data preparation and customer analytics

August 11, 2020

[1]:
```python
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns; sns.set(palette='Spectral')
from scipy import stats
import xlrd
import datetime
%matplotlib inline
```

1 Load Data

[2]:
```python
purchase = pd.read_csv('data/QVI_purchase_behaviour.csv')
purchase.head()
```

[2]:
```
   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
```

[3]:
```python
transaction = pd.read_excel('data/QVI_transaction_data.xlsx')
transaction.head()
```

[3]:
```
    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
```

Converting 'DATE' to date type

[4]:
```
book = xlrd.open_workbook("data/QVI_transaction_data.xlsx")
datemode = book.datemode
transaction["DATE"].map(lambda x:xlrd.xldate_as_tuple(x, datemode))
transaction['DATE'] =  transaction["DATE"].map(lambda x:datetime.datetime(*xlrd.
↪xldate_as_tuple(x,datemode)))
```

[5]:
```
transaction.head()
```

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

                                   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
```

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

[6]:
```
          STORE_NBR  LYLTY_CARD_NBR        TXN_ID       PROD_NBR  \
count  264836.00000    2.648360e+05  2.648360e+05  264836.000000
mean      135.08011    1.355495e+05  1.351583e+05      56.583157
std        76.78418    8.057998e+04  7.813303e+04      32.826638
min         1.00000    1.000000e+03  1.000000e+00       1.000000
25%        70.00000    7.002100e+04  6.760150e+04      28.000000
50%       130.00000    1.303575e+05  1.351375e+05      56.000000
75%       203.00000    2.030942e+05  2.027012e+05      85.000000
max       272.00000    2.373711e+06  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
```

[7]:
```
df = pd.merge(transaction,purchase)
df.head()
```

[7]:
```
         DATE  STORE_NBR  LYLTY_CARD_NBR  TXN_ID  PROD_NBR  \
0  2018-10-17          1            1000       1         5
1  2019-05-14          1            1307     348        66
2  2018-11-10          1            1307     346        96
3  2019-03-09          1            1307     347        54
4  2019-05-20          1            1343     383        61

                               PROD_NAME  PROD_QTY  TOT_SALES  \
0  Natural Chip        Compny SeaSalt175g         2        6.0
1                CCs Nacho Cheese    175g         3        6.3
2          WW Original Stacked Chips 160g         2        3.8
3                      CCs Original 175g          1        2.1
4  Smiths Crinkle Cut  Chips Chicken 170g         2        2.9

                LIFESTAGE PREMIUM_CUSTOMER
0   YOUNG SINGLES/COUPLES          Premium
1  MIDAGE SINGLES/COUPLES           Budget
2  MIDAGE SINGLES/COUPLES           Budget
3  MIDAGE SINGLES/COUPLES           Budget
4  MIDAGE SINGLES/COUPLES           Budget
```

2 Data Manipulation

Removing Salsa products from dataframe

[8]:
```
df.drop(df[df['PROD_NAME'].apply(lambda x: True if 'salsa' in x.lower().split()␣
 ↪else False)].index,inplace=True)
```

Creating Size and Brand columns

[9]:
```
df['SIZE'] = df['PROD_NAME'].apply(lambda x: x[-4:-1])
df['BRAND'] = df['PROD_NAME'].apply(lambda x: x.split(' ')[0])
df.head()
```

[9]:
```
         DATE  STORE_NBR  LYLTY_CARD_NBR  TXN_ID  PROD_NBR  \
0  2018-10-17          1            1000       1         5
1  2019-05-14          1            1307     348        66
2  2018-11-10          1            1307     346        96
3  2019-03-09          1            1307     347        54
4  2019-05-20          1            1343     383        61

                               PROD_NAME  PROD_QTY  TOT_SALES  \
0  Natural Chip        Compny SeaSalt175g         2        6.0
1                CCs Nacho Cheese    175g         3        6.3
```

```
2          WW Original Stacked Chips 160g         2        3.8
3                      CCs Original 175g         1        2.1
4  Smiths Crinkle Cut  Chips Chicken 170g         2        2.9

                LIFESTAGE PREMIUM_CUSTOMER SIZE    BRAND
0   YOUNG SINGLES/COUPLES          Premium  175  Natural
1  MIDAGE SINGLES/COUPLES           Budget  175      CCs
2  MIDAGE SINGLES/COUPLES           Budget  160       WW
3  MIDAGE SINGLES/COUPLES           Budget  175      CCs
4  MIDAGE SINGLES/COUPLES           Budget  170   Smiths
```

Checking to see if there's any error in the created columns.

```
[10]: df['SIZE'].unique()
```

```
[10]: array(['175', '160', '170', '150', '165', '380', '330', '110', '210',
       '180', '200', '134', '270', '220', '125', ' 70', 'Sal', '250',
       ' 90', '190'], dtype=object)
```

See what products have been assigned as 'Sal'.

