quantinum-forage-intership-1

April 8, 2024

0.0.1 Library

[4]:
```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
import datetime
import xlrd
import re
from mlxtend.frequent_patterns import apriori
from mlxtend.frequent_patterns import association_rules
from sklearn.preprocessing import OneHotEncoder
```

```
/usr/local/lib/python3.10/dist-packages/ipykernel/ipkernel.py:283:
DeprecationWarning: `should_run_async` will not call `transform_cell`
automatically in the future. Please pass the result to `transformed_cell`
argument and any exception that happen during thetransform in
`preprocessing_exc_tuple` in IPython 7.17 and above.
  and should_run_async(code)
```

0.0.2 Importing Data

[5]:
```
customer = pd.read_csv('/content/QVI_purchase_behaviour.csv')
transaction = pd.read_excel('/content/QVI_transaction_data.xlsx')
```

```
/usr/local/lib/python3.10/dist-packages/ipykernel/ipkernel.py:283:
DeprecationWarning: `should_run_async` will not call `transform_cell`
automatically in the future. Please pass the result to `transformed_cell`
argument and any exception that happen during thetransform in
`preprocessing_exc_tuple` in IPython 7.17 and above.
  and should_run_async(code)
```

0.0.3 Analysing Data

Exploratory Data Analysis

1. Transection

[6]:
```
transaction.head()
```

```
/usr/local/lib/python3.10/dist-packages/ipykernel/ipykernel.py:283:
DeprecationWarning: `should_run_async` will not call `transform_cell`
automatically in the future. Please pass the result to `transformed_cell`
argument and any exception that happen during thetransform in
`preprocessing_exc_tuple` in IPython 7.17 and above.
  and should_run_async(code)
```

[6]:
```
    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
```

[7]:
```python
transaction.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
```

```
/usr/local/lib/python3.10/dist-packages/ipykernel/ipykernel.py:283:
DeprecationWarning: `should_run_async` will not call `transform_cell`
automatically in the future. Please pass the result to `transformed_cell`
argument and any exception that happen during thetransform in
`preprocessing_exc_tuple` in IPython 7.17 and above.
  and should_run_async(code)
```

[8]:
```python
transaction.isnull().sum()
```

```
/usr/local/lib/python3.10/dist-packages/ipykernel/ipkernel.py:283:
DeprecationWarning: `should_run_async` will not call `transform_cell`
automatically in the future. Please pass the result to `transformed_cell`
argument and any exception that happen during thetransform in
`preprocessing_exc_tuple` in IPython 7.17 and above.
  and should_run_async(code)
```

[8]:
```
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
```

[9]:
```
transaction.nunique().sort_values()
```

```
/usr/local/lib/python3.10/dist-packages/ipykernel/ipkernel.py:283:
DeprecationWarning: `should_run_async` will not call `transform_cell`
automatically in the future. Please pass the result to `transformed_cell`
argument and any exception that happen during thetransform in
`preprocessing_exc_tuple` in IPython 7.17 and above.
  and should_run_async(code)
```

[9]:
```
PROD_QTY               6
TOT_SALES            112
PROD_NBR             114
PROD_NAME            114
STORE_NBR            272
DATE                 364
LYLTY_CARD_NBR     72637
TXN_ID            263127
dtype: int64
```

2. Customer

[10]:
```
customer.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

/usr/local/lib/python3.10/dist-packages/ipykernel/ipkernel.py:283:
DeprecationWarning: `should_run_async` will not call `transform_cell`
automatically in the future. Please pass the result to `transformed_cell`
argument and any exception that happen during thetransform in
`preprocessing_exc_tuple` in IPython 7.17 and above.
  and should_run_async(code)
```

[11]:
```
customer.isnull().sum()
```

```
/usr/local/lib/python3.10/dist-packages/ipykernel/ipkernel.py:283:
DeprecationWarning: `should_run_async` will not call `transform_cell`
automatically in the future. Please pass the result to `transformed_cell`
argument and any exception that happen during thetransform in
`preprocessing_exc_tuple` in IPython 7.17 and above.
  and should_run_async(code)
```

[11]:
```
LYLTY_CARD_NBR      0
LIFESTAGE           0
PREMIUM_CUSTOMER    0
dtype: int64
```

[12]:
```
customer.head()
```

```
/usr/local/lib/python3.10/dist-packages/ipykernel/ipkernel.py:283:
DeprecationWarning: `should_run_async` will not call `transform_cell`
automatically in the future. Please pass the result to `transformed_cell`
argument and any exception that happen during thetransform in
`preprocessing_exc_tuple` in IPython 7.17 and above.
  and should_run_async(code)
```

[12]:
```
   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
```

[13]:
```
customer.nunique().sort_values()
```

```
/usr/local/lib/python3.10/dist-packages/ipykernel/ipkernel.py:283:
DeprecationWarning: `should_run_async` will not call `transform_cell`
automatically in the future. Please pass the result to `transformed_cell`
argument and any exception that happen during thetransform in
`preprocessing_exc_tuple` in IPython 7.17 and above.
  and should_run_async(code)
```

[13]:
```
PREMIUM_CUSTOMER        3
LIFESTAGE               7
LYLTY_CARD_NBR      72637
dtype: int64
```

[14]:
```
print(customer.PREMIUM_CUSTOMER.unique())
print(customer.LIFESTAGE.unique())
```

```
/usr/local/lib/python3.10/dist-packages/ipykernel/ipkernel.py:283:
DeprecationWarning: `should_run_async` will not call `transform_cell`
automatically in the future. Please pass the result to `transformed_cell`
argument and any exception that happen during thetransform in
`preprocessing_exc_tuple` in IPython 7.17 and above.
  and should_run_async(code)
```

```
['Premium' 'Mainstream' 'Budget']
['YOUNG SINGLES/COUPLES' 'YOUNG FAMILIES' 'OLDER SINGLES/COUPLES'
 'MIDAGE SINGLES/COUPLES' 'NEW FAMILIES' 'OLDER FAMILIES' 'RETIREES']
```

0.0.4 Date Data Types

[15]:
```
df_transaction = transaction.copy()
```

```
/usr/local/lib/python3.10/dist-packages/ipykernel/ipkernel.py:283:
DeprecationWarning: `should_run_async` will not call `transform_cell`
automatically in the future. Please pass the result to `transformed_cell`
argument and any exception that happen during thetransform in
`preprocessing_exc_tuple` in IPython 7.17 and above.
  and should_run_async(code)
```

[16]:
```
#change date into date data type
df_transaction.DATE = pd.to_datetime(df_transaction.DATE, unit='D',␣
 ↪origin='1899-12-30')
df_transaction.info()
```

```
/usr/local/lib/python3.10/dist-packages/ipykernel/ipkernel.py:283:
DeprecationWarning: `should_run_async` will not call `transform_cell`
automatically in the future. Please pass the result to `transformed_cell`
argument and any exception that happen during thetransform in
`preprocessing_exc_tuple` in IPython 7.17 and above.
  and should_run_async(code)
```

```
<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  datetime64[ns]
 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: datetime64[ns](1), float64(1), int64(5), object(1)
memory usage: 16.2+ MB
```

pack size and brand name from the data and define metrics of interest to enable you to draw insights on who spends on chips and what drives spends for each customer segment.

0.0.5 Find Chips

[17]:
```
df_transaction.PROD_NAME.unique()
```

```
/usr/local/lib/python3.10/dist-packages/ipykernel/ipkernel.py:283:
DeprecationWarning: `should_run_async` will not call `transform_cell`
automatically in the future. Please pass the result to `transformed_cell`
argument and any exception that happen during thetransform in
`preprocessing_exc_tuple` in IPython 7.17 and above.
  and should_run_async(code)
```

