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

Customer Purchase Behaviour

```
df = pd.read_csv('/content/QVI_purchase_behaviour.csv')
```

```
df.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

```
df.shape
```

```
(72637, 3)
```

```
len(df['LYLTY_CARD_NBR'].unique())
```

```
72637
```

```
df['LIFESTAGE'].value_counts()
```

```
RETIREES                  14805
OLDER SINGLES/COUPLES     14609
YOUNG SINGLES/COUPLES     14441
OLDER FAMILIES             9780
YOUNG FAMILIES             9178
MIDAGE SINGLES/COUPLES     7275
NEW FAMILIES               2549
Name: LIFESTAGE, dtype: int64
```

```
df.isnull().sum() # So this is a cleaned dataset with no Nan(null) values
```

```
LYLTY_CARD_NBR      0
```

The total number of customers are 72,637 out of which majority of the customers are Retirees

or Older Singles/Couples or Young Singles/Couples(14K each). New families,Young families and midage singles/couples are less compared to the others but customers of new families are very less. Generally we think young couples/singles purchase frequently and purchase more often but its also interesting to note that even older singles and couples purchase a lot. There are no missing/empty values in our dataset and the dataset is completely clean.

```
df['PREMIUM_CUSTOMER'].value_counts()
```

```
Mainstream    29245
Budget        24470
Premium       18922
Name: PREMIUM_CUSTOMER, dtype: int64
```

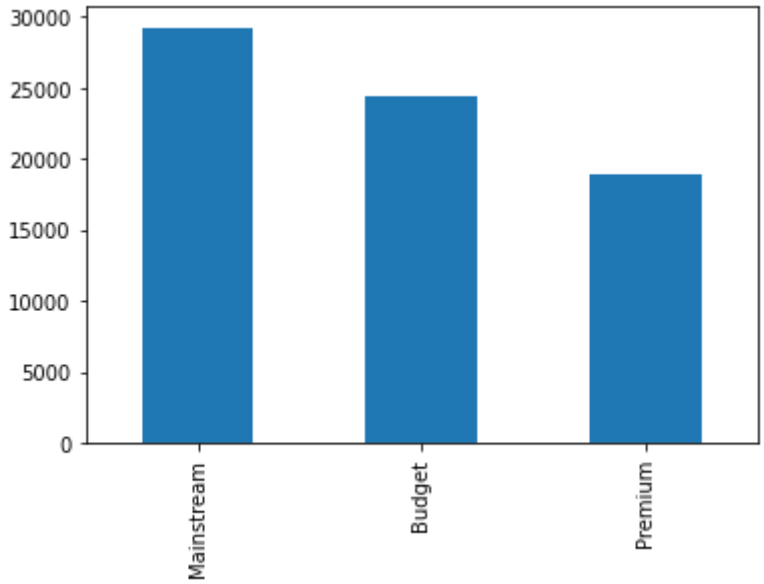
```
df['PREMIUM_CUSTOMER'].value_counts(normalize=True)*100
```

```
Mainstream    40.261850
Budget        33.688065
Premium       26.050085
Name: PREMIUM_CUSTOMER, dtype: float64
```

So the number of premium customers are actually less when compared to the other types of customers. 26% of the total customers are premium customers.

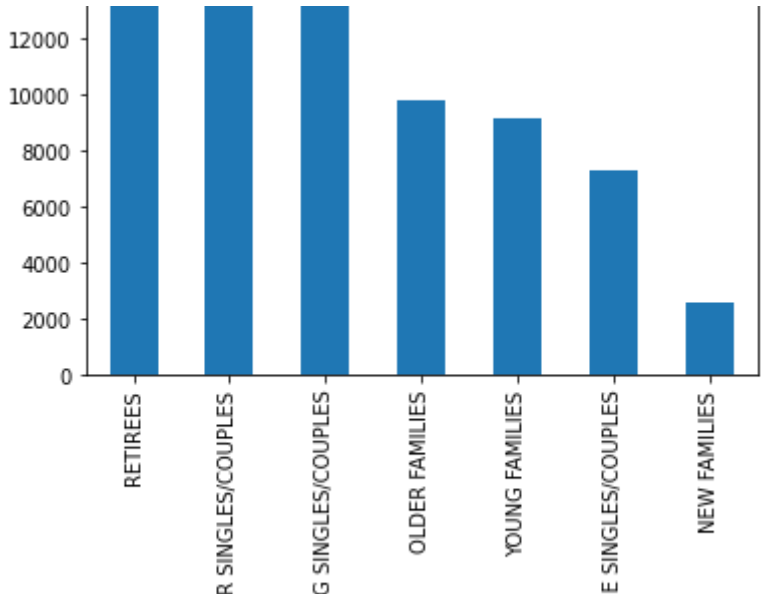
```
df['PREMIUM_CUSTOMER'].value_counts().plot.bar() # Distribution of the Premium Cus
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f07e493ef98>
```