```
[11]: df[df['SIZE']=='Sal'].head(3)
```

```
[11]:           DATE  STORE_NBR  LYLTY_CARD_NBR  TXN_ID  PROD_NBR  \
101 2019-04-30         39           39167   35644        63
132 2018-11-23         45           45127   41120        63
152 2019-04-01         55           55072   48881        63

                  PROD_NAME  PROD_QTY  TOT_SALES  \
101  Kettle 135g Swt Pot Sea Salt         2        8.4
132  Kettle 135g Swt Pot Sea Salt         2        8.4
152  Kettle 135g Swt Pot Sea Salt         2        8.4

                  LIFESTAGE PREMIUM_CUSTOMER SIZE   BRAND
101  MIDAGE SINGLES/COUPLES           Budget  Sal  Kettle
132  MIDAGE SINGLES/COUPLES           Budget  Sal  Kettle
152  MIDAGE SINGLES/COUPLES           Budget  Sal  Kettle
```

Apparently all the items assigned 'Sal' as their size need to be assigned 135.

```
[12]: df.loc[df['SIZE']=='Sal','SIZE'] = '135'
```

```
[13]: df['SIZE'].unique()
```

```
[13]: array(['175', '160', '170', '150', '165', '380', '330', '110', '210',
       '180', '200', '134', '270', '220', '125', ' 70', '135', '250',
       ' 90', '190'], dtype=object)
```

Checking if BRAND's column has any uncorrectly assings

[14]:
```
df['BRAND'].unique()
```

[14]:
```
array(['Natural', 'CCs', 'WW', 'Smiths', 'Kettle', 'Tyrrells', 'Dorito',
       'Doritos', 'Infuzions', 'Grain', 'Thins', 'Red', 'GrnWves',
       'Tostitos', 'Pringles', 'Cobs', 'Twisties', 'RRD', 'Infzns',
       'Burger', 'NCC', 'Cheezels', 'Smith', 'French', 'Sunbites',
       'Cheetos', 'Woolworths', 'Snbts'], dtype=object)
```

Apparently Dorito, Infzns, Red, and Snbts have been misassigned.

[15]:
```
df.loc[df['BRAND']=='Dorito','BRAND'] = 'Doritos'
df.loc[df['BRAND']=='Snbts','BRAND'] = 'Sunbites'
df.loc[df['BRAND']=='Infzns','BRAND'] = 'Infuzions'
