Task: I am part of Quantium's retail analytics team and have been approached by a client, the Category Manager for Chips, who wants to better understand the types of customers who purchase Chips and their purchasing behaviour within the region. The insights from this analysis will feed into the supermarket's strategic plan for the chip category in the next half year.

In [1]:
```
#imports necessary for analysis
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)
pd.set_option('display.max_columns', 1000)
pd.set_option('display.width', 1000)

from pandas import Series
from pandas import DataFrame

import os # accessing directory structure

import numpy as np # linear algebra
from numpy import ndarray
from numpy.random import randn
from numpy import loadtxt, where

import matplotlib as plt
import matplotlib.pyplot as plt
%matplotlib inline
from matplotlib.patches import Patch
from matplotlib.lines import Line2D
plt.rc("font", size=14)
get_ipython().run_line_magic('matplotlib', 'inline')

from mpl_toolkits.mplot3d import Axes3D

import seaborn as sns

import sklearn
import sklearn.preprocessing
from sklearn.preprocessing import StandardScaler

from scipy import stats

import plotly.express as px
```

In [2]:
```
#confirm package versions
print("pandas version " + pd.__version__)
print("numpy version " + np.__version__)
print("seaborn version " + sns.__version__)
```

```
pandas version 1.4.2
numpy version 1.21.5
seaborn version 0.11.2
```

In [3]:
```
#ignore future warnings
import warnings
warnings.filterwarnings('ignore')
```

In [4]:
```
#read customer csv into pandas dataframe
df_customer = pd.read_csv('QVI_purchase_behaviour.csv')
```

In [5]:
```
#evaluate top of loaded data
df_customer.head()
```

Out[5]:

	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

In [6]:
```
#determine data types and get the shape of the data (# of columns and rows)
df_customer.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 72637 entries, 0 to 72636
Data columns (total 3 columns):
 #   Column            Non-Null Count  Dtype 
---  ------            --------------  ----- 
 0   LYLTY_CARD_NBR    72637 non-null  int64 
 1   LIFESTAGE         72637 non-null  object
 2   PREMIUM_CUSTOMER  72637 non-null  object
dtypes: int64(1), object(2)
memory usage: 1.7+ MB
```

In [7]:
```
#evalute for null values
df_customer.isnull().sum()
```

Out[7]:
```
LYLTY_CARD_NBR      0
LIFESTAGE           0
PREMIUM_CUSTOMER    0
dtype: int64
```

In [8]:
```
#check for duplicates
df_customer.duplicated().sum()
```

Out[8]:
```
0
```

In [9]:
```
df_customer.PREMIUM_CUSTOMER.unique()
```

Out[9]:
```
array(['Premium', 'Mainstream', 'Budget'], dtype=object)
```

In [10]:
```
#Convert categorical to binary where "Premium"=1 or yes and "Budget" and "Mainstrain"
df_customer['PREMIUM_CUSTOMER'].replace(['Premium', 'Budget', 'Mainstream'],
                        [1, 0, 0], inplace=True)
```

In [11]:
```
#evaluate top of loaded data
df_customer.head()
```

Out[11]:

	LYLTY_CARD_NBR	LIFESTAGE	PREMIUM_CUSTOMER
0	1000	YOUNG SINGLES/COUPLES	1
1	1002	YOUNG SINGLES/COUPLES	0
2	1003	YOUNG FAMILIES	0
3	1004	OLDER SINGLES/COUPLES	0
4	1005	MIDAGE SINGLES/COUPLES	0

In [12]:
```
#determine data types
df_customer.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 72637 entries, 0 to 72636
Data columns (total 3 columns):
 #   Column            Non-Null Count  Dtype 
---  ------            --------------  ----- 
 0   LYLTY_CARD_NBR    72637 non-null  int64 
 1   LIFESTAGE         72637 non-null  object
 2   PREMIUM_CUSTOMER  72637 non-null  int64 
dtypes: int64(2), object(1)
memory usage: 1.7+ MB
```

In [13]:
```
#read transaction csv into pandas dataframe
#file converted to csv from .xlsx
df_transaction = pd.read_csv('QVI_transaction_data.csv')
```

In [14]:
```
#evaluate top of loaded data
df_transaction.head()
```

Out[14]:

	DATE	STORE_NBR	LYLTY_CARD_NBR	TXN_ID	PROD_NBR	PROD_NAME	PROD_QTY	TOT
0	10/17/2018	1	1000	1	5	Natural Chip Compny SeaSalt175g	2	
1	5/14/2019	1	1307	348	66	CCs Nacho Cheese 175g	3	
2	5/20/2019	1	1343	383	61	Smiths Crinkle Cut Chips Chicken 170g	2	
3	8/17/2018	2	2373	974	69	Smiths Chip Thinly S/Cream&Onion 175g	5	
4	8/18/2018	2	2426	1038	108	Kettle Tortilla ChpsHny&Jlpno Chili 150g	3	

In [15]:
```
#determine data types and get shape of data (# of columns and rows)
df_transaction.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 264836 entries, 0 to 264835
Data columns (total 8 columns):
 #   Column          Non-Null Count   Dtype
---  ------          --------------   -----
 0   DATE            264836 non-null  object
 1   STORE_NBR       264836 non-null  int64
 2   LYLTY_CARD_NBR  264836 non-null  int64
 3   TXN_ID          264836 non-null  int64
 4   PROD_NBR        264836 non-null  int64
 5   PROD_NAME       264836 non-null  object
 6   PROD_QTY        264836 non-null  int64
 7   TOT_SALES       264836 non-null  float64
dtypes: float64(1), int64(5), object(2)
memory usage: 16.2+ MB
```

In [16]:
```
#convert "DATE" to datetime
df_transaction['DATE'] = pd.to_datetime(df_transaction['DATE'])
#show the change
df_transaction.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 264836 entries, 0 to 264835
Data columns (total 8 columns):
 #   Column          Non-Null Count   Dtype
---  ------          --------------   -----
 0   DATE            264836 non-null  datetime64[ns]
 1   STORE_NBR       264836 non-null  int64
 2   LYLTY_CARD_NBR  264836 non-null  int64
 3   TXN_ID          264836 non-null  int64
 4   PROD_NBR        264836 non-null  int64
 5   PROD_NAME       264836 non-null  object
 6   PROD_QTY        264836 non-null  int64
 7   TOT_SALES       264836 non-null  float64
dtypes: datetime64[ns](1), float64(1), int64(5), object(1)
memory usage: 16.2+ MB
```

In [17]:
```
#change "DATE" column display
df_transaction['DATE'] = pd.to_datetime(df_transaction["DATE"].dt.strftime('%Y-%m'))
```

In [18]:
```
#view "DATE" change
df_transaction.head()
```

Out[18]:

	DATE	STORE_NBR	LYLTY_CARD_NBR	TXN_ID	PROD_NBR	PROD_NAME	PROD_QTY	TOT_SALE
0	2018-10-01	1	1000	1	5	Natural Chip Compny SeaSalt175g	2	6
1	2019-05-01	1	1307	348	66	CCs Nacho Cheese 175g	3	6
2	2019-05-01	1	1343	383	61	Smiths Crinkle Cut Chips Chicken 170g	2	2
3	2018-08-01	2	2373	974	69	Smiths Chip Thinly S/Cream&Onion 175g	5	15
4	2018-08-01	2	2426	1038	108	Kettle Tortilla ChpsHny&Jlpno Chili 150g	3	13

In [19]:
```
#evalute for null values
df_transaction.isnull().sum()
```

Out[19]:
```
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
```

In [20]:
```
#evaluate for duplicate values
df_transaction.duplicated().sum()
```

Out[20]:
```
1
```

In [21]:
```
#remove duplicate-use keep='first' to keep the first occurance
df_transaction= df_transaction.drop_duplicates(subset=None, keep="first", inplace=Fals
```

In [22]:
```
#evaluate for duplicate values
df_transaction.duplicated().sum()
```

Out[22]:
```
0
```

In [23]:
```
#join two csv files at loyalty card number
df = df_transaction.merge(df_customer,on=["LYLTY_CARD_NBR"])
df.head()
```

Out[23]:

	DATE	STORE_NBR	LYLTY_CARD_NBR	TXN_ID	PROD_NBR	PROD_NAME	PROD_QTY	TOT_SALES
0	2018-10-01	1	1000	1	5	Natural Chip Compny SeaSalt175g	2	6.0
1	2019-05-01	1	1307	348	66	CCs Nacho Cheese 175g	3	6.3
2	2018-11-01	1	1307	346	96	WW Original Stacked Chips 160g	2	3.8
3	2019-03-01	1	1307	347	54	CCs Original 175g	1	2.1
4	2019-05-01	1	1343	383	61	Smiths Crinkle Cut Chips Chicken 170g	2	2.9

In [24]:
```
#determine data types and get shape of data
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 264835 entries, 0 to 264834
Data columns (total 10 columns):
 #   Column            Non-Null Count   Dtype
---  ------            --------------   -----
 0   DATE              264835 non-null  datetime64[ns]
 1   STORE_NBR         264835 non-null  int64
 2   LYLTY_CARD_NBR    264835 non-null  int64
 3   TXN_ID            264835 non-null  int64
 4   PROD_NBR          264835 non-null  int64
 5   PROD_NAME         264835 non-null  object
 6   PROD_QTY          264835 non-null  int64
 7   TOT_SALES         264835 non-null  float64
 8   LIFESTAGE         264835 non-null  object
 9   PREMIUM_CUSTOMER  264835 non-null  int64
dtypes: datetime64[ns](1), float64(1), int64(6), object(2)
memory usage: 22.2+ MB
```

In [25]:
```
#check for nulls
df.isnull().sum()
```

Out[25]:
```
DATE              0
STORE_NBR         0
LYLTY_CARD_NBR    0
TXN_ID            0
PROD_NBR          0
PROD_NAME         0
PROD_QTY          0
TOT_SALES         0
LIFESTAGE         0
PREMIUM_CUSTOMER  0
dtype: int64
```

In [26]:
```
#evaluate for duplicate values
df_transaction.duplicated().sum()
```

Out[26]:
```
0
```

In [27]:
```
#save clean data csv
df.to_csv('Quantium_clean_customer_Transaction.csv')
```

Analysis

In [28]:
```
import pandas_profiling as pp
from pandas_profiling import ProfileReport
from pandas_profiling.utils.cache import cache_file
profile = ProfileReport(df)
profile
```

```
Summarize dataset:   0%|          | 0/5 [00:00<?, ?it/s]
Generate report structure:   0%|          | 0/1 [00:00<?, ?it/s]
Render HTML:   0%|          | 0/1 [00:00<?, ?it/s]
```

Overview

Dataset statistics

Number of variables	10
Number of observations	264835
Missing cells	0
Missing cells (%)	0.0%
Duplicate rows	0
Duplicate rows (%)	0.0%
Total size in memory	22.2 MiB
Average record size in memory	88.0 B

Variable types

DateTime	1
Numeric	6
Categorical	3

Alerts

`PROD_NAME` has a high cardinality: 114 distinct values	**High cardinality**
`STORE_NBR` is highly correlated with `LYLTY_CARD_NBR` and 1 other fields (LYLTY_CARD_NBR, TXN_ID)	**High correlation**

Out[28]:

In [29]:
```
profile.to_notebook_iframe()
```

Overview

Dataset statistics

Number of variables	10
Number of observations	264835
Missing cells	0
Missing cells (%)	0.0%
Duplicate rows	0
Duplicate rows (%)	0.0%
Total size in memory	22.2 MiB
Average record size in memory	88.0 B

Variable types

DateTime	1
Numeric	6
Categorical	3

Alerts

`PROD_NAME` has a high cardinality: 114 distinct values	**High cardinality**
`STORE_NBR` is highly correlated with `LYLTY_CARD_NBR` and 1 other fields (LYLTY_CARD_NBR, TXN_ID)	**High correlation**

