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

```
tran_data = pd.read_excel("/content/QVI_transaction_data.xlsx")
```

```
tran_data.head()
```

| | DATE | STORE_NBR | LYLTY_CARD_NBR | TXN_ID | PROD_NBR | PROD_NAME | PROD_QTY | TOT_SALES |
|---|---|---|---|---|---|---|---|---|
| **0** | 43390 | 1 | 1000 | 1 | 5 | Natural Chip Compny SeaSalt175g | 2 | 6.0 |
| **1** | 43599 | 1 | 1307 | 348 | 66 | CCs Nacho Cheese 175g | 3 | 6.3 |
| **2** | 43605 | 1 | 1343 | 383 | 61 | Smiths Crinkle Cut Chips Chicken 170g | 2 | 2.9 |
| **3** | 43329 | 2 | 2373 | 974 | 69 | Smiths Chip Thinly S/Cream&Onion 175g | 5 | 15.0 |
| **4** | 43330 | 2 | 2426 | 1038 | 108 | Kettle Tortilla ChpsHny&Jlpno Chili 150g | 3 | 13.8 |

```
tran_data.describe()
```

| | DATE | STORE_NBR | LYLTY_CARD_NBR | TXN_ID | PROD_NBR | PROD_QTY | TOT_SALES |
|---|---|---|---|---|---|---|---|
| **count** | 264836.000000 | 264836.00000 | 2.648360e+05 | 2.648360e+05 | 264836.000000 | 264836.000000 | 264836.000000 |
| **mean** | 43464.036260 | 135.08011 | 1.355495e+05 | 1.351583e+05 | 56.583157 | 1.907309 | 7.304200 |
| **std** | 105.389282 | 76.78418 | 8.057998e+04 | 7.813303e+04 | 32.826638 | 0.643654 | 3.083226 |
| **min** | 43282.000000 | 1.00000 | 1.000000e+03 | 1.000000e+00 | 1.000000 | 1.000000 | 1.500000 |
| **25%** | 43373.000000 | 70.00000 | 7.002100e+04 | 6.760150e+04 | 28.000000 | 2.000000 | 5.400000 |
| **50%** | 43464.000000 | 130.00000 | 1.303575e+05 | 1.351375e+05 | 56.000000 | 2.000000 | 7.400000 |
| **75%** | 43555.000000 | 203.00000 | 2.030942e+05 | 2.027012e+05 | 85.000000 | 2.000000 | 9.200000 |
| **max** | 43646.000000 | 272.00000 | 2.373711e+06 | 2.415841e+06 | 114.000000 | 200.000000 | 650.000000 |

```
pur_bvr = pd.read_csv("/content/QVI_purchase_behaviour.csv")
```

```
pur_bvr.head()
```

| | 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 |

```
pur_bvr.describe()
```

| | 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 |

```
tran_data.isnull().sum()
```

```
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
```

```
pur_bvr.isnull().sum()
```

```
LYLTY_CARD_NBR      0
LIFESTAGE           0
PREMIUM_CUSTOMER    0
dtype: int64
```

```
merged_data = pd.merge(pur_bvr, tran_data, on = 'LYLTY_CARD_NBR', how = 'right')
merged_data.head()
```

| | LYLTY_CARD_NBR | LIFESTAGE | PREMIUM_CUSTOMER | DATE | STORE_NBR | TXN_ID | PROD_NBR | PROD_NAME | PROD_QTY | TOT_SALES |
|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 1000 | YOUNG SINGLES/COUPLES | Premium | 43390 | 1 | 1 | 5 | Natural Chip Compny SeaSalt175g | 2 | 6.0 |
| **1** | 1307 | MIDAGE SINGLES/COUPLES | Budget | 43599 | 1 | 348 | 66 | CCs Nacho Cheese 175g | 3 | 6.3 |
| **2** | 1343 | MIDAGE SINGLES/COUPLES | Budget | 43605 | 1 | 383 | 61 | Smiths Crinkle Cut Chips Chicken 170g | 2 | 2.9 |
| **3** | 2373 | MIDAGE | Budget | 43329 | 2 | 974 | 69 | Smiths Chip Thinly | 5 | 15.0 |

```
print(len(merged_data))
print(len(tran_data))
```

```
264836
264836
```

```
merged_data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 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: 22.2+ MB
```

```
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)


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

```
datetime64[ns]
```

```
merged_data["PROD_NAME"].unique()
```

```
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',
```

```
split_prods = merged_data["PROD_NAME"].str.replace(r'([0-9]+[gG])','').str.replace(r'[^\w]',' ').str.split()
```

```
<ipython-input-22-b22b46bd0072>:1: FutureWarning: The default value of regex will change from True to False in a future version.
  split_prods = merged_data["PROD_NAME"].str.replace(r'([0-9]+[gG])','').str.replace(r'[^\w]',' ').str.split()
```

```
word_counts = {}
def count_words(line):
  for word in line:
    if word not in word_counts:
      word_counts[word] = 1
    else:
      word_counts[word] += 1
split_prods.apply(lambda line: count_words(line))
print(pd.Series(word_counts).sort_values(ascending = False))
```

```
Chips       49770
Kettle      41288
Smiths      28860
Salt        27976
Cheese      27890
            ...
Sunbites     1432
Pc           1431
Garden       1419
NCC          1419
Fries        1418
Length: 198, dtype: int64
```

```
print(merged_data.describe(), '\n')
print(merged_data.info())
```

```
        LYLTY_CARD_NBR     STORE_NBR        TXN_ID       PROD_NBR  \
count    2.648360e+05  264836.00000  2.648360e+05  264836.000000
mean     1.355495e+05     135.08011  1.351583e+05      56.583157
std      8.057998e+04      76.78418  7.813303e+04      32.826638
min      1.000000e+03       1.00000  1.000000e+00       1.000000
25%      7.002100e+04      70.00000  6.760150e+04      28.000000
50%      1.303575e+05     130.00000  1.351375e+05      56.000000
75%      2.030942e+05     203.00000  2.027012e+05      85.000000
max      2.373711e+06     272.00000  2.415841e+06     114.000000