df.loc[df['BRAND']=='Red','BRAND'] = 'RRD'
```

[16]:
```
df['BRAND'].unique()
```

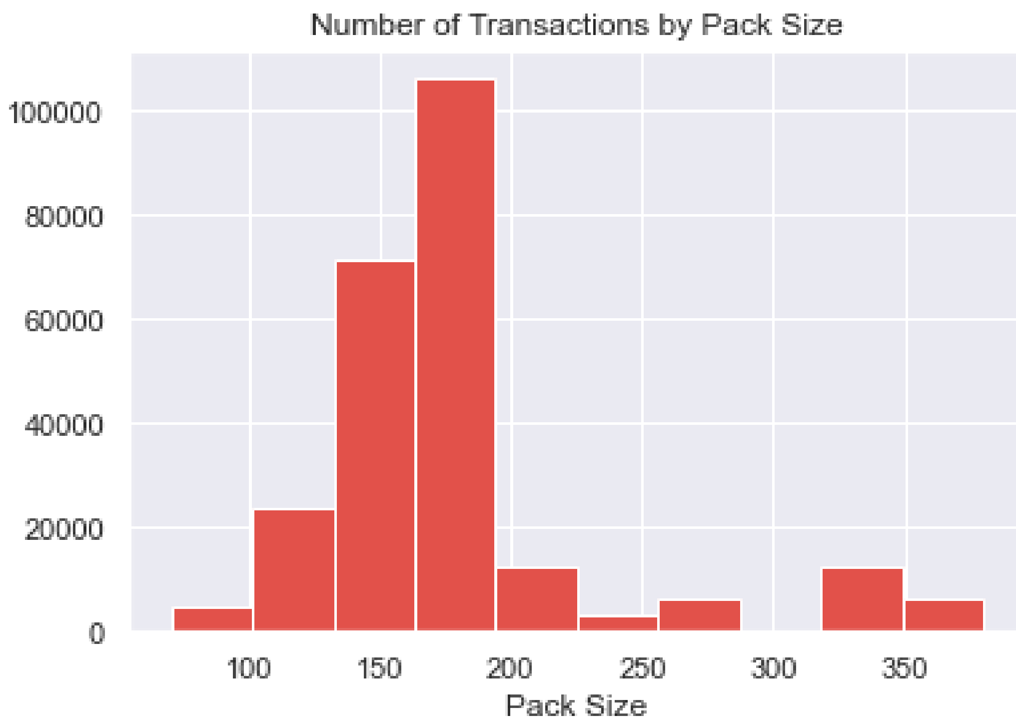
[16]:
```
array(['Natural', 'CCs', 'WW', 'Smiths', 'Kettle', 'Tyrrells', 'Doritos',
       'Infuzions', 'Grain', 'Thins', 'RRD', 'GrnWves', 'Tostitos',
       'Pringles', 'Cobs', 'Twisties', 'Burger', 'NCC', 'Cheezels',
       'Smith', 'French', 'Sunbites', 'Cheetos', 'Woolworths'],
      dtype=object)
```

Setting 'SIZE' column as integer.

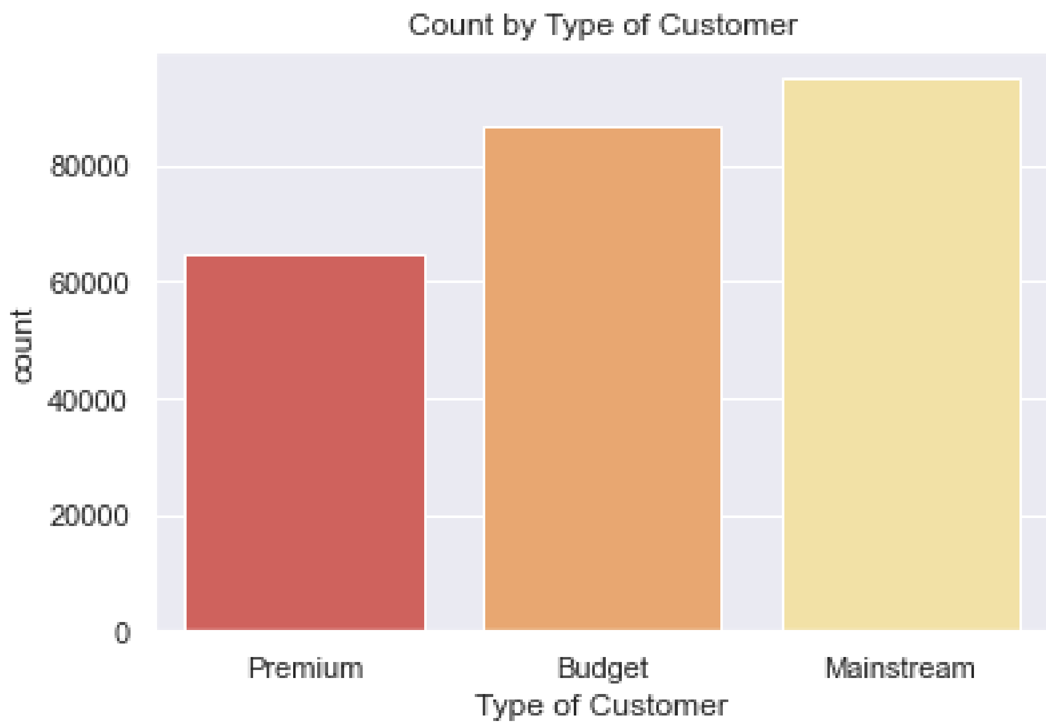
[17]:
```
df['SIZE'] = df['SIZE'].astype(int)
```

3 Data Exploration

[18]:
```
df.hist('SIZE')
plt.title("Number of Transactions by Pack Size")
plt.xlabel('Pack Size');
```

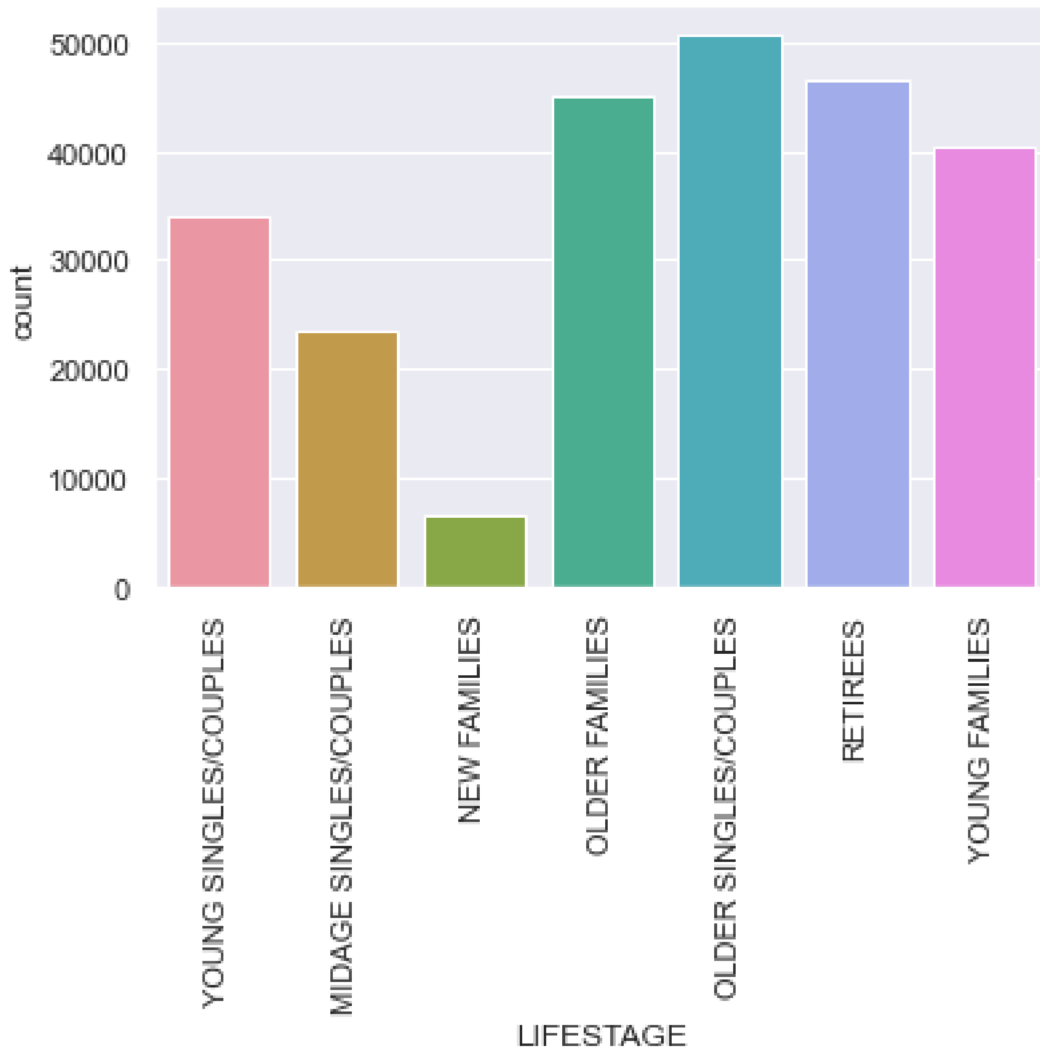


[19]:
```
sns.countplot(data=df,x='PREMIUM_CUSTOMER')
plt.title("Count by Type of Customer")
plt.xlabel("Type of Customer");
```



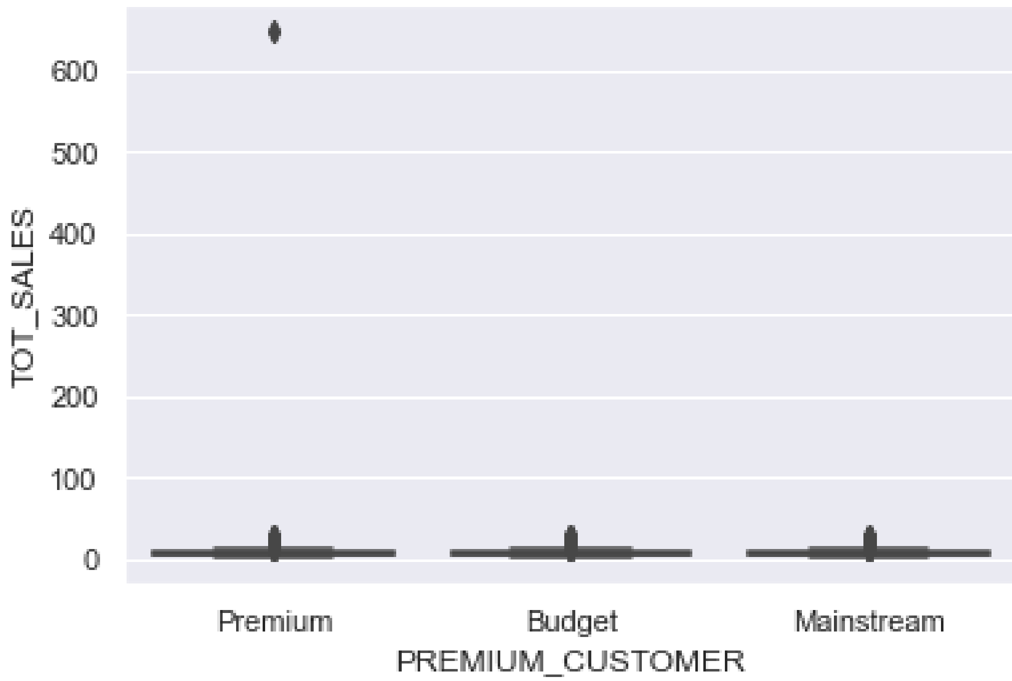
[20]:

```
sns.countplot(data=df,x='LIFESTAGE')
plt.xticks(rotation=90);
```



[21]:
```
sns.boxplot(x="PREMIUM_CUSTOMER", y="TOT_SALES", data=df,palette='rainbow')
```

[21]: <matplotlib.axes._subplots.AxesSubplot at 0x1285eff2e08>



Checking who's the outlier.

[22]:
```
df['TOT_SALES'].sort_values(ascending=False)
```

[22]:
```
71457     650.0
71456     650.0
171914     29.5
5745       29.5
119732     29.5
          …
181600      1.7
18434       1.7
235438      1.7
80905       1.7
149351      1.7
Name: TOT_SALES, Length: 246742, dtype: float64
```

[23]:
```
df.iloc[71456:71458]
```

[23]:
```
             DATE  STORE_NBR  LYLTY_CARD_NBR  TXN_ID  PROD_NBR  \
76786 2019-03-30        149          149089  148716        93
76787 2018-10-01        149          149120  148769        30

                                    PROD_NAME  PROD_QTY  TOT_SALES  \
76786  Doritos Corn Chip Southern Chicken 150g         2        7.8
```

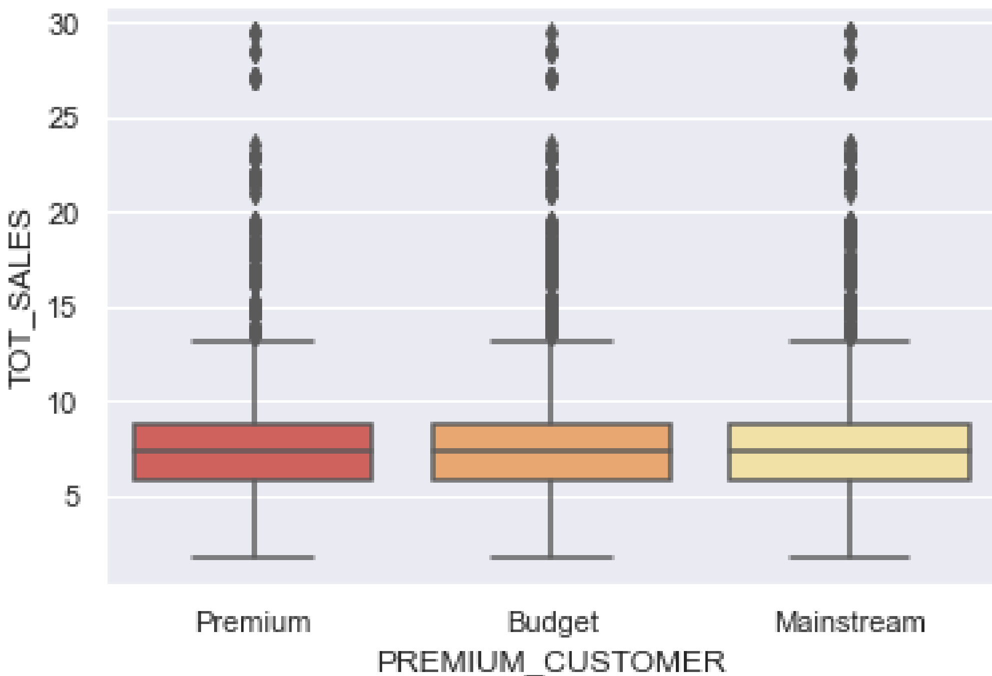
```
76787  Doritos Corn Chips  Cheese Supreme 170g        2        8.8