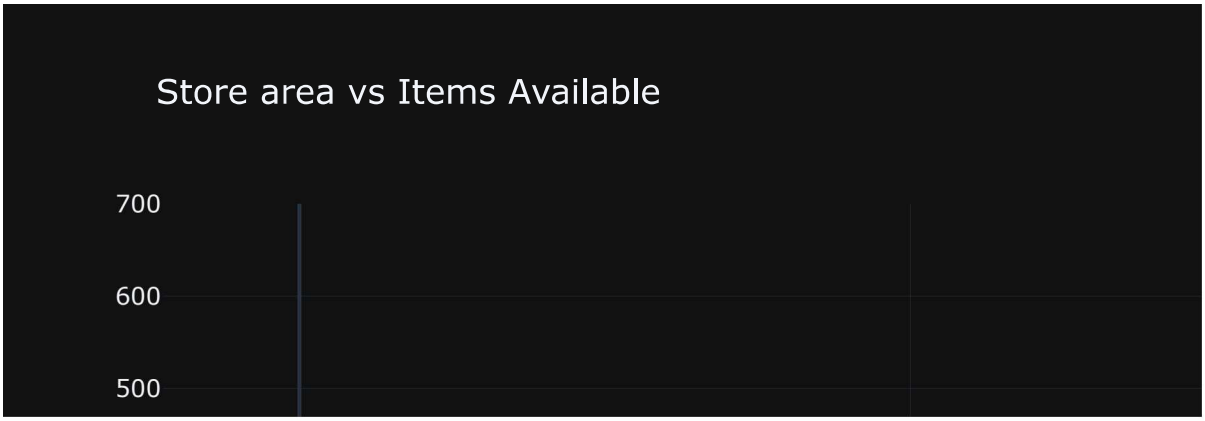
In [30]:
```
profile.to_file(output_file="quantium_report.html")
```

```
Export report to file:   0%|          | 0/1 [00:00<?, ?it/s]
```

In [31]:
```
df.skew().sort_values(ascending=False)
```

Out[31]:
```
PROD_QTY            220.102685
TOT_SALES            68.569567
LYLTY_CARD_NBR        2.466686
PREMIUM_CUSTOMER      1.075787
TXN_ID                0.093379
PROD_NBR              0.070136
STORE_NBR             0.030784
dtype: float64
```

In [32]:
```
fig=px.scatter(df,
               x="PROD_QTY",
               y="TOT_SALES",
               title="Store area vs Items Available",
               template="plotly_dark")
fig.show()
```

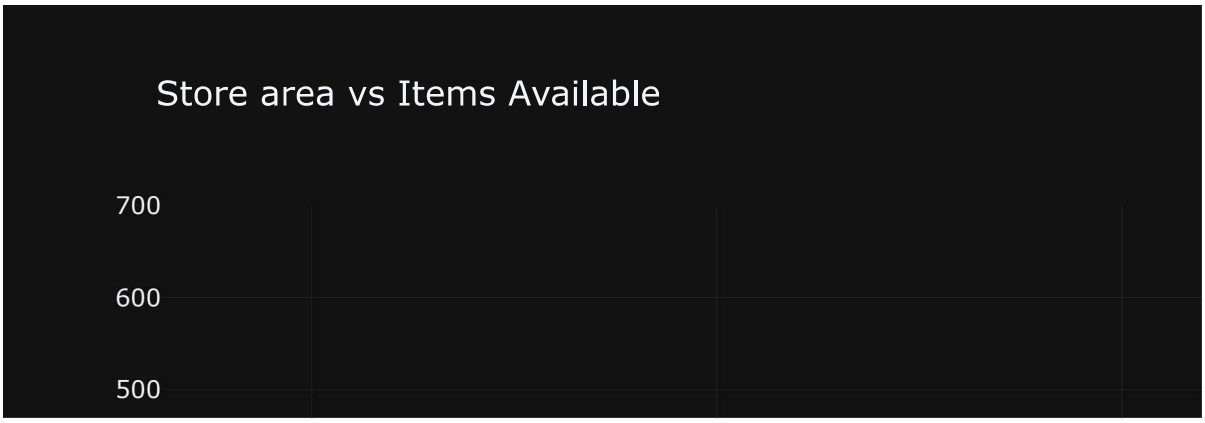


In [33]:
```
df.columns
```

Out[33]:
```
Index(['DATE', 'STORE_NBR', 'LYLTY_CARD_NBR', 'TXN_ID', 'PROD_NBR', 'PROD_NAME', 'PROD_QTY', 'TOT_SALES', 'LIFESTAGE', 'PREMIUM_CUSTOMER'], dtype='object')
```

