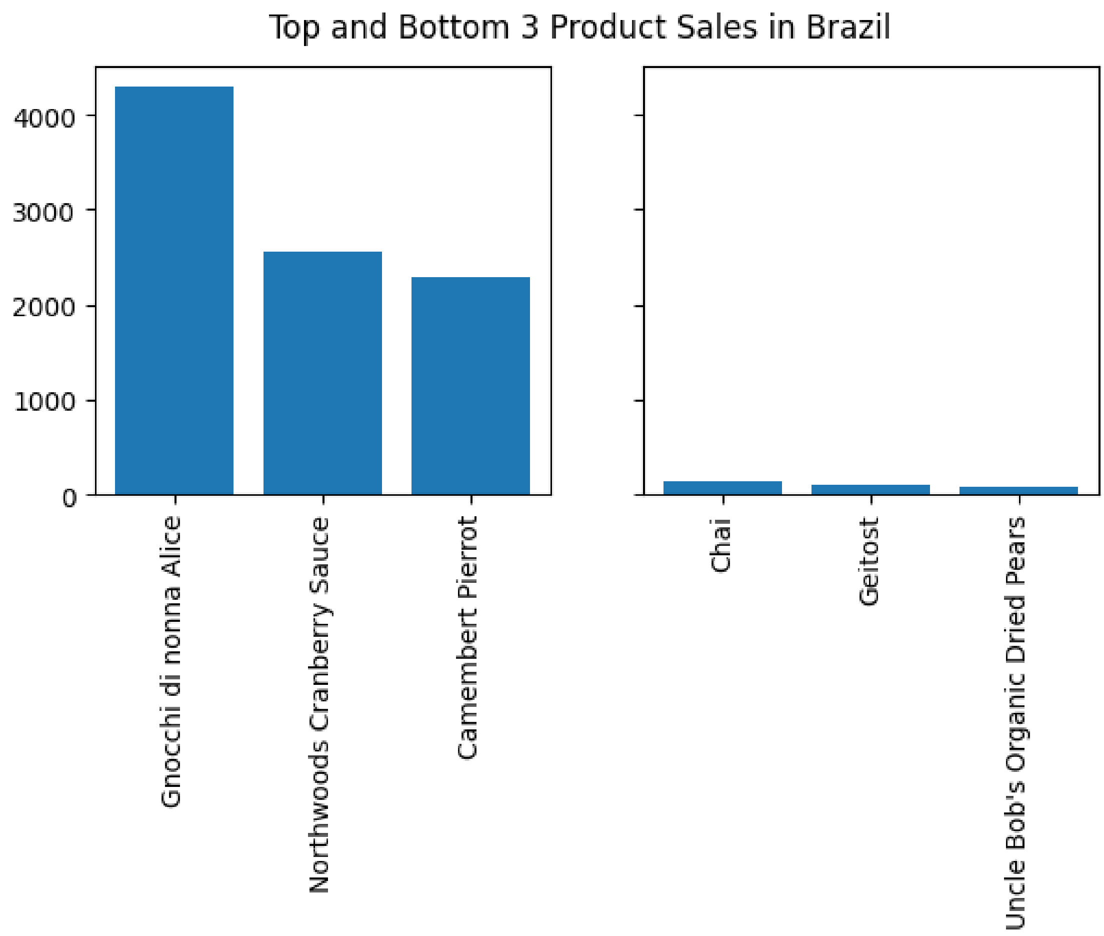
```
C:\Users\asus\AppData\Local\Temp\ipykernel_73944\1843367482.py:5: UserWarning: FixedFormatter should onl
  ax1.set_xticklabels(ax1.get_xticklabels(), rotation = 90)
C:\Users\asus\AppData\Local\Temp\ipykernel_73944\1843367482.py:8: UserWarning: FixedFormatter should onl
  ax2.set_xticklabels(ax2.get_xticklabels(), rotation = 90)
```

```
<Figure size 640x480 with 0 Axes>
```



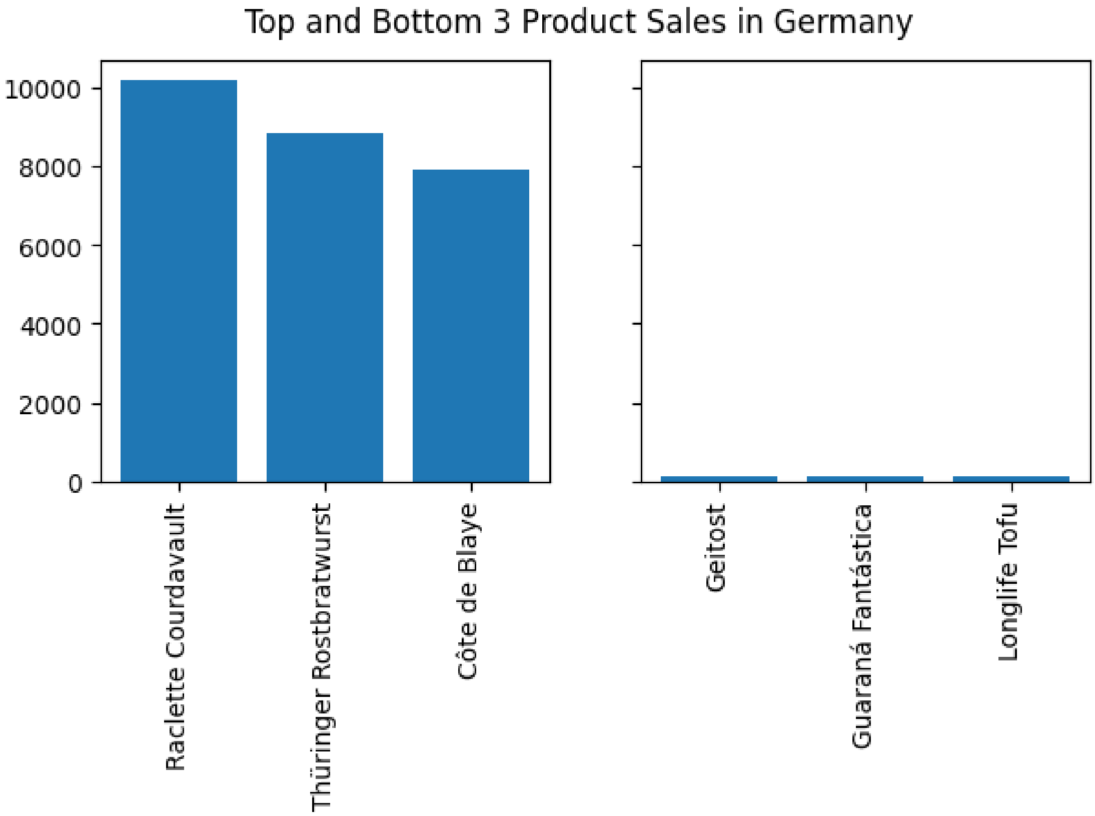
```
C:\Users\asus\AppData\Local\Temp\ipykernel_73944\1843367482.py:5: UserWarning: FixedFormatter should onl
  ax1.set_xticklabels(ax1.get_xticklabels(), rotation = 90)
C:\Users\asus\AppData\Local\Temp\ipykernel_73944\1843367482.py:8: UserWarning: FixedFormatter should onl
  ax2.set_xticklabels(ax2.get_xticklabels(), rotation = 90)
```

```
<Figure size 640x480 with 0 Axes>
```



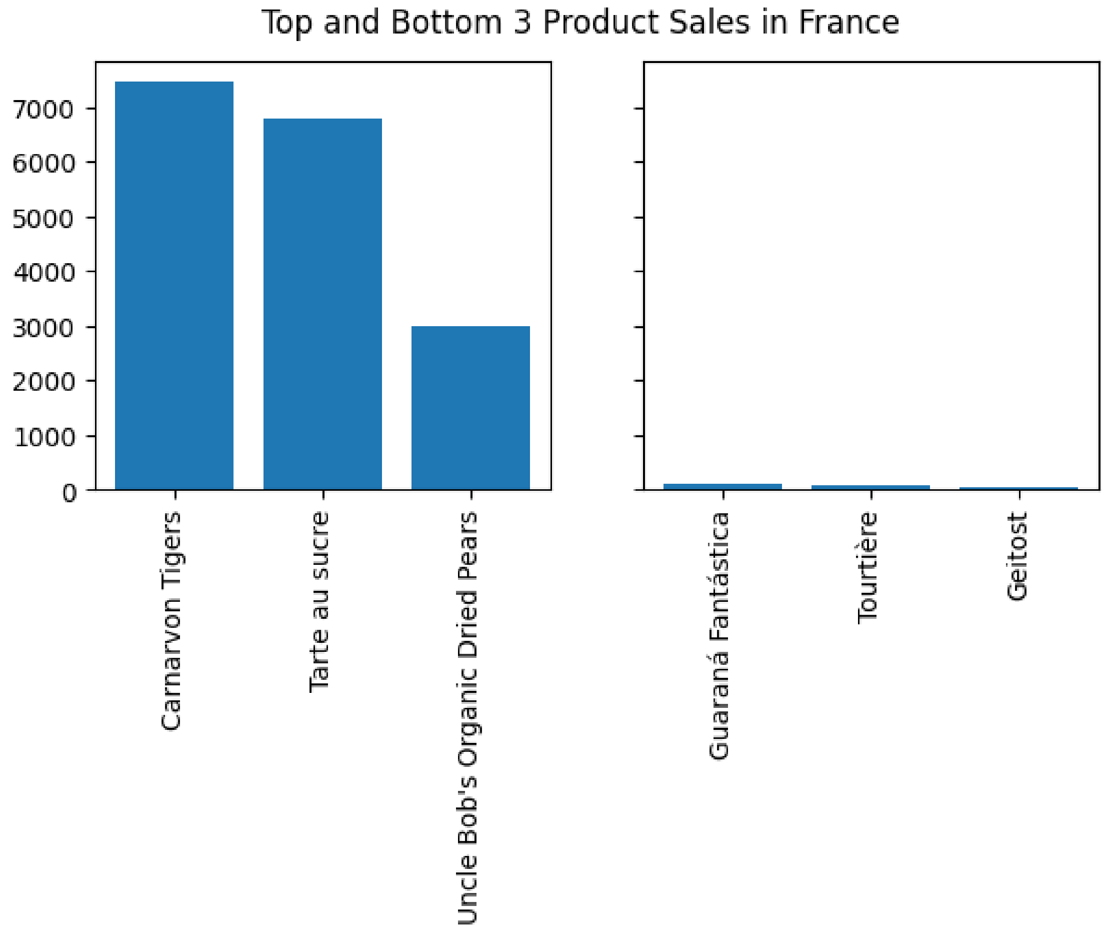
```
C:\Users\asus\AppData\Local\Temp\ipykernel_73944\1843367482.py:5: UserWarning: FixedFormatter should onl
  ax1.set_xticklabels(ax1.get_xticklabels(), rotation = 90)
C:\Users\asus\AppData\Local\Temp\ipykernel_73944\1843367482.py:8: UserWarning: FixedFormatter should onl
  ax2.set_xticklabels(ax2.get_xticklabels(), rotation = 90)
```

```
<Figure size 640x480 with 0 Axes>
```



