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

pur_bhvr = pd.read_csv('QVI_purchase_behaviour.csv')
print(pur_bhvr.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
```

```
tran_data = pd.read_csv('QVI_transaction_data.csv')
print(tran_data.head())
```

```
    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
```

```
merged_data = pd.merge(pur_bhvr,tran_data,on="LYLTY_CARD_NBR",
how="right")
print(merged_data.head())
```

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

   TXN_ID  PROD_NBR                                PROD_NAME
PROD_QTY  \
```

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

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

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

```
264836
264836
```

```
merged_data.info()
```

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

```
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',
       '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)
```

```
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))
```

```
175g        60561
Chips       49770
150g        41633
Kettle      41288
&           35565
            ...
Sunbites     1432
Pc           1431
NCC          1419
Garden       1419
Fries        1418
Length: 220, dtype: int64
```

```
merged_data =
merged_data[~merged_data["PROD_NAME"].str.contains(r"[Ss]alsa")]
```

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

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

              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

<class 'pandas.core.frame.DataFrame'>
Index: 246742 entries, 0 to 264835
Data columns (total 10 columns):
 #   Column            Non-Null Count   Dtype
---  ------            --------------   -----
 0   LYLTY_CARD_NBR    246742 non-null  int64
 1   LIFESTAGE         246742 non-null  object
 2   PREMIUM_CUSTOMER  246742 non-null  object
 3   DATE              246742 non-null  datetime64[ns]
 4   STORE_NBR         246742 non-null  int64
 5   TXN_ID            246742 non-null  int64
 6   PROD_NBR          246742 non-null  int64
 7   PROD_NAME         246742 non-null  object
 8   PROD_QTY          246742 non-null  int64
 9   TOT_SALES         246742 non-null  float64
dtypes: datetime64[ns](1), float64(1), int64(5), object(3)
memory usage: 20.7+ MB
None
```

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

```
(0.8, 50.75]       246740
(50.75, 100.5]          0
(100.5, 150.25]         0
(150.25, 200.0]         2
Name: count, dtype: int64
```

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

```
        LYLTY_CARD_NBR       LIFESTAGE PREMIUM_CUSTOMER       DATE
STORE_NBR  \
69763           226000  OLDER FAMILIES          Premium 2019-05-20
226
69762           226000  OLDER FAMILIES          Premium 2018-08-19
226
135225           46296        RETIREES           Budget 2019-05-15
46
69523            71142  OLDER FAMILIES          Premium 2019-05-15
71
69502            55144  OLDER FAMILIES          Premium 2018-08-18
55

        TXN_ID  PROD_NBR                          PROD_NAME
PROD_QTY  \
69763   226210         4  Dorito Corn Chp     Supreme 380g       200

69762   226201         4  Dorito Corn Chp     Supreme 380g       200

135225   42138        81   Pringles Original   Crisps 134g         5

69523    69852        96    WW Original Stacked Chips 160g         5

69502    49328        44    Thins Chips Light&  Tangy 175g         5


        TOT_SALES
69763       650.0
69762       650.0
135225       18.5
69523         9.5
69502        16.5
```

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

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

```
0
```

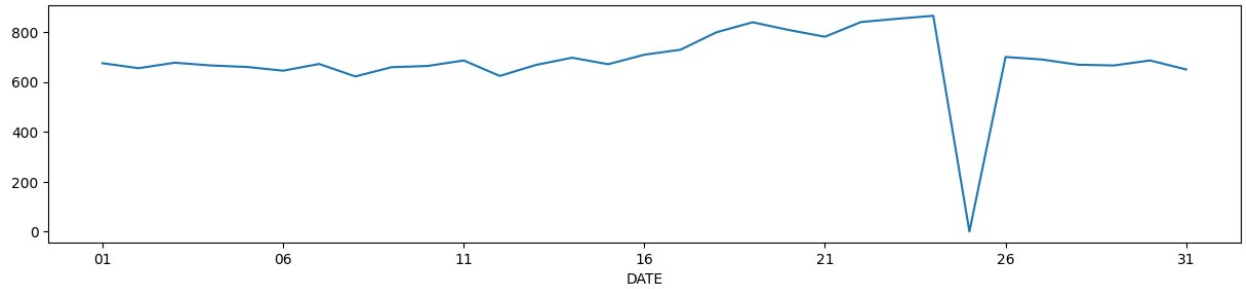
```
merged_data["DATE"].describe()
```

```
count                            246740
mean     2018-12-30 01:18:58.448569344
min                2018-07-01 00:00:00
25%                2018-09-30 00:00:00
50%                2018-12-30 00:00:00
75%                2019-03-31 00:00:00
max                2019-06-30 00:00:00
Name: DATE, dtype: object
```

```
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='D')
```

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

trans_by_date = check_null_date["DATE"].value_counts()
dec = trans_by_date[(trans_by_date.index >= pd.to_datetime('2018-12-
01')) & (trans_by_date.index < pd.to_datetime('2019-01-
01'))].sort_index()
dec.index = dec.index.strftime('%d')
ax = dec.plot(figsize=(15, 3))
```




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

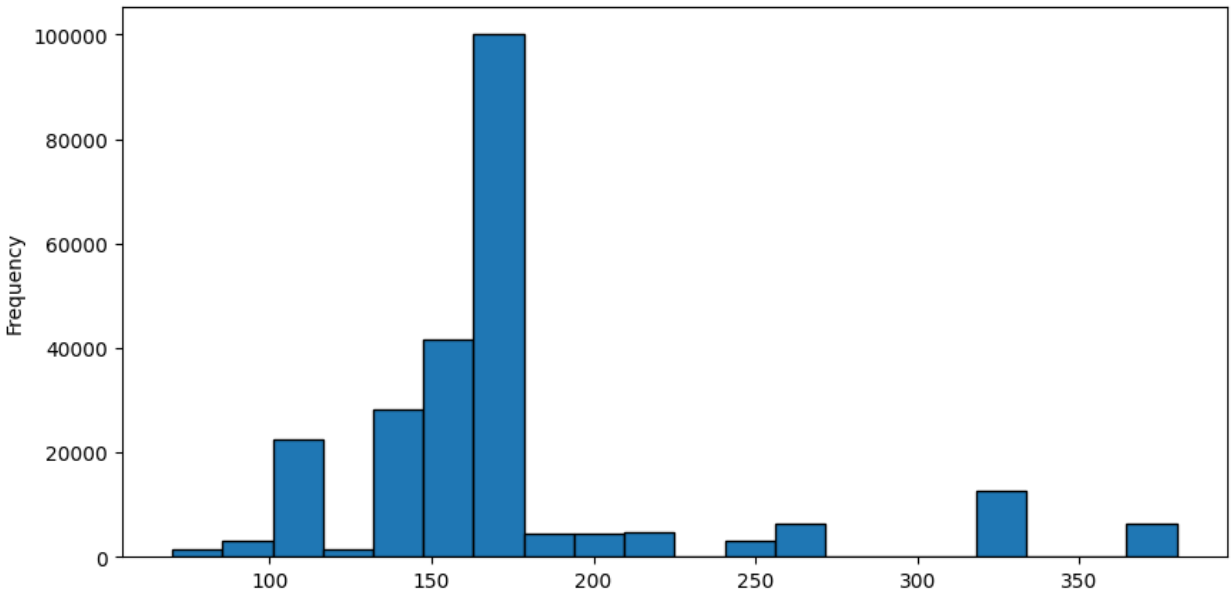
```
DATE
2018-12-25      1
2019-06-13    607
2018-09-22    609
2018-11-25    610
2018-10-18    611
Name: count, dtype: int64
```

```
# Fix "G" to "g" after the number
merged_data["PROD_NAME"] = merged_data["PROD_NAME"].str.replace(r'[0-
9]+(G)', 'g', regex=True)

# Extract pack size (numbers followed by g or G)
pack_sizes = merged_data["PROD_NAME"].str.extract(r'([0-9]+[gG])')[0]

# Remove 'g' and convert to float
pack_sizes = pack_sizes.str.replace('g', '',
regex=False).astype(float)

# Summary statistics
print(pack_sizes.describe())
```

```
# Plotting histogram
pack_sizes.plot.hist(bins=20, figsize=(10,5), edgecolor='black')
```

```
count    240676.000000
mean        175.302286
std          60.014468
min          70.000000
25%         150.000000
50%         170.000000
75%         175.000000
max         380.000000
Name: 0, dtype: float64
```

```
<Axes: ylabel='Frequency'>
```




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

```
PROD_NAME
Burger         1564
CCs            4551
Cheetos        2927
Cheezels       4603
Cobs           9693
Dorito         3183
Doritos       22041
French         1418
Grain          6272
GrnWves        1468
Infuzions     11057
Infzns         3144
```

```
Kettle        41288
NCC            1419
Natural        6050
Pringles      25102
RRD           11894
Red            4427
Smith          2963
Smiths        27390
Snbts          1576
Sunbites       1432
Thins         14075
Tostitos       9471
Twisties       9454
Tyrrells       6442
WW            10320
Woolworths     1516
Name: count, dtype: int64
```

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

```
PROD_NAME
[Grain, Waves, Sweet, Chilli, 210g]      3167
[Grain, Waves, Sour, Cream&Chives, g]    3105
Name: count, dtype: int64
```

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

```
PROD_NAME
[Natural, Chip, Co, Tmato, Hrb&Spce, 175g]       1572
[Natural, ChipCo, Sea, Salt, &, Vinegr, 175g]    1550
[Natural, Chip, Compny, SeaSalt175g]             1468
[Natural, ChipCo, Hony, Soy, Chckn175g]          1460
Name: count, dtype: int64
```