[17]:
```
array(['Natural Chip        Compny SeaSalt175g',
       'CCs Nacho Cheese    175g',
       'Smiths Crinkle Cut  Chips Chicken 170g',
       'Smiths Chip Thinly  S/Cream&Onion 175g',
       'Kettle Tortilla ChpsHny&Jlpno Chili 150g',
       'Old El Paso Salsa   Dip Tomato Mild 300g',
       'Smiths Crinkle Chips Salt & Vinegar 330g',
       'Grain Waves         Sweet Chilli 210g',
       'Doritos Corn Chip Mexican Jalapeno 150g',
       'Grain Waves Sour    Cream&Chives 210G',
       'Kettle Sensations   Siracha Lime 150g',
       'Twisties Cheese     270g', 'WW Crinkle Cut      Chicken 175g',
       'Thins Chips Light&  Tangy 175g', 'CCs Original 175g',
       'Burger Rings 220g', 'NCC Sour Cream &    Garden Chives 175g',
       'Doritos Corn Chip Southern Chicken 150g',
       'Cheezels Cheese Box 125g', 'Smiths Crinkle      Original 330g',
       'Infzns Crn Crnchers Tangy Gcamole 110g',
       'Kettle Sea Salt     And Vinegar 175g',
       'Smiths Chip Thinly  Cut Original 175g', 'Kettle Original 175g',
       'Red Rock Deli Thai  Chilli&Lime 150g',
       'Pringles Sthrn FriedChicken 134g', 'Pringles Sweet&Spcy BBQ 134g',
       'Red Rock Deli SR    Salsa & Mzzrlla 150g',
       'Thins Chips         Originl saltd 175g',
       'Red Rock Deli Sp    Salt & Truffle 150G',
       'Smiths Thinly       Swt Chli&S/Cream175G', 'Kettle Chilli 175g',
       'Doritos Mexicana    170g',
       'Smiths Crinkle Cut  French OnionDip 150g',
       'Natural ChipCo      Hony Soy Chckn175g',
       'Dorito Corn Chp     Supreme 380g', 'Twisties Chicken270g',
       'Smiths Thinly Cut   Roast Chicken 175g',
       'Smiths Crinkle Cut  Tomato Salsa 150g',
       'Kettle Mozzarella   Basil & Pesto 175g',
       'Infuzions Thai SweetChili PotatoMix 110g',
       'Kettle Sensations   Camembert & Fig 150g',
       'Smith Crinkle Cut   Mac N Cheese 150g',
       'Kettle Honey Soy    Chicken 175g',
       'Thins Chips Seasonedchicken 175g',
       'Smiths Crinkle Cut  Salt & Vinegar 170g',
       'Infuzions BBQ Rib   Prawn Crackers 110g',
       'GrnWves Plus Btroot & Chilli Jam 180g',
       'Tyrrells Crisps     Lightly Salted 165g',
       'Kettle Sweet Chilli And Sour Cream 175g',
       'Doritos Salsa       Medium 300g', 'Kettle 135g Swt Pot Sea Salt',
       'Pringles SourCream  Onion 134g',
       'Doritos Corn Chips  Original 170g',
       'Twisties Cheese     Burger 250g',
       'Old El Paso Salsa   Dip Chnky Tom Ht300g',
       'Cobs Popd Swt/Chlli &Sr/Cream Chips 110g',
       'Woolworths Mild     Salsa 300g',
       'Natural Chip Co     Tmato Hrb&Spce 175g',
       'Smiths Crinkle Cut  Chips Original 170g',
       'Cobs Popd Sea Salt  Chips 110g',
       'Smiths Crinkle Cut  Chips Chs&Onion170g',
       'French Fries Potato Chips 175g',
       'Old El Paso Salsa   Dip Tomato Med 300g',
       'Doritos Corn Chips  Cheese Supreme 170g',
       'Pringles Original   Crisps 134g',
       'RRD Chilli&         Coconut 150g',
       'WW Original Corn    Chips 200g',
       'Thins Potato Chips  Hot & Spicy 175g',
       'Cobs Popd Sour Crm  &Chives Chips 110g',
       'Smiths Crnkle Chip  Orgnl Big Bag 380g',
       'Doritos Corn Chips  Nacho Cheese 170g',
       'Kettle Sensations   BBQ&Maple 150g',
       'WW D/Style Chip     Sea Salt 200g',
       'Pringles Chicken    Salt Crips 134g',
       'WW Original Stacked Chips 160g',
       'Smiths Chip Thinly  CutSalt/Vinegr175g', 'Cheezels Cheese 330g',
       'Tostitos Lightly    Salted 175g',
       'Thins Chips Salt &  Vinegar 175g',
       'Smiths Crinkle Cut  Chips Barbecue 170g', 'Cheetos Puffs 165g',
       'RRD Sweet Chilli &  Sour Cream 165g',
       'WW Crinkle Cut      Original 175g',
       'Tostitos Splash Of  Lime 175g', 'Woolworths Medium   Salsa 300g',
       'Kettle Tortilla ChpsBtroot&Ricotta 150g',
       'CCs Tasty Cheese    175g', 'Woolworths Cheese   Rings 190g',
       'Tostitos Smoked     Chipotle 175g', 'Pringles Barbeque   134g',
       'WW Supreme Cheese   Corn Chips 200g',
       'Pringles Mystery    Flavour 134g',
       'Tyrrells Crisps     Ched & Chives 165g',
       'Snbts Whlgrn Crisps Cheddr&Mstrd 90g',
       'Cheetos Chs & Bacon Balls 190g', 'Pringles Slt Vingar 134g',
       'Infuzions SourCream&Herbs Veg Strws 110g',
       'Kettle Tortilla ChpsFeta&Garlic 150g',
       'Infuzions Mango     Chutny Papadums 70g',
       'RRD Steak &         Chimuchurri 150g',
       'RRD Honey Soy       Chicken 165g',
       'Sunbites Whlegrn    Crisps Frch/Onin 90g',
       'RRD Salt & Vinegar  165g', 'Doritos Cheese      Supreme 330g',
       'Smiths Crinkle Cut  Snag&Sauce 150g',
       'WW Sour Cream &OnionStacked Chips 160g',
       'RRD Lime & Pepper   165g',
       'Natural ChipCo Sea  Salt & Vinegr 175g',
       'Red Rock Deli Chikn&Garlic Aioli 150g',
       'RRD SR Slow Rst     Pork Belly 150g', 'RRD Pc Sea Salt     165g',
       'Smith Crinkle Cut   Bolognese 150g', 'Doritos Salsa Mild  300g'],
      dtype=object)
```

[18]:
```
# Remove digits from the product names
prod_name = df_transaction['PROD_NAME'].str.replace(r'[0-9]+[gG]','');

# Remove & characters and replace with a space to separate flavours
prod_name = prod_name.str.replace(r'&',' ');
```

```
/usr/local/lib/python3.10/dist-packages/ipykernel/ipkernel.py:283:
DeprecationWarning: `should_run_async` will not call `transform_cell`
automatically in the future. Please pass the result to `transformed_cell`
argument and any exception that happen during thetransform in
`preprocessing_exc_tuple` in IPython 7.17 and above.
  and should_run_async(code)
```

[19]:
```
prod_name.info()
```

```
/usr/local/lib/python3.10/dist-packages/ipykernel/ipkernel.py:283:
DeprecationWarning: `should_run_async` will not call `transform_cell`
automatically in the future. Please pass the result to `transformed_cell`
argument and any exception that happen during thetransform in
`preprocessing_exc_tuple` in IPython 7.17 and above.
  and should_run_async(code)
```

```
<class 'pandas.core.series.Series'>
RangeIndex: 264836 entries, 0 to 264835
Series name: PROD_NAME
Non-Null Count   Dtype
--------------   -----
264836 non-null  object
dtypes: object(1)
memory usage: 2.0+ MB
```

[20]:
```
# Count the frequencies of words in product names and display counts in␣
↪descending order
prod_counts = pd.Series(' '.join(prod_name).split()).value_counts()

with pd.option_context('display.max_rows', None): # show all rows
  display(prod_counts)
```

```
/usr/local/lib/python3.10/dist-packages/ipykernel/ipkernel.py:283:
DeprecationWarning: `should_run_async` will not call `transform_cell`
automatically in the future. Please pass the result to `transformed_cell`
argument and any exception that happen during thetransform in
`preprocessing_exc_tuple` in IPython 7.17 and above.
  and should_run_async(code)
```

```
175g                      60561
Chips                     49770
150g                      41633
Kettle                    41288
Smiths                    28860
Salt                      27976
Cheese                    27890
Pringles                  25102
134g                      25102
Doritos                   24962
Crinkle                   23960
110g                      22387
Corn                      22063
Original                  21560
Cut                       20754
Chip                      18645
170g                      18502
Salsa                     18094
Chicken                   15407
Chilli                    15390
165g                      15297
Sea                       14145
Thins                     14075
Sour                      13882
Crisps                    12607
```

```
330g                    12540
Vinegar                 12402
300g                    12041
RRD                     11894
Sweet                   11060
Infuzions               11057
Supreme                 10963
Chives                  10951
Cream                   10723
WW                      10320
Popd                     9693
Cobs                     9693
Tortilla                 9580
Tostitos                 9471
Twisties                 9454
BBQ                      9434
Sensations               9429
Lime                     9347
Dip                      9324
Old                      9324
El                       9324
Paso                     9324
Tomato                   7669
Thinly                   7507
Tyrrells                 6442
380g                     6418
And                      6373
Tangy                    6332
SourCream                6296
Grain                    6272
Waves                    6272
Salted                   6248
Lightly                  6248
Soy                      6121
Natural                  6050
Mild                     6048
Deli                     5885
Red                      5885
Rock                     5885
Thai                     4737
Burger                   4733
Swt                      4718
Honey                    4661
Nacho                    4658
Potato                   4647
Onion                    4635
Cheezels                 4603
Garlic                   4572
```

CCs	4551
200g	4473
Woolworths	4437
Pesto	3304
Mozzarella	3304
Basil	3304
Jlpno	3296
Chili	3296
ChpsHny	3296
Swt/Chlli	3269
Sr/Cream	3269
Ched	3268
Pot	3257
135g	3257
Of	3252
Splash	3252
SweetChili	3242
PotatoMix	3242
Crnkle	3233
Orgnl	3233
Big	3233
Bag	3233
Hot	3229
Spicy	3229
Fig	3219
Camembert	3219
Barbeque	3210
Mexican	3204
Jalapeno	3204
Light	3188
Chp	3185
Dorito	3185
Spcy	3177
Rib	3174
Crackers	3174
Prawn	3174
Southern	3172
Chicken270g	3170
250g	3169
210g	3167
Crm	3159
Ricotta	3146
ChpsBtroot	3146
Chipotle	3145
Smoked	3145
Infzns	3144
Crn	3144
Crnchers	3144