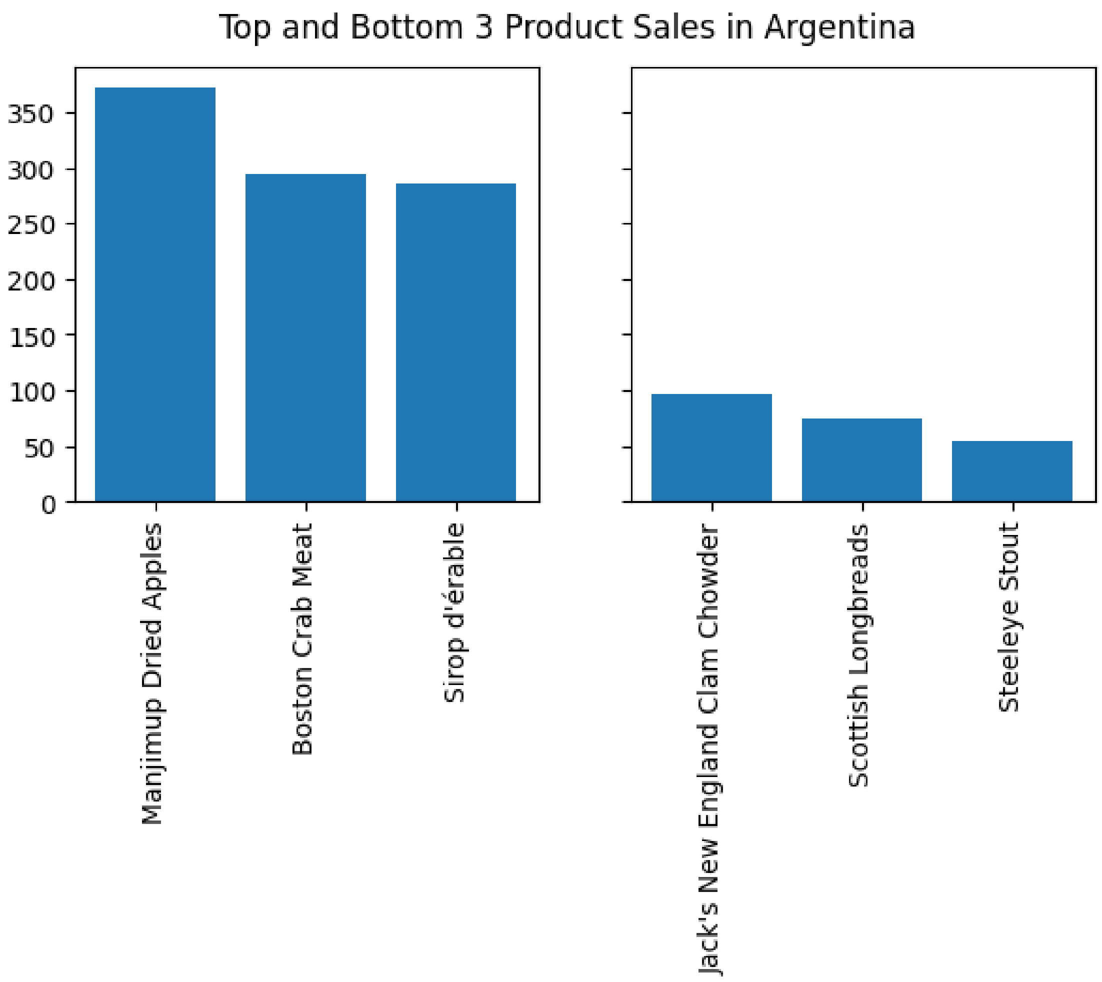
```
C:\Users\asus\AppData\Local\Temp\ipykernel_73944\1843367482.py:5: UserWarning: FixedFormatter should onl
  ax1.set_xticklabels(ax1.get_xticklabels(), rotation = 90)
C:\Users\asus\AppData\Local\Temp\ipykernel_73944\1843367482.py:8: UserWarning: FixedFormatter should onl
  ax2.set_xticklabels(ax2.get_xticklabels(), rotation = 90)
```

```
<Figure size 640x480 with 0 Axes>
```

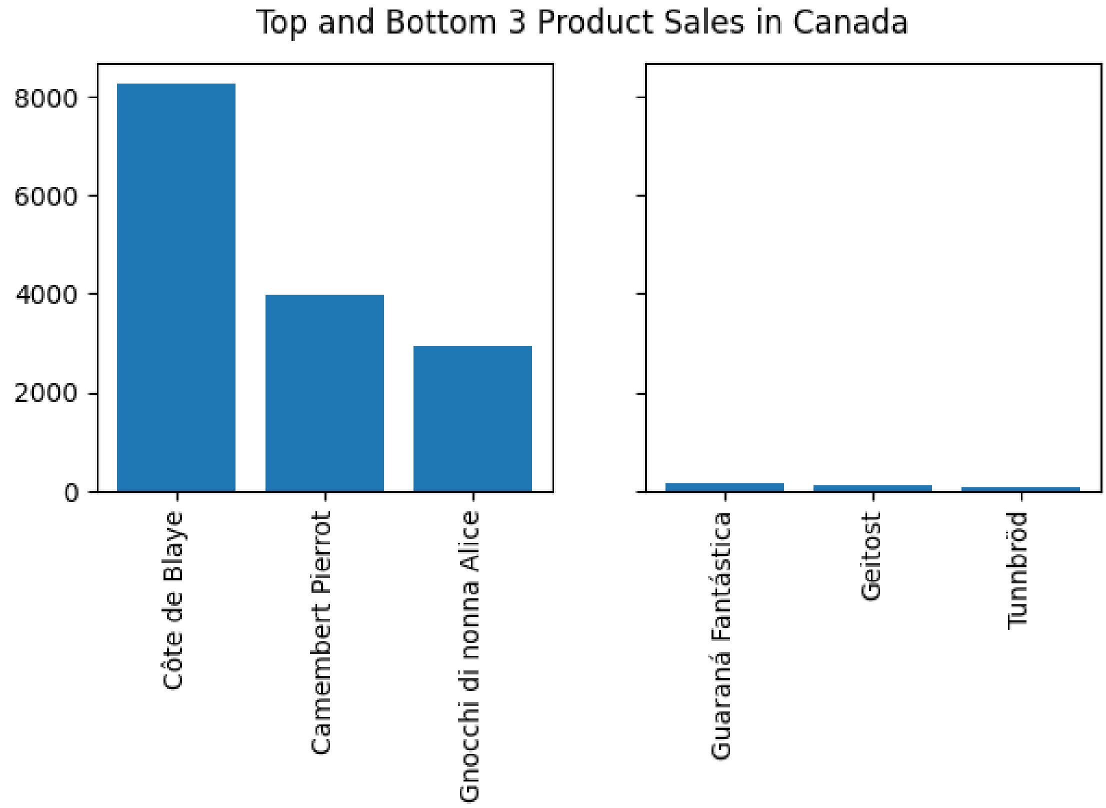


```
C:\Users\asus\AppData\Local\Temp\ipykernel_73944\1843367482.py:5: UserWarning: FixedFormatter should onl
  ax1.set_xticklabels(ax1.get_xticklabels(), rotation = 90)
C:\Users\asus\AppData\Local\Temp\ipykernel_73944\1843367482.py:8: UserWarning: FixedFormatter should onl
  ax2.set_xticklabels(ax2.get_xticklabels(), rotation = 90)
```

```
<Figure size 640x480 with 0 Axes>
```