             LIFESTAGE PREMIUM_CUSTOMER  SIZE    BRAND
76786  OLDER FAMILIES          Premium   150  Doritos
76787  OLDER FAMILIES          Premium   170  Doritos
```

Removing the outliers.

[24]:
```
df.drop(index=[71456,71457],inplace=True)
```

[25]:
```
sns.boxplot(x="PREMIUM_CUSTOMER", y="TOT_SALES", data=df)
```

[25]: <matplotlib.axes._subplots.AxesSubplot at 0x1285d57b088>



[26]:
```
sns.scatterplot(data=df.sort_values(by='SIZE'),x='SIZE',y='TOT_SALES')
plt.title("Total Sales in function of Size")
plt.xlabel("Size")
plt.ylabel("Total Sale");
```

4 Data Analysis

Average total spent by type of customer

[27]:
```
df.pivot_table('TOT_SALES','PREMIUM_CUSTOMER',aggfunc={'TOT_SALES':
 ↪['mean','count','sum']})
```

[27]:

	count	mean	sum
PREMIUM_CUSTOMER			
Budget	86762	7.277458	631406.85
Mainstream	95043	7.374193	700865.40
Premium	64935	7.282751	472905.45

As expected Mainstream customers represent the majority of customers buying chips, followed by Budget customers. We can also see that, on average, beeing a Budget, Mainstream or Premium customer doesns't affect the value spent. Although Mainstream and Budget customers represent 74% of sales.

Top 5 selling brands and their mean sale value

[28]:
```
df.pivot_table('TOT_SALES','BRAND',aggfunc={'TOT_SALES':
 ↪['count','mean','sum']}).sort_values('count',ascending=False).head(5)
```

[28]:

	count	mean	sum
BRAND			
Kettle	41288	9.451652	390239.8
Smiths	27390	7.408127	202908.6
Doritos	25224	8.972800	226329.9
Pringles	25102	7.077344	177655.5
RRD	16321	5.367778	87607.5

Kettle chips not only sells almost double the amount compared to the second highest selling brand, it also has a higher mean value spent.

Total sales by LIFESTAGE and PREMIUM_CUSTOMER

[29]:
```
df.
↪pivot_table('TOT_SALES',['LIFESTAGE','PREMIUM_CUSTOMER'],aggfunc={'TOT_SALES':
↪['sum']}).sort_values('sum',ascending=False).head(3)
```

[29]:

		sum
LIFESTAGE	PREMIUM_CUSTOMER	
OLDER FAMILIES	Budget	156863.75
YOUNG SINGLES/COUPLES	Mainstream	147582.20
RETIREES	Mainstream	145168.95

Sales are coming mainly from Budget - older families, Mainstream - young singles/couples, and Mainstream - retirees.

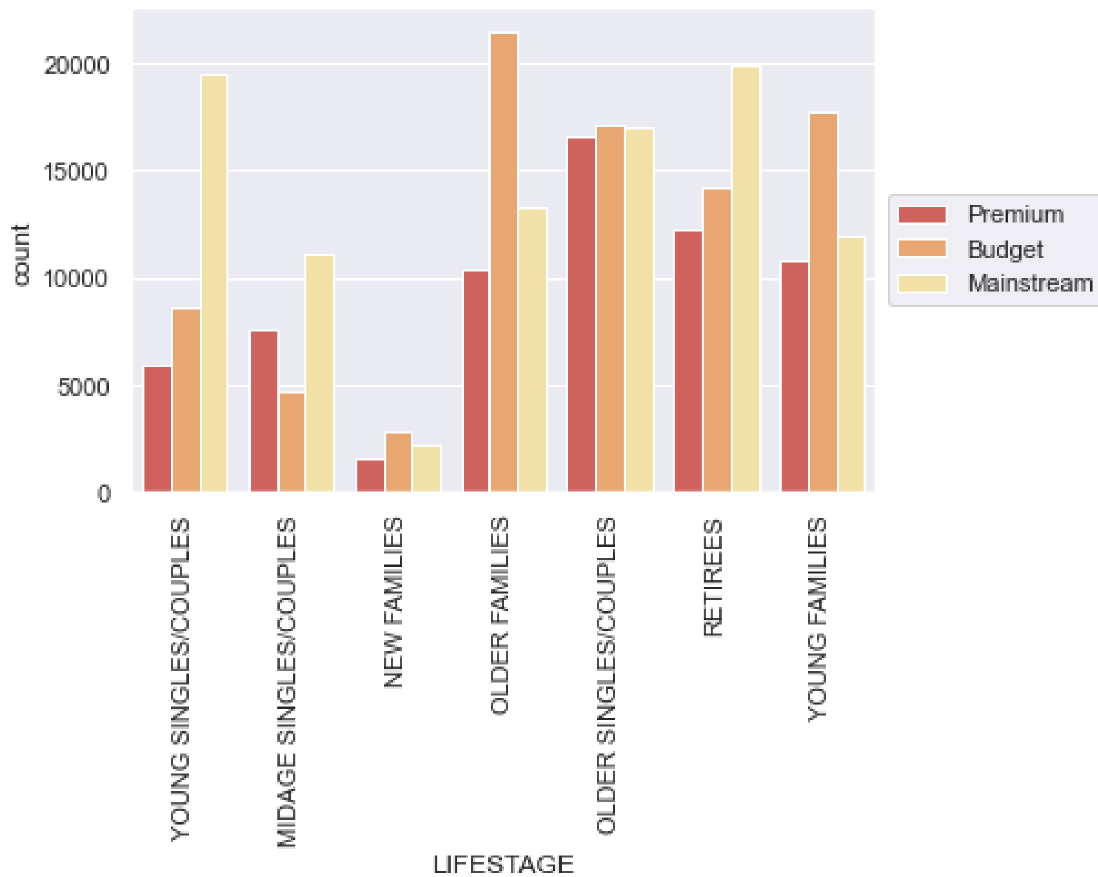
Number of customers by LIFESTAGE and PREMIUM_CUSTOMER

[30]:
```
df.
↪pivot_table('TOT_SALES',['LIFESTAGE','PREMIUM_CUSTOMER'],aggfunc={'TOT_SALES':
↪['count']}).sort_values('count',ascending=False).head(3)
```

[30]:

		count
LIFESTAGE	PREMIUM_CUSTOMER	
OLDER FAMILIES	Budget	21514
RETIREES	Mainstream	19970
YOUNG SINGLES/COUPLES	Mainstream	19544

[31]:
```
sns.countplot(data=df,x='LIFESTAGE',hue='PREMIUM_CUSTOMER')
plt.xticks(rotation=90)
plt.legend(loc='center left', bbox_to_anchor=(1, 0.5));
```



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

Average number of units bought per customer

[32]:
```
df.pivot_table('PROD_QTY',['LIFESTAGE','PREMIUM_CUSTOMER'],aggfunc={'PROD_QTY':
 ↪['mean']}).sort_values('mean',ascending=False).head(6)
```

[32]:

LIFESTAGE	PREMIUM_CUSTOMER	mean
OLDER FAMILIES	Mainstream	1.948795
	Premium	1.945496
	Budget	1.945384
YOUNG FAMILIES	Mainstream	1.941408
	Budget	1.941226
	Premium	1.938149

Older and young families in general buy more chips per customer.

Average price per unit chips bought for each customer