            PROD_QTY      TOT_SALES
count  264836.000000  264836.000000
mean        1.907309       7.304200
std         0.643654       3.083226
min         1.000000       1.500000
25%         2.000000       5.400000
50%         2.000000       7.400000
75%         2.000000       9.200000
max       200.000000     650.000000

<class 'pandas.core.frame.DataFrame'>
Int64Index: 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  datetime64[ns]
 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: datetime64[ns](1), float64(1), int64(5), object(3)
memory usage: 22.2+ MB
None
```

```
merged_data["PROD_QTY"].value_counts(bins=4).sort_index()
```

```
(0.8, 50.75]       264834
(50.75, 100.5]          0
(100.5, 150.25]         0
(150.25, 200.0]         2
Name: PROD_QTY, dtype: int64
```

```
merged_data.sort_values(by="PROD_QTY", ascending=False).head()
```

| | LYLTY_CARD_NBR | LIFESTAGE | PREMIUM_CUSTOMER | DATE | STORE_NBR | TXN_ID | PROD_NBR | PROD_NAME | PROD_QTY | TOT_SALES |
|---|---|---|---|---|---|---|---|---|---|---|
| **69762** | 226000 | OLDER FAMILIES | Premium | 2018-08-19 | 226 | 226201 | 4 | Dorito Corn Chp Supreme 380g | 200 | 650.0 |
| **69763** | 226000 | OLDER FAMILIES | Premium | 2019-05-20 | 226 | 226210 | 4 | Dorito Corn Chp Supreme 380g | 200 | 650.0 |
| **217237** | 201060 | YOUNG FAMILIES | Premium | 2019-05-18 | 201 | 200202 | 26 | Pringles Sweet&Spcy BBQ 134g | 5 | 18.5 |

```
merged_data = merged_data[merged_data["PROD_QTY"] < 6]
```

```
len(merged_data[merged_data["LYLTY_CARD_NBR"]==226000])
```

```
0
```

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

```
<ipython-input-29-d551bd00c70c>:1: FutureWarning: Treating datetime data as categorical rather than numeric in `.describe` is deprecatec
  merged_data["DATE"].describe()
count                  264834
unique                    364
top       2018-12-24 00:00:00
freq                      939
first     2018-07-01 00:00:00
last      2019-06-30 00:00:00
Name: DATE, dtype: object
```

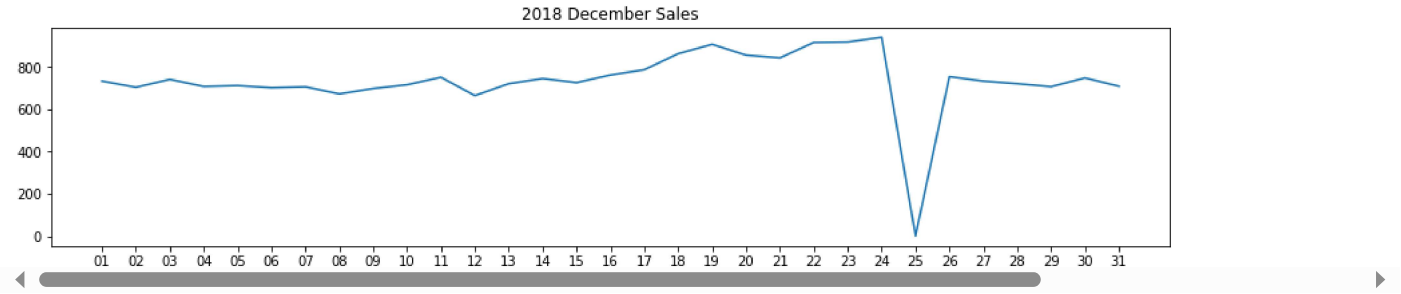
```
pd.date_range(start=merged_data["DATE"].min(), end=merged_data["DATE"].max()).difference(merged_data["DATE"])
```

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

```
check_null_date = pd.merge(pd.Series(pd.date_range(start=merged_data["DATE"].min(), end = merged_data["DATE"].max()), name="DATE"), merged_c
```

```
trans_by_date = check_null_date["DATE"].value_counts()
dec = trans_by_date[(trans_by_date.index >= pd.datetime(2018,12,1)) & (trans_by_date.index < pd.datetime(2019,1,1))].sort_index()
dec.index = dec.index.strftime('%d')
ax = dec.plot(figsize=(15,3))
ax.set_xticks(np.arange(len(dec)))
ax.set_xticklabels(dec.index)
plt.title("2018 December Sales")
plt.savefig("2018 December Sales.png", bbox_inches="tight")
plt.show()
```

```
<ipython-input-32-502b977c9a27>:2: FutureWarning: The pandas.datetime class is deprecated and will be removed from pandas in a future ve
  dec = trans_by_date[(trans_by_date.index >= pd.datetime(2018,12,1)) & (trans_by_date.index < pd.datetime(2019,1,1))].sort_index()
```

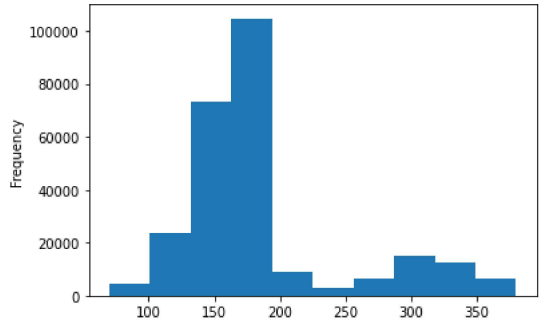



```
check_null_date["DATE"].value_counts().sort_values().head()
```

```
2018-12-25      1
2018-11-25    648
2018-10-18    658
2019-06-13    659
2019-06-24    662
Name: DATE, dtype: int64
```