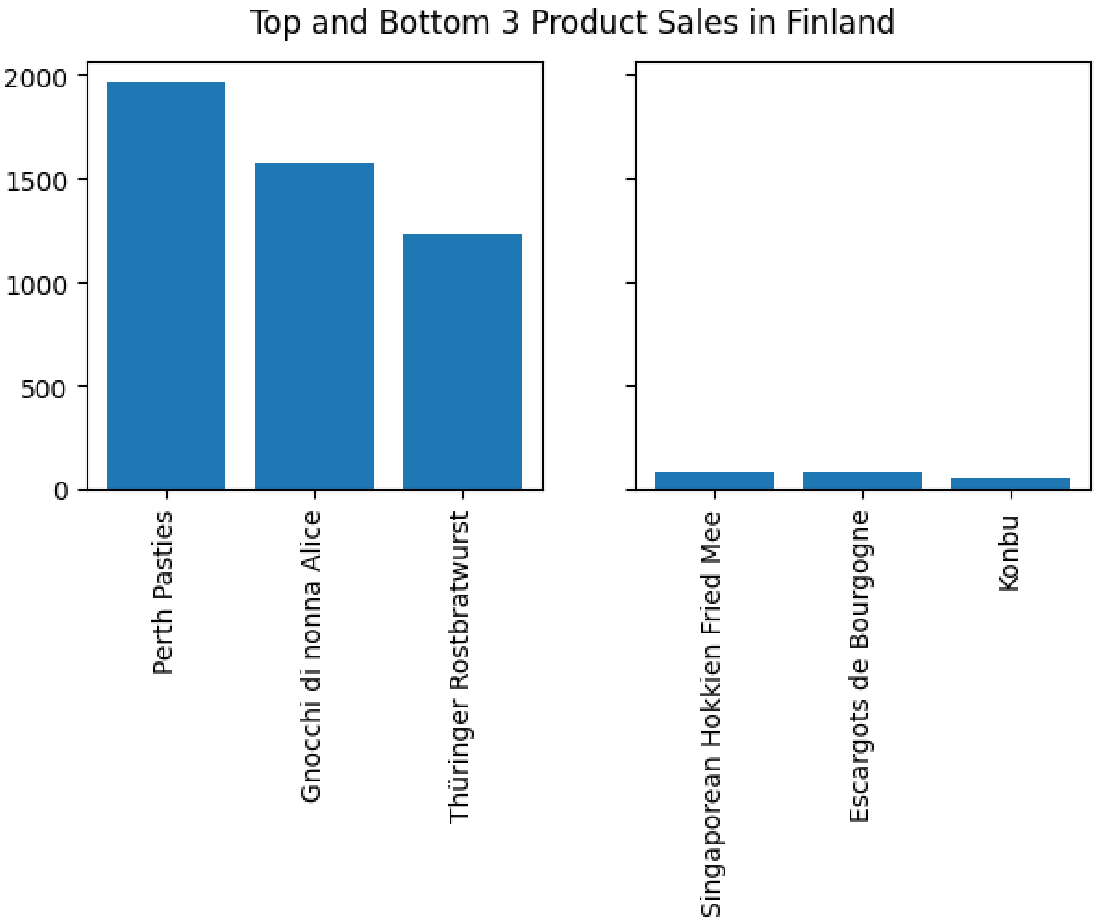


```
C:\Users\asus\AppData\Local\Temp\ipykernel_73944\1843367482.py:5: UserWarning: FixedFormatter should onl
  ax1.set_xticklabels(ax1.get_xticklabels(), rotation = 90)
C:\Users\asus\AppData\Local\Temp\ipykernel_73944\1843367482.py:8: UserWarning: FixedFormatter should onl
  ax2.set_xticklabels(ax2.get_xticklabels(), rotation = 90)


<Figure size 640x480 with 0 Axes>
```



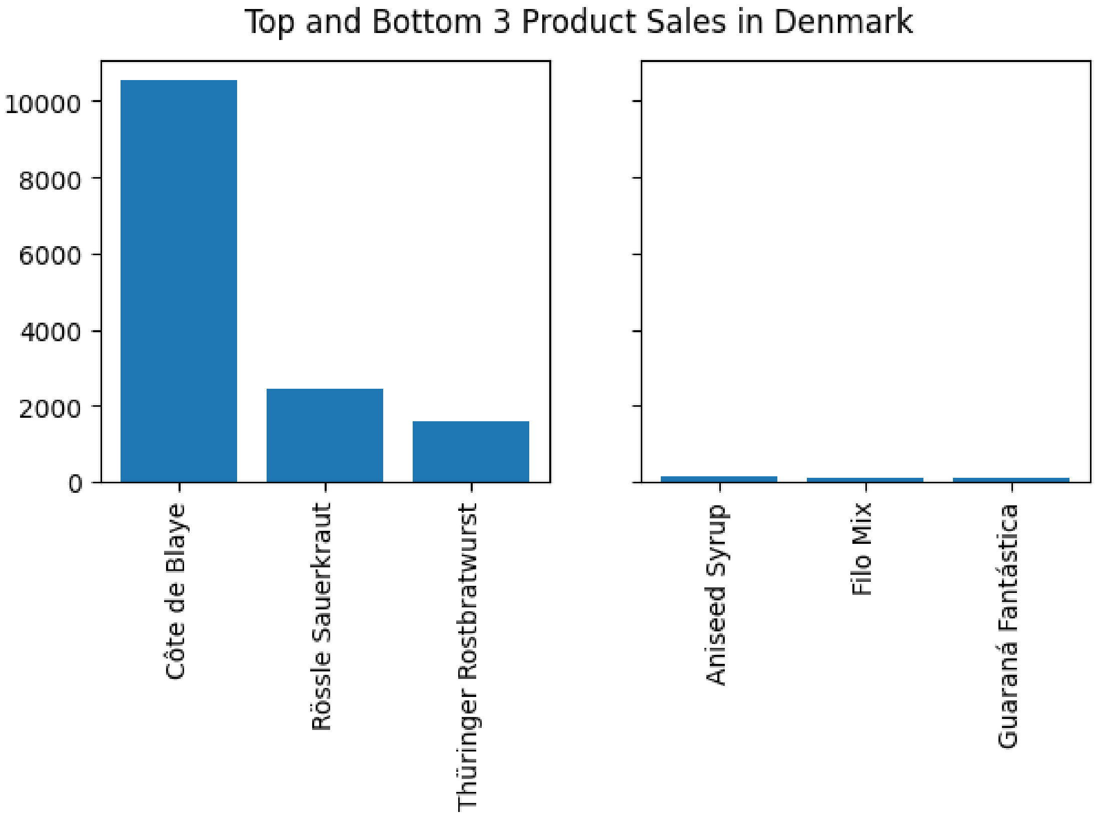
```
C:\Users\asus\AppData\Local\Temp\ipykernel_73944\1843367482.py:5: UserWarning: FixedFormatter should onl
  ax1.set_xticklabels(ax1.get_xticklabels(), rotation = 90)
C:\Users\asus\AppData\Local\Temp\ipykernel_73944\1843367482.py:8: UserWarning: FixedFormatter should onl
  ax2.set_xticklabels(ax2.get_xticklabels(), rotation = 90)
```

```
<Figure size 640x480 with 0 Axes>
```



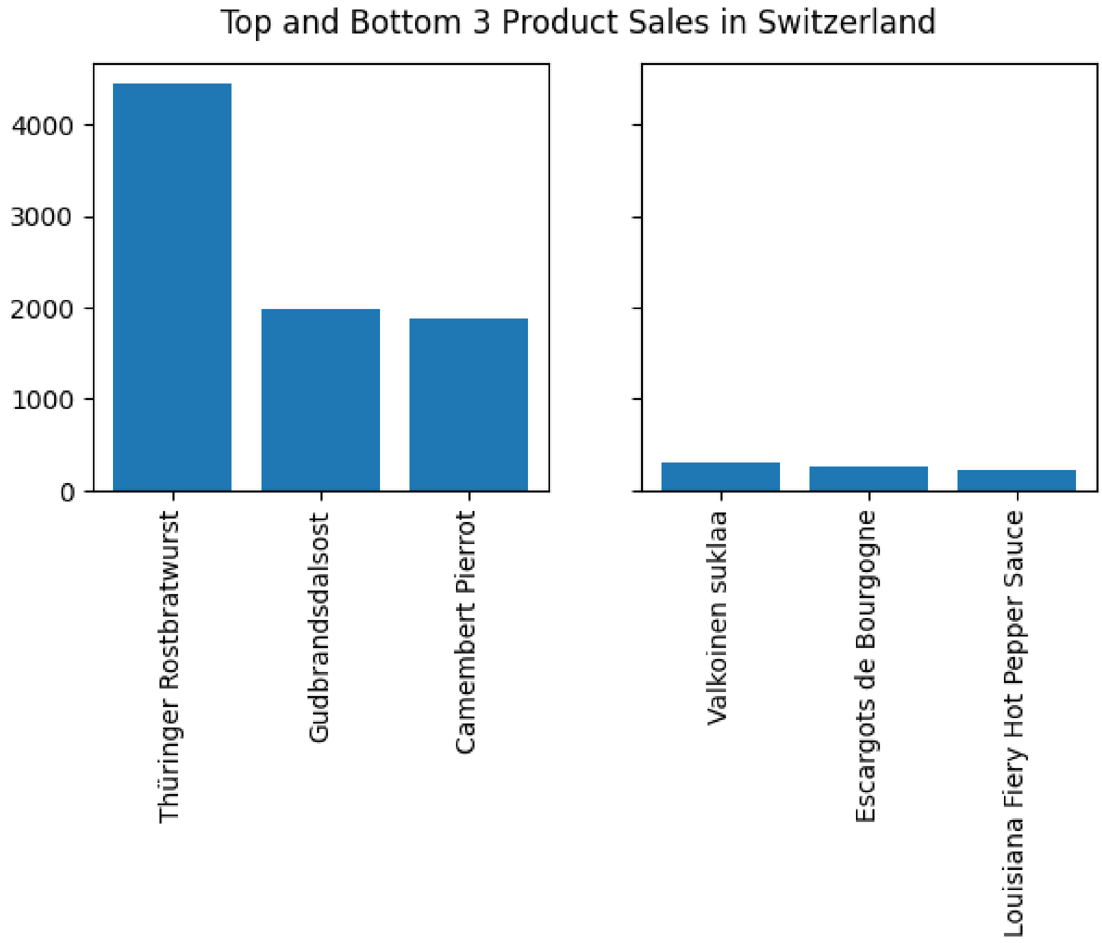
```
C:\Users\asus\AppData\Local\Temp\ipykernel_73944\1843367482.py:5: UserWarning: FixedFormatter should onl
  ax1.set_xticklabels(ax1.get_xticklabels(), rotation = 90)
C:\Users\asus\AppData\Local\Temp\ipykernel_73944\1843367482.py:8: UserWarning: FixedFormatter should onl
  ax2.set_xticklabels(ax2.get_xticklabels(), rotation = 90)
```

```
<Figure size 640x480 with 0 Axes>
```