```
[33]: df.
       ↪pivot_table('TOT_SALES',['LIFESTAGE','PREMIUM_CUSTOMER'],aggfunc={'TOT_SALES':
       ↪['mean','count']}).sort_values('mean',ascending=False)
```

```
[33]:                                          count      mean
      LIFESTAGE              PREMIUM_CUSTOMER
      MIDAGE SINGLES/COUPLES Mainstream        11095  7.637156
      YOUNG SINGLES/COUPLES  Mainstream        19544  7.551279
      RETIREES               Premium           12236  7.461315
      OLDER SINGLES/COUPLES  Premium           16560  7.459997
      RETIREES               Budget            14225  7.445786
      OLDER SINGLES/COUPLES  Budget            17172  7.444305
      NEW FAMILIES           Mainstream         2185  7.313364
      OLDER SINGLES/COUPLES  Mainstream        17061  7.306049
      YOUNG FAMILIES         Budget            17763  7.302705
      NEW FAMILIES           Budget             2824  7.297256
      OLDER FAMILIES         Budget            21514  7.291241
      YOUNG FAMILIES         Premium           10784  7.285951
      OLDER FAMILIES         Mainstream        13241  7.281440
      RETIREES               Mainstream        19970  7.269352
      OLDER FAMILIES         Premium           10403  7.232779
      NEW FAMILIES           Premium            1488  7.231720
      YOUNG FAMILIES         Mainstream        11947  7.226772
      MIDAGE SINGLES/COUPLES Premium            7612  7.152371
                             Budget             4691  7.108442
      YOUNG SINGLES/COUPLES  Premium            5852  6.673325
                             Budget             8573  6.663023
```

Mainstream - midage and young singles and couples are more willing to pay more per packet of chips compared to their budget and premium counterparts. This may be due to premium shoppers being more likely to buy healthy snacks. As the difference in average price per unit isn't large, we can check if this difference is statistically different.

T-test statistics

```
[34]: stats.ttest_ind(df[((df['LIFESTAGE']=='MIDAGE SINGLES/
       ↪COUPLES')|(df['LIFESTAGE']=='YOUNG SINGLES/
       ↪COUPLES'))&(df['PREMIUM_CUSTOMER']=='Mainstream')]['TOT_SALES'],\
                      df[((df['LIFESTAGE']=='MIDAGE SINGLES/
       ↪COUPLES')|(df['LIFESTAGE']=='YOUNG SINGLES/
       ↪COUPLES'))&(df['PREMIUM_CUSTOMER']!='Mainstream')]['TOT_SALES'])
```