Gcamole	3144
ChpsFeta	3138
Veg	3134
Herbs	3134
Strws	3134
Siracha	3127
Tom	3125
Chnky	3125
Ht300g	3125
270g	3115
Mexicana	3115
Flavour	3114
Mystery	3114
Seasonedchicken	3114
Med	3114
210G	3105
Crips	3104
Slt	3095
Vingar	3095
Maple	3083
Sthrn	3083
FriedChicken	3083
Rings	3080
ChipCo	3010
90g	3008
190g	2995
SR	2984
160g	2970
Smith	2963
Chs	2960
Cheetos	2927
Medium	2879
French	2856
Snbts	1576
Whlgrn	1576
Cheddr	1576
Mstrd	1576
Spce	1572
Tmato	1572
Co	1572
Hrb	1572
220g	1564
Vinegr	1550
Tasty	1539
Belly	1526
Pork	1526
Rst	1526
Slow	1526

Roast	1519
N	1512
Mac	1512
Mango	1507
70g	1507
Chutny	1507
Papadums	1507
Coconut	1506
Sauce	1503
Snag	1503
Truffle	1498
Sp	1498
150G	1498
Barbecue	1489
Stacked	1487
OnionStacked	1483
Onion170g	1481
Balls	1479
Bacon	1479
S/Cream	1473
Pepper	1473
D/Style	1469
Compny	1468
SeaSalt175g	1468
Jam	1468
GrnWves	1468
Plus	1468
Btroot	1468
180g	1468
Chli	1461
S/Cream175G	1461
Hony	1460
Chckn175g	1460
Mzzrlla	1458
Steak	1455
Chimuchurri	1455
Box	1454
125g	1454
Bolognese	1451
Puffs	1448
Originl	1441
saltd	1441
CutSalt/Vinegr175g	1440
OnionDip	1438
Chikn	1434
Aioli	1434
Frch/Onin	1432
Whlegrn	1432

```
Sunbites               1432
Pc                     1431
Garden                 1419
NCC                    1419
Fries                  1418
Name: count, dtype: int64
```

There are salsa products in the dataset but we are only interested in the chips category, so let's remove these.

```
[21]: # remove salsa
      df_transaction = df_transaction[df_transaction.PROD_NAME.str.
       ↪contains(r"[Ss]alsa") == False]
```

```
/usr/local/lib/python3.10/dist-packages/ipykernel/ipkernel.py:283:
DeprecationWarning: `should_run_async` will not call `transform_cell`
automatically in the future. Please pass the result to `transformed_cell`
argument and any exception that happen during thetransform in
`preprocessing_exc_tuple` in IPython 7.17 and above.
  and should_run_async(code)
```

```
[22]: #check null
      df_transaction.isnull().values.any()
```

```
/usr/local/lib/python3.10/dist-packages/ipykernel/ipkernel.py:283:
DeprecationWarning: `should_run_async` will not call `transform_cell`
automatically in the future. Please pass the result to `transformed_cell`
argument and any exception that happen during thetransform in
`preprocessing_exc_tuple` in IPython 7.17 and above.
  and should_run_async(code)
```

```
[22]: False
```

```
[23]: df_transaction.shape
```

```
/usr/local/lib/python3.10/dist-packages/ipykernel/ipkernel.py:283:
DeprecationWarning: `should_run_async` will not call `transform_cell`
automatically in the future. Please pass the result to `transformed_cell`
argument and any exception that happen during thetransform in
`preprocessing_exc_tuple` in IPython 7.17 and above.
  and should_run_async(code)
```

```
[23]: (246742, 8)
```

```
[24]: df_transaction.describe()
```

```
/usr/local/lib/python3.10/dist-packages/ipykernel/ipkernel.py:283:
DeprecationWarning: `should_run_async` will not call `transform_cell`
automatically in the future. Please pass the result to `transformed_cell`
```

```
argument and any exception that happen during thetransform in
`preprocessing_exc_tuple` in IPython 7.17 and above.
  and should_run_async(code)
```

[24]:
```
                                DATE      STORE_NBR  LYLTY_CARD_NBR  \
count                         246742  246742.000000    2.467420e+05
mean   2018-12-30 01:19:01.211467520     135.051098    1.355310e+05
min              2018-07-01 00:00:00       1.000000    1.000000e+03
25%              2018-09-30 00:00:00      70.000000    7.001500e+04
50%              2018-12-30 00:00:00     130.000000    1.303670e+05
75%              2019-03-31 00:00:00     203.000000    2.030840e+05
max              2019-06-30 00:00:00     272.000000    2.373711e+06
std                              NaN      76.787096    8.071528e+04

             TXN_ID       PROD_NBR       PROD_QTY      TOT_SALES
count  2.467420e+05  246742.000000  246742.000000  246742.000000
mean   1.351311e+05      56.351789       1.908062       7.321322
min    1.000000e+00       1.000000       1.000000       1.700000
25%    6.756925e+04      26.000000       2.000000       5.800000
50%    1.351830e+05      53.000000       2.000000       7.400000
75%    2.026538e+05      87.000000       2.000000       8.800000
max    2.415841e+06     114.000000     200.000000     650.000000
std    7.814772e+04      33.695428       0.659831       3.077828
```

[25]:
```python
# Filter the entries that have 200 packets.
df_transaction.loc[df_transaction['PROD_QTY'] == 200.0]
```

```
/usr/local/lib/python3.10/dist-packages/ipykernel/ipkernel.py:283:
DeprecationWarning: `should_run_async` will not call `transform_cell`
automatically in the future. Please pass the result to `transformed_cell`
argument and any exception that happen during thetransform in
`preprocessing_exc_tuple` in IPython 7.17 and above.
  and should_run_async(code)
```

[25]:
```
            DATE  STORE_NBR  LYLTY_CARD_NBR  TXN_ID  PROD_NBR  \
69762 2018-08-19        226          226000  226201         4
69763 2019-05-20        226          226000  226210         4

                              PROD_NAME  PROD_QTY  TOT_SALES
69762  Dorito Corn Chp     Supreme 380g       200      650.0
69763  Dorito Corn Chp     Supreme 380g       200      650.0
```

The same customer has made these transactions. They could have been for commercial purposes so we can check to see if they made any other purchases.

[26]:
```python
# Filter the entires by the customer
df_transaction.loc[df_transaction['LYLTY_CARD_NBR'] == 226000]
```

```
/usr/local/lib/python3.10/dist-packages/ipykernel/ipkernel.py:283:
DeprecationWarning: `should_run_async` will not call `transform_cell`
automatically in the future. Please pass the result to `transformed_cell`
argument and any exception that happen during thetransform in
`preprocessing_exc_tuple` in IPython 7.17 and above.
  and should_run_async(code)
```

[26]:
```
             DATE  STORE_NBR  LYLTY_CARD_NBR  TXN_ID  PROD_NBR  \
69762 2018-08-19        226          226000  226201         4
69763 2019-05-20        226          226000  226210         4

                               PROD_NAME  PROD_QTY  TOT_SALES
69762  Dorito Corn Chp     Supreme 380g       200      650.0
69763  Dorito Corn Chp     Supreme 380g       200      650.0
```

It looks like this is the only purchase they have made so we will remove these transactions from the dataset.

[27]:
```python
# Remove the transactions
trans_df = df_transaction[df_transaction['LYLTY_CARD_NBR'] != 226000]
trans_df.shape # check for a reduction of 2 rows (before = 246742 rows)
```

```
/usr/local/lib/python3.10/dist-packages/ipykernel/ipkernel.py:283:
DeprecationWarning: `should_run_async` will not call `transform_cell`
automatically in the future. Please pass the result to `transformed_cell`
argument and any exception that happen during thetransform in
`preprocessing_exc_tuple` in IPython 7.17 and above.
  and should_run_async(code)
```

[27]:
```
(246740, 8)
```

[28]:
```python
df_transaction.describe()
```

```
/usr/local/lib/python3.10/dist-packages/ipykernel/ipkernel.py:283:
DeprecationWarning: `should_run_async` will not call `transform_cell`
automatically in the future. Please pass the result to `transformed_cell`
argument and any exception that happen during thetransform in
`preprocessing_exc_tuple` in IPython 7.17 and above.
  and should_run_async(code)
```

[28]:
```
                                DATE      STORE_NBR  LYLTY_CARD_NBR  \
count                         246742  246742.000000    2.467420e+05
mean   2018-12-30 01:19:01.211467520     135.051098    1.355310e+05
min              2018-07-01 00:00:00       1.000000    1.000000e+03
25%              2018-09-30 00:00:00      70.000000    7.001500e+04
50%              2018-12-30 00:00:00     130.000000    1.303670e+05
75%              2019-03-31 00:00:00     203.000000    2.030840e+05
max              2019-06-30 00:00:00     272.000000    2.373711e+06
```

```
std                                 NaN      76.787096    8.071528e+04

              TXN_ID       PROD_NBR       PROD_QTY      TOT_SALES
count  2.467420e+05  246742.000000  246742.000000  246742.000000
mean   1.351311e+05      56.351789       1.908062       7.321322
min    1.000000e+00       1.000000       1.000000       1.700000
25%    6.756925e+04      26.000000       2.000000       5.800000
50%    1.351830e+05      53.000000       2.000000       7.400000
75%    2.026538e+05      87.000000       2.000000       8.800000
max    2.415841e+06     114.000000     200.000000     650.000000
std    7.814772e+04      33.695428       0.659831       3.077828
```

The summaries now look reasonable. Now look at the number of transaction lines over time to see if there are any obvious data issues such as missing data from particular days.