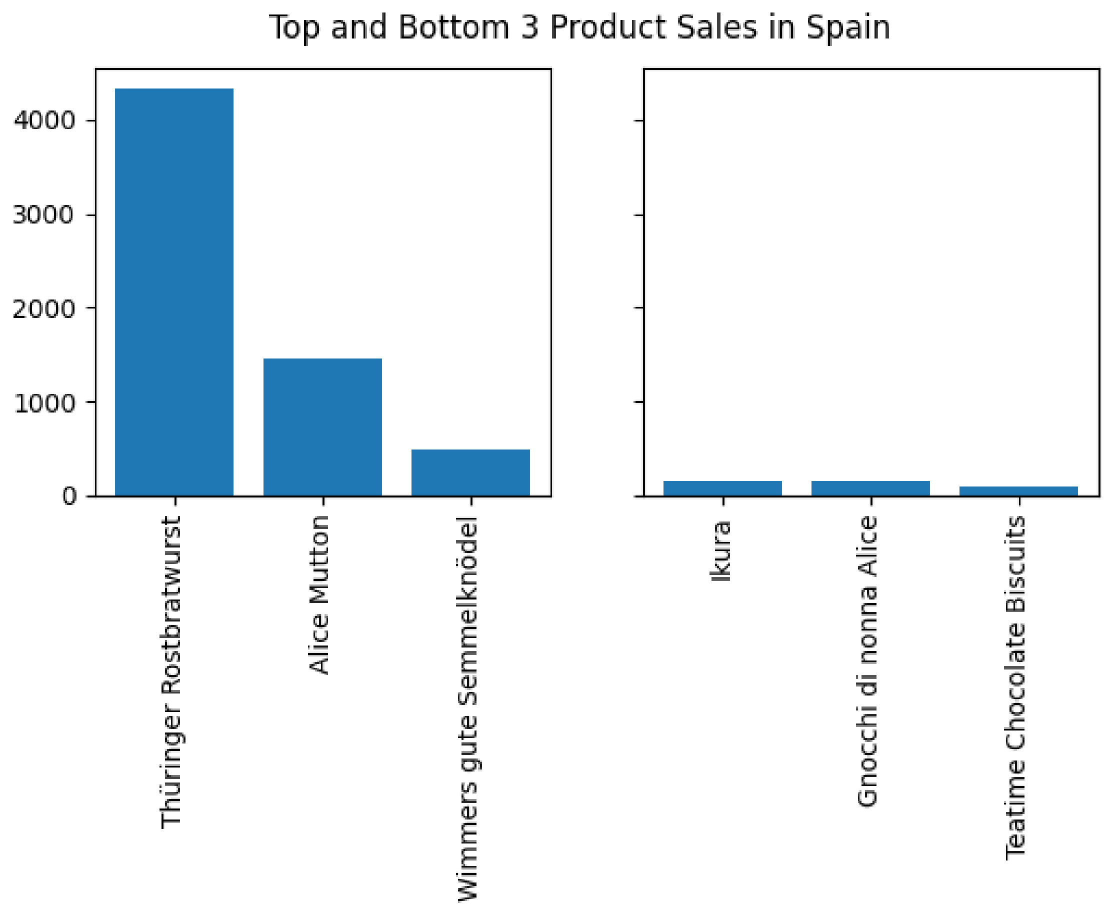


```
C:\Users\asus\AppData\Local\Temp\ipykernel_73944\1843367482.py:5: UserWarning: FixedFormatter should onl
  ax1.set_xticklabels(ax1.get_xticklabels(), rotation = 90)
C:\Users\asus\AppData\Local\Temp\ipykernel_73944\1843367482.py:8: UserWarning: FixedFormatter should onl
  ax2.set_xticklabels(ax2.get_xticklabels(), rotation = 90)
```

```
<Figure size 640x480 with 0 Axes>
```

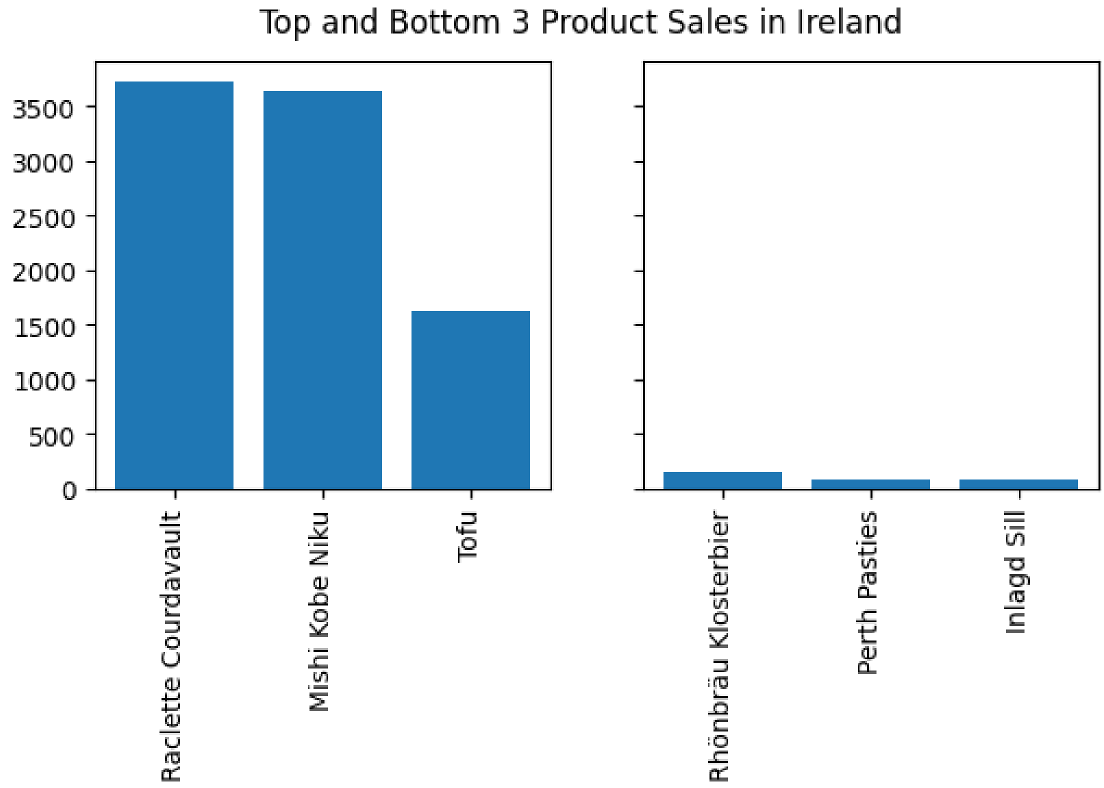


```
C:\Users\asus\AppData\Local\Temp\ipykernel_73944\1843367482.py:5: UserWarning: FixedFormatter should onl
  ax1.set_xticklabels(ax1.get_xticklabels(), rotation = 90)
C:\Users\asus\AppData\Local\Temp\ipykernel_73944\1843367482.py:8: UserWarning: FixedFormatter should onl
  ax2.set_xticklabels(ax2.get_xticklabels(), rotation = 90)


<Figure size 640x480 with 0 Axes>
```