```
[34]: Ttest_indResult(statistic=33.200521751400665, pvalue=1.9916804791067727e-239)
```

As we can see from all the p-values the unit price for mainstream, young and mid-age singles and couples are significantly higher than that of budget or premium, young and midage singles and couples.

4.1 Deeper dive into highest spender on average, the Mainstream - midage singles/couples.

[35]:
```
midage_main = df[(df['LIFESTAGE']=='MIDAGE SINGLES/
↪COUPLES')&(df['PREMIUM_CUSTOMER']=='Mainstream')]
other = df.drop(df[(df['LIFESTAGE']=='MIDAGE SINGLES/
↪COUPLES')&(df['PREMIUM_CUSTOMER']=='Mainstream')].index)
```

Target audience's preferred brand

[36]:
```
midage_main.groupby('BRAND')['PROD_QTY'].count().sort_values(ascending=False).
↪head(3)
```

[36]:
```
BRAND
Kettle     2136
Doritos    1210
Smiths     1176
Name: PROD_QTY, dtype: int64
```

Our target's top 3 brands of chip are the same as our total customers the only diferrence being that our target prefers Doritos over Smiths.

Perfoming an affinity analysis on the brand

[37]:
```
qty_segment1 = midage_main['PROD_QTY'].sum()
qty_segment2 = other['PROD_QTY'].sum()
qty_seg_1_by_brand = midage_main.groupby('BRAND').sum()['PROD_QTY'] /␣
↪qty_segment1
qty_seg_2_by_brand = other.groupby('BRAND').sum()['PROD_QTY'] / qty_segment2
brand_affinity = pd.
↪merge(qty_seg_1_by_brand,qty_seg_2_by_brand,suffixes=('_target','_other'),on='BRAND')
brand_affinity['AFFINITY'] = brand_affinity['PROD_QTY_target'] /␣
↪brand_affinity['PROD_QTY_other']
brand_affinity.sort_values(by='AFFINITY',ascending=False)
```

[37]:

BRAND	PROD_QTY_target	PROD_QTY_other	AFFINITY
Kettle	0.192571	0.166893	1.153857
Twisties	0.043935	0.038260	1.148326
Cobs	0.044831	0.039227	1.142875
Tostitos	0.043558	0.038314	1.136882
Grain	0.027719	0.025321	1.094683
Infuzions	0.061755	0.057457	1.074792
Cheezels	0.019846	0.018536	1.070705
Doritos	0.108895	0.102454	1.062870
Tyrrells	0.026917	0.026107	1.031035
Pringles	0.104181	0.101982	1.021564
Thins	0.057182	0.057250	0.998807

```
Smiths          0.106114     0.110694  0.958630
NCC             0.005280     0.005721  0.922803
Cheetos         0.010135     0.011833  0.856563
RRD             0.054259     0.066209  0.819518
Natural         0.019375     0.024518  0.790242
CCs             0.014425     0.018485  0.780388
Smith           0.009051     0.012060  0.750527
GrnWves         0.004243     0.005953  0.712697
Woolworths      0.004337     0.006189  0.700757
Burger          0.004337     0.006407  0.676895
French          0.003818     0.005704  0.669468
WW              0.027012     0.042049  0.642381
Sunbites        0.006223     0.012378  0.502717
```

In a more in-deepth look into our target preferency's we notice that they're 15% more likely to purchase Kettle chips and 50% less likely to purchase Sunbites compared to the rest of the population.

Target audience's preferred size of chips

[38]:
```
midage_main.groupby('SIZE')['PROD_QTY'].count().sort_values(ascending=False).
↪head(3)
```

[38]:
```
SIZE
175    2975
150    1777
134    1159
Name: PROD_QTY, dtype: int64
```

[39]:
```
other.groupby('SIZE')['PROD_QTY'].count().sort_values(ascending=False).head(3)
```

[39]:
```
SIZE
175    63415
150    38426
134    23943
Name: PROD_QTY, dtype: int64
```

Our targeted segment preferred size of chips doesn't seem to differ from the rest of the customers.

Perfoming an affinity analysis on the size of chips

[40]:
```
qty_seg_1_by_size = midage_main.groupby('SIZE').sum()['PROD_QTY'] / qty_segment1
qty_seg_2_by_size = other.groupby('SIZE').sum()['PROD_QTY'] / qty_segment2
brand_affinity = pd.
↪merge(qty_seg_1_by_size,qty_seg_2_by_size,suffixes=('_target','_other'),on='SIZE')
brand_affinity['AFFINITY'] = brand_affinity['PROD_QTY_target'] /␣
↪brand_affinity['PROD_QTY_other']
brand_affinity.sort_values(by='AFFINITY',ascending=False)
```

[40]:

SIZE	PROD_QTY_target	PROD_QTY_other	AFFINITY
270	0.030736	0.025373	1.211382
330	0.059728	0.050607	1.180220
110	0.102060	0.090542	1.127218
135	0.014519	0.013144	1.104660
210	0.027719	0.025321	1.094683
380	0.028426	0.025980	1.094134
250	0.013199	0.012888	1.024185
134	0.104181	0.101982	1.021564
175	0.268562	0.268864	0.998875
150	0.160420	0.163093	0.983615
170	0.079385	0.081044	0.979529
165	0.057088	0.061979	0.921084
190	0.010324	0.012142	0.850263
160	0.009051	0.012048	0.751221
70	0.004526	0.006142	0.736790
180	0.004243	0.005953	0.712697
220	0.004337	0.006407	0.676895
200	0.012068	0.018186	0.663583
125	0.003206	0.005926	0.540910
90	0.006223	0.012378	0.502717

As it seems Mainstream midage singles/couples are 21% more likely to purchase a 270g pack of chips compared to the rest of the population and 50% less likely to purchase a 90g pack compared to the rest of the population.