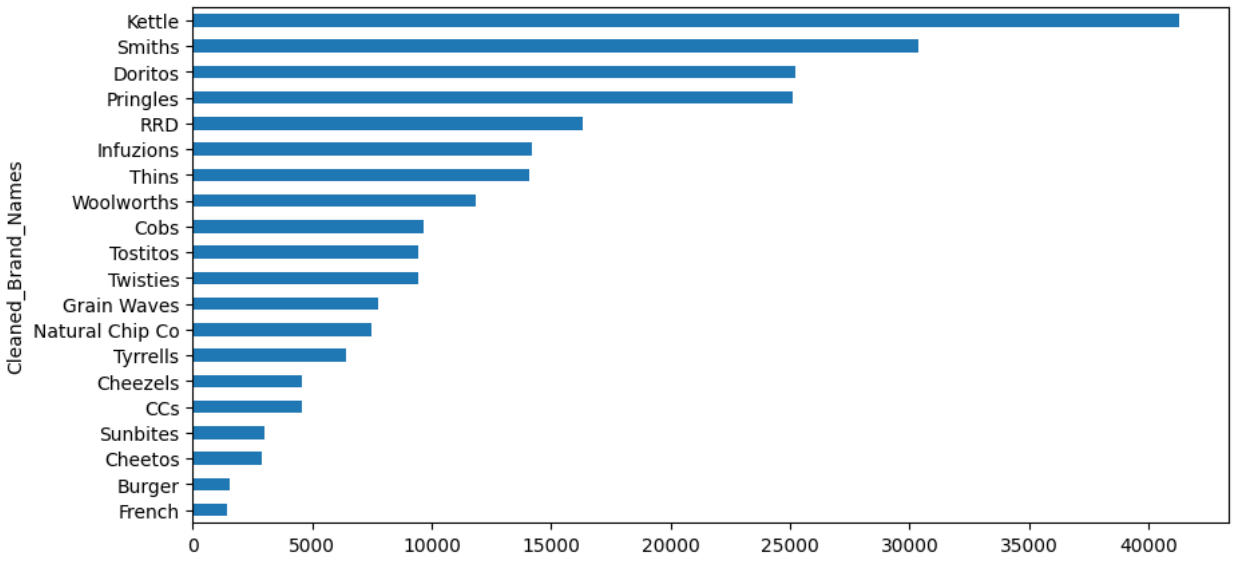
```
merged_data["PROD_NAME"].str.split()
[merged_data["PROD_NAME"].str.split().str[0] == "Red"].value_counts()
```

```
PROD_NAME
[Red, Rock, Deli, Sp, Salt, &, Truffle, g]      1498
[Red, Rock, Deli, Thai, Chilli&Lime, 150g]      1495
[Red, Rock, Deli, Chikn&Garlic, Aioli, 150g]    1434
Name: count, 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.b
arh(figsize=(10,5))
```

```
<Axes: ylabel='Cleaned_Brand_Names'>
```




```
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")
```

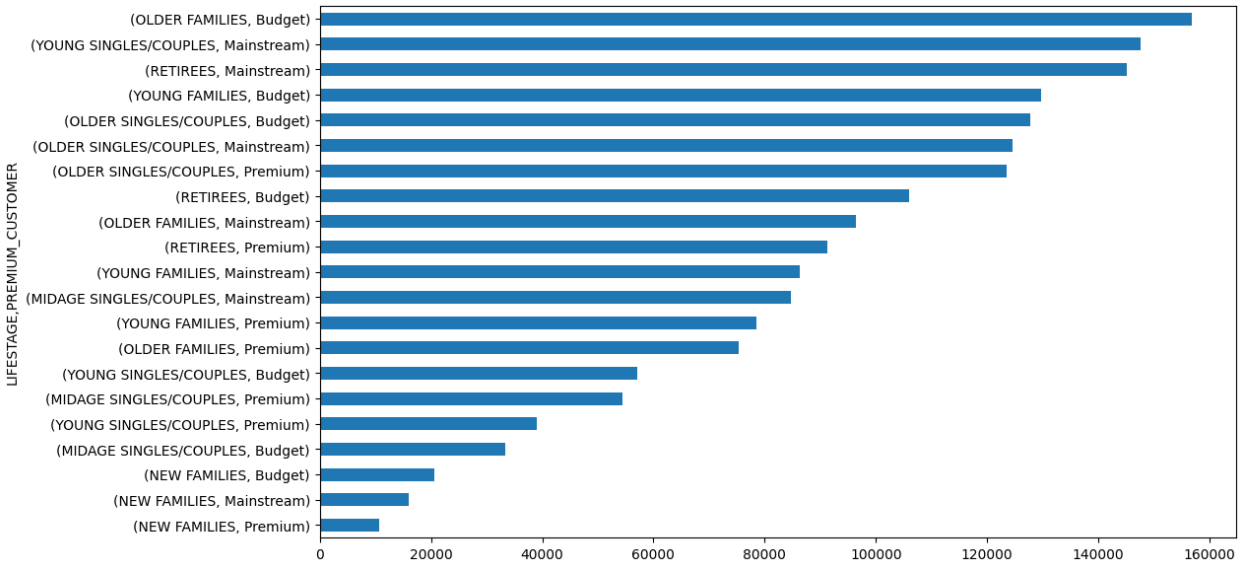
		sum	mean
LIFESTAGE	PREMIUM_CUSTOMER		
OLDER FAMILIES	Budget	156863.75	7.291241
YOUNG SINGLES/COUPLES	Mainstream	147582.20	7.551279
RETIREES	Mainstream	145168.95	7.269352
YOUNG FAMILIES	Budget	129717.95	7.302705
OLDER SINGLES/COUPLES	Budget	127833.60	7.444305
	Mainstream	124648.50	7.306049
	Premium	123537.55	7.459997
RETIREES	Budget	105916.30	7.445786
OLDER FAMILIES	Mainstream	96413.55	7.281440
RETIREES	Premium	91296.65	7.461315
YOUNG FAMILIES	Mainstream	86338.25	7.226772
MIDAGE SINGLES/COUPLES	Mainstream	84734.25	7.637156
YOUNG FAMILIES	Premium	78571.70	7.285951
OLDER FAMILIES	Premium	75242.60	7.232779
YOUNG SINGLES/COUPLES	Budget	57122.10	6.663023
MIDAGE SINGLES/COUPLES	Premium	54443.85	7.152371
YOUNG SINGLES/COUPLES	Premium	39052.30	6.673325
MIDAGE SINGLES/COUPLES	Budget	33345.70	7.108442
NEW FAMILIES	Budget	20607.45	7.297256
	Mainstream	15979.70	7.313364
	Premium	10760.80	7.231720

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

```
1805177.7
```

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

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

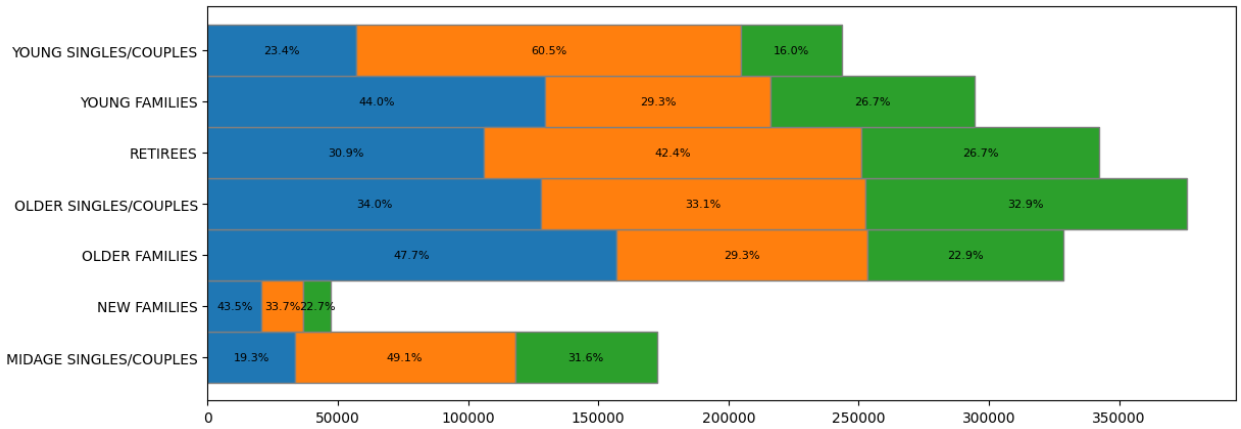



```
# Pivot the data first
pivot_sales = grouped_sales.reset_index().pivot(index='LIFESTAGE',
columns='PREMIUM_CUSTOMER', values='sum').fillna(0)

# Now your bars will always align perfectly
bars1 = pivot_sales["Budget"]
bars2 = pivot_sales["Mainstream"]
bars3 = pivot_sales["Premium"]

# Recalculate the percentages
total_sum = bars1 + bars2 + bars3
bars1_text = (bars1 / total_sum).apply("{:.1%}".format)
bars2_text = (bars2 / total_sum).apply("{:.1%}".format)
bars3_text = (bars3 / total_sum).apply("{:.1%}".format)

# x-axis labels
names = pivot_sales.index
r = np.arange(len(names))

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

budget_bar = plt.barh(r, bars1, edgecolor='grey', height=1,
label="Budget")
mains_bar = plt.barh(r, bars2, left=bars1, edgecolor='grey', height=1,
label="Mainstream")
prem_bar = plt.barh(r, bars3, left=bars1 + bars2, edgecolor='grey',
height=1, label="Premium")

for i in range(len(names)):
    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.iloc[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)

plt.yticks(r, names)
```

```
C:\Users\Lenovo\AppData\Local\Temp\ipykernel_3732\231116412.py:29:
FutureWarning: Series.__getitem__ treating keys as positions is
deprecated. In a future version, integer keys will always be treated
as labels (consistent with DataFrame behavior). To access a value by
position, use `ser.iloc[pos]`
  plt.text(budget_width + mains_bar[i].get_width()/2, i,
bars2_text[i], va='center', ha='center', size=8)
C:\Users\Lenovo\AppData\Local\Temp\ipykernel_3732\231116412.py:30:
FutureWarning: Series.__getitem__ treating keys as positions is
deprecated. In a future version, integer keys will always be treated
as labels (consistent with DataFrame behavior). To access a value by
position, use `ser.iloc[pos]`
  plt.text(budget_main_width + prem_bar[i].get_width()/2, i,
bars3_text[i], va='center', ha='center', size=8)
```

```
([<matplotlib.axis.YTick at 0x1fca8e0ab70>,
  <matplotlib.axis.YTick at 0x1fca8e111f0>,
  <matplotlib.axis.YTick at 0x1fca8e03cb0>,
  <matplotlib.axis.YTick at 0x1fca8df1bb0>,
  <matplotlib.axis.YTick at 0x1fca8df12e0>,
  <matplotlib.axis.YTick at 0x1fca8df05c0>,
  <matplotlib.axis.YTick at 0x1fca8debe30>],
 [Text(0, 0, 'MIDAGE SINGLES/COUPLES'),
  Text(0, 1, 'NEW FAMILIES'),
  Text(0, 2, 'OLDER FAMILIES'),
  Text(0, 3, 'OLDER SINGLES/COUPLES'),
  Text(0, 4, 'RETIREES'),
  Text(0, 5, 'YOUNG FAMILIES'),
  Text(0, 6, 'YOUNG SINGLES/COUPLES')])
```




```
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
```