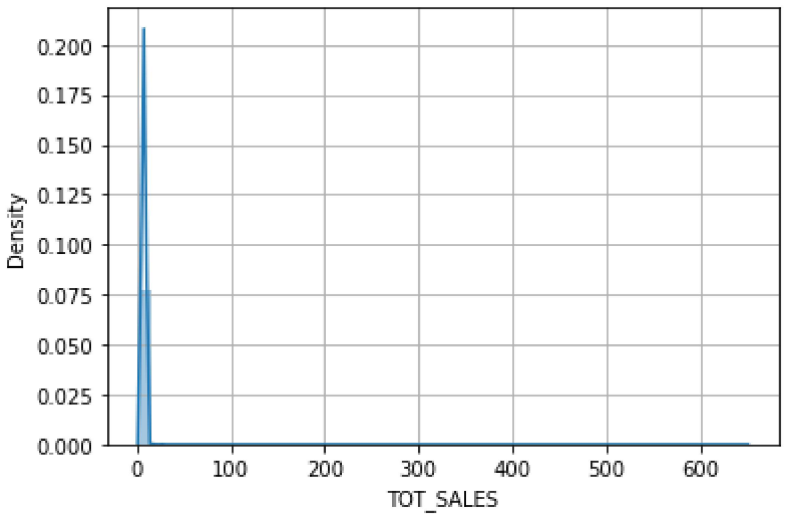
In [34]:
```
fig=px.scatter(df,
               x="LIFESTAGE",
               y="TOT_SALES",
               title="Store area vs Items Available",
```

```
                template="plotly_dark")
fig.show()
```