```
df['LIFESTAGE'].value_counts().plot.bar() # Distribution of the different Lifestag
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f07e4873278>
```




```
grouped = df.groupby(['LIFESTAGE','PREMIUM_CUSTOMER'])
grouped.size()
```

```
LIFESTAGE               PREMIUM_CUSTOMER
MIDAGE SINGLES/COUPLES  Budget             1504
                        Mainstream         3340
                        Premium            2431
NEW FAMILIES            Budget             1112
                        Mainstream          849
                        Premium             588
OLDER FAMILIES          Budget             4675
                        Mainstream         2831
                        Premium            2274
OLDER SINGLES/COUPLES   Budget             4929
                        Mainstream         4930
                        Premium            4750
RETIREES                Budget             4454
                        Mainstream         6479
                        Premium            3872
YOUNG FAMILIES          Budget             4017
                        Mainstream         2728
                        Premium            2433
YOUNG SINGLES/COUPLES   Budget             3779
                        Mainstream         8088
                        Premium            2574
dtype: int64
```

This table shows us the distribution of different stages with respect to the premium customers. So the maximum number of premium customers are present are Older Singles/Couples.

```
customer_table = pd.crosstab(index=df["LIFESTAGE"],
                             columns=df["PREMIUM_CUSTOMER"])

customer_table # A 2 way table between lifestage and premium customers
```

PREMIUM_CUSTOMER	Budget	Mainstream	Premium
LIFESTAGE			

MIDAGE SINGLES/COUPLES	1504	3340	2431
NEW FAMILIES	1112	849	588
OLDER FAMILIES	4675	2831	2274
OLDER SINGLES/COUPLES	4929	4930	4750
RETIREES	4454	6479	3872
YOUNG FAMILIES	4017	2728	2433
YOUNG SINGLES/COUPLES	3779	8088	2574

This is a 2 way table and is easier to visualize than the groupby function. We can easily compare the premium customer based on different lifestages.

Transaction Data

```
df1 = pd.read_csv('/content/QVI_transaction_data.csv')
```

```
df1.head()
```

	DATE	STORE_NBR	LYLTY_CARD_NBR	TXN_ID	PROD_NBR	PROD_NAME	PROD_QTY
0	43390	1	1000	1	5	Natural Chip Compny SeaSalt175g	2
1	43599	1	1307	348	66	CCs Nacho Cheese 175g	3
2	43605	1	1343	383	61	Smiths Crinkle Cut Chips Chicken 170g	2

```
df1.shape # 2,64,836 entries
```

```
(264836, 8)
```

```
df1.dtypes
```

```
DATE                int64
STORE_NBR           int64
LYLTY_CARD_NBR      int64
```

The datatype for date has to be changed as it is in int64(it has to be converted to Date datatype). The values of DATE do not signify any meaning and this is a very important thing which needs to be handled.

```
dtype: object
```

```
print(len(df1['DATE'].unique()))
print(len(df1['STORE_NBR'].unique()))
print(len(df1['LYLTY_CARD_NBR'].unique()))
print(len(df1['PROD_NBR'].unique()))
print(len(df1['TOT_SALES'].unique()))
```

```
364
272
72637
114
112
```

```
df1.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
```

```
df1['TOT_SALES'].sum() # The total number of items sold in a particular year is 19
```

```
1934415.0000000002
```

So this dataset consists of 2,64,836 instances which shows all the transactions of a particular year.There are 272 stores for purchasing with a total of 114 products.The number of customers have matched with the previous dataset where the total number of customers are 72,637. Also the dataset has no missing values so its a structured and clean dataset.