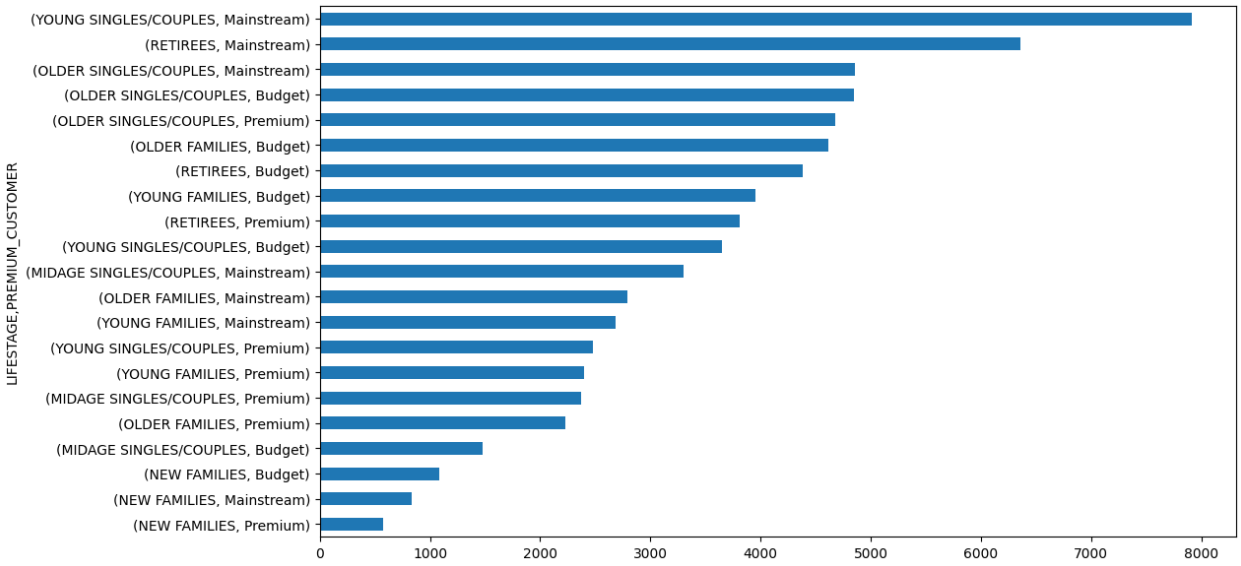
```
unique_cust = merged_data.groupby(["LIFESTAGE", "PREMIUM_CUSTOMER"])
["LYLTY_CARD_NBR"].nunique().sort_values(ascending=False)
pd.DataFrame(unique_cust)
```

LIFESTAGE	PREMIUM_CUSTOMER	LYLTY_CARD_NBR
YOUNG SINGLES/COUPLES	Mainstream	7917
RETIREES	Mainstream	6358
OLDER SINGLES/COUPLES	Mainstream	4858
	Budget	4849
	Premium	4682
OLDER FAMILIES	Budget	4611
RETIREES	Budget	4385
YOUNG FAMILIES	Budget	3953
RETIREES	Premium	3812
YOUNG SINGLES/COUPLES	Budget	3647
MIDAGE SINGLES/COUPLES	Mainstream	3298
OLDER FAMILIES	Mainstream	2788
YOUNG FAMILIES	Mainstream	2685
YOUNG SINGLES/COUPLES	Premium	2480

```
YOUNG FAMILIES          Premium                    2398
MIDAGE SINGLES/COUPLES  Premium                    2369
OLDER FAMILIES          Premium                    2231
MIDAGE SINGLES/COUPLES  Budget                     1474
NEW FAMILIES            Budget                     1087
                        Mainstream                  830
                        Premium                     575
```

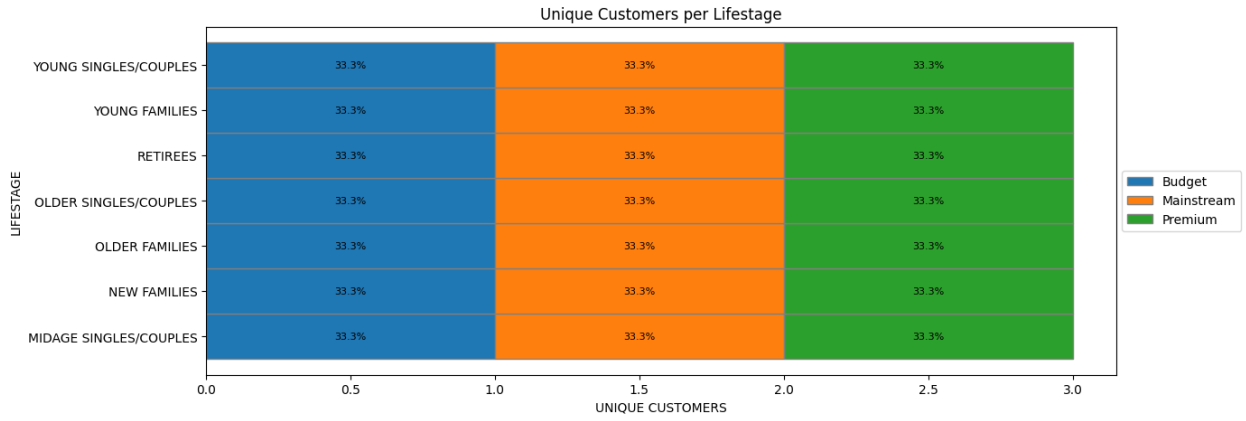
```
unique_cust.sort_values().plot.barh(figsize=(12,7))
```

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




```
# Pivot the data first
pivot_cust = unique_cust.reset_index().pivot_table(index='LIFESTAGE',
columns='PREMIUM_CUSTOMER', aggfunc='size', fill_value=0)

# Now bars will align
ncust_bars1 = pivot_cust["Budget"]
ncust_bars2 = pivot_cust["Mainstream"]
ncust_bars3 = pivot_cust["Premium"]

# Recalculate percentages per row
total_customers = ncust_bars1 + ncust_bars2 + ncust_bars3
ncust_bars1_text = (ncust_bars1 /
total_customers).apply("{:.1%}".format)
ncust_bars2_text = (ncust_bars2 /
total_customers).apply("{:.1%}".format)
ncust_bars3_text = (ncust_bars3 /
total_customers).apply("{:.1%}".format)

# Set names and positions
names = pivot_cust.index
```

```
r = np.arange(len(names))

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

# Create the bars
budget_bar = plt.barh(r, ncust_bars1, edgecolor='grey', height=1,
label="Budget")
mains_bar = plt.barh(r, ncust_bars2, left=ncust_bars1,
edgecolor='grey', height=1, label="Mainstream")
prem_bar = plt.barh(r, ncust_bars3, left=ncust_bars1 + ncust_bars2,
edgecolor='grey', height=1, label="Premium")

# Add text labels inside bars
for i in range(len(names)):
    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.iloc[i], va='center',
ha='center', size=8)
    plt.text(budget_width + mains_bar[i].get_width()/2, i,
ncust_bars2_text.iloc[i], va='center', ha='center', size=8)
    plt.text(budget_main_width + prem_bar[i].get_width()/2, i,
ncust_bars3_text.iloc[i], va='center', ha='center', size=8)

# Customizing
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")
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	4.749283	2788
	Budget	4.665799	4611
	Premium	4.662931	2231
YOUNG FAMILIES	Premium	4.497081	2398
	Budget	4.493549	3953
	Mainstream	4.449534	2685
OLDER SINGLES/COUPLES	Budget	3.541349	4849
	Premium	3.536950	4682
	Mainstream	3.511939	4858
MIDAGE SINGLES/COUPLES	Mainstream	3.364160	3298
RETIREES	Budget	3.244014	4385
MIDAGE SINGLES/COUPLES	Premium	3.213170	2369
RETIREES	Premium	3.209864	3812
MIDAGE SINGLES/COUPLES	Budget	3.182497	1474
RETIREES	Mainstream	3.140925	6358
NEW FAMILIES	Mainstream	2.632530	830
	Budget	2.597976	1087
	Premium	2.587826	575
YOUNG SINGLES/COUPLES	Mainstream	2.468612	7917
	Premium	2.359677	2480
	Budget	2.350699	3647

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

		sum	mean
LIFESTAGE	PREMIUM_CUSTOMER		
MIDAGE SINGLES/COUPLES	Mainstream	84734.25	7.637156
YOUNG SINGLES/COUPLES	Mainstream	147582.20	7.551279
RETIREES	Premium	91296.65	7.461315
OLDER SINGLES/COUPLES	Premium	123537.55	7.459997
RETIREES	Budget	105916.30	7.445786
OLDER SINGLES/COUPLES	Budget	127833.60	7.444305
NEW FAMILIES	Mainstream	15979.70	7.313364
OLDER SINGLES/COUPLES	Mainstream	124648.50	7.306049
YOUNG FAMILIES	Budget	129717.95	7.302705
NEW FAMILIES	Budget	20607.45	7.297256
OLDER FAMILIES	Budget	156863.75	7.291241
YOUNG FAMILIES	Premium	78571.70	7.285951
OLDER FAMILIES	Mainstream	96413.55	7.281440
RETIREES	Mainstream	145168.95	7.269352
OLDER FAMILIES	Premium	75242.60	7.232779
NEW FAMILIES	Premium	10760.80	7.231720
YOUNG FAMILIES	Mainstream	86338.25	7.226772
MIDAGE SINGLES/COUPLES	Premium	54443.85	7.152371
	Budget	33345.70	7.108442
YOUNG SINGLES/COUPLES	Premium	39052.30	6.673325
	Budget	57122.10	6.663023