```
merged_data["PROD_NAME"] = merged_data["PROD_NAME"].str.replace(r'[0-9]+(G)','g')
pack_sizes = merged_data["PROD_NAME"].str.extract(r'([0-9]+[gG])')[0].str.replace("g","").astype("float")
print(pack_sizes.describe())
pack_sizes.plot.hist()
```

```
<ipython-input-34-c0b8f769a815>:1: FutureWarning: The default value of regex will change from True to False in a future version.
  merged_data["PROD_NAME"] = merged_data["PROD_NAME"].str.replace(r'[0-9]+(G)','g')
count    258770.000000
mean        182.324276
std          64.955035
min          70.000000
25%         150.000000
50%         170.000000
75%         175.000000
max         380.000000
Name: 0, dtype: float64
<AxesSubplot:ylabel='Frequency'>
```




```
merged_data["PROD_NAME"].str.split().str[0].value_counts().sort_index()
```

```
Burger         1564
CCs            4551
Cheetos        2927
Cheezels       4603
Cobs           9693
Dorito         3183
Doritos       24962
French         1418
Grain          6272
GrnWves        1468
Infuzions     11057
Infzns         3144
Kettle        41288
NCC            1419
Natural        6050
Old            9324
Pringles      25102
RRD           11894
Red            5885
Smith          2963
```

```
Smiths        28860
Snbts          1576
Sunbites       1432
Thins         14075
Tostitos       9471
Twisties       9454
Tyrrells       6442
WW            10320
Woolworths     4437
Name: PROD_NAME, dtype: int64
```

```
merged_data["PROD_NAME"].str.split()[merged_data["PROD_NAME"].str.split().str[0] == "Red"].value_counts()
```

```
[Red, Rock, Deli, Sp, Salt, &, Truffle, g]        1498
[Red, Rock, Deli, Thai, Chilli&Lime, 150g]        1495
[Red, Rock, Deli, SR, Salsa, &, Mzzrlla, 150g]    1458
[Red, Rock, Deli, Chikn&Garlic, Aioli, 150g]      1434
Name: PROD_NAME, dtype: int64
```

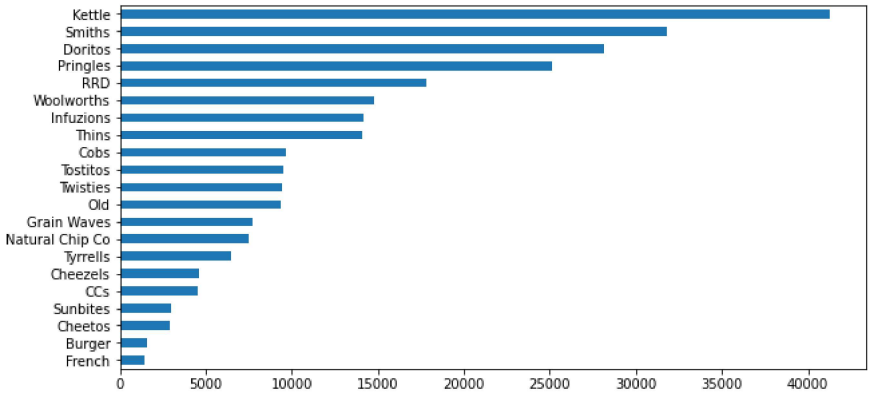
```
merged_data["Cleaned_Brand_Names"] = merged_data["PROD_NAME"].str.split().str[0]
```

```
def clean_brand_names(line):
    brand = line["Cleaned_Brand_Names"]
    if brand == "Dorito":
        return "Doritos"
    elif brand == "GrnWves" or brand == "Grain":
        return "Grain Waves"
    elif brand == "Infzns":
        return "Infuzions"
    elif brand == "Natural" or brand == "NCC":
        return "Natural Chip Co"
    elif brand == "Red":
        return "RRD"
    elif brand == "Smith":
        return "Smiths"
    elif brand == "Snbts":
        return "Sunbites"
    elif brand == "WW":
        return "Woolworths"
    else:
        return brand
```

```
merged_data["Cleaned_Brand_Names"] = merged_data.apply(lambda line: clean_brand_names(line), axis=1)
```

```
merged_data["Cleaned_Brand_Names"].value_counts(ascending=True).plot.barh(figsize=(10,5))
```

```
<AxesSubplot:>
```




```
merged_data.isnull().sum()
```

```
LYLTY_CARD_NBR        0
LIFESTAGE             0
PREMIUM_CUSTOMER      0
DATE                  0
STORE_NBR             0
TXN_ID                0
PROD_NBR              0
PROD_NAME             0
PROD_QTY              0
```