[29]:
```
# Missing day by counting transactions by date
count = df_transaction.groupby(df_transaction['DATE'].dt.date).size().
 ↪reset_index(name = 'COUNT')
count.shape
```

```
/usr/local/lib/python3.10/dist-packages/ipykernel/ipkernel.py:283:
DeprecationWarning: `should_run_async` will not call `transform_cell`
automatically in the future. Please pass the result to `transformed_cell`
argument and any exception that happen during thetransform in
`preprocessing_exc_tuple` in IPython 7.17 and above.
  and should_run_async(code)
```

[29]:
```
(364, 2)
```

[30]:
```
# 1. See the date range
df_transaction.sort_values(by='DATE')
```

```
/usr/local/lib/python3.10/dist-packages/ipykernel/ipkernel.py:283:
DeprecationWarning: `should_run_async` will not call `transform_cell`
automatically in the future. Please pass the result to `transformed_cell`
argument and any exception that happen during thetransform in
`preprocessing_exc_tuple` in IPython 7.17 and above.
  and should_run_async(code)
```

[30]:
```
             DATE  STORE_NBR  LYLTY_CARD_NBR  TXN_ID  PROD_NBR  \
136301 2018-07-01          9            9341    8808        45
157526 2018-07-01         86           86016   84237        48
126416 2018-07-01        129          129046  132474        82
121423 2018-07-01         58           58072   53145        99
73583  2018-07-01         97           97164   97311        92
…             …      …               …       …       …
245590 2019-06-30         91           91076   89519        40
231677 2019-06-30         84           84116   83704        77
```

```
186851 2019-06-30        24          24115   20917        100
13810  2019-06-30        199         199117  198068        77
147420 2019-06-30        220         220032  219497         4

                                        PROD_NAME  PROD_QTY  TOT_SALES
136301    Smiths Thinly Cut    Roast Chicken 175g         2        6.0
157526   Red Rock Deli Sp     Salt & Truffle 150G         2        5.4
126416     Smith Crinkle Cut    Mac N Cheese 150g         2        5.2
121423         Pringles Sthrn FriedChicken 134g         2        7.4
73583          WW Crinkle Cut      Chicken 175g         2        3.4
…                                            …        …         …
245590         Thins Chips Seasonedchicken 175g         2        6.6
231677     Doritos Corn Chips   Nacho Cheese 170g         2        8.8
186851   Smiths Crinkle Cut   Chips Chs&Onion170g         2        5.8
13810      Doritos Corn Chips   Nacho Cheese 170g         2        8.8
147420         Dorito Corn Chp     Supreme 380g         2       13.0

[246742 rows x 8 columns]
```

the date range start from 1 July 2018 to 30 June 2019

[31]:
```
#2. check the missing date
missing_date = df_transaction.groupby('DATE').size()
pd.date_range(start = '2018-07-01', end = '2019-06-30' ).
  ↪difference(missing_date.index)
```

```
/usr/local/lib/python3.10/dist-packages/ipykernel/ipkernel.py:283:
DeprecationWarning: `should_run_async` will not call `transform_cell`
automatically in the future. Please pass the result to `transformed_cell`
argument and any exception that happen during thetransform in
`preprocessing_exc_tuple` in IPython 7.17 and above.
  and should_run_async(code)
```

[31]:
```
DatetimeIndex(['2018-12-25'], dtype='datetime64[ns]', freq=None)
```

the missing date is Christmast day in 2018, it is expected to be no sales in that day because it was a holiday

Now we move onto creating other features such as the pack size, and checking this for any outliers.

[32]:
```
# Add a new column to data with packet sizes and extract sizes from product␣
  ↪name column
df_transaction.insert(8, "PACK_SIZE", df_transaction['PROD_NAME'].str.
  ↪extract('(\d+)').astype(float), True)

# Sort by packet sizes to check for outliers
df_transaction.sort_values(by='PACK_SIZE')
```

```
/usr/local/lib/python3.10/dist-packages/ipykernel/ipkernel.py:283:
```

```
DeprecationWarning: `should_run_async` will not call `transform_cell`
automatically in the future. Please pass the result to `transformed_cell`
argument and any exception that happen during thetransform in
`preprocessing_exc_tuple` in IPython 7.17 and above.
  and should_run_async(code)
<>:2: DeprecationWarning: invalid escape sequence '\d'
<>:2: DeprecationWarning: invalid escape sequence '\d'
<ipython-input-32-eabca21f2c3a>:2: DeprecationWarning: invalid escape sequence
'\d'
  df_transaction.insert(8, "PACK_SIZE",
df_transaction['PROD_NAME'].str.extract('(\d+)').astype(float), True)
```

[32]:

```
              DATE  STORE_NBR  LYLTY_CARD_NBR  TXN_ID  PROD_NBR  \
46694   2019-02-04        162          162202  163019        38
203179  2019-01-04        258          258051  257352        38
165827  2019-05-25        197          197343  197307        38
198867  2018-08-28        196          196012  195458        38
2690    2018-11-02        136          136253  138630        38
…              …         …              …       …         …
63293   2019-05-03        145          145104  145308         4
197954  2018-11-19        180          180070  181430        14
120796  2019-04-26         47           47199   42610        14
197986  2019-02-27        180          180143  181934        14
207990  2018-08-23         84           84190   83832         4

                                       PROD_NAME  PROD_QTY  TOT_SALES  \
46694    Infuzions Mango     Chutny Papadums 70g         2        4.8
203179   Infuzions Mango     Chutny Papadums 70g         2        4.8
165827   Infuzions Mango     Chutny Papadums 70g         2        4.8
198867   Infuzions Mango     Chutny Papadums 70g         2        4.8
2690     Infuzions Mango     Chutny Papadums 70g         1        2.4
…                                              …       …          …
63293           Dorito Corn Chp     Supreme 380g         2       13.0
197954    Smiths Crnkle Chip  Orgnl Big Bag 380g         2       11.8
120796    Smiths Crnkle Chip  Orgnl Big Bag 380g         2       11.8
197986    Smiths Crnkle Chip  Orgnl Big Bag 380g         2       11.8
207990          Dorito Corn Chp     Supreme 380g         2       13.0

        PACK_SIZE
46694        70.0
203179       70.0
165827       70.0
198867       70.0
2690         70.0
…              …
63293       380.0
197954      380.0
```

```
120796      380.0
197986      380.0
207990      380.0

[246742 rows x 9 columns]
```

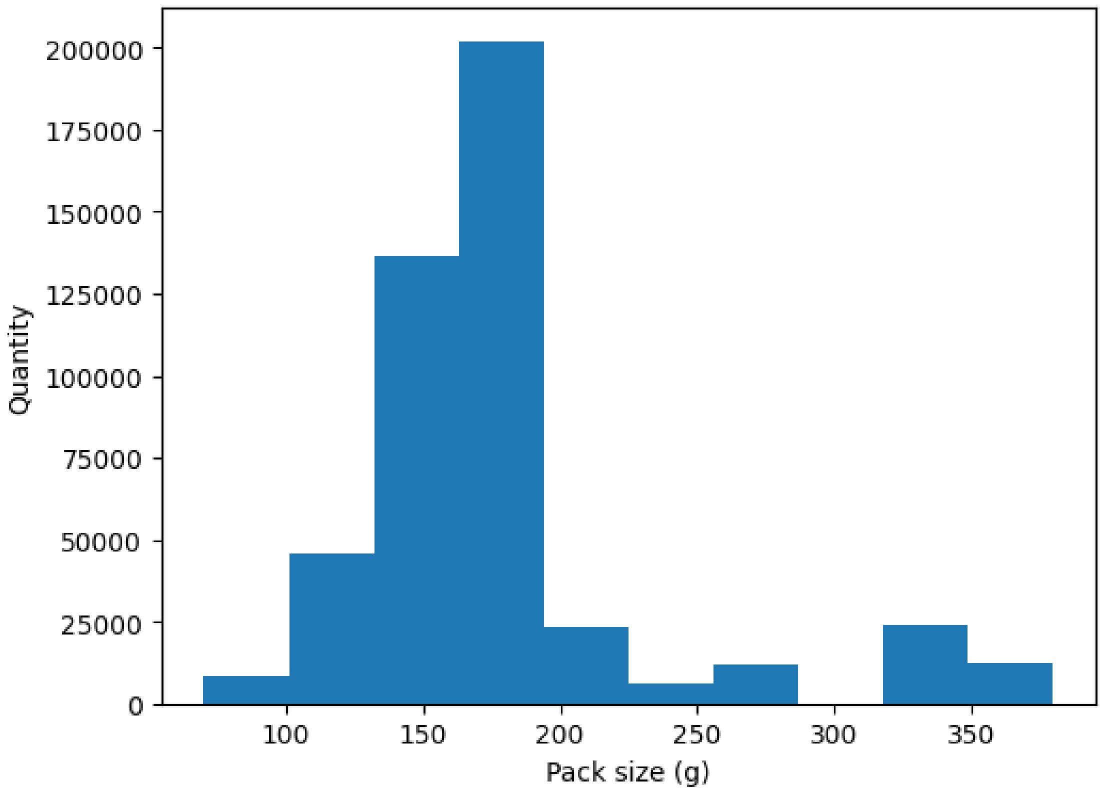
[33]:
```
df_transaction.PACK_SIZE.describe()
```

```
/usr/local/lib/python3.10/dist-packages/ipykernel/ipykernel.py:283:
DeprecationWarning: `should_run_async` will not call `transform_cell`
automatically in the future. Please pass the result to `transformed_cell`
argument and any exception that happen during thetransform in
`preprocessing_exc_tuple` in IPython 7.17 and above.
  and should_run_async(code)
```