```
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
```

```
1.834645908180742e-237
```

```
True
```

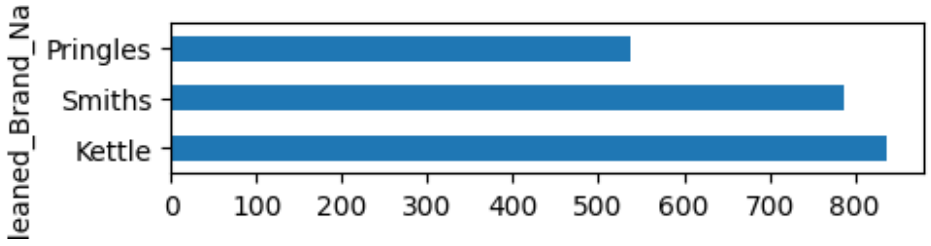
```
merged_data.groupby(["LIFESTAGE", "PREMIUM_CUSTOMER"])
["Cleaned_Brand_Names"].agg(pd.Series.mode).sort_values()
```

```
LIFESTAGE                PREMIUM_CUSTOMER
MIDAGE SINGLES/COUPLES   Budget              Kettle
YOUNG 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           Premium             Kettle
                         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
Name: Cleaned_Brand_Names, dtype: object
```

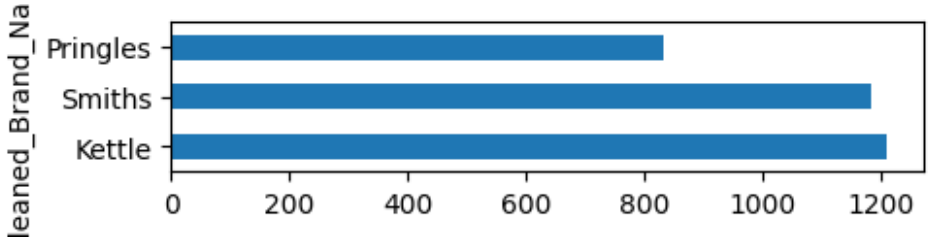
```
for stage in merged_data["LIFESTAGE"].unique():
    for prem in merged_data["PREMIUM_CUSTOMER"].unique():
        print('==========',stage, '-', prem,'==========')
        summary = merged_data[(merged_data["LIFESTAGE"] == stage) &
```

```
(merged_data["PREMIUM_CUSTOMER"] == prem)]
["Cleaned_Brand_Names"].value_counts().head(3)
        print(summary)
        plt.figure()
        summary.plot.barh(figsize=(5,1))
        plt.show()
```

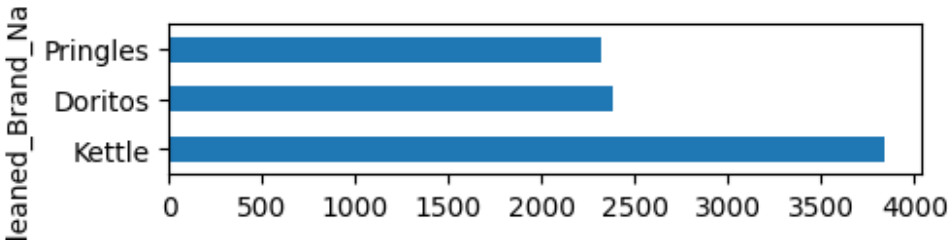
```
========== YOUNG SINGLES/COUPLES - Premium ==========
Cleaned_Brand_Names
Kettle      838
Smiths      787
Pringles    537
Name: count, dtype: int64
```



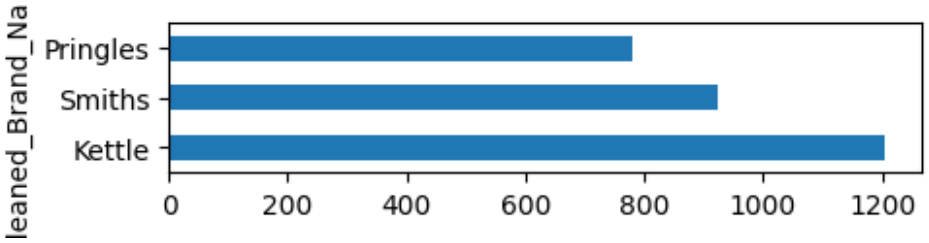

```
========== YOUNG SINGLES/COUPLES - Budget ==========
Cleaned_Brand_Names
Kettle      1211
Smiths      1185
Pringles     832
Name: count, dtype: int64
```



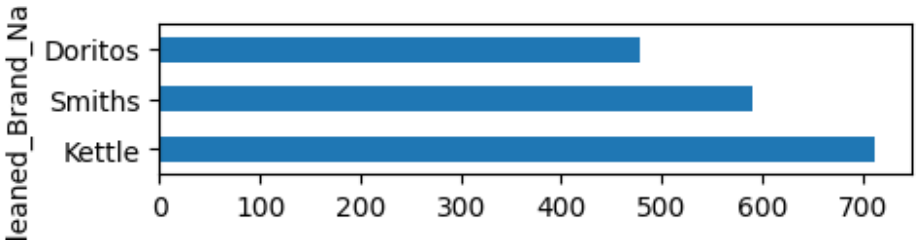

```
========== YOUNG SINGLES/COUPLES - Mainstream ==========
Cleaned_Brand_Names
Kettle      3844
Doritos     2379
Pringles    2315
Name: count, dtype: int64
```



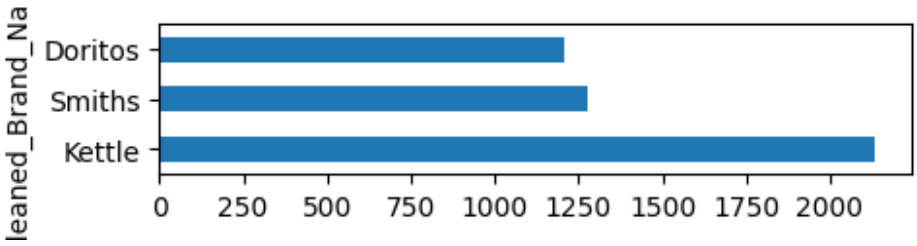

```
========== MIDAGE SINGLES/COUPLES - Premium ==========
Cleaned_Brand_Names
Kettle      1206
Smiths       923
Pringles     781
Name: count, dtype: int64
```



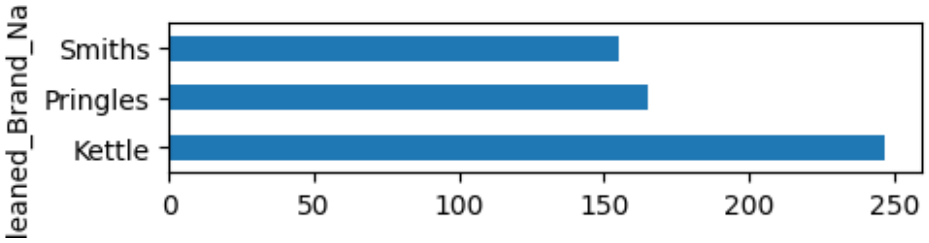

```
========== MIDAGE SINGLES/COUPLES - Budget ==========
Cleaned_Brand_Names
Kettle     713
Smiths     591
Doritos    479
Name: count, dtype: int64
```



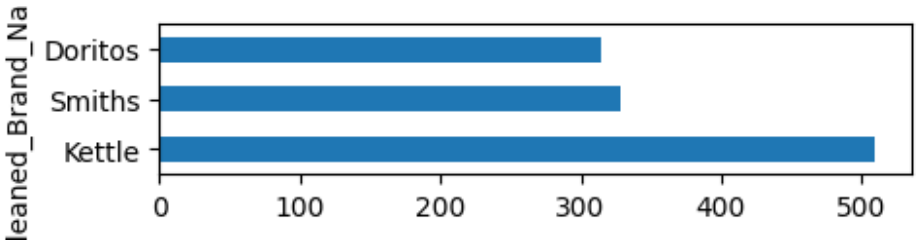

```
========== MIDAGE SINGLES/COUPLES - Mainstream ==========
Cleaned_Brand_Names
Kettle     2136
Smiths     1276
Doritos    1210
Name: count, dtype: int64
```



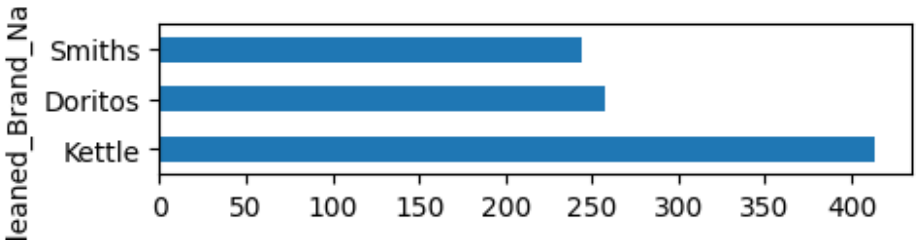