```
TOT_SALES             0
Cleaned_Brand_Names   0
dtype: int64
```

```
grouped_sales = pd.DataFrame(merged_data.groupby(["LIFESTAGE", "PREMIUM_CUSTOMER"])["TOT_SALES"].agg(["sum", "mean"]))
grouped_sales.sort_values(ascending=False, by="sum")
```

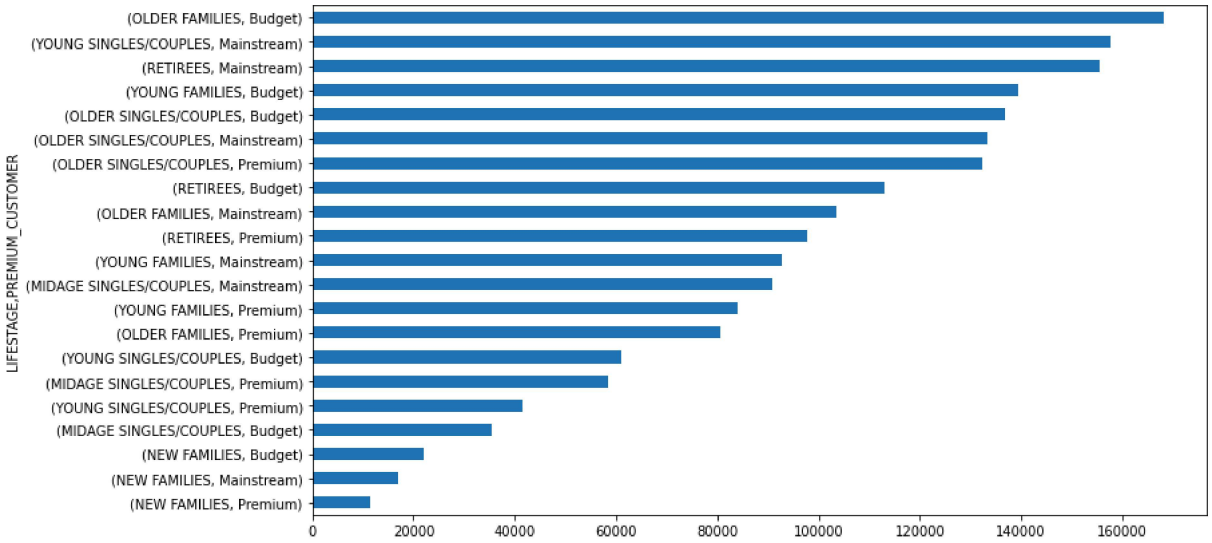
| | | sum | mean |
|---|---|---|---|
| LIFESTAGE | PREMIUM_CUSTOMER | | |
| OLDER FAMILIES | Budget | 168363.25 | 7.269570 |
| YOUNG SINGLES/COUPLES | Mainstream | 157621.60 | 7.558339 |
| RETIREES | Mainstream | 155677.05 | 7.252262 |
| YOUNG FAMILIES | Budget | 139345.85 | 7.287201 |
| OLDER SINGLES/COUPLES | Budget | 136769.80 | 7.430315 |
| | Mainstream | 133393.80 | 7.282116 |
| | Premium | 132263.15 | 7.449766 |
| RETIREES | Budget | 113147.80 | 7.443445 |
| OLDER FAMILIES | Mainstream | 103445.55 | 7.262395 |
| RETIREES | Premium | 97646.05 | 7.456174 |
| YOUNG FAMILIES | Mainstream | 92788.75 | 7.189025 |
| MIDAGE SINGLES/COUPLES | Mainstream | 90803.85 | 7.647284 |
| YOUNG FAMILIES | Premium | 84025.50 | 7.266756 |
| OLDER FAMILIES | Premium | 80658.40 | 7.208079 |
| YOUNG SINGLES/COUPLES | Budget | 61141.60 | 6.615624 |
| MIDAGE SINGLES/COUPLES | Premium | 58432.65 | 7.112056 |
| YOUNG SINGLES/COUPLES | Premium | 41642.10 | 6.629852 |
| MIDAGE SINGLES/COUPLES | Budget | 35514.80 | 7.074661 |
| NEW FAMILIES | Budget | 21928.45 | 7.297321 |
| | Mainstream | 17013.90 | 7.317806 |
| | Premium | 11491.10 | 7.231655 |

```
grouped_sales["sum"].sum()
```

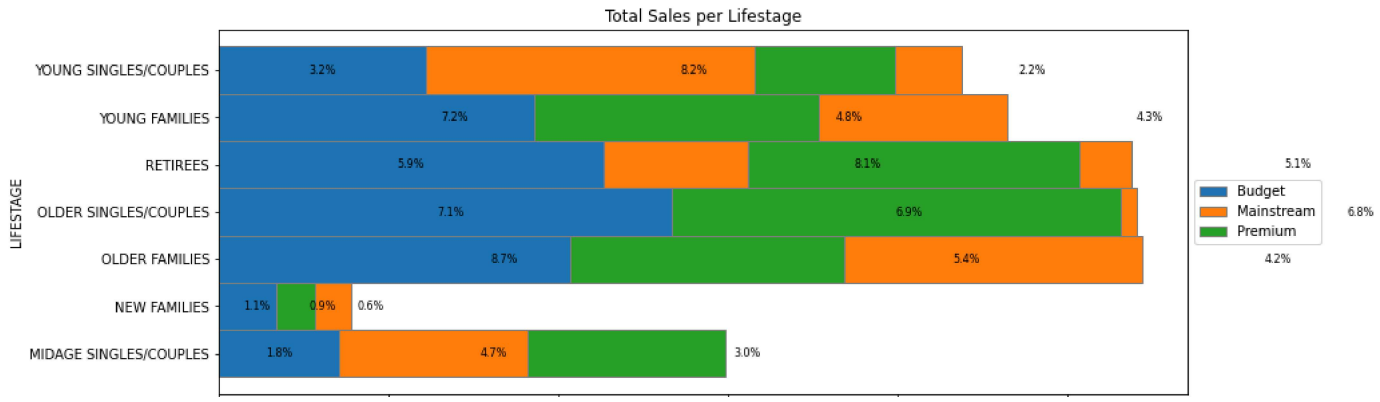
```
1933115.0000000002
```

```
grouped_sales["sum"].sort_values().plot.barh(figsize=(12,7))
```

```
<AxesSubplot:ylabel='LIFESTAGE,PREMIUM_CUSTOMER'>
```



```
# Values of each group
bars1 = grouped_sales[grouped_sales.index.get_level_values("PREMIUM_CUSTOMER") == "Budget"]["sum"]
bars2 = grouped_sales[grouped_sales.index.get_level_values("PREMIUM_CUSTOMER") == "Mainstream"]["sum"]
bars3 = grouped_sales[grouped_sales.index.get_level_values("PREMIUM_CUSTOMER") == "Premium"]["sum"]

bars1_text = (bars1 / sum(grouped_sales["sum"])).apply("{:.1%}".format)
bars2_text = (bars2 / sum(grouped_sales["sum"])).apply("{:.1%}".format)
bars3_text = (bars3 / sum(grouped_sales["sum"])).apply("{:.1%}".format)

# Names of group and bar width
names = grouped_sales.index.get_level_values("LIFESTAGE").unique()

# The position of the bars on the x-axis
r = np.arange(len(names))

plt.figure(figsize=(13,5))