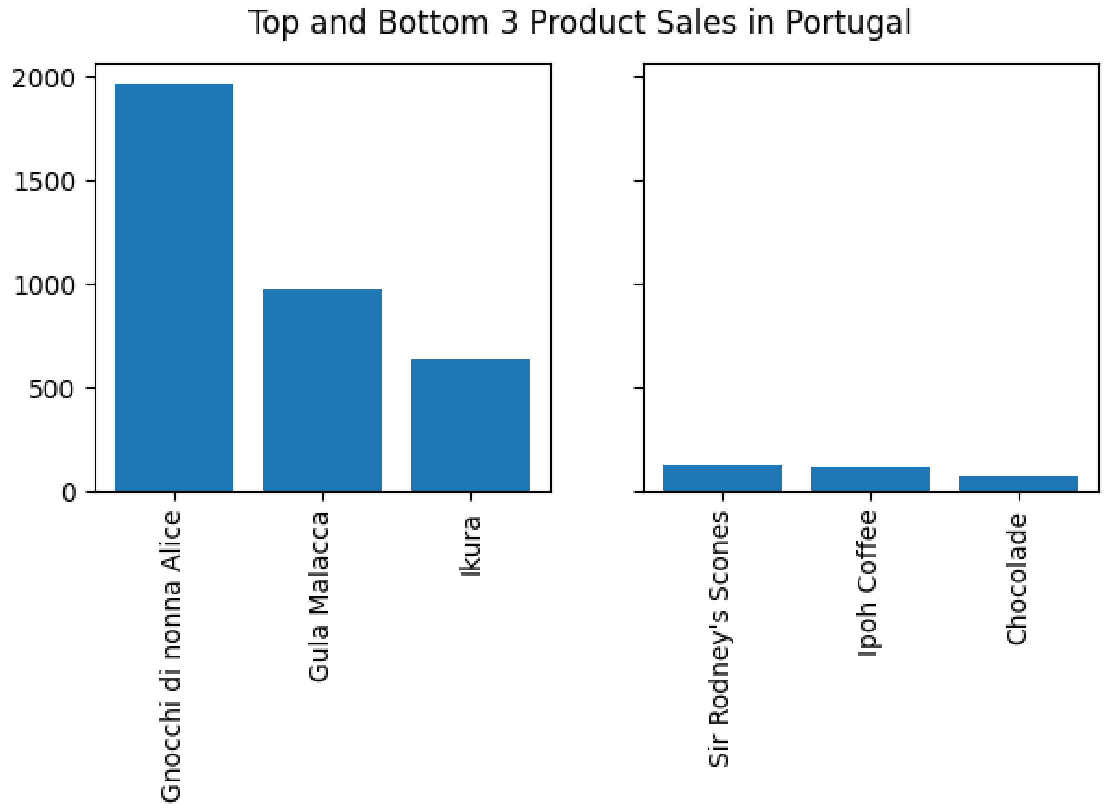
```
C:\Users\asus\AppData\Local\Temp\ipykernel_73944\1843367482.py:5: UserWarning: FixedFormatter should onl
  ax1.set_xticklabels(ax1.get_xticklabels(), rotation = 90)
C:\Users\asus\AppData\Local\Temp\ipykernel_73944\1843367482.py:8: UserWarning: FixedFormatter should onl
  ax2.set_xticklabels(ax2.get_xticklabels(), rotation = 90)
```

```
<Figure size 640x480 with 0 Axes>
```



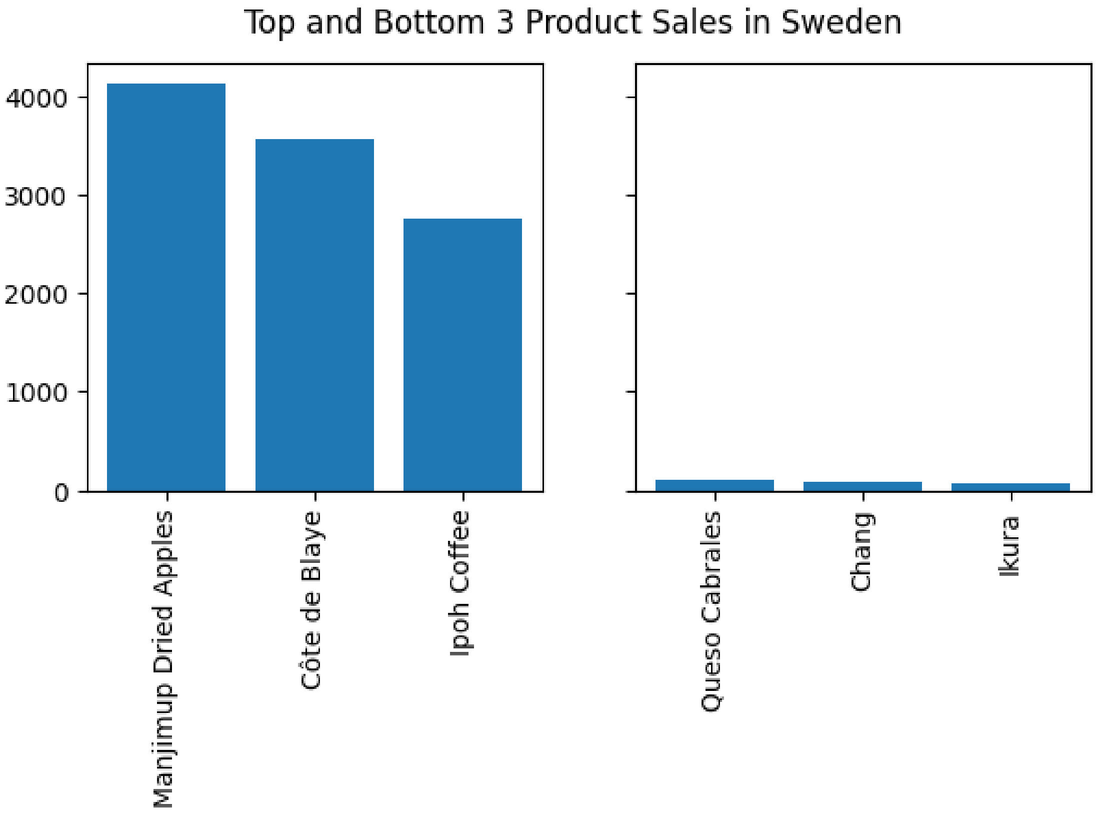
```
C:\Users\asus\AppData\Local\Temp\ipykernel_73944\1843367482.py:5: UserWarning: FixedFormatter should onl
  ax1.set_xticklabels(ax1.get_xticklabels(), rotation = 90)
C:\Users\asus\AppData\Local\Temp\ipykernel_73944\1843367482.py:8: UserWarning: FixedFormatter should onl
  ax2.set_xticklabels(ax2.get_xticklabels(), rotation = 90)
```

```
<Figure size 640x480 with 0 Axes>
```



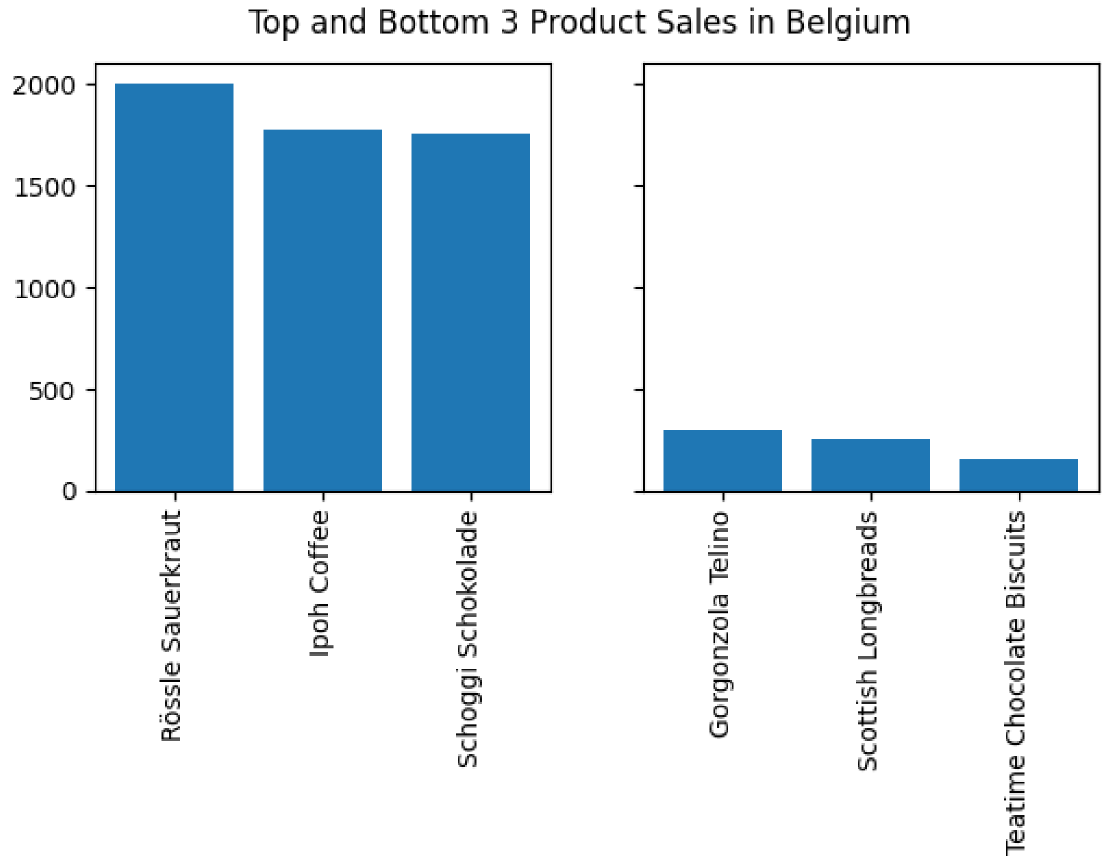
```
C:\Users\asus\AppData\Local\Temp\ipykernel_73944\1843367482.py:5: UserWarning: FixedFormatter should onl
  ax1.set_xticklabels(ax1.get_xticklabels(), rotation = 90)
C:\Users\asus\AppData\Local\Temp\ipykernel_73944\1843367482.py:8: UserWarning: FixedFormatter should onl
  ax2.set_xticklabels(ax2.get_xticklabels(), rotation = 90)
```

```
<Figure size 640x480 with 0 Axes>
```