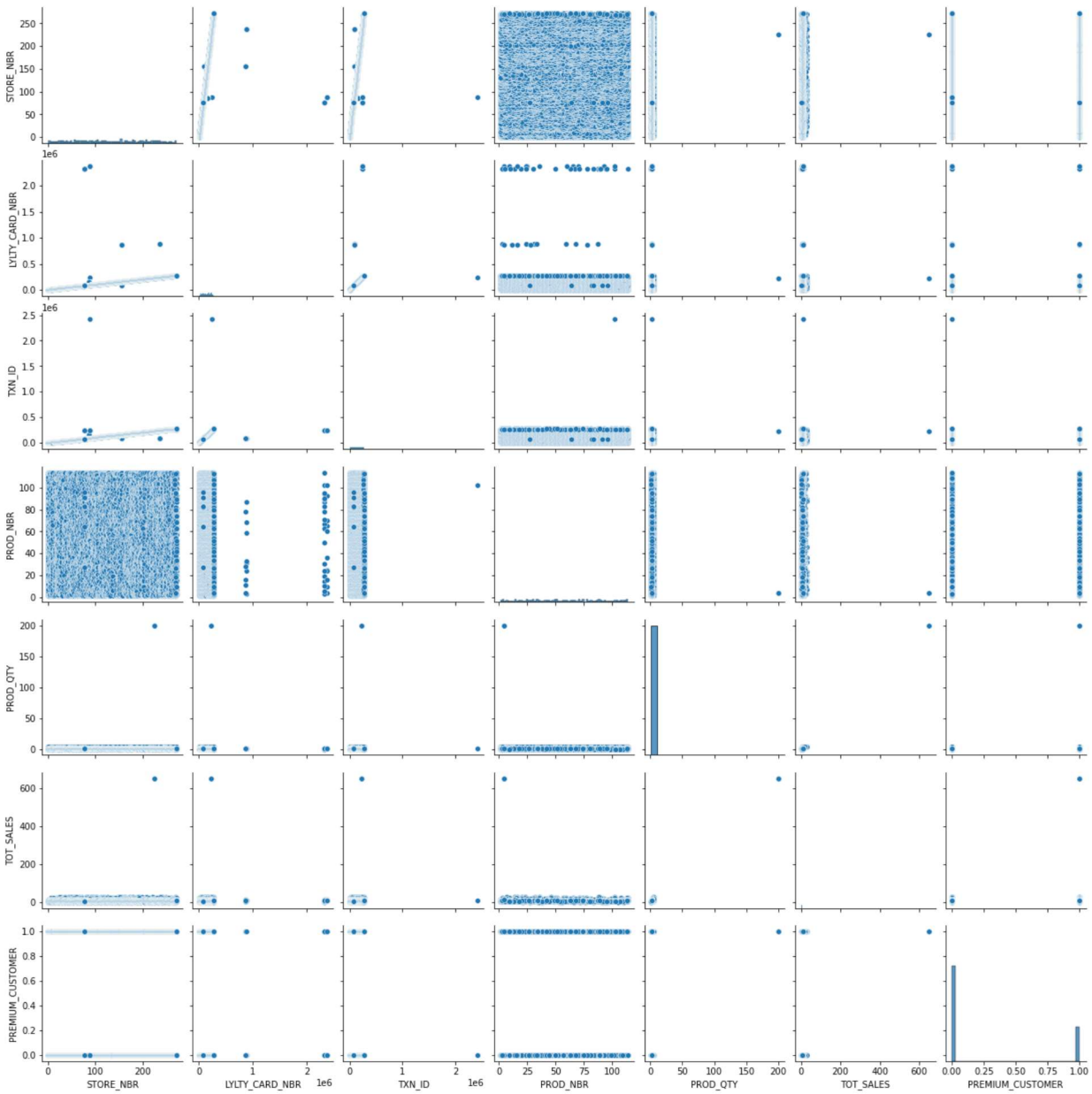


In [35]:
```
sns.distplot(df['TOT_SALES'])

plt.grid()
plt.show()
```



In [36]:
```
sns.pairplot(df)
plt.show()
```



Data Cleaning:

QVI_purchase_behaviour.csv has 3 columns with 72637 rows. There are 0 duplicates and 0 missing values in this dataset.

LYLTY_CARD_NBR is an int64, LIFESTAGE is an object and PREMIUM_CUSTOMER is an object.

PREMIUM_CUSTOMER has been converted to binary where "Premium"=1 or yes and "Budget" and "Mainstrain" = 0 or no.

PREMIUM_CUSTOMER is now an int64.

QVI_transaction_data.csv has 8 columns with 264836 rows. There are no missing values, there was one duplicate value found, the

last duplicate value was removed from the dataset leaving no duplicates. The column “DATE” was converted from int64 to

datetime64, and the column view was converted to year, month, day.

The two csv files were joined at loyalty card number. There are now 10 unique variables with 264835 observations in the

combined dataset. There are zero missing values. There are zero duplicates. There are 3 variable types 1 datetime, 6 numeric,

and 3 categorical.

Univariate Analysis:

"DATE" = there are 12 unique dates in this dataset beginning on 2018-07-01 and ending on 2019-06-01 meaning we have 11 months

of data in this dataset.

“STORE_NBR” = there are 272 unique values aka data on 272 different store locations.

“LYLTY_CARD_NBR” = there are 72,637 distinct records aka customers. Some loyalty card numbers are found multiple times in the

dataset. Below is their count.

Loyalty Card # Count 172032 18

162039 18

13138 17

116181 17

128178 17

230078 17

94185 16

129050 16

113080 16

104117 16

These customers used their loyalty card number the most in the given dataset.

“PROD_NBR” = there are 114 distinct values aka products in the dataset.

Value Count 102 3304

108 3296

33 3269

112 3268

75 3265

63 3257

74 3252

104 3242

14 3233

28 3229

PROD_NBR 114 shows up the most with a count of 3127

PROD_NBR 113 shows up 3170 times

PRD_NBR 112 shows up 3268 times

PRD_NBR 109 shows up 3210 times

PRD_NBR 108 shows up 3296 times

PROD_NAME =

Value	Count
Kettle Mozzarella Basil & Pesto 175g	3304
Kettle Tortilla ChpsHny&Jlpno Chili 150g	3296
Cobs Popd Swt/Chlli &Sr/Cream Chips 110g	3269
Tyrrells Crisps Ched & Chives 165g	3268
Cobs Popd Sea Salt Chips 110g	3265
Kettle 135g Swt Pot Sea Salt	3257
Tostitos Splash Of Lime 175g	3252
Infuzions Thai SweetChili PotatoMix 110g	3242
Smiths Crnkle Chip Orgnl