# Create brown bars
budget_bar = plt.barh(r, bars1, edgecolor='grey', height=1, label="Budget")
# Create green bars (middle), on top of the firs ones
mains_bar = plt.barh(r, bars2, left=bars1, edgecolor='grey', height=1, label="Mainstream")
# Create green bars (top)
tmp_bar = np.add(bars1, bars2)
prem_bar = plt.barh(r, bars3, left=bars2, edgecolor='grey', height=1, label="Premium")

for i in range(7):
    budget_width = budget_bar[i].get_width()
    budget_main_width = budget_width + mains_bar[i].get_width()
    plt.text(budget_width/2, i, bars1_text[i], va='center', ha='center', size=8)
    plt.text(budget_width + mains_bar[i].get_width()/2, i, bars2_text[i], va='center', ha='center', size=8)
    plt.text(budget_main_width + prem_bar[i].get_width()/2, i, bars3_text[i], va='center', ha='center', size=8)

# Custom X axis
plt.yticks(r, names)
plt.ylabel("LIFESTAGE")
plt.xlabel("TOTAL SALES")
plt.legend(loc='center left', bbox_to_anchor=(1.0, 0.5))

plt.title("Total Sales per Lifestage")

plt.savefig("lifestage_sales.png", bbox_inches="tight")

# Show graphic
plt.show()
```




```
stage_agg_prem = merged_data.groupby("LIFESTAGE")["PREMIUM_CUSTOMER"].agg(pd.Series.mode).sort_values()
print("Top contributor per LIFESTAGE by PREMIUM category")
print(stage_agg_prem)
```

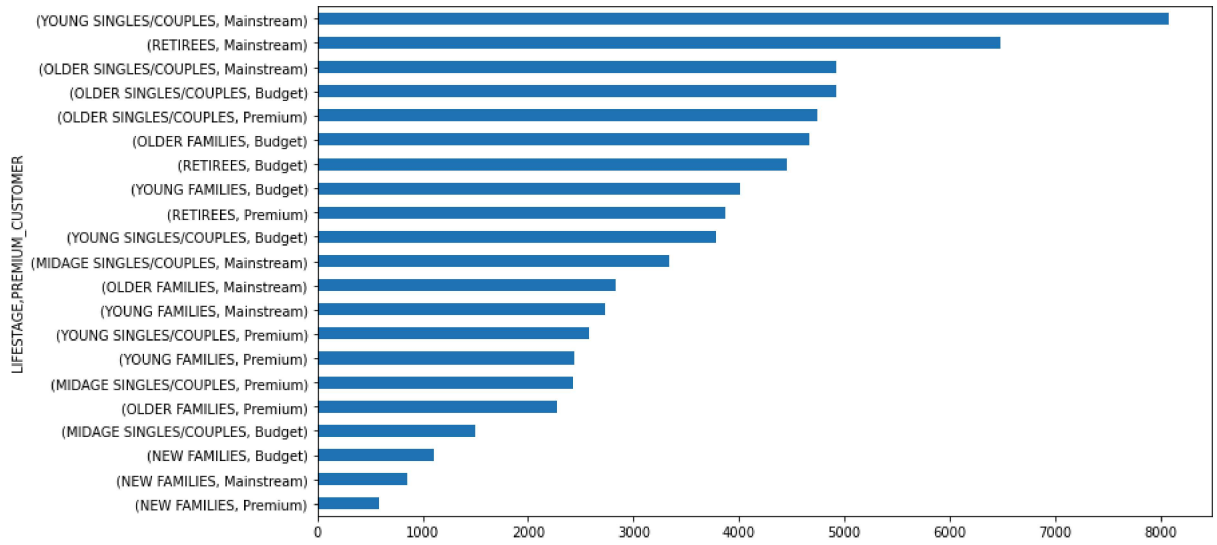
```
Top contributor per LIFESTAGE by PREMIUM category
LIFESTAGE
NEW FAMILIES                Budget
OLDER FAMILIES              Budget
OLDER SINGLES/COUPLES       Budget
YOUNG FAMILIES              Budget
MIDAGE SINGLES/COUPLES    Mainstream
RETIREES                  Mainstream
YOUNG SINGLES/COUPLES     Mainstream
Name: PREMIUM_CUSTOMER, dtype: object
```

```
unique_cust = merged_data.groupby(["LIFESTAGE", "PREMIUM_CUSTOMER"])["LYLTY_CARD_NBR"].nunique().sort_values(ascending=False)
pd.DataFrame(unique_cust)
```

| LIFESTAGE | PREMIUM_CUSTOMER | LYLTY_CARD_NBR |
|---|---|---|
| YOUNG SINGLES/COUPLES | Mainstream | 8088 |
| RETIREES | Mainstream | 6479 |
| OLDER SINGLES/COUPLES | Mainstream | 4930 |
| | Budget | 4929 |
| | Premium | 4750 |
| OLDER FAMILIES | Budget | 4675 |
| RETIREES | Budget | 4454 |
| YOUNG FAMILIES | Budget | 4017 |
| RETIREES | Premium | 3872 |
| YOUNG SINGLES/COUPLES | Budget | 3779 |
| MIDAGE SINGLES/COUPLES | Mainstream | 3340 |
| OLDER FAMILIES | Mainstream | 2831 |
| YOUNG FAMILIES | Mainstream | 2728 |
| YOUNG SINGLES/COUPLES | Premium | 2574 |
| YOUNG FAMILIES | Premium | 2433 |
| MIDAGE SINGLES/COUPLES | Premium | 2431 |
| OLDER FAMILIES | Premium | 2273 |
| MIDAGE SINGLES/COUPLES | Budget | 1504 |
| NEW FAMILIES | Budget | 1112 |
| | Mainstream | 849 |
| | Premium | 588 |

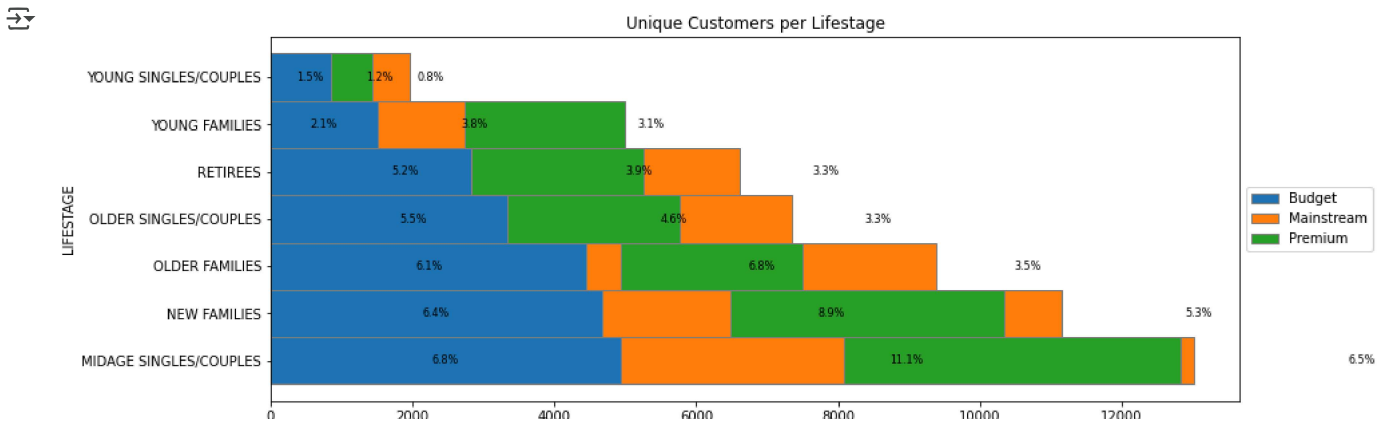
```
unique_cust.sort_values().plot.barh(figsize=(12,7))
```