[33]:
```
count    246742.000000
mean        175.585178
std          59.434727
min          70.000000
25%         150.000000
50%         170.000000
75%         175.000000
max         380.000000
Name: PACK_SIZE, dtype: float64
```

[34]:
```
# Plot a histogram to visualise distribution of pack sizes.
plt.hist(df_transaction['PACK_SIZE'], weights=df_transaction['PROD_QTY']);
plt.xlabel('Pack size (g)');
plt.ylabel('Quantity');
```

```
/usr/local/lib/python3.10/dist-packages/ipykernel/ipykernel.py:283:
DeprecationWarning: `should_run_async` will not call `transform_cell`
automatically in the future. Please pass the result to `transformed_cell`
argument and any exception that happen during thetransform in
`preprocessing_exc_tuple` in IPython 7.17 and above.
  and should_run_async(code)
```



Now that the pack size looks reasonable, we can create the brand names using the first word of each product name.

```
[35]: # Add a column to extract each product name in the first word
df_transaction.insert(9, "BRAND_NAME",df_transaction['PROD_NAME'].str.split().
  ↪str.get(0), True)
df_transaction
```

```
/usr/local/lib/python3.10/dist-packages/ipykernel/ipkernel.py:283:
DeprecationWarning: `should_run_async` will not call `transform_cell`
automatically in the future. Please pass the result to `transformed_cell`
argument and any exception that happen during thetransform in
`preprocessing_exc_tuple` in IPython 7.17 and above.
  and should_run_async(code)
```

```
[35]:             DATE  STORE_NBR  LYLTY_CARD_NBR  TXN_ID  PROD_NBR  \
0       2018-10-17          1            1000       1         5
1       2019-05-14          1            1307     348        66
2       2019-05-20          1            1343     383        61
3       2018-08-17          2            2373     974        69
4       2018-08-18          2            2426    1038       108
…            …        …              …      …       …
```

```
264831 2019-03-09        272          272319  270088        89
264832 2018-08-13        272          272358  270154        74
264833 2018-11-06        272          272379  270187        51
264834 2018-12-27        272          272379  270188        42
264835 2018-09-22        272          272380  270189        74

                                       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
…                                            …       …         …
264831   Kettle Sweet Chilli And Sour Cream 175g         2       10.8
264832             Tostitos Splash Of  Lime 175g         1        4.4
264833                  Doritos Mexicana    170g         2        8.8
264834   Doritos Corn Chip Mexican Jalapeno 150g         2        7.8
264835             Tostitos Splash Of  Lime 175g         2        8.8

        PACK_SIZE BRAND_NAME
0           175.0    Natural
1           175.0        CCs
2           170.0     Smiths
3           175.0     Smiths
4           150.0     Kettle
…             …        …
264831      175.0     Kettle
264832      175.0   Tostitos
264833      170.0    Doritos
264834      150.0    Doritos
264835      175.0   Tostitos

[246742 rows x 10 columns]
```

[36]:
```
# Check Brand Name
df_transaction["BRAND_NAME"].unique()
```

```
/usr/local/lib/python3.10/dist-packages/ipykernel/ipkernel.py:283:
DeprecationWarning: `should_run_async` will not call `transform_cell`
automatically in the future. Please pass the result to `transformed_cell`
argument and any exception that happen during thetransform in
`preprocessing_exc_tuple` in IPython 7.17 and above.
  and should_run_async(code)
```

[36]:
```
array(['Natural', 'CCs', 'Smiths', 'Kettle', 'Grain', 'Doritos',
       'Twisties', 'WW', 'Thins', 'Burger', 'NCC', 'Cheezels', 'Infzns',
       'Red', 'Pringles', 'Dorito', 'Infuzions', 'Smith', 'GrnWves',
       'Tyrrells', 'Cobs', 'French', 'RRD', 'Tostitos', 'Cheetos',
       'Woolworths', 'Snbts', 'Sunbites'], dtype=object)
```

Some brand names have been doubled up. Replace all contractions and double ups with their full name.

```
[37]: # Function to identify to map the brand names into the same brand name
def replace_name(line):
  name = line['BRAND_NAME']
  if name == "Infzns":
        return "Infuzions"
  elif name == "Red":
        return "Red Rock Deli"
  elif name == "RRD":
        return "Red Rock Deli"
  elif name == "Grain":
        return "Grain Waves"
  elif name == "GrnWves":
        return "Grain Waves"
  elif name == "Snbts":
        return "Sunbites"
  elif name == "Natural":
        return "Natural Chip Co"
  elif name == "NCC":
        return "Natural Chip Co"
  elif name == "WW":
        return "Woolworths"
  elif name == "Smith":
        return "Smiths"
  elif name == "Dorito":
        return "Doritos"
  else:
        return name
#Apply function in clean brand names
df_transaction.BRAND_NAME = df_transaction.apply(lambda line:␣
 ↪replace_name(line), axis=1)

#check duplicate
df_transaction.BRAND_NAME.unique()
```

```
/usr/local/lib/python3.10/dist-packages/ipykernel/ipkernel.py:283:
DeprecationWarning: `should_run_async` will not call `transform_cell`
automatically in the future. Please pass the result to `transformed_cell`
argument and any exception that happen during thetransform in
`preprocessing_exc_tuple` in IPython 7.17 and above.
  and should_run_async(code)
```

```
[37]: array(['Natural Chip Co', 'CCs', 'Smiths', 'Kettle', 'Grain Waves',
             'Doritos', 'Twisties', 'Woolworths', 'Thins', 'Burger', 'Cheezels',
             'Infuzions', 'Red Rock Deli', 'Pringles', 'Tyrrells', 'Cobs',
             'French', 'Tostitos', 'Cheetos', 'Sunbites'], dtype=object)
```

The brand names seme reasonable, without duplicates.

Now we want to examine the customer data. We can generate summaries and check the categories in this dataset.

```
[38]: # Examine Customer Data
      df_customer = customer.copy()
      df_customer.head()
```

```
/usr/local/lib/python3.10/dist-packages/ipykernel/ipkernel.py:283:
DeprecationWarning: `should_run_async` will not call `transform_cell`
automatically in the future. Please pass the result to `transformed_cell`
argument and any exception that happen during thetransform in
`preprocessing_exc_tuple` in IPython 7.17 and above.
  and should_run_async(code)
```

```
[38]:    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
```

```
[39]: #Rename column name into 'MEMBER_TYPE'
      df_customer = df_customer.rename(columns={'PREMIUM_CUSTOMER' : 'MEMBER_TYPE'})
```

```
/usr/local/lib/python3.10/dist-packages/ipykernel/ipkernel.py:283:
DeprecationWarning: `should_run_async` will not call `transform_cell`
automatically in the future. Please pass the result to `transformed_cell`
argument and any exception that happen during thetransform in
`preprocessing_exc_tuple` in IPython 7.17 and above.
  and should_run_async(code)
```

```
[40]: df_customer.describe()
```

```
/usr/local/lib/python3.10/dist-packages/ipykernel/ipkernel.py:283:
DeprecationWarning: `should_run_async` will not call `transform_cell`
automatically in the future. Please pass the result to `transformed_cell`
argument and any exception that happen during thetransform in
`preprocessing_exc_tuple` in IPython 7.17 and above.
  and should_run_async(code)
```

```
[40]:        LYLTY_CARD_NBR
       count    7.263700e+04
       mean     1.361859e+05
       std      8.989293e+04
       min      1.000000e+03
       25%      6.620200e+04
       50%      1.340400e+05
       75%      2.033750e+05
       max      2.373711e+06
```

```
[41]: # Chek the member type
      df_customer.MEMBER_TYPE.unique()
```

```
/usr/local/lib/python3.10/dist-packages/ipykernel/ipkernel.py:283:
DeprecationWarning: `should_run_async` will not call `transform_cell`
automatically in the future. Please pass the result to `transformed_cell`
argument and any exception that happen during thetransform in
`preprocessing_exc_tuple` in IPython 7.17 and above.
  and should_run_async(code)
```

```
[41]: array(['Premium', 'Mainstream', 'Budget'], dtype=object)
```

```
[42]: df_customer.LIFESTAGE.unique()
```

```
/usr/local/lib/python3.10/dist-packages/ipykernel/ipkernel.py:283:
DeprecationWarning: `should_run_async` will not call `transform_cell`
automatically in the future. Please pass the result to `transformed_cell`
argument and any exception that happen during thetransform in
`preprocessing_exc_tuple` in IPython 7.17 and above.
  and should_run_async(code)
```

```
[42]: array(['YOUNG SINGLES/COUPLES', 'YOUNG FAMILIES', 'OLDER SINGLES/COUPLES',
              'MIDAGE SINGLES/COUPLES', 'NEW FAMILIES', 'OLDER FAMILIES',
              'RETIREES'], dtype=object)
```