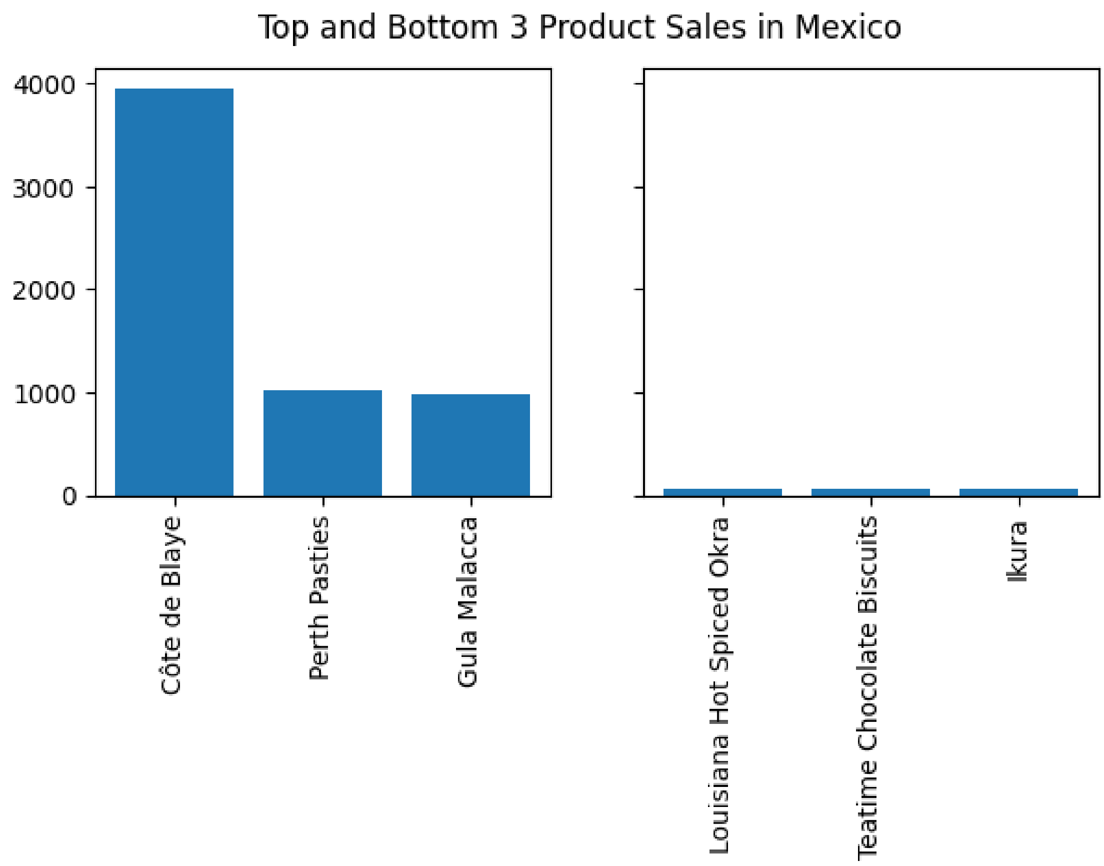
```
C:\Users\asus\AppData\Local\Temp\ipykernel_73944\1843367482.py:5: UserWarning: FixedFormatter should onl
  ax1.set_xticklabels(ax1.get_xticklabels(), rotation = 90)
C:\Users\asus\AppData\Local\Temp\ipykernel_73944\1843367482.py:8: UserWarning: FixedFormatter should onl
  ax2.set_xticklabels(ax2.get_xticklabels(), rotation = 90)
```

```
<Figure size 640x480 with 0 Axes>
```

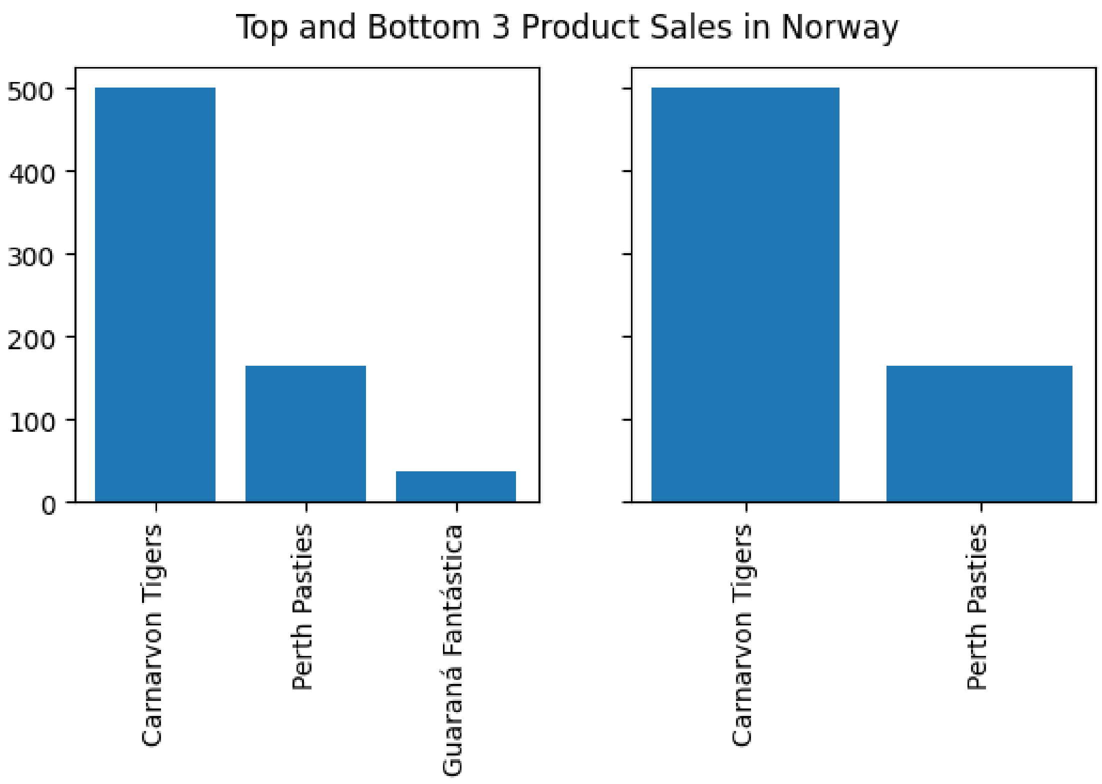


```
C:\Users\asus\AppData\Local\Temp\ipykernel_73944\1843367482.py:5: UserWarning: FixedFormatter should onl
  ax1.set_xticklabels(ax1.get_xticklabels(), rotation = 90)
C:\Users\asus\AppData\Local\Temp\ipykernel_73944\1843367482.py:8: UserWarning: FixedFormatter should onl
  ax2.set_xticklabels(ax2.get_xticklabels(), rotation = 90)
```

```
<Figure size 640x480 with 0 Axes>
```

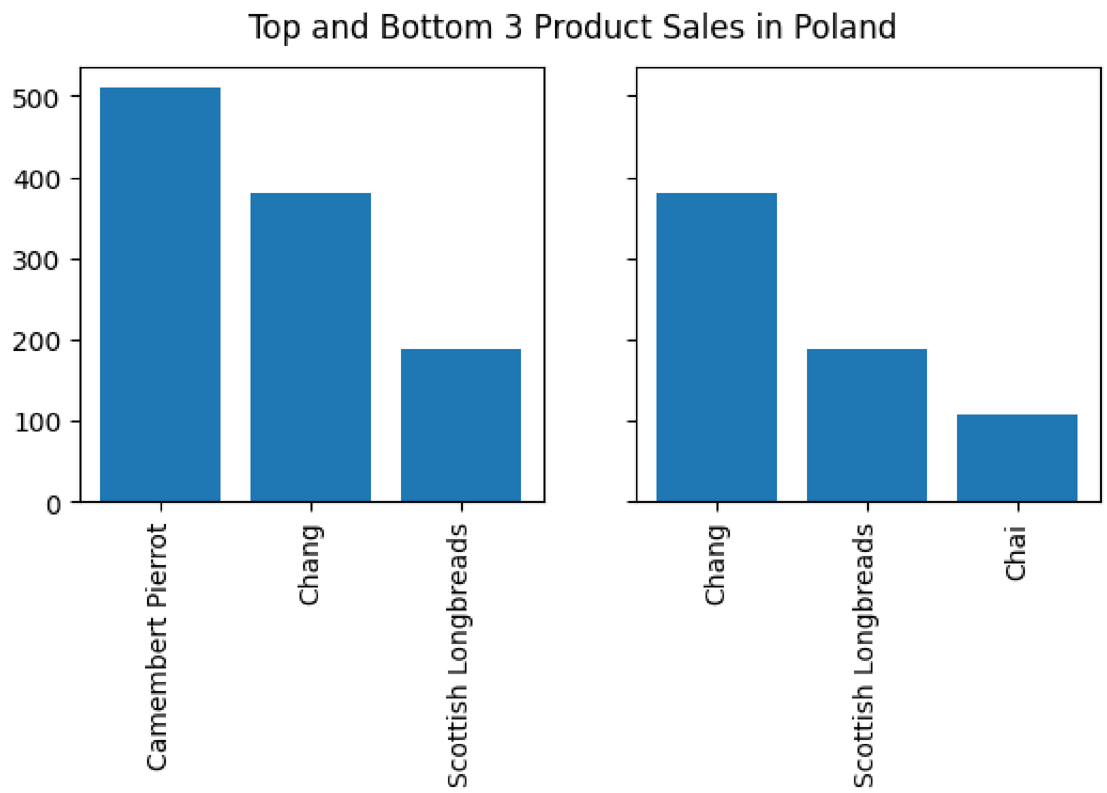


```
C:\Users\asus\AppData\Local\Temp\ipykernel_73944\1843367482.py:5: UserWarning: FixedFormatter should onl
  ax1.set_xticklabels(ax1.get_xticklabels(), rotation = 90)
C:\Users\asus\AppData\Local\Temp\ipykernel_73944\1843367482.py:8: UserWarning: FixedFormatter should onl
  ax2.set_xticklabels(ax2.get_xticklabels(), rotation = 90)


<Figure size 640x480 with 0 Axes>
```



```
C:\Users\asus\AppData\Local\Temp\ipykernel_73944\1843367482.py:5: UserWarning: FixedFormatter should onl
  ax1.set_xticklabels(ax1.get_xticklabels(), rotation = 90)
C:\Users\asus\AppData\Local\Temp\ipykernel_73944\1843367482.py:8: UserWarning: FixedFormatter should onl
  ax2.set_xticklabels(ax2.get_xticklabels(), rotation = 90)
```

```
<Figure size 640x480 with 0 Axes>
```



```
<Figure size 640x480 with 0 Axes>
```