```
<AxesSubplot:ylabel='LIFESTAGE,PREMIUM_CUSTOMER'>
```




```
# Values of each group
ncust_bars1 = unique_cust[unique_cust.index.get_level_values("PREMIUM_CUSTOMER") == "Budget"]
ncust_bars2 = unique_cust[unique_cust.index.get_level_values("PREMIUM_CUSTOMER") == "Mainstream"]
ncust_bars3 = unique_cust[unique_cust.index.get_level_values("PREMIUM_CUSTOMER") == "Premium"]

ncust_bars1_text = (ncust_bars1 / sum(unique_cust)).apply("{:.1%}".format)
ncust_bars2_text = (ncust_bars2 / sum(unique_cust)).apply("{:.1%}".format)
ncust_bars3_text = (ncust_bars3 / sum(unique_cust)).apply("{:.1%}".format)
```

```
# # Names of group and bar width
#names = unique_cust.index.get_level_values("LIFESTAGE").unique()

# # The position of the bars on the x-axis
#r = np.arange(len(names))

plt.figure(figsize=(13,5))

# # Create brown bars
budget_bar = plt.barh(r, ncust_bars1, edgecolor='grey', height=1, label="Budget")
# # Create green bars (middle), on top of the firs ones
mains_bar = plt.barh(r, ncust_bars2, left=ncust_bars1, edgecolor='grey', height=1, label="Mainstream")
# # Create green bars (top)
prem_bar = plt.barh(r, ncust_bars3, left=ncust_bars2, edgecolor='grey', height=1, label="Premium")

for i in range(7):
    budget_width = budget_bar[i].get_width()
    budget_main_width = budget_width + mains_bar[i].get_width()
    plt.text(budget_width/2, i, ncust_bars1_text[i], va='center', ha='center', size=8)
    plt.text(budget_width + mains_bar[i].get_width()/2, i, ncust_bars2_text[i], va='center', ha='center', size=8)
    plt.text(budget_main_width + prem_bar[i].get_width()/2, i, ncust_bars3_text[i], va='center', ha='center', size=8)

# Custom X axis
plt.yticks(r, names)
plt.ylabel("LIFESTAGE")
plt.xlabel("UNIQUE CUSTOMERS")
plt.legend(loc='center left', bbox_to_anchor=(1.0, 0.5))

plt.title("Unique Customers per Lifestage")

plt.savefig("lifestage_customers.png", bbox_inches="tight")

# # Show graphic
plt.show()
```




```
freq_per_cust = merged_data.groupby(["LYLTY_CARD_NBR", "LIFESTAGE", "PREMIUM_CUSTOMER"]).count()["DATE"]
freq_per_cust.groupby(["LIFESTAGE", "PREMIUM_CUSTOMER"]).agg(["mean", "count"]).sort_values(ascending=False, by="mean")
```

| | | mean | count |
|---|---|---|---|
| LIFESTAGE | PREMIUM_CUSTOMER | | |
| OLDER FAMILIES | Mainstream | 5.031438 | 2831 |
| | Budget | 4.954011 | 4675 |
| | Premium | 4.923009 | 2273 |
| YOUNG FAMILIES | Budget | 4.760269 | 4017 |
| | Premium | 4.752569 | 2433 |
| | Mainstream | 4.731305 | 2728 |
| OLDER SINGLES/COUPLES | Premium | 3.737684 | 4750 |
| | Budget | 3.734429 | 4929 |
| | Mainstream | 3.715619 | 4930 |
| MIDAGE SINGLES/COUPLES | Mainstream | 3.555090 | 3340 |
| RETIREES | Budget | 3.412887 | 4454 |
| | Premium | 3.382231 | 3872 |
| MIDAGE SINGLES/COUPLES | Premium | 3.379679 | 2431 |
| | Budget | 3.337766 | 1504 |
| RETIREES | Mainstream | 3.313166 | 6479 |
| NEW FAMILIES | Mainstream | 2.738516 | 849 |
| | Premium | 2.702381 | 588 |
| | Budget | 2.702338 | 1112 |
| YOUNG SINGLES/COUPLES | Mainstream | 2.578388 | 8088 |
| | Budget | 2.445621 | 3779 |
| | Premium | 2.440171 | 2574 |