```
========== NEW FAMILIES - Premium ==========
Cleaned_Brand_Names
Kettle      247
Pringles    165
Smiths      155
Name: count, dtype: int64
```



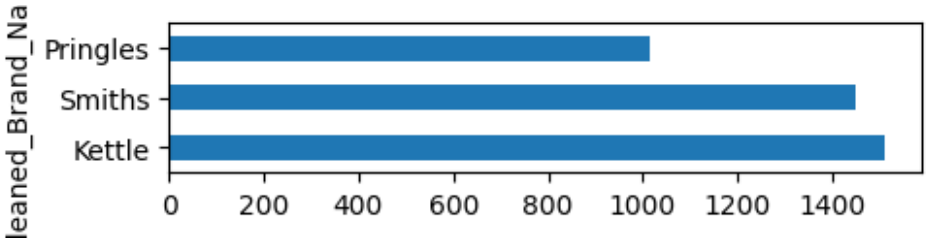

```
========== NEW FAMILIES - Budget ==========
Cleaned_Brand_Names
Kettle     510
Smiths     328
Doritos    315
Name: count, dtype: int64
```



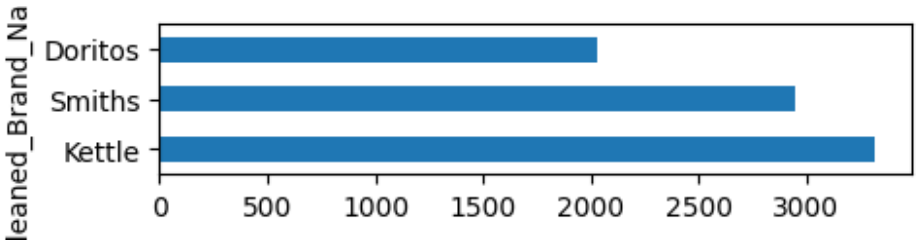

```
========== NEW FAMILIES - Mainstream ==========
Cleaned_Brand_Names
Kettle     414
Doritos    257
Smiths     244
Name: count, dtype: int64
```



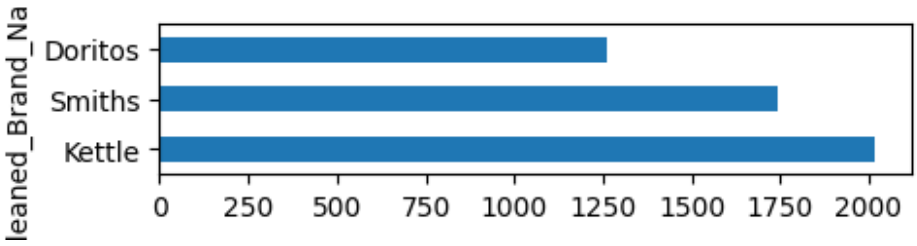

```
========== OLDER FAMILIES - Premium ==========
Cleaned_Brand_Names
Kettle      1512
Smiths      1448
Pringles    1014
Name: count, dtype: int64
```



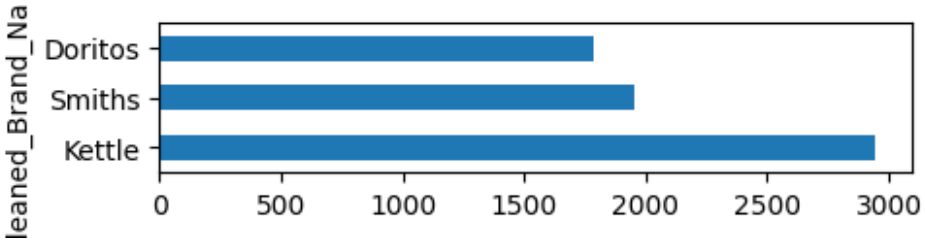

```
========== OLDER FAMILIES - Budget ==========
Cleaned_Brand_Names
Kettle     3320
Smiths     2948
Doritos    2032
Name: count, dtype: int64
```



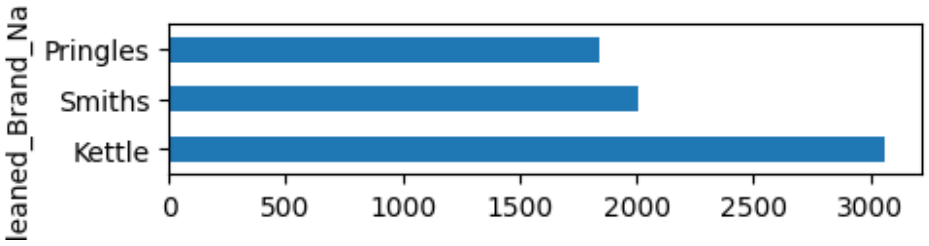

```
========== OLDER FAMILIES - Mainstream ==========
Cleaned_Brand_Names
Kettle     2019
Smiths     1742
Doritos    1263
Name: count, dtype: int64
```



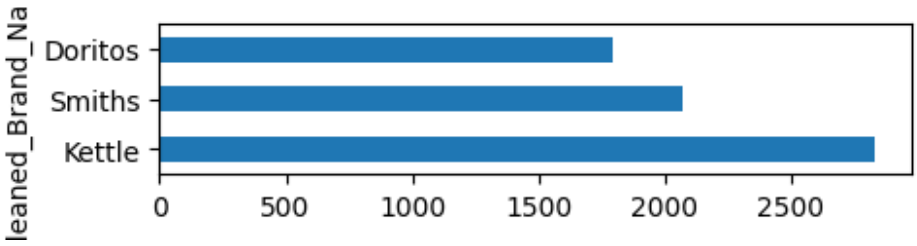

```
========== OLDER SINGLES/COUPLES - Premium ==========
Cleaned_Brand_Names
Kettle     2947
Smiths     1952
Doritos    1784
Name: count, dtype: int64
```



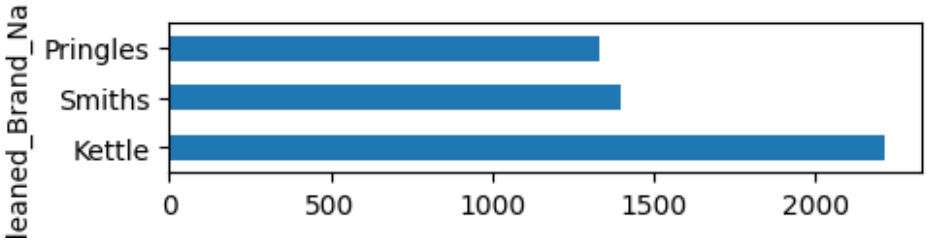

```
========== OLDER SINGLES/COUPLES - Budget ==========
Cleaned_Brand_Names
Kettle      3065
Smiths      2010
Pringles    1843
Name: count, dtype: int64
```



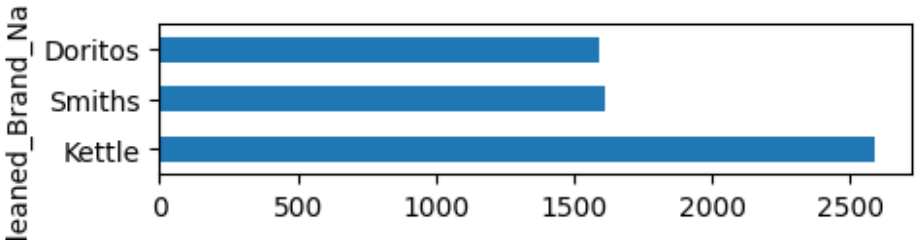

```
========== OLDER SINGLES/COUPLES - Mainstream ==========
Cleaned_Brand_Names
Kettle     2835
Smiths     2070
Doritos    1791
Name: count, dtype: int64
```



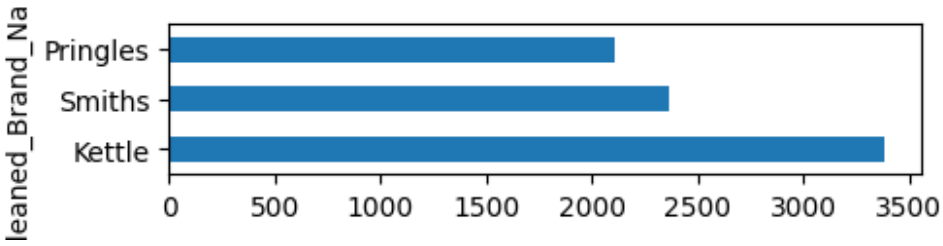

```
========== RETIREES - Premium ==========
Cleaned_Brand_Names
Kettle      2216
Smiths      1395
Pringles    1331
Name: count, dtype: int64
```



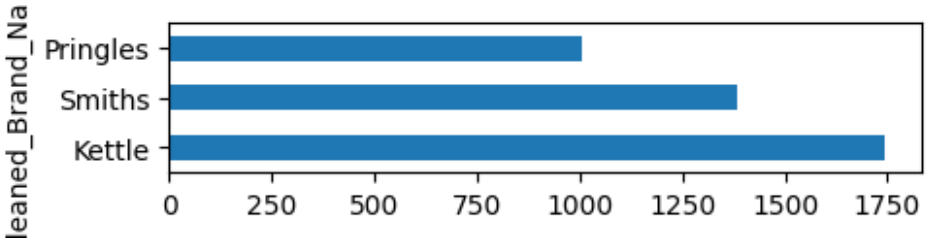

```
========== RETIREES - Budget ==========
Cleaned_Brand_Names
Kettle     2592
Smiths     1612
Doritos    1592
Name: count, dtype: int64
```



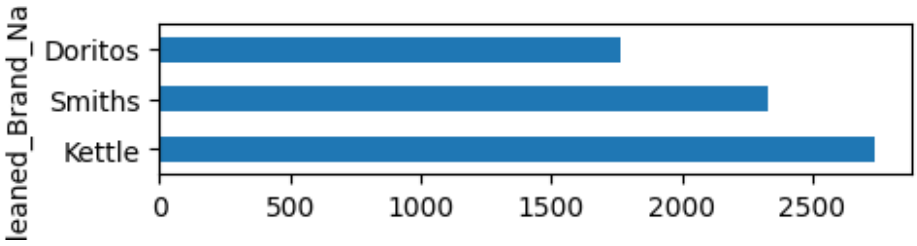