Now that the customer dataset looks fine, we want to add this information to the transactions dataset.

```
[43]: #Join customer data and transaction data then short transaction by date
      df_all = df_transaction.set_index('LYLTY_CARD_NBR').join(df_customer.
        ↪set_index('LYLTY_CARD_NBR'))
      df_all = df_all.reset_index()
      df_all = df_all.sort_values(by='DATE').reset_index(drop=True)
      df_all
```

```
/usr/local/lib/python3.10/dist-packages/ipykernel/ipkernel.py:283:
DeprecationWarning: `should_run_async` will not call `transform_cell`
automatically in the future. Please pass the result to `transformed_cell`
```

```
argument and any exception that happen during thetransform in
`preprocessing_exc_tuple` in IPython 7.17 and above.
  and should_run_async(code)
```

[43]:
```
        LYLTY_CARD_NBR       DATE  STORE_NBR  TXN_ID  PROD_NBR  \
0               207165 2018-07-01        207  205566        16
1                58195 2018-07-01         58   53678        26
2                58201 2018-07-01         58   53702        47
3                58242 2018-07-01         58   53871        62
4               207184 2018-07-01        207  205693        32
…                  …          …        …       …       …
246737          104100 2019-06-30        104  104327        18
246738          125122 2019-06-30        125  128751        20
246739          130290 2019-06-30        130  134866        16
246740           80151 2019-06-30         80   78870         3
246741           55142 2019-06-30         55   49322        78

                                       PROD_NAME  PROD_QTY  TOT_SALES  \
0       Smiths Crinkle Chips Salt & Vinegar 330g         2       11.4
1                   Pringles Sweet&Spcy BBQ 134g         2        7.4
2              Doritos Corn Chips  Original 170g         2        8.8
3                Pringles Mystery    Flavour 134g        2        7.4
4           Kettle Sea Salt     And Vinegar 175g         2       10.8
…                                             …       …         …
246737            Cheetos Chs & Bacon Balls 190g         2        6.6
246738          Doritos Cheese      Supreme 330g         2       11.4
246739  Smiths Crinkle Chips Salt & Vinegar 330g         2       11.4
246740  Kettle Sensations   Camembert & Fig 150g         2        9.2
246741          Thins Chips Salt &  Vinegar 175g         2        6.6

        PACK_SIZE BRAND_NAME              LIFESTAGE MEMBER_TYPE
0           330.0     Smiths  MIDAGE SINGLES/COUPLES  Mainstream
1           134.0   Pringles  MIDAGE SINGLES/COUPLES  Mainstream
2           170.0    Doritos                RETIREES      Budget
3           134.0   Pringles   OLDER SINGLES/COUPLES  Mainstream
4           175.0     Kettle                RETIREES     Premium
…             …         …                    …          …
246737      190.0    Cheetos   OLDER SINGLES/COUPLES  Mainstream
246738      330.0    Doritos   YOUNG SINGLES/COUPLES  Mainstream
246739      330.0     Smiths                RETIREES      Budget
246740      150.0     Kettle   OLDER SINGLES/COUPLES  Mainstream
246741      175.0      Thins                RETIREES  Mainstream

[246742 rows x 12 columns]
```

[44]:
```python
#check null
df_all.isnull().values.any()
```

```
/usr/local/lib/python3.10/dist-packages/ipykernel/ipkernel.py:283:
DeprecationWarning: `should_run_async` will not call `transform_cell`
automatically in the future. Please pass the result to `transformed_cell`
argument and any exception that happen during thetransform in
`preprocessing_exc_tuple` in IPython 7.17 and above.
  and should_run_async(code)
```

[44]:
```
False
```

[45]:
```python
df_all.to_csv('QVI_alldata.csv')
```

```
/usr/local/lib/python3.10/dist-packages/ipykernel/ipkernel.py:283:
DeprecationWarning: `should_run_async` will not call `transform_cell`
automatically in the future. Please pass the result to `transformed_cell`
argument and any exception that happen during thetransform in
`preprocessing_exc_tuple` in IPython 7.17 and above.
  and should_run_async(code)
```

0.1 Data Anlaysis on Customer Segments

Now that the data has been cleaned, we want to look for interesting insights in the chip market to help recommend a business strategy.

To do so, some metrics we want to consider are:

- Who spends the most on chips (total sales), describing customers by lifestage and how premium their general purchasing behaviour is
- How many customers are in each segment
- How many chips are bought per customer by segment
- What's the average chip price by customer segment

Some more information from the data team that we could ask for, to analyse with the chip information for more insight includes

- The customer's total spend over the period and total spend for each transaction to understand what proportion of their grocery spend is on chips.
- Spending on other snacks, such as crackers and biscuits, to determine the preference and the purchase frequency of chips compared to other snacks
- Proportion of customers in each customer segment overall to compare against the mix of customers who purchase chips

Firstly, we want to take a look at the split of the total sales by LIFESTAGE and MEMBER_TYPE.

[48]:
```python
# calculate total sales by lifestage and member type
customer_totsales = df_all.groupby(['LIFESTAGE','MEMBER_TYPE'],␣
↪as_index=False)['TOT_SALES'].agg(['sum'])
customer_totsales = customer_totsales.rename(columns={'sum' : 'sum_totsales'})
customer_totsales.sort_values(by='sum_totsales', ascending = False)
```

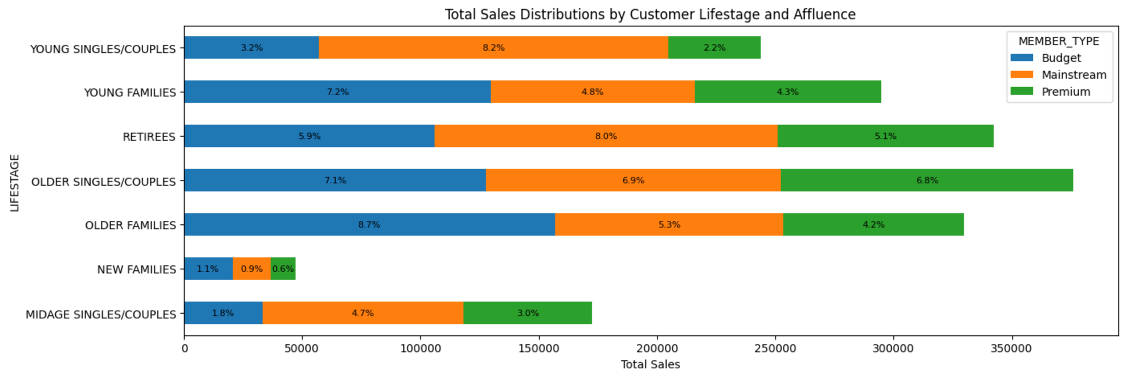
```
/usr/local/lib/python3.10/dist-packages/ipykernel/ipkernel.py:283:
DeprecationWarning: `should_run_async` will not call `transform_cell`
```

```
automatically in the future. Please pass the result to `transformed_cell`
argument and any exception that happen during thetransform in
`preprocessing_exc_tuple` in IPython 7.17 and above.
  and should_run_async(code)
```

[48]:
```
                  LIFESTAGE MEMBER_TYPE  sum_totsales
6            OLDER FAMILIES      Budget     156863.75
19   YOUNG SINGLES/COUPLES  Mainstream     147582.20
13                 RETIREES  Mainstream     145168.95
15           YOUNG FAMILIES      Budget     129717.95
9    OLDER SINGLES/COUPLES      Budget     127833.60
10   OLDER SINGLES/COUPLES  Mainstream     124648.50
11   OLDER SINGLES/COUPLES     Premium     123537.55
12                 RETIREES      Budget     105916.30
7            OLDER FAMILIES  Mainstream      96413.55
14                 RETIREES     Premium      91296.65
16           YOUNG FAMILIES  Mainstream      86338.25
1   MIDAGE SINGLES/COUPLES  Mainstream      84734.25
17           YOUNG FAMILIES     Premium      78571.70
8            OLDER FAMILIES     Premium      76542.60
18   YOUNG SINGLES/COUPLES      Budget      57122.10
2   MIDAGE SINGLES/COUPLES     Premium      54443.85
20   YOUNG SINGLES/COUPLES     Premium      39052.30
0   MIDAGE SINGLES/COUPLES      Budget      33345.70
3              NEW FAMILIES      Budget      20607.45
4              NEW FAMILIES  Mainstream      15979.70
5              NEW FAMILIES     Premium      10760.80
```

[50]:
```python
# Total Sales
totsales= df_all['TOT_SALES'].agg(['sum'])['sum']

# Breakdown the total sales by lifestage and member type
totsales_breakdown = df_all.groupby(['LIFESTAGE', 'MEMBER_TYPE'], as_index=␣
 ↪True)['TOT_SALES'].agg(['sum', 'mean']).unstack('MEMBER_TYPE').fillna(0)
ax= totsales_breakdown['sum'].plot(kind='barh', stacked=True, figsize=(15, 5))

# Add % of the summed total sales
for rect in ax.patches:
  #find where each label is located
  height = rect.get_height()
  width = rect.get_width()
  label = width / totsales*100
  x= rect.get_x()
  y=rect.get_y()

  label_text = f'{(label):.1f}%'
  #set label positions
  label_x = x + width / 2
  label_y = y + height / 2

  # plot labels > given width
  if width >0 :
    ax.text(label_x, label_y, label_text, ha='center', va='center', fontsize =8)
ax.set_xlabel('Total Sales')
ax.set_title('Total Sales Distributions by Customer Lifestage and Affluence')
plt.show()
```

```
/usr/local/lib/python3.10/dist-packages/ipykernel/ipkernel.py:283:
DeprecationWarning: `should_run_async` will not call `transform_cell`
automatically in the future. Please pass the result to `transformed_cell`
argument and any exception that happen during thetransform in
`preprocessing_exc_tuple` in IPython 7.17 and above.
  and should_run_async(code)
```



