Task1

June 5, 2025

[1]:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
import seaborn as sns
```

Importing Dataset

[3]:

```
purchase_data = pd.read_csv('QVI_purchase_behaviour.csv')
purchase_data.head()
```

[3]:

```
   LYLTY_CARD_NBR               LIFESTAGE PREMIUM_CUSTOMER
0            1000   YOUNG SINGLES/COUPLES          Premium
1            1002   YOUNG SINGLES/COUPLES       Mainstream
2            1003          YOUNG FAMILIES           Budget
3            1004   OLDER SINGLES/COUPLES       Mainstream
4            1005  MIDAGE SINGLES/COUPLES       Mainstream
```

[4]:

```
transaction_data = pd.read_excel('QVI_transaction_data.xlsx')
transaction_data.head()
```

[4]:

```
    DATE  STORE_NBR  LYLTY_CARD_NBR  TXN_ID  PROD_NBR  \
0  43390          1            1000       1         5
1  43599          1            1307     348        66
2  43605          1            1343     383        61
3  43329          2            2373     974        69
4  43330          2            2426    1038       108

                                  PROD_NAME  PROD_QTY  TOT_SALES
0    Natural Chip        Compny SeaSalt175g         2        6.0
1                  CCs Nacho Cheese    175g         3        6.3
2    Smiths Crinkle Cut  Chips Chicken 170g         2        2.9
3    Smiths Chip Thinly  S/Cream&Onion 175g         5       15.0
4  Kettle Tortilla ChpsHny&Jlpno Chili 150g         3       13.8
```

[5]:

```
purchase_data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 72637 entries, 0 to 72636
```

```
Data columns (total 3 columns):
 #   Column            Non-Null Count  Dtype
---  ------            --------------  -----
 0   LYLTY_CARD_NBR    72637 non-null  int64
 1   LIFESTAGE         72637 non-null  object
 2   PREMIUM_CUSTOMER  72637 non-null  object
dtypes: int64(1), object(2)
memory usage: 1.7+ MB
```

[6]:

```
transaction_data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 264836 entries, 0 to 264835
Data columns (total 8 columns):
 #   Column          Non-Null Count   Dtype
---  ------          --------------   -----
 0   DATE            264836 non-null  int64
 1   STORE_NBR       264836 non-null  int64
 2   LYLTY_CARD_NBR  264836 non-null  int64
 3   TXN_ID          264836 non-null  int64
 4   PROD_NBR        264836 non-null  int64
 5   PROD_NAME       264836 non-null  object
 6   PROD_QTY        264836 non-null  int64
 7   TOT_SALES       264836 non-null  float64
dtypes: float64(1), int64(6), object(1)
memory usage: 16.2+ MB
```

[7]:

```
purchase_data.describe().T
```

[7]:

```
                  count          mean            std     min      25%  \
LYLTY_CARD_NBR  72637.0  136185.93177  89892.932014  1000.0  66202.0

                     50%       75%        max
LYLTY_CARD_NBR  134040.0  203375.0  2373711.0
```

[9]:

```
transaction_data.describe().T
```

[9]:

```
                   count           mean           std      min      25%  \
DATE            264836.0   43464.036260    105.389282  43282.0  43373.0
STORE_NBR       264836.0     135.080110     76.784180      1.0     70.0
LYLTY_CARD_NBR  264836.0  135549.476404  80579.978022   1000.0  70021.0
TXN_ID          264836.0  135158.310815  78133.026026      1.0  67601.5
PROD_NBR        264836.0      56.583157     32.826638      1.0     28.0
PROD_QTY        264836.0       1.907309      0.643654      1.0      2.0
TOT_SALES       264836.0       7.304200      3.083226      1.5      5.4

                     50%       75%       max
DATE             43464.0  43555.00   43646.0
```

```
STORE_NBR          130.0      203.00       272.0
LYLTY_CARD_NBR  130357.5  203094.25  2373711.0
TXN_ID          135137.5  202701.25  2415841.0
PROD_NBR            56.0      85.00       114.0
PROD_QTY             2.0       2.00       200.0
TOT_SALES            7.4       9.20       650.0
```

Checking Missing Values

[10]:

```
transaction_data.isnull().sum()
```

[10]:

```
DATE              0
STORE_NBR         0
LYLTY_CARD_NBR    0
TXN_ID            0
PROD_NBR          0
PROD_NAME         0
PROD_QTY          0
TOT_SALES         0
dtype: int64
```

Analyzing and Removing Outliers

[11]:

```
#merging both dataset
merged_data = pd.merge(purchase_data, transaction_data, on = 'LYLTY_CARD_NBR',␣
  ↪how = 'right')
merged_data.head()
```

[11]:

```
   LYLTY_CARD_NBR               LIFESTAGE PREMIUM_CUSTOMER   DATE  STORE_NBR  \
0            1000   YOUNG SINGLES/COUPLES          Premium  43390          1
1            1307  MIDAGE SINGLES/COUPLES           Budget  43599          1
2            1343  MIDAGE SINGLES/COUPLES           Budget  43605          1
3            2373  MIDAGE SINGLES/COUPLES           Budget  43329          2
4            2426  MIDAGE SINGLES/COUPLES           Budget  43330          2

   TXN_ID  PROD_NBR                                 PROD_NAME  PROD_QTY  \
0       1         5    Natural Chip        Compny SeaSalt175g         2
1     348        66                  CCs Nacho Cheese    175g         3
2     383        61    Smiths Crinkle Cut  Chips Chicken 170g         2
3     974        69    Smiths Chip Thinly  S/Cream&Onion 175g         5
4    1038       108  Kettle Tortilla ChpsHny&Jlpno Chili 150g         3

   TOT_SALES
0        6.0
1        6.3
2        2.9
3       15.0
4       13.8
```

[12]:

```
print(len(merged_data))
print(len(transaction_data))
```

```
264836
264836
```

[13]:

```
merged_data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 264836 entries, 0 to 264835
Data columns (total 10 columns):
 #   Column            Non-Null Count   Dtype
---  ------            --------------   -----
 0   LYLTY_CARD_NBR    264836 non-null  int64
 1   LIFESTAGE         264836 non-null  object
 2   PREMIUM_CUSTOMER  264836 non-null  object
 3   DATE              264836 non-null  int64
 4   STORE_NBR         264836 non-null  int64
 5   TXN_ID            264836 non-null  int64
 6   PROD_NBR          264836 non-null  int64
 7   PROD_NAME         264836 non-null  object
 8   PROD_QTY          264836 non-null  int64
 9   TOT_SALES         264836 non-null  float64
dtypes: float64(1), int64(6), object(3)
memory usage: 20.2+ MB
```

Date column is not in proper format

[14]:

```
from datetime import date, timedelta

start = date(1899, 12, 30)
new_date_format = []
for date in merged_data["DATE"]:
  delta = timedelta(date)
  new_date_format.append(start + delta)
```

[15]:

```
merged_data["DATE"] = pd.to_datetime(pd.Series(new_date_format))
print(merged_data["DATE"].dtype)
```

```
datetime64[ns]
```

[16]:

```
merged_data["DATE"].describe()
```

[16]:

```
count                          264836
mean     2018-12-30 00:52:12.879215616
min                2018-07-01 00:00:00
25%                2018-09-30 00:00:00
50%                2018-12-30 00:00:00
75%                2019-03-31 00:00:00
```

```
max                2019-06-30 00:00:00
Name: DATE, dtype: object
```

[18]:

```
pd.date_range(start=merged_data["DATE"].min(),
              end=merged_data["DATE"].max()).difference(merged_data["DATE"])
```

[18]:

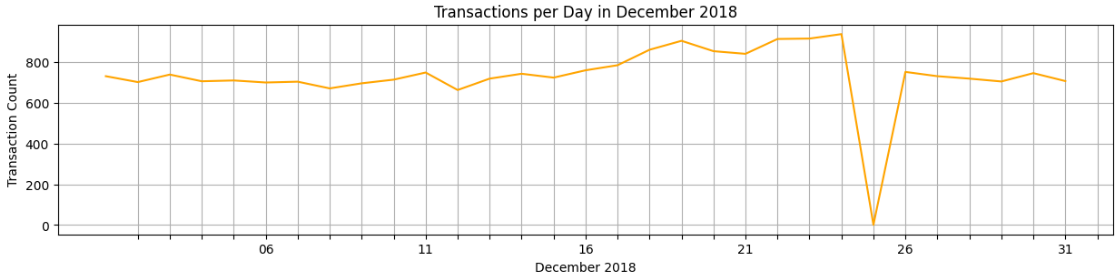
```
DatetimeIndex(['2018-12-25'], dtype='datetime64[ns]', freq='D')
```

[19]:

```
check_null_date = pd.merge(pd.Series(pd.date_range(start=merged_data["DATE"].
 ↪min(),
                                                   end = merged_data["DATE"].
 ↪max()),
                                     name="DATE"), merged_data, on = "DATE",␣
 ↪how = "left")
```

[23]:

```
import pandas as pd
from datetime import datetime

trans_by_date = check_null_date["DATE"].value_counts()
trans_by_date.index = pd.to_datetime(trans_by_date.index)

dec = trans_by_date[(trans_by_date.index >= datetime(2018, 12, 1)) &
                    (trans_by_date.index < datetime(2019, 1, 1))].sort_index()

dec.index = dec.index.strftime('%d')
ax = dec.plot(figsize=(15, 3), color='orange')
ax.set_xticks(range(1, 32))
ax.set_xlabel("December 2018")
ax.set_ylabel("Transaction Count")
ax.set_title("Transactions per Day in December 2018")
plt.grid(True)
plt.show()
```



Finding Average Purchase Quantity

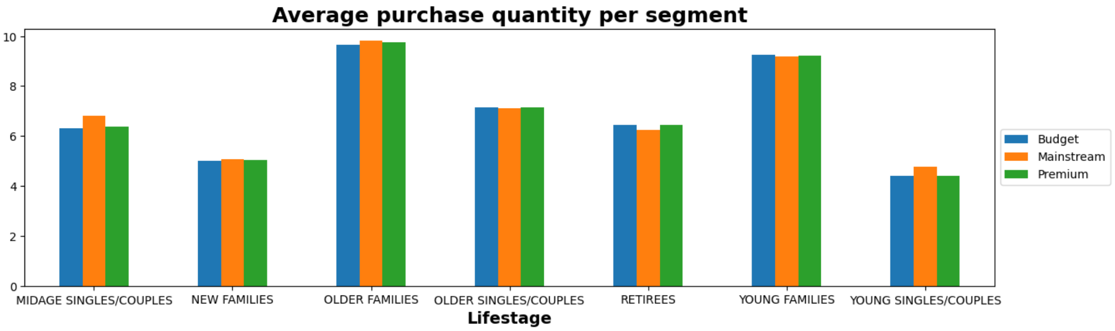
[27]:

```
temp = check_null_date.copy()

avg_qty_per_customer = (
    temp.groupby(["LIFESTAGE", "PREMIUM_CUSTOMER"])["PROD_QTY"].sum() /
    temp.groupby(["LIFESTAGE", "PREMIUM_CUSTOMER"])["LYLTY_CARD_NBR"].
 ↪nunique()).sort_values(ascending=False)

print(avg_qty_per_customer)
```

```
LIFESTAGE               PREMIUM_CUSTOMER
OLDER FAMILIES          Mainstream          9.804309
                        Premium             9.749780
                        Budget              9.639572
YOUNG FAMILIES          Budget              9.238486
                        Premium             9.209207
                        Mainstream          9.180352
OLDER SINGLES/COUPLES   Premium             7.154947
                        Budget              7.145466
                        Mainstream          7.098783
MIDAGE SINGLES/COUPLES  Mainstream          6.796108
RETIREES                Budget              6.458015
                        Premium             6.426653
MIDAGE SINGLES/COUPLES  Premium             6.386672
                        Budget              6.313830
RETIREES                Mainstream          6.253743
NEW FAMILIES            Mainstream          5.087161
                        Premium             5.028912
                        Budget              5.009892
YOUNG SINGLES/COUPLES   Mainstream          4.776459
                        Budget              4.411485
                        Premium             4.402098
dtype: float64
```

[30]:

```
(temp.groupby(["LIFESTAGE", "PREMIUM_CUSTOMER"])["PROD_QTY"].sum()
 / temp.groupby(["LIFESTAGE", "PREMIUM_CUSTOMER"])["LYLTY_CARD_NBR"].nunique()).
 ↪unstack().plot.bar(figsize=(15,4), rot=0)

plt.title("Average purchase quantity per segment", fontsize=18,␣
 ↪fontweight='bold', color='black')
plt.xlabel("Lifestage", fontsize=14, fontweight='bold', color='black')
plt.legend(loc="center left", bbox_to_anchor=(1.0, 0.5))
plt.savefig("Average purchase quantity per segment.png", bbox_inches="tight")
plt.show()
```


Average purchase quantity per segment
10
8
6
4
2
0
MIDAGE SINGLES/COUPLES
NEW FAMILIES
OLDER FAMILIES
OLDER SINGLES/COUPLES
RETIREES
YOUNG FAMILIES
YOUNG SINGLES/COUPLES
Lifestage
Budget
Mainstream
Premium