```
========== RETIREES - Mainstream ==========
Cleaned_Brand_Names
Kettle      3386
Smiths      2367
Pringles    2103
Name: count, dtype: int64
```



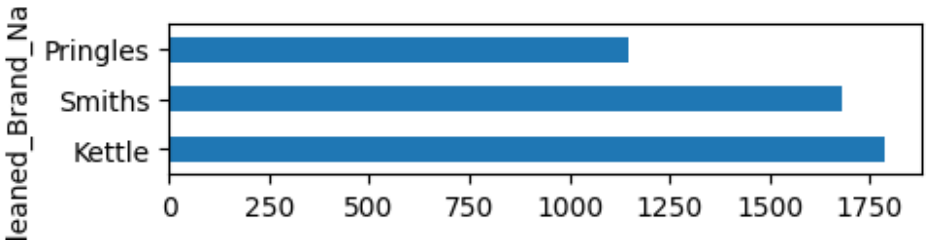

```
========== YOUNG FAMILIES - Premium ==========
Cleaned_Brand_Names
Kettle      1745
Smiths      1384
Pringles    1007
Name: count, dtype: int64
```




```
========== YOUNG FAMILIES - Budget ==========
Cleaned_Brand_Names
Kettle     2743
Smiths     2334
Doritos    1767
Name: count, dtype: int64
```



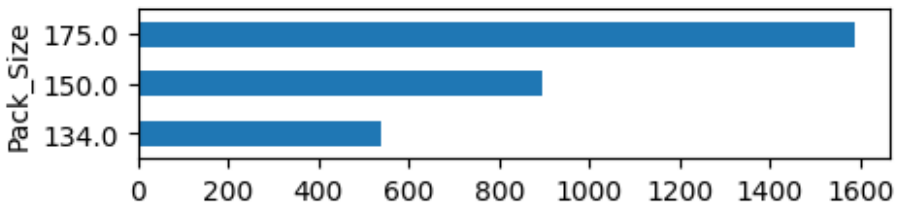

```
========== YOUNG FAMILIES - Mainstream ==========
Cleaned_Brand_Names
Kettle      1789
Smiths      1681
Pringles    1148
Name: count, dtype: int64
```



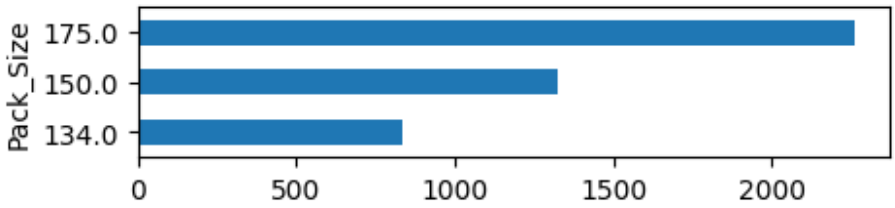

```
#!pip install mlxtend

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  \
0              (OLDER FAMILIES - Budget)    (Smiths)
0.087193
3       (OLDER SINGLES/COUPLES - Budget)    (Kettle)
0.069596
5      (OLDER SINGLES/COUPLES - Premium)    (Kettle)
0.067115
7                    (RETIREES - Budget)    (Kettle)
0.057652
9                (RETIREES - Mainstream)    (Kettle)
0.080935
11  (YOUNG SINGLES/COUPLES - Mainstream)    (Kettle)
0.079209

    consequent support   support  confidence      lift
representativity  \
0             0.123016  0.011948    0.137027  1.113895
1.0
3             0.167334  0.012422    0.178488  1.066658
1.0
5             0.167334  0.011944    0.177959  1.063495
1.0
```

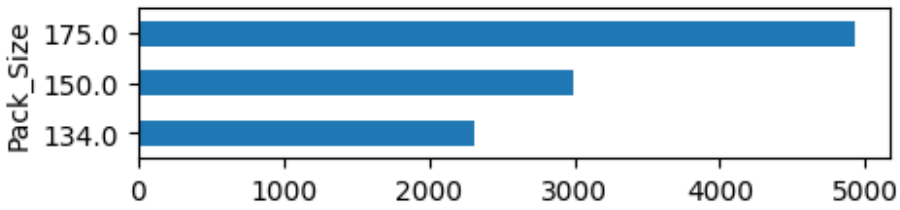
```
7             0.167334  0.010505    0.182214  1.088926
1.0
9             0.167334  0.013723    0.169554  1.013269
1.0
11            0.167334  0.015579    0.196684  1.175400
1.0

    leverage  conviction  zhangs_metric   jaccard  certainty
kulczynski
0   0.001222    1.016236       0.112016  0.060263   0.015976
0.117075
3   0.000776    1.013578       0.067167  0.055330   0.013396
0.126361
5   0.000713    1.012925       0.064000  0.053678   0.012760
0.124668
7   0.000858    1.018196       0.086660  0.048979   0.017871
0.122496
9   0.000180    1.002674       0.014248  0.058508   0.002666
0.125782
11  0.002325    1.036537       0.162062  0.067453   0.035249
0.144893

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().head(3).sort_index()
        print(summary)
        plt.figure()
        summary.plot.barh(figsize=(5,1))
        plt.show()

========== YOUNG SINGLES/COUPLES - Premium ==========
Pack_Size
134.0     537
150.0     896
175.0    1587
Name: count, dtype: int64
```



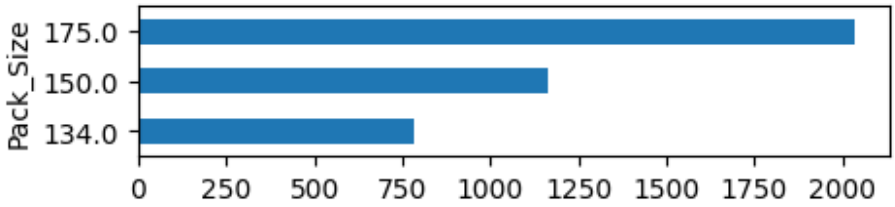
```
========== YOUNG SINGLES/COUPLES - Budget ==========
Pack_Size
134.0     832
150.0    1325
175.0    2262
Name: count, dtype: int64
```




```
========== YOUNG SINGLES/COUPLES - Mainstream ==========
Pack_Size
134.0    2315
150.0    2998
175.0    4928
Name: count, dtype: int64
```

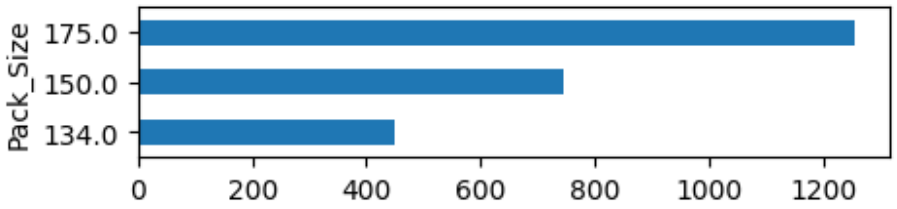



```
========== MIDAGE SINGLES/COUPLES - Premium ==========
Pack_Size
134.0     781
150.0    1163
175.0    2034
Name: count, dtype: int64
```

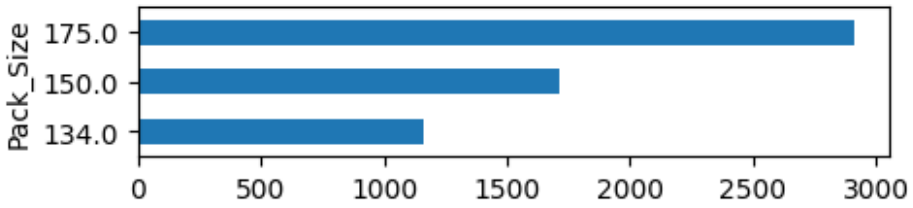



```
========== MIDAGE SINGLES/COUPLES - Budget ==========
Pack_Size
134.0     449
150.0     746
```

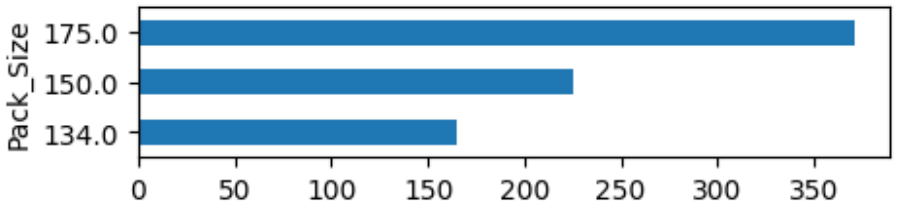
```
175.0    1256
Name: count, dtype: int64
```



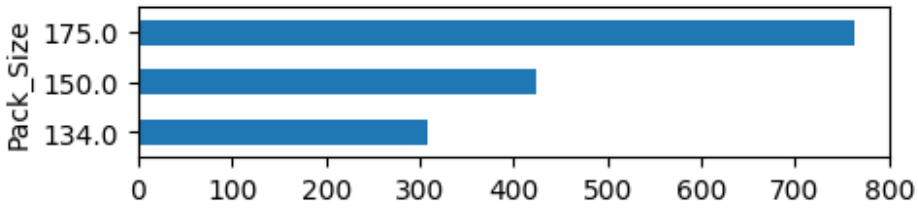

```
========== MIDAGE SINGLES/COUPLES - Mainstream ==========
Pack_Size
134.0    1159
150.0    1714
175.0    2912
Name: count, dtype: int64
```



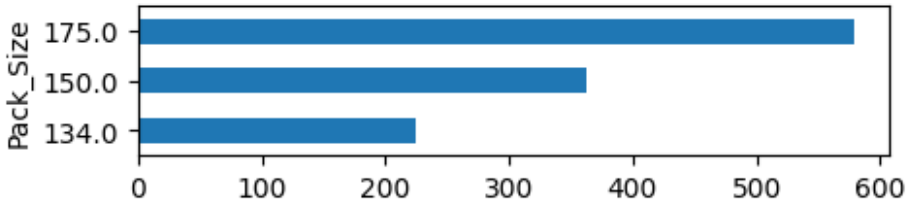