Here, we can see the most sales are from Older families - Budget, Young singles/couples - Mainstream and Retirees - Mainstream. We can see if this is because of the customer numbers in each segment.

```
[51]: # Check all rows are unique in customer info
len(df_customer['LYLTY_CARD_NBR'].unique()) == df_customer.shape[0]
```

```
/usr/local/lib/python3.10/dist-packages/ipykernel/ipkernel.py:283:
DeprecationWarning: `should_run_async` will not call `transform_cell`
automatically in the future. Please pass the result to `transformed_cell`
argument and any exception that happen during thetransform in
`preprocessing_exc_tuple` in IPython 7.17 and above.
  and should_run_async(code)
```

```
[51]: True
```

[52]:
```
# Check if all customers made chip purschase
len(df_customer['LYLTY_CARD_NBR'].unique()) == len(df_all['LYLTY_CARD_NBR'].
 ↪unique())
```

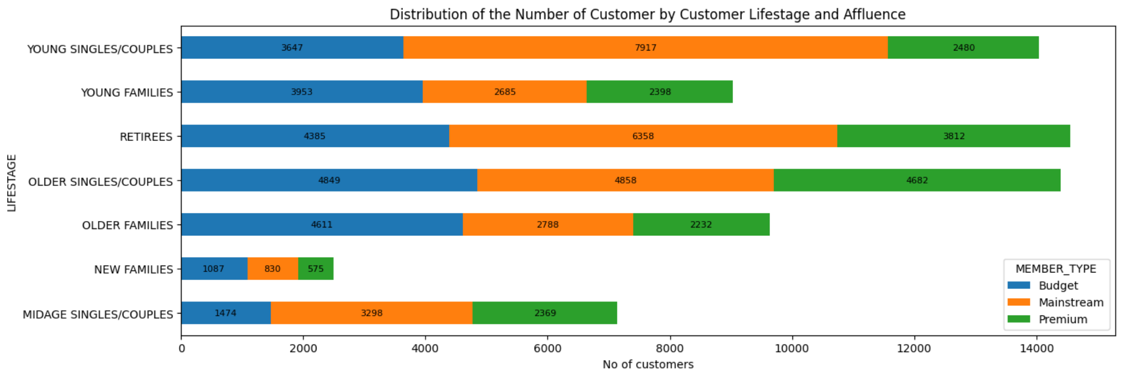
```
/usr/local/lib/python3.10/dist-packages/ipykernel/ipykernel.py:283:
DeprecationWarning: `should_run_async` will not call `transform_cell`
automatically in the future. Please pass the result to `transformed_cell`
argument and any exception that happen during thetransform in
`preprocessing_exc_tuple` in IPython 7.17 and above.
  and should_run_async(code)
```

[52]:
```
False
```

[56]:
```
# Plot the numbers of customers in each segment by counting the unique␣
 ↪LYLTY_CARD_NBR entries
sum_customer = df_all.groupby(['LIFESTAGE', 'MEMBER_TYPE'])['LYLTY_CARD_NBR'].
 ↪agg('nunique').unstack('MEMBER_TYPE').fillna(0)
ax = sum_customer.plot(kind='barh', stacked=True, figsize=(15, 5))

# Add customer numbers as label
for rect in ax.patches:
  #find where everything is located
  height = rect.get_height()
  width = rect.get_width()
  x = rect.get_x()
  y = rect.get_y()

  label_text = f'{(width):.0f}'

  #set label positions
  label_x = x + width / 2
  label_y = y + height / 2

  #only plot labels > given width
  if width > 0:
    ax.text(label_x, label_y, label_text, ha='center', va='center', fontsize=8)

ax.set_xlabel('No of customers')
ax.set_title('Distribution of the Number of Customer by Customer Lifestage and␣
 ↪Affluence')
plt.show()
```

```
/usr/local/lib/python3.10/dist-packages/ipykernel/ipykernel.py:283:
DeprecationWarning: `should_run_async` will not call `transform_cell`
automatically in the future. Please pass the result to `transformed_cell`
argument and any exception that happen during thetransform in
`preprocessing_exc_tuple` in IPython 7.17 and above.
```

```
  and should_run_async(code)
```

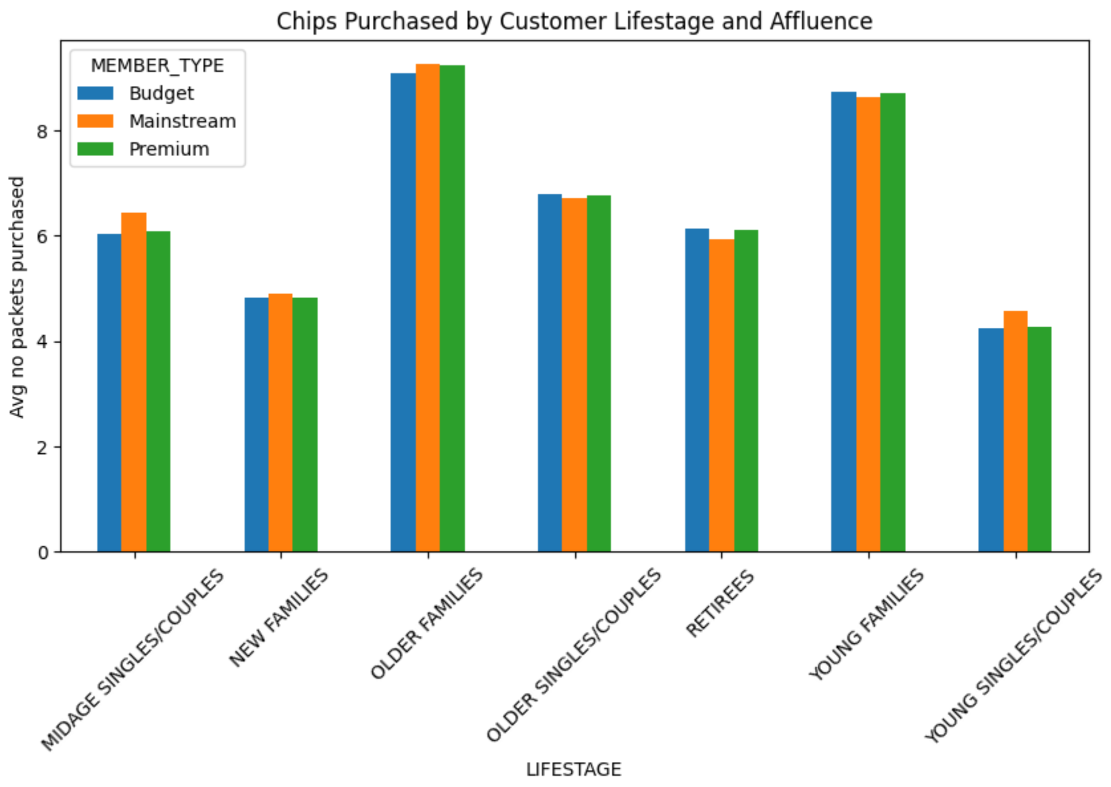


There are more Young singles/couples - mainstream and Retirees - mainstream who buy chips. This contributes to there being more sales to these customer segments but this is not a major driver for the Older families - budget segment.

We can then take a look at the total and average units of chips bought per customer by LIFESTAGE and MEMBER_TYPE.

```
[57]: # Plot the average no of chip pack bought per customer by LIFESTAGE and␣
       ↪MEMBER_TYPE.
      no_pack_data = df_all.groupby(['LIFESTAGE','MEMBER_TYPE'])['PROD_QTY'].sum()/
       ↪df_all.groupby(['LIFESTAGE','MEMBER_TYPE'])['LYLTY_CARD_NBR'].nunique(0)
      ax = no_pack_data.unstack('MEMBER_TYPE').fillna(0).plot.bar(stacked =␣
       ↪False,figsize=(10, 5))
      ax.set_ylabel("Avg no packets purchased")
      ax.set_title('Chips Purchased by Customer Lifestage and Affluence')
      plt.xticks(rotation=45)
      plt.show()
```

```
/usr/local/lib/python3.10/dist-packages/ipykernel/ipkernel.py:283:
DeprecationWarning: `should_run_async` will not call `transform_cell`
automatically in the future. Please pass the result to `transformed_cell`
argument and any exception that happen during thetransform in
`preprocessing_exc_tuple` in IPython 7.17 and above.
  and should_run_async(code)
```