Findings:

Tidak, setiap negara memiliki preferensi yang berbeda-beda terhadap produk favorit mereka, sehingga cara melakukan bundling harus berbeda-beda setiap negara

Chapter 4

Customer Analysis

Pada analisis customer akan dilakukan segmentasi berdasarkan RFM (Recency, Frequency, dan Monetary) Dengan simulasi bahwa analisis ini dilakukan pada awal tahun 1998 dan menggunakan data historis 1 tahun ke belakang (1997)

```
In [19]: today = datetime(1998,1,1)
```

```
In [20]: df_cst = df_cst[(df_cst.OrderDate >= '1997-01-01') &
                         (df_cst.OrderDate < '1998-01-01')]
         df_cst
```

```
Out[20]:      CustomerID              ContactName          Address      City Region  \
         0         ALFKI             Maria Anders    Obere Str. 57    Berlin   None
         1         ALFKI             Maria Anders    Obere Str. 57    Berlin   None
         2         ALFKI             Maria Anders    Obere Str. 57    Berlin   None
         3         ALFKI             Maria Anders    Obere Str. 57    Berlin   None
         4         ALFKI             Maria Anders    Obere Str. 57    Berlin   None
         …         …                      …              …       …    …
         2144      WOLZA  Zbyszek Piestrzeniewicz  ul. Filtrowa 68  Warszawa   None
         2145      WOLZA  Zbyszek Piestrzeniewicz  ul. Filtrowa 68  Warszawa   None
         2146      WOLZA  Zbyszek Piestrzeniewicz  ul. Filtrowa 68  Warszawa   None
         2147      WOLZA  Zbyszek Piestrzeniewicz  ul. Filtrowa 68  Warszawa   None
         2148      WOLZA  Zbyszek Piestrzeniewicz  ul. Filtrowa 68  Warszawa   None

              PostalCode  Country  OrderDate ShippedDate RequiredDate  \
         0         12209  Germany 1997-08-25  1997-09-02   1997-09-22
         1         12209  Germany 1997-08-25  1997-09-02   1997-09-22
         2         12209  Germany 1997-08-25  1997-09-02   1997-09-22
         3         12209  Germany 1997-10-03  1997-10-13   1997-10-31
         4         12209  Germany 1997-10-13  1997-10-21   1997-11-24
         …         …      …        …         …          …
         2144     01-012   Poland 1997-07-25  1997-08-01   1997-08-22
         2145     01-012   Poland 1997-07-25  1997-08-01   1997-08-22
         2146     01-012   Poland 1997-12-23  1997-12-31   1998-01-20
         2147     01-012   Poland 1997-12-23  1997-12-31   1998-01-20
         2148     01-012   Poland 1997-12-23  1997-12-31   1998-01-20

                       ProductName  Quantity  UnitPrice  Discount  TotalSales  \
         0        Chartreuse verte      21.0      18.00      0.25      378.00
         1       Rössle Sauerkraut      15.0      45.60      0.25      684.00
         2               Spegesild       2.0      12.00      0.25       24.00
```

```
3           Vegie-spread      20.0      43.90      0.00      878.00
4          Aniseed Syrup       6.0      10.00      0.00       60.00
…                     …       …        …          …          …
2144                Chai       6.0      18.00      0.00      108.00
2145               Chang      10.0      19.00      0.00      190.00
2146               Chang      10.0      19.00      0.00      190.00
2147  Scottish Longbreads     15.0      12.50      0.00      187.50
2148           Tourtière       3.0       7.45      0.00       22.35

      FinalSales
0         283.50
1         513.00
2          18.00
3         878.00
4          60.00
…          …
2144      108.00
2145      190.00
2146      190.00
2147      187.50
2148       22.35

[1059 rows x 16 columns]
```

4.1 Recency

```
In [21]: recency = (today - df_cst.groupby("CustomerID").agg({"OrderDate":"max"}))
```

```
In [22]: recency.rename(columns={'OrderDate':'Recency'}, inplace=True)
```

```
In [23]: recency = recency["Recency"].apply(lambda x: x.days)
```

4.2 Frequency

```
In [24]: freq = df_cst.groupby("CustomerID").agg({"CustomerID":"count"})
         freq.rename(columns={'CustomerID' : 'Frequency'}, inplace=True)
         freq.head()
```

```
Out[24]:             Frequency
         CustomerID
         ALFKI               6
         ANATR               4
         ANTON              14
         AROUT              18
         BERGS              27
```

4.3 Monetary

```
In [25]: monetary = df_cst.groupby('CustomerID').agg({'FinalSales':np.sum})
         monetary.rename(columns = {'FinalSales':'Monetary'}, inplace=True)
         monetary.head()
```

```
Out[25]:             Monetary
         CustomerID
```

```
ALFKI        2022.50
ANATR         799.75
ANTON        5960.77
AROUT        6406.90
BERGS       13849.02
```

4.4 RFM Analysis

In [26]:
```
rfm = pd.concat([recency, freq, monetary], axis=1)
rfm.head()
```

Out[26]:
```
            Recency  Frequency  Monetary
CustomerID
ALFKI            80          6   2022.50
ANATR            34          4    799.75
ANTON            98         14   5960.77
AROUT             8         18   6406.90
BERGS            16         27  13849.02
```

Setelah dilakukan concatenate pada dataframe recency, frequency, dan monetary. Selanjutnya akan dilakukan scoring terhadap RFM menggunakan quantile dengan rentang skor 1 s.d 5 penjabaran sebagai berikut :

1. Pada kolom recency, customer dengan recency terendah (baru membeli) akan mendapat skor tertinggi.
2. Pada kolom frequency, akan dilakukan rank lowest frequency (jarang membeli) akan mendapat skor rendah, dan sebaliknya.
3. Pada kolom monetary, akan monetary terendah (spending paling rendah) akan mendapatkan skor rendah, dan sebaliknya.

In [27]:
```
rfm["RecencyScore"] = pd.qcut(rfm["Recency"], 5, labels = [5, 4 , 3, 2, 1])
rfm["FrequencyScore"]= pd.qcut(rfm["Frequency"].rank(method="first"),5, labels=[1,2,3,4,5])
rfm["MonetaryScore"] = pd.qcut(rfm['Monetary'], 5, labels = [1, 2, 3, 4, 5])

rfm.head()
```

Out[27]:
```
            Recency  Frequency  Monetary RecencyScore FrequencyScore  \
CustomerID
ALFKI            80          6   2022.50            2              2
ANATR            34          4    799.75            3              1
ANTON            98         14   5960.77            2              4
AROUT             8         18   6406.90            5              5
BERGS            16         27  13849.02            4              5

           MonetaryScore
CustomerID
ALFKI                  2
ANATR                  1
ANTON                  4
AROUT                  4
BERGS                  5
```

In [28]:
```
rfm['RFM_SCORE'] = rfm['RecencyScore'].astype(str) + rfm['FrequencyScore'].astype(str) + rfm['
```

In [29]:
```
rfm.head()
```

```
Out[29]:             Recency  Frequency  Monetary RecencyScore FrequencyScore  \
         CustomerID
         ALFKI            80          6   2022.50            2              2
         ANATR            34          4    799.75            3              1
         ANTON            98         14   5960.77            2              4
         AROUT             8         18   6406.90            5              5
         BERGS            16         27  13849.02            4              5

                    MonetaryScore RFM_SCORE
         CustomerID
         ALFKI                  2       222
         ANATR                  1       311
         ANTON                  4       244
         AROUT                  4       554
         BERGS                  5       455
```