```
========== NEW FAMILIES - Premium ==========
Pack_Size
134.0    165
150.0    225
175.0    371
Name: count, dtype: int64
```



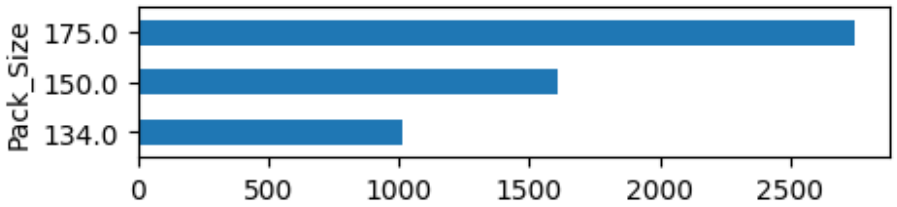

```
========== NEW FAMILIES - Budget ==========
Pack_Size
134.0    309
150.0    425
175.0    763
Name: count, dtype: int64
```



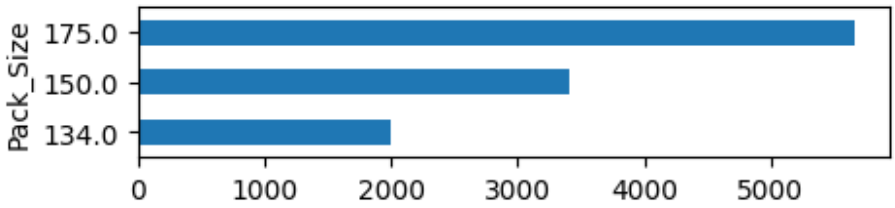

```
========== NEW FAMILIES - Mainstream ==========
Pack_Size
134.0    224
150.0    362
175.0    579
Name: count, dtype: int64
```



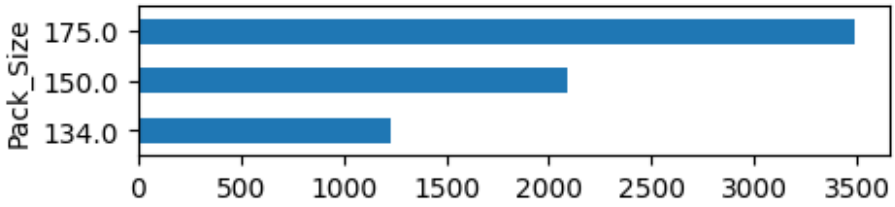

```
========== OLDER FAMILIES - Premium ==========
Pack_Size
134.0    1014
150.0    1607
175.0    2747
Name: count, dtype: int64
```



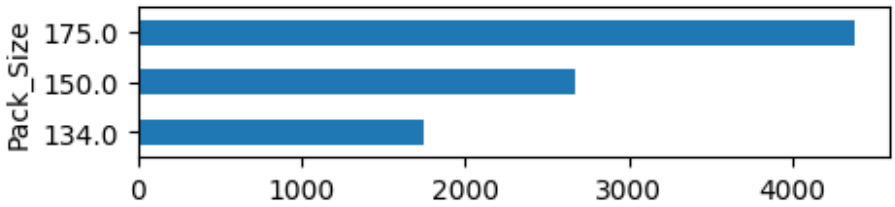

```
========== OLDER FAMILIES - Budget ==========
Pack_Size
134.0    1996
150.0    3414
175.0    5662
Name: count, dtype: int64
```




```
========== OLDER FAMILIES - Mainstream ==========
Pack_Size
134.0    1234
150.0    2091
175.0    3489
Name: count, dtype: int64
```

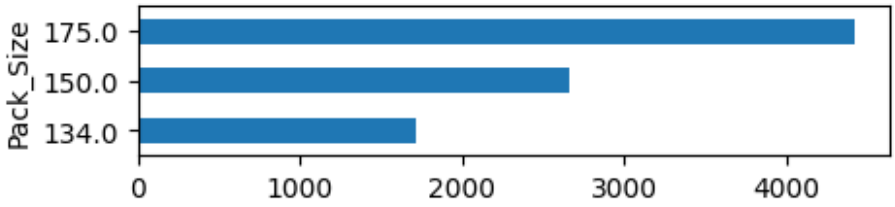



```
========== OLDER SINGLES/COUPLES - Premium ==========
Pack_Size
134.0    1744
150.0    2672
175.0    4382
Name: count, dtype: int64
```

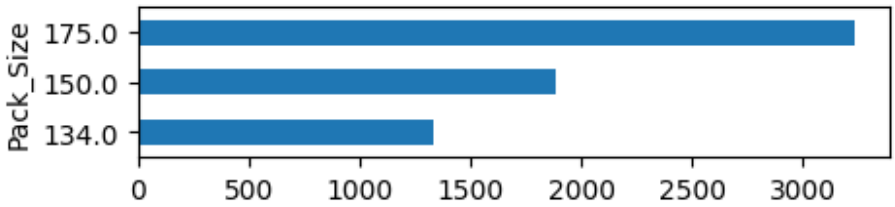



```
========== OLDER SINGLES/COUPLES - Budget ==========
Pack_Size
134.0    1843
150.0    2726
175.0    4535
Name: count, dtype: int64
```

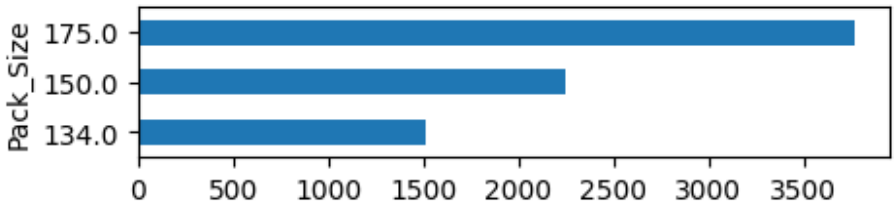

```
========== OLDER SINGLES/COUPLES - Mainstream ==========
Pack_Size
134.0    1720
150.0    2660
175.0    4422
Name: count, dtype: int64
```



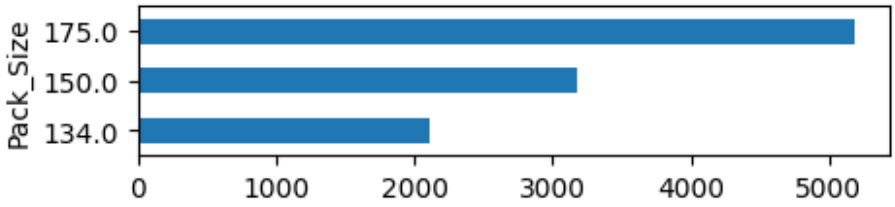

```
========== RETIREES - Premium ==========
Pack_Size
134.0    1331
150.0    1883
175.0    3232
Name: count, dtype: int64
```



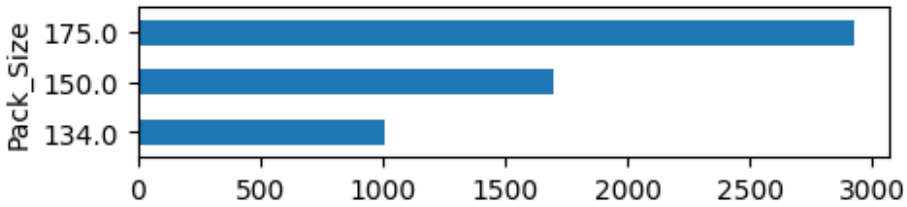

```
========== RETIREES - Budget ==========
Pack_Size
134.0    1517
150.0    2242
175.0    3768
Name: count, dtype: int64
```



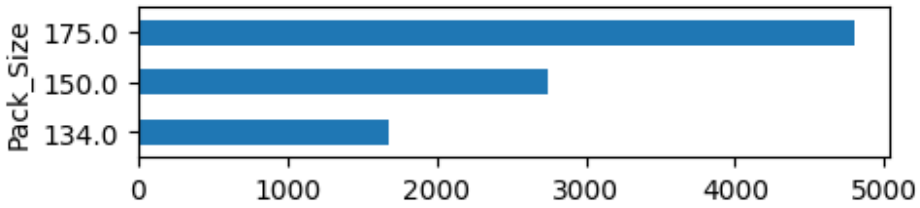

```
========== RETIREES - Mainstream ==========
Pack_Size
134.0    2103
150.0    3183
175.0    5187
Name: count, dtype: int64
```



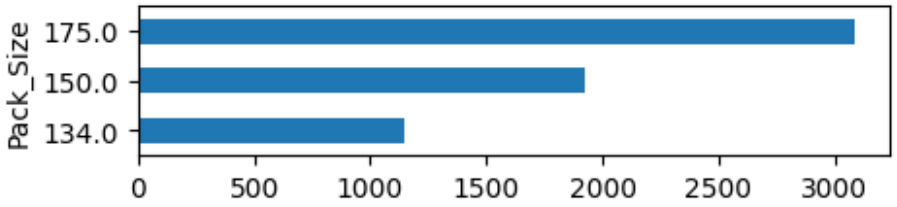

```
========== YOUNG FAMILIES - Premium ==========
Pack_Size
134.0    1007
150.0    1697
175.0    2926
Name: count, dtype: int64
```




```
========== YOUNG FAMILIES - Budget ==========
Pack_Size
134.0    1674
150.0    2749
175.0    4800
Name: count, dtype: int64
```



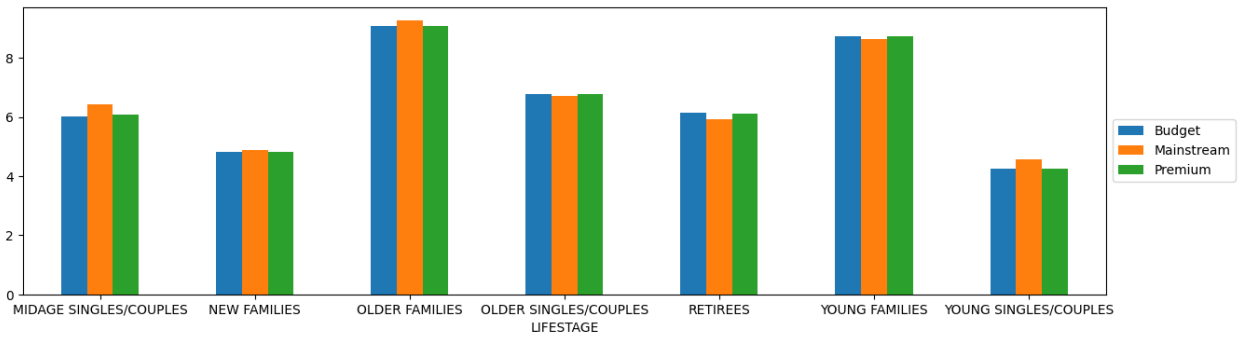