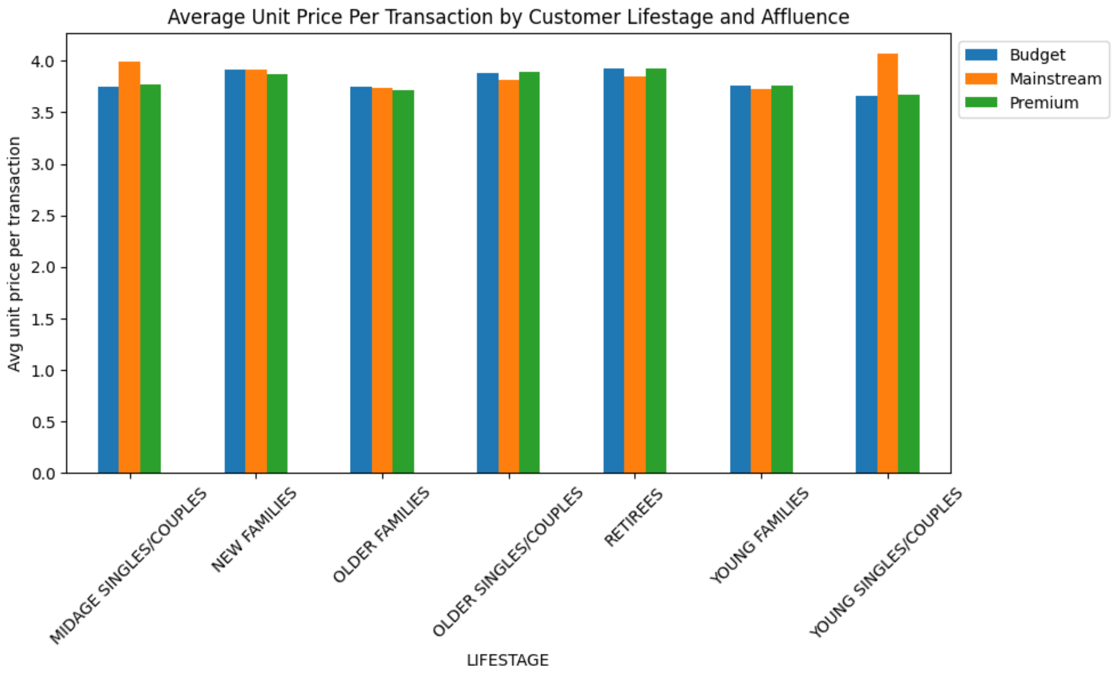
Older families and young families in general buy more chips per customer. We can also investigate the average price per unit sold by LIFESTAGE and MEMBER_TYPE.

[59]:
```
# Column for the unit price of chips purchased per transaction
df_all['UNIT_PRICE'] = df_all['TOT_SALES']/df_all['PROD_QTY']
```

```
/usr/local/lib/python3.10/dist-packages/ipykernel/ipkernel.py:283:
DeprecationWarning: `should_run_async` will not call `transform_cell`
automatically in the future. Please pass the result to `transformed_cell`
argument and any exception that happen during thetransform in
`preprocessing_exc_tuple` in IPython 7.17 and above.
  and should_run_async(code)
```

[61]:
```
# Plot the distribution of the average unit price per transaction by LIFESTAGE␣
↪and MEMBER_TYPE.
avg_priceperunit = df_all.groupby(['LIFESTAGE', 'MEMBER_TYPE'], as_index =␣
↪True)['UNIT_PRICE'].agg(['mean']).unstack('MEMBER_TYPE').fillna(0)
ax = avg_priceperunit['mean'].plot.bar(stacked=False, figsize=(10, 5))
ax.set_ylabel("Avg unit price per transaction")
ax.set_title('Average Unit Price Per Transaction by Customer Lifestage and␣
↪Affluence')
plt.legend(loc = "upper left",bbox_to_anchor=(1.0, 1.0))
plt.xticks(rotation=45)
plt.show()
```

```
/usr/local/lib/python3.10/dist-packages/ipykernel/ipkernel.py:283:
DeprecationWarning: `should_run_async` will not call `transform_cell`
automatically in the future. Please pass the result to `transformed_cell`
argument and any exception that happen during thetransform in
`preprocessing_exc_tuple` in IPython 7.17 and above.
  and should_run_async(code)
```



For young and midage singles/couples, the mainstream group are more willing to pay more for a packet of chips than their budget and premium counterpart. Given the total sales, as well as the number of customers buying chips, is higher in these groups compared to the non-mainstream groups, this suggests that chips may not be the choice of snack for these groups. Further information on shopping habits would be useful in this case.

As the difference in average price per unit isn't large, we can check if this difference is statistically different, with a t-test.

[62]:
```
# Check the difference in the average price unit between the mainstream and␣
 ↪premium/budget groups for young/midage singles/couples
from scipy.stats import ttest_ind

# Identify the groups to test the hypthesis with
mainstream = df_all["MEMBER_TYPE"] == "Mainstream"
```

```
young_midage = (df_all["LIFESTAGE"] == "MIDAGE SINGLES/COUPLES") |␣
↪(df_all["LIFESTAGE"] == "YOUNG SINGLES/COUPLES")
premium_budget = df_all["MEMBER_TYPE"] != "Mainstream"

group1 = df_all[mainstream & young_midage]["UNIT_PRICE"]
group2 = df_all[premium_budget & young_midage]["UNIT_PRICE"]

# Generate the t-test
stat, pval = ttest_ind(group1.values, group2.values, equal_var=False)

print(pval, stat)
```

```
/usr/local/lib/python3.10/dist-packages/ipykernel/ipkernel.py:283:
DeprecationWarning: `should_run_async` will not call `transform_cell`
automatically in the future. Please pass the result to `transformed_cell`
argument and any exception that happen during thetransform in
`preprocessing_exc_tuple` in IPython 7.17 and above.
  and should_run_async(code)
```

```
6.967354232991983e-306 37.6243885962296
```

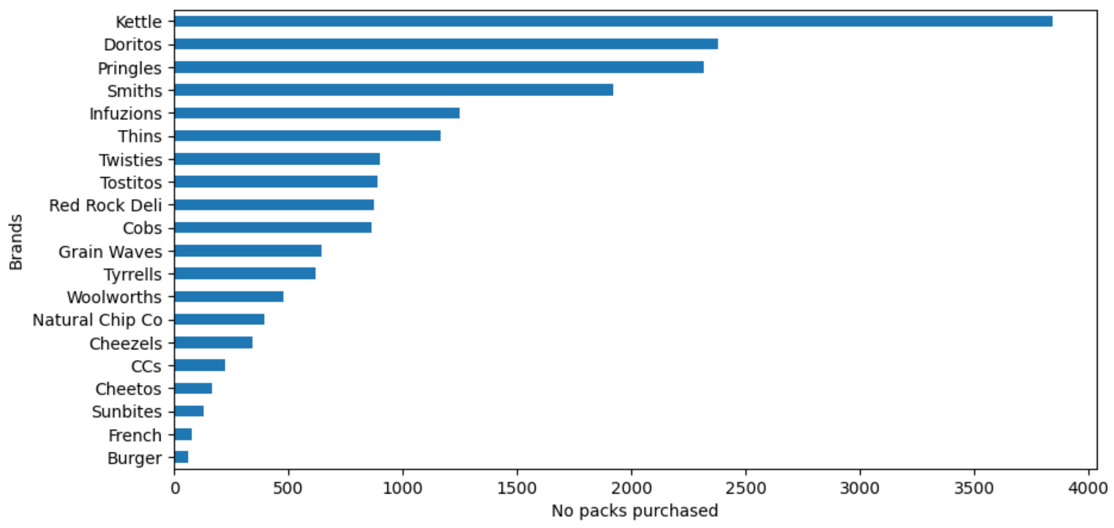
The **t-test results in a p-value of 6.97e-306, being close to 0,** indicates that the unit price for mainstream, young and mid-age singles and couples ARE significantly higher than that of budget or premium, young and midage singles and couples.

0.2 Deep Dive into Specific Customer Segments for Insights

We have found quite a few interesting insights that we can dive deeper into. We might want to target customer segments that contribute the most to sales to retain them or further increase sales. Let's look at Mainstream - young singles/couples. For instance, let's find out if they tend to buy a particular brand of chips.

```
[63]: # Visual of what kind of brand young singles/couples are purchasing the most
young_mainstream = df_all.loc[df_all['LIFESTAGE'] == 'YOUNG SINGLES/COUPLES']
young_mainstream = young_mainstream.
  ↪loc[young_mainstream['MEMBER_TYPE']=='Mainstream']
ax = young_mainstream['BRAND_NAME'].value_counts().sort_values(ascending=True).
  ↪plot.barh(figsize=(10,5))
ax.set_xlabel('No packs purchased')
ax.set_ylabel('Brands')
plt.show()
```

```
/usr/local/lib/python3.10/dist-packages/ipykernel/ipkernel.py:283:
DeprecationWarning: `should_run_async` will not call `transform_cell`
automatically in the future. Please pass the result to `transformed_cell`
argument and any exception that happen during thetransform in
`preprocessing_exc_tuple` in IPython 7.17 and above.
  and should_run_async(code)
```



We can see that : * Mainstream young singles/couples are 23% more likely to purchase Tyrrells chips compared to the rest of the population * Mainstream young singles/couples are 56% less likely to purchase Burger Rings compared to the rest of the population

Let's also find out if our target segment tends to buy larger packs of chips.

Conclusion Let's recap what we've found! Sales have mainly been due to Budget - older families, Mainstream - young singles/couples, and Mainstream -retirees shoppers. We found that the high spend in chips for mainstream young singles/couples and retirees is due to there being more of them than other buyers. Mainstream, midage and young singles and couples are also more likely to pay more per packet of chips. This is indicative of impulse buying behaviour. We've also found that Mainstream young singles and couples are 23% more likely to purchase Tyrrells chips compared to the rest of the population. The Category Manager may want to increase the category's performance by off-locating some Tyrrells and smaller packs of chips in discretionary space near segments where young singles and couples frequent more often to increase visibilty and impulse behaviour. Quantium can help the Category Manager with recommendations of where these segments are and further help them with measuring the impact of the changed placement. We'll work on measuring the impact of trials in the next task and putting all these together in the third task

```
[ ]:
```