4.4.1 Segmentasi

Segmentasi akan dilakukan menggunakan recency dan frequensi dengan refensi segmentasi Referensi Segmentasi

```
In [30]: segmentation_mapping = {
             r'[1-2][1-2]': 'Hibernating',
             r'[1-2][3-4]': 'At Risk',
             r'[1-2]5': 'Can\'t Loose',
             r'3[1-2]': 'About to Sleep',
             r'33': 'Need Attention',
             r'[3-4][4-5]': 'Loyal Customers',
             r'41': 'Promising',
             r'51': 'New Customers',
             r'[4-5][2-3]': 'Potential Loyalists',
             r'5[4-5]': 'Champions'
         }
```

```
In [31]: rfm['Segment'] = rfm['RecencyScore'].astype(str) + rfm['FrequencyScore'].astype(str)
         rfm['Segment'] = rfm['Segment'].replace(segmentation_mapping, regex=True)
         rfm
```

```
Out[31]:             Recency  Frequency  Monetary RecencyScore FrequencyScore  \
         CustomerID
         ALFKI            80          6   2022.50            2              2
         ANATR            34          4    799.75            3              1
         ANTON            98         14   5960.77            2              4
         AROUT             8         18   6406.90            5              5
         BERGS            16         27  13849.02            4              5
         BLAUS           156          7   1079.80            1              2
         BLONP           100         15   7817.88            2              4
         BOLID             3          2   3026.85            5              1
         BONAP            36         21  11208.37            3              5
         BOTTM            48         13   7630.25            3              4
         BSBEV           170         14   3179.50            1              4
         CACTU            15          4    238.00            4              1
         CHOPS            43          9   6516.40            3              3
         COMMI           274          3   1128.00            1              1
         CONSH           304          5    787.60            1              1
```

DRACD	7	1	420.00	5	1
DUMON	97	4	487.00	2	1
EASTC	59	7	4514.35	2	2
ERNSH	8	44	48096.27	5	5
FAMIA	62	13	3127.13	2	4
FOLIG	10	16	11666.90	5	4
FOLKO	21	18	13314.67	4	5
FRANK	9	24	11829.79	5	5
FRANR	106	3	920.10	2	1
FRANS	1	6	249.70	5	2
FURIB	113	14	5065.33	2	4
GALED	202	3	493.20	1	1
GODOS	142	6	3458.35	1	2
GOURL	10	16	8008.78	5	4
GREAL	98	15	8565.33	2	4
GROSR	14	2	387.50	4	1
HANAR	14	11	6022.77	4	3
HILAA	7	25	13482.74	5	5
HUNGC	115	5	2283.20	1	2
HUNGO	51	26	20454.41	3	5
ISLAT	6	12	2560.50	5	3
KOENE	6	18	9664.21	5	5
LAMAI	13	17	6923.87	4	4
LAUGB	149	5	335.50	1	2
LAZYK	224	2	357.00	1	1
LEHMS	22	23	13076.13	4	5
LETSS	52	7	1698.40	3	2
LILAS	16	9	5175.20	4	3
LINOD	58	15	7359.48	3	4
LONEP	112	6	1837.20	2	2
MAGAA	14	12	4695.88	4	3
MAISD	31	8	5297.00	3	3
MEREP	63	25	23332.32	2	5
MORGK	16	7	3596.40	4	2
NORTS	38	5	604.00	3	2
OCEAN	238	3	429.20	1	1
OLDWO	77	9	5475.38	2	3
OTTIK	27	18	8254.27	4	5
PERIC	266	7	2065.40	1	2
PICCO	43	16	9305.58	3	4
PRINI	290	4	1409.20	1	1
QUEDE	8	13	3502.41	5	4
QUEEN	13	24	10132.77	4	5
QUICK	10	44	61109.92	5	5
RANCH	69	5	1149.40	2	2
RATTC	30	17	19383.75	3	4
REGGC	59	10	3000.84	2	3
RICAR	126	11	4283.78	1	3
RICSU	34	16	11864.42	3	4
SANTG	134	3	700.00	1	1
SAVEA	35	64	57713.57	3	5
SEVES	2	14	9021.25	5	4
SIMOB	3	12	16232.42	5	3
SPECD	50	1	52.35	3	1

```
SPLIR            35          6   2475.00            3              2
SUPRD            27         11   6137.48            4              3
THEBI             2          4   2955.40            5              1
THECR            20          5   1621.24            4              2
TOMSP           162         10   2004.34            1              3
TORTU           101          9   5523.35            2              3
TRADH           163          4   1320.40            1              1
TRAIH           192          7   1333.30            1              2
VAFFE            24         17   8960.12            4              4
VICTE             1         12   5807.12            5              3
VINET            50          4    379.80            3              1
WANDK           108         12   4262.83            2              3
WARTH            15         28  12262.94            4              5
WELLI             2         11   4415.15            5              3
WHITC            49         23   9146.50            3              5
WILMK            86          7   1174.35            2              3
WOLZA             9          6   1207.85            5              2

           MonetaryScore RFM_SCORE              Segment
CustomerID
ALFKI                  2       222          Hibernating
ANATR                  1       311      About to Sleep
ANTON                  4       244              At Risk
AROUT                  4       554            Champions
BERGS                  5       455      Loyal Customers
BLAUS                  1       121          Hibernating
BLONP                  4       244              At Risk
BOLID                  3       513        New Customers
BONAP                  5       355      Loyal Customers
BOTTM                  4       344      Loyal Customers
BSBEV                  3       143              At Risk
CACTU                  1       411            Promising
CHOPS                  4       334       Need Attention
COMMI                  1       111          Hibernating
CONSH                  1       111          Hibernating
DRACD                  1       511        New Customers
DUMON                  1       211          Hibernating
EASTC                  3       223          Hibernating
ERNSH                  5       555            Champions
FAMIA                  3       243              At Risk
FOLIG                  5       545            Champions
FOLKO                  5       455      Loyal Customers
FRANK                  5       555            Champions
FRANR                  1       211          Hibernating
FRANS                  1       521  Potential Loyalists
FURIB                  3       243              At Risk
GALED                  1       111          Hibernating
GODOS                  3       123          Hibernating
GOURL                  4       544            Champions
GREAL                  4       244              At Risk
GROSR                  1       411            Promising
HANAR                  4       434  Potential Loyalists
HILAA                  5       555            Champions
HUNGC                  2       122          Hibernating
```

```
HUNGO                  5       355     Loyal Customers
ISLAT                  2       532  Potential Loyalists
KOENE                  4       554           Champions
LAMAI                  4       444     Loyal Customers
LAUGB                  1       121         Hibernating
LAZYK                  1       111         Hibernating
LEHMS                  5       455     Loyal Customers
LETSS                  2       322      About to Sleep
LILAS                  3       433  Potential Loyalists
LINOD                  4       344     Loyal Customers
LONEP                  2       222         Hibernating
MAGAA                  3       433  Potential Loyalists
MAISD                  3       333      Need Attention
MEREP                  5       255         Can't Loose
MORGK                  3       423  Potential Loyalists
NORTS                  1       321      About to Sleep
OCEAN                  1       111         Hibernating
OLDWO                  3       233             At Risk
OTTIK                  4       454     Loyal Customers
PERIC                  2       122         Hibernating
PICCO                  4       344     Loyal Customers
PRINI                  2       112         Hibernating
QUEDE                  3       543           Champions
QUEEN                  5       455     Loyal Customers
QUICK                  5       555           Champions
RANCH                  2       222         Hibernating
RATTC                  5       345     Loyal Customers
REGGC                  2       232             At Risk
RICAR                  3       133             At Risk
RICSU                  5       345     Loyal Customers
SANTG                  1       111         Hibernating
SAVEA                  5       355     Loyal Customers
SEVES                  4       544           Champions
SIMOB                  5       535  Potential Loyalists
SPECD                  1       311      About to Sleep
SPLIR                  2       322      About to Sleep
SUPRD                  4       434  Potential Loyalists
THEBI                  2       512       New Customers
THECR                  2       422  Potential Loyalists
TOMSP                  2       132             At Risk
TORTU                  3       233             At Risk
TRADH                  2       112         Hibernating
TRAIH                  2       122         Hibernating
VAFFE                  4       444     Loyal Customers
VICTE                  3       533  Potential Loyalists
VINET                  1       311      About to Sleep
WANDK                  3       233             At Risk
WARTH                  5       455     Loyal Customers
WELLI                  3       533  Potential Loyalists
WHITC                  4       354     Loyal Customers
WILMK                  2       232             At Risk
WOLZA                  2       522  Potential Loyalists
```

```
In [39]: fig = px.treemap(pd.DataFrame(rfm['Segment'].value_counts().reset_index()),
                 path=['index'],
                 values='Segment', width=800, height=400)
fig.update_layout(
    treemapcolorway =['#fae588','#f79d65','#f9dc5c','#e8ac65','#e76f51','#ef233c','#b7094c'],
    margin = dict(t=50, l=25, r=25, b=25))
fig.show()
```

Findings:

-

Chapter 5

Cohort Analysis

Setelah diketahui bahwa sebagian besar konsumen kita berada pada segmentasi Hibernating dan At Risk maka perlu dilakukan analisis lebih jauh menggunakan cohort analisis untuk retention rate dar customer-customer tersebut

```
In [ ]:
```