```
grouped_sales.sort_values(ascending=False, by="mean")
```

| | | sum | mean |
|---|---|---|---|
| LIFESTAGE | PREMIUM_CUSTOMER | | |
| MIDAGE SINGLES/COUPLES | Mainstream | 90803.85 | 7.647284 |
| YOUNG SINGLES/COUPLES | Mainstream | 157621.60 | 7.558339 |
| RETIREES | Premium | 97646.05 | 7.456174 |
| OLDER SINGLES/COUPLES | Premium | 132263.15 | 7.449766 |
| RETIREES | Budget | 113147.80 | 7.443445 |
| OLDER SINGLES/COUPLES | Budget | 136769.80 | 7.430315 |
| NEW FAMILIES | Mainstream | 17013.90 | 7.317806 |
| | Budget | 21928.45 | 7.297321 |
| YOUNG FAMILIES | Budget | 139345.85 | 7.287201 |
| OLDER SINGLES/COUPLES | Mainstream | 133393.80 | 7.282116 |
| OLDER FAMILIES | Budget | 168363.25 | 7.269570 |
| YOUNG FAMILIES | Premium | 84025.50 | 7.266756 |
| OLDER FAMILIES | Mainstream | 103445.55 | 7.262395 |
| RETIREES | Mainstream | 155677.05 | 7.252262 |
| NEW FAMILIES | Premium | 11491.10 | 7.231655 |
| OLDER FAMILIES | Premium | 80658.40 | 7.208079 |
| YOUNG FAMILIES | Mainstream | 92788.75 | 7.189025 |
| MIDAGE SINGLES/COUPLES | Premium | 58432.65 | 7.112056 |
| | Budget | 35514.80 | 7.074661 |
| YOUNG SINGLES/COUPLES | Premium | 41642.10 | 6.629852 |
| | Budget | 61141.60 | 6.615624 |

```
from scipy.stats import ttest_ind
mainstream = merged_data["PREMIUM_CUSTOMER"] == "Mainstream"
young_midage = (merged_data["LIFESTAGE"] == "MIDAGE SINGLES/COUPLES") | (merged_data["LIFESTAGE"] == "YOUNG SINGLES/COUPLES")

budget_premium = (merged_data["PREMIUM_CUSTOMER"] == "Budget") | (merged_data["PREMIUM_CUSTOMER"] == "Premium")

a = merged_data[young_midage & mainstream]["TOT_SALES"]
b = merged_data[young_midage & budget_premium]["TOT_SALES"]
stat, pval = ttest_ind(a.values, b.values, equal_var=False)

print(pval)
pval < 0.0000001
```

```
1.8542040107536954e-281
True
```

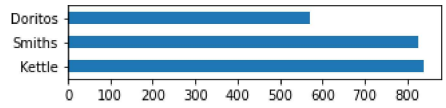
```
merged_data.groupby(["LIFESTAGE", "PREMIUM_CUSTOMER"])["Cleaned_Brand_Names"].agg(pd.Series.mode).sort_values()
```

```
LIFESTAGE               PREMIUM_CUSTOMER
MIDAGE SINGLES/COUPLES  Budget             Kettle
YOUNG FAMILIES          Premium            Kettle
                        Mainstream         Kettle
                        Budget             Kettle
RETIREES                Premium            Kettle
                        Mainstream         Kettle
                        Budget             Kettle
OLDER SINGLES/COUPLES   Premium            Kettle
YOUNG SINGLES/COUPLES   Mainstream         Kettle
OLDER SINGLES/COUPLES   Mainstream         Kettle
OLDER FAMILIES          Mainstream         Kettle
                        Budget             Kettle
NEW FAMILIES            Premium            Kettle
                        Mainstream         Kettle
                        Budget             Kettle
MIDAGE SINGLES/COUPLES  Premium            Kettle
                        Mainstream         Kettle
OLDER SINGLES/COUPLES   Budget             Kettle
YOUNG SINGLES/COUPLES   Premium            Kettle
```

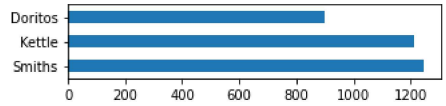
```
OLDER FAMILIES          Premium             Smiths
YOUNG SINGLES/COUPLES   Budget              Smiths
Name: Cleaned_Brand_Names, dtype: object
```

```
for stage in merged_data["LIFESTAGE"].unique():
    for prem in merged_data["PREMIUM_CUSTOMER"].unique():
        print('==========',stage, '-', prem,'==========')
        summary = merged_data[(merged_data["LIFESTAGE"] == stage) & (merged_data["PREMIUM_CUSTOMER"] == prem)]["Cleaned_Brand_Names"].value_
        print(summary)
        plt.figure()
        summary.plot.barh(figsize=(5,1))
        plt.show()
```

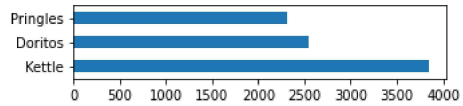
```
========== YOUNG SINGLES/COUPLES - Premium ==========
Kettle     838
Smiths     826
Doritos    570
Name: Cleaned_Brand_Names, dtype: int64
```



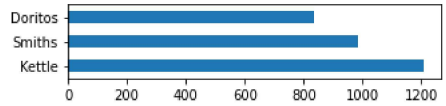

```
========== YOUNG SINGLES/COUPLES - Budget ==========
Smiths     1245
Kettle     1211
Doritos     899
Name: Cleaned_Brand_Names, dtype: int64
```



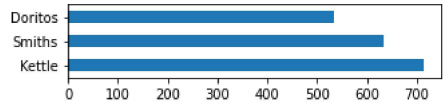

```
========== YOUNG SINGLES/COUPLES - Mainstream ==========
Kettle      3844
Doritos     2541
Pringles    2315
Name: Cleaned_Brand_Names, dtype: int64
```




```
========== MIDAGE SINGLES/COUPLES - Premium ==========
Kettle     1206
Smiths      986
Doritos     837
Name: Cleaned_Brand_Names, dtype: int64
```