```
========== YOUNG FAMILIES - Mainstream ==========
Pack_Size
134.0    1148
150.0    1927
175.0    3087
Name: count, dtype: int64
```




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

```
LIFESTAGE                PREMIUM_CUSTOMER
OLDER FAMILIES           Mainstream          9.255380
                         Budget              9.076773
                         Premium             9.071717
YOUNG FAMILIES           Budget              8.722995
                         Premium             8.716013
                         Mainstream          8.638361
OLDER SINGLES/COUPLES    Budget              6.781398
                         Premium             6.769543
                         Mainstream          6.712021
MIDAGE SINGLES/COUPLES   Mainstream          6.432080
RETIREES                 Budget              6.141847
                         Premium             6.103358
MIDAGE SINGLES/COUPLES   Premium             6.078514
                         Budget              6.026459
RETIREES                 Mainstream          5.925920
NEW FAMILIES             Mainstream          4.891566
                         Budget              4.821527
                         Premium             4.815652
YOUNG SINGLES/COUPLES    Mainstream          4.575597
                         Premium             4.264113
                         Budget              4.250069
dtype: float64
```

```
(temp.groupby(["LIFESTAGE", "PREMIUM_CUSTOMER"])["PROD_QTY"].sum() /
temp.groupby(["LIFESTAGE", "PREMIUM_CUSTOMER"])
["LYLTY_CARD_NBR"].nunique()).unstack().plot.bar(figsize=(15,4),
rot=0)
plt.legend(loc="center left", bbox_to_anchor=(1.0, 0.5))
plt.savefig("Average purchase quantity per segment.png",
bbox_inches="tight")
```




```
print(temp.dtypes)
```

```
transaction                      int64
LYLTY_CARD_NBR                   int64
LIFESTAGE                       object
PREMIUM_CUSTOMER                object
DATE                    datetime64[ns]
STORE_NBR                        int64
TXN_ID                           int64
PROD_NBR                         int64
PROD_NAME                       object
PROD_QTY                         int64
TOT_SALES                      float64
Cleaned_Brand_Names             object
Segment                         object
dtype: object
```

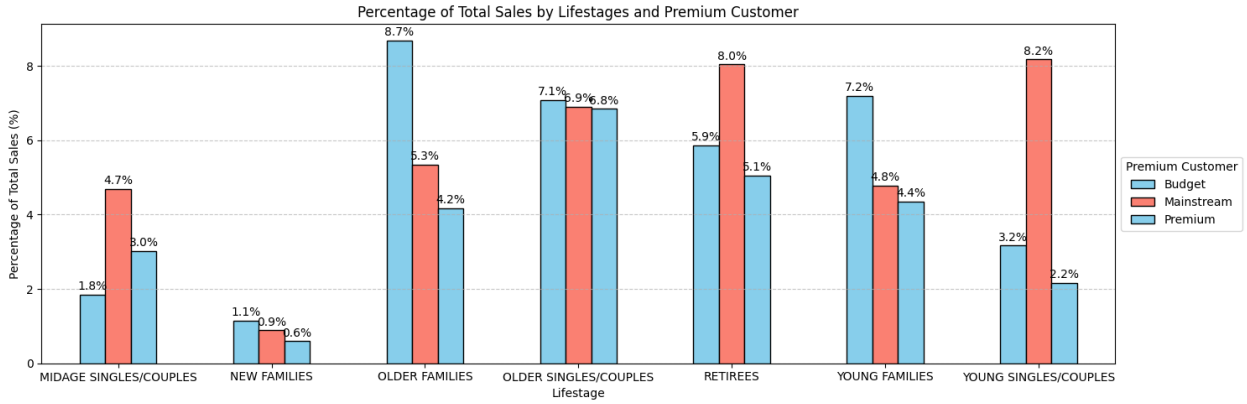
```
# Step 1: Ensure TOT_SALES and PROD_QTY are numeric
temp["TOT_SALES"] = pd.to_numeric(temp["TOT_SALES"], errors='coerce')
temp["PROD_QTY"] = pd.to_numeric(temp["PROD_QTY"], errors='coerce')

# Step 2: Create Unit_Price
temp["Unit_Price"] = temp["TOT_SALES"] / temp["PROD_QTY"]

# Step 3: Remove inf or NaN values from Unit_Price
temp = temp[temp["Unit_Price"].notna() & (temp["Unit_Price"] !=
float('inf'))]

# Step 4: Now group only 'Unit_Price' and calculate mean
segment_avg_price = temp.groupby("Segment")
["Unit_Price"].mean().sort_values(ascending=False)
```

```python
# Step 5: Print the result
print(segment_avg_price)
```

```
Segment
YOUNG SINGLES/COUPLES - Mainstream     4.065642
MIDAGE SINGLES/COUPLES - Mainstream    3.994241
RETIREES - Budget                      3.924404
RETIREES - Premium                     3.920942
NEW FAMILIES - Budget                  3.917688
NEW FAMILIES - Mainstream              3.916133
OLDER SINGLES/COUPLES - Premium        3.893182
OLDER SINGLES/COUPLES - Budget         3.882096
NEW FAMILIES - Premium                 3.872110
RETIREES - Mainstream                  3.844294
OLDER SINGLES/COUPLES - Mainstream     3.814665
MIDAGE SINGLES/COUPLES - Premium       3.770698
YOUNG FAMILIES - Premium               3.762150
YOUNG FAMILIES - Budget                3.760737
OLDER FAMILIES - Budget                3.745340
MIDAGE SINGLES/COUPLES - Budget        3.743328
OLDER FAMILIES - Mainstream            3.737077
YOUNG FAMILIES - Mainstream            3.724533
OLDER FAMILIES - Premium               3.717000
YOUNG SINGLES/COUPLES - Premium        3.665414
YOUNG SINGLES/COUPLES - Budget         3.657366
Name: Unit_Price, dtype: float64
```

```python
# Step 1: Group by Lifestage and Premium Customer, aur Total Sales ka sum nikal
sales_by_group = temp.groupby(["LIFESTAGE", "PREMIUM_CUSTOMER"])["TOT_SALES"].sum()

# Step 2: Overall total sales
total_sales = sales_by_group.sum()

# Step 3: Percentage nikaalna
percentage_sales = (sales_by_group / total_sales) * 100

# Step 4: Plot karna
ax = percentage_sales.unstack().plot.bar(
    figsize=(15,5),
    rot=0,
    color=["skyblue", "salmon"],
    edgecolor='black'
)

# Step 5: Bars ke upar percentage likhna
for container in ax.containers:
```

```
    ax.bar_label(container, fmt='%.1f%%', label_type='edge',
fontsize=10, padding=2)

# Step 6: Chart Decoration
plt.title('Percentage of Total Sales by Lifestages and Premium
Customer')
plt.ylabel('Percentage of Total Sales (%)')
plt.xlabel('Lifestage')
plt.legend(title="Premium Customer", loc="center left",
bbox_to_anchor=(1.0, 0.5))
plt.grid(axis='y', linestyle='--', alpha=0.7)
plt.tight_layout()
plt.show()
```




```
# Step 1: Sirf 'TOT_SALES' ka groupby-sum lena
z = temp.groupby(["Segment", "Cleaned_Brand_Names"])
["TOT_SALES"].sum().sort_values(ascending=False).reset_index()

# Step 2: Ab filter karo specific 'Segment' ke liye
young_singles_mainstream = z[z["Segment"] == "YOUNG SINGLES/COUPLES -
Mainstream"]

# Step 3: Dekho result
print(young_singles_mainstream)
```

	Segment	Cleaned_Brand_Names	TOT_SALES
0	YOUNG SINGLES/COUPLES - Mainstream	Kettle	35423.6
8	YOUNG SINGLES/COUPLES - Mainstream	Doritos	20925.9
22	YOUNG SINGLES/COUPLES - Mainstream	Pringles	16006.2
24	YOUNG SINGLES/COUPLES - Mainstream	Smiths	14958.9
54	YOUNG SINGLES/COUPLES - Mainstream	Infuzions	8749.4
61	YOUNG SINGLES/COUPLES - Mainstream	Twisties	7539.8
69	YOUNG SINGLES/COUPLES - Mainstream	Tostitos	7238.0
70	YOUNG SINGLES/COUPLES - Mainstream	Thins	7217.1
84	YOUNG SINGLES/COUPLES - Mainstream	Cobs	6144.6

115	YOUNG SINGLES/COUPLES - Mainstream	Tyrrells	4800.6
122	YOUNG SINGLES/COUPLES - Mainstream	RRD	4509.9
136	YOUNG SINGLES/COUPLES - Mainstream	Grain Waves	4201.0
172	YOUNG SINGLES/COUPLES - Mainstream	Cheezels	3318.3
227	YOUNG SINGLES/COUPLES - Mainstream	Natural Chip Co	2130.0
258	YOUNG SINGLES/COUPLES - Mainstream	Woolworths	1605.8
298	YOUNG SINGLES/COUPLES - Mainstream	Cheetos	898.8
308	YOUNG SINGLES/COUPLES - Mainstream	CCs	850.5
362	YOUNG SINGLES/COUPLES - Mainstream	French	429.0
371	YOUNG SINGLES/COUPLES - Mainstream	Sunbites	391.0
394	YOUNG SINGLES/COUPLES - Mainstream	Burger	243.8