```
grouped = df1.groupby(['LYLTY_CARD_NBR','TOT_SALES'])
grouped.size()
```

```
LYLTY_CARD_NBR  TOT_SALES
1000            6.0          1
1002            2.7          1
1003            3.0          1
```

```
1003          3.0          1
              3.6          1
```

```
customer = df1.groupby(['LYLTY_CARD_NBR', 'TOT_SALES'])
customer.first() # This gives us the details of every customer,the number of sales
```

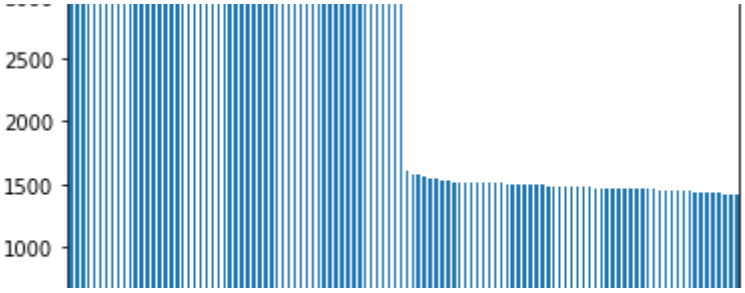
		DATE	STORE_NBR	TXN_ID	PROD_NBR	PROD_NAME	P
LYLTY_CARD_NBR	**TOT_SALES**						
1000	**6.0**	43390	1	1	5	Natural Chip Compny SeaSalt175g	
1002	**2.7**	43359	1	2	58	Red Rock Deli Chikn&Garlic Aioli 150g	
1003	**3.0**	43532	1	4	106	Natural ChipCo Hony Soy Chckn175g	
	3.6	43531	1	3	52	Grain Waves Sour Cream&Chives 210G	
1004	**1.9**	43406	1	5	96	WW Original Stacked Chips 160g	
...	**...**	...	...	...	...	...	

```
df1['PROD_NBR'].value_counts()
```

```
102    3304
108    3296
33     3269
112    3268
75     3265
       ...
11     1431
76     1430
98     1419
29     1418
72     1410
Name: PROD_NBR, Length: 114, dtype: int64
```

```
df1['PROD_NBR'].value_counts().plot.bar() # This plot shows us which product is bo
# It does not give us the quantity but the no of custoers who have bought that pro
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f07d83f7ef0>
```



The product number 102 has 3304 entries which is the highest. This indicates that this was the product bought by maximum number of customers. The above plot shows us which product is bought by how many number of customers.It does not give us the quantity but the no of customers who have bought that product.

```
df1.groupby('PROD_NBR')['TOT_SALES'].sum()
```

```
PROD_NBR
1       8125.8
2      22944.4
3      28308.4
4      40352.0
5       8331.0
        ...
110     5367.5
111     9135.0
112    26149.2
113    27853.0
114    27567.8
Name: TOT_SALES, Length: 114, dtype: float64
```

```
df1.groupby('PROD_NBR')['TOT_SALES'].sum().idxmax()
```

```
4
```

```
df1[df1['PROD_NBR']==4]['PROD_NAME']
```

```
41        Dorito Corn Chp     Supreme 380g
67        Dorito Corn Chp     Supreme 380g
157       Dorito Corn Chp     Supreme 380g
199       Dorito Corn Chp     Supreme 380g
236       Dorito Corn Chp     Supreme 380g
                          ...
264457    Dorito Corn Chp     Supreme 380g
264470    Dorito Corn Chp     Supreme 380g
264536    Dorito Corn Chp     Supreme 380g
264584    Dorito Corn Chp     Supreme 380g
264807    Dorito Corn Chp     Supreme 380g
Name: PROD_NAME, Length: 3185, dtype: object
```

Double-click (or enter) to edit

So this gives us the total sales for every product and we can see that the maxiumum number of

sales for a particular product is 40,352 and this is for the product number 4 which is Dorito Corn Chp Supreme 380g.

```
df1.groupby('PROD_NBR')['PROD_QTY'].sum()
```

```
PROD_NBR
1      2802
2      6038
3      6157
4      6509
5      2777
       ...
110    2825
111    3045
112    6227
113    6055
114    5993
Name: PROD_QTY, Length: 114, dtype: int64
```

```
df1.groupby('PROD_NBR')['PROD_QTY'].sum().idxmax()
```

```
4
```

The maximum number of quantity for a particular product is 6509 and the product is Dorito Corn Chp Supreme 380g. So as we can see there is a huge demand for this chips product when compared to other products. So the company has to manufacture more products of this kind as the customers are more willing to buy this chips product.