```
========== MIDAGE SINGLES/COUPLES - Budget ==========
Kettle     713
Smiths     633
Doritos    533
Name: Cleaned_Brand_Names, dtype: int64
```




```
========== MIDAGE SINGLES/COUPLES - Mainstream ==========
Kettle     2136
Smiths     1337
Doritos    1291
Name: Cleaned_Brand_Names, dtype: int64
```

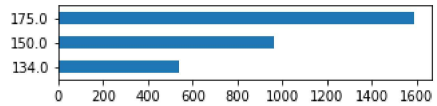

```
from mlxtend.frequent_patterns import apriori
from mlxtend.frequent_patterns import association_rules

temp = merged_data.reset_index().rename(columns = {"index": "transaction"})
temp["Segment"] = temp["LIFESTAGE"] + ' - ' + temp['PREMIUM_CUSTOMER']
segment_brand_encode = pd.concat([pd.get_dummies(temp["Segment"]), pd.get_dummies(temp["Cleaned_Brand_Names"])], axis=1)

frequent_sets = apriori(segment_brand_encode, min_support=0.01, use_colnames=True)
rules = association_rules(frequent_sets, metric="lift", min_threshold=1)
```

```
set_temp = temp["Segment"].unique()
rules[rules["antecedents"].apply(lambda x: list(x)).apply(lambda x: x in set_temp)]
```

| | antecedents | consequents | antecedent support | consequent support | support | confidence | lift | leverage | conviction |
|---|---|---|---|---|---|---|---|---|---|
| **0** | (OLDER FAMILIES - Budget) | (Smiths) | 0.087451 | 0.120162 | 0.011679 | 0.133549 | 1.111409 | 0.001171 | 1.015451 |
| **2** | (OLDER SINGLES/COUPLES - Budget) | (Kettle) | 0.069504 | 0.155901 | 0.011573 | 0.166513 | 1.068064 | 0.000738 | 1.012731 |
| **5** | (OLDER SINGLES/COUPLES - Premium) | (Kettle) | 0.067038 | 0.155901 | 0.011128 | 0.165991 | 1.064716 | 0.000676 | 1.012097 |
| **7** | (RETIREES - Mainstream) | (Kettle) | 0.081055 | 0.155901 | 0.012785 | 0.157738 | 1.011779 | 0.000149 | 1.002180 |

```
merged_pack = pd.concat([merged_data, pack_sizes.rename("Pack_Size")], axis=1)

for stage in merged_data["LIFESTAGE"].unique():
    for prem in merged_data["PREMIUM_CUSTOMER"].unique():
        print('==========',stage, '-', prem,'==========')
        summary = merged_pack[(merged_pack["LIFESTAGE"] == stage) & (merged_pack["PREMIUM_CUSTOMER"] == prem)]["Pack_Size"].value_counts().h
        print(summary)
        plt.figure()
        summary.plot.barh(figsize=(5,1))
        plt.show()
```

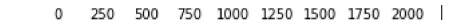
```
========== YOUNG SINGLES/COUPLES - Premium ==========
134.0     537
150.0     961
175.0    1587
Name: Pack_Size, dtype: int64
```



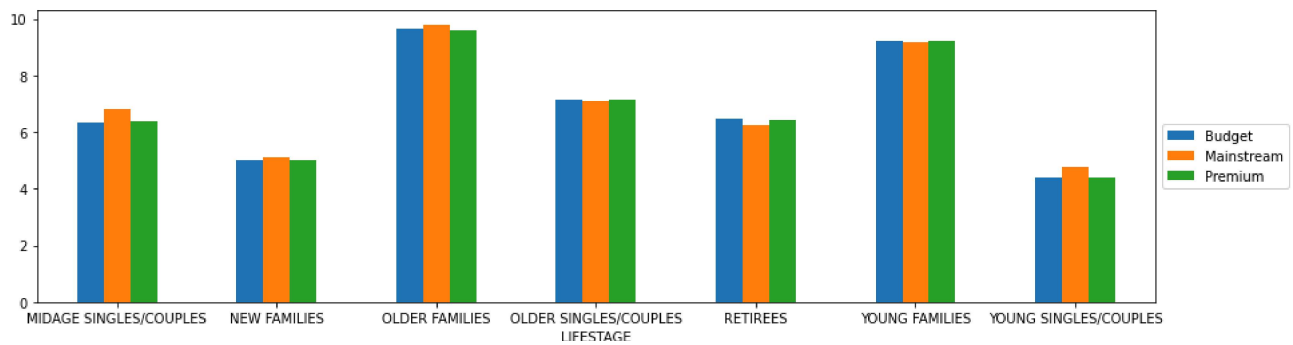

```
========== YOUNG SINGLES/COUPLES - Budget ==========
134.0     832
```

```
(temp.groupby(["LIFESTAGE", "PREMIUM_CUSTOMER"])["PROD_QTY"].sum() / temp.groupby(["LIFESTAGE", "PREMIUM_CUSTOMER"])["LYLTY_CARD_NBR"].nunic
```

```
LIFESTAGE               PREMIUM_CUSTOMER
OLDER FAMILIES          Mainstream          9.804309
                        Budget              9.639572
                        Premium             9.578091
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
```




```
(temp.groupby(["LIFESTAGE", "PREMIUM_CUSTOMER"])["PROD_QTY"].sum() / temp.groupby(["LIFESTAGE", "PREMIUM_CUSTOMER"])["LYLTY_CARD_NBR"].nunic
plt.legend(loc="center left", bbox_to_anchor=(1.0, 0.5))
plt.savefig("Average purchase quantity per segment.png", bbox_inches="tight")
```




```
#Average chips price per transaction by segments
temp["Unit_Price"] = temp["TOT_SALES"] / temp["PROD_QTY"]
temp.groupby(["Segment"]).mean()["Unit_Price"].sort_values(ascending=False)
```

```
Segment
YOUNG SINGLES/COUPLES - Mainstream     4.071485
MIDAGE SINGLES/COUPLES - Mainstream    4.000101
RETIREES - Budget                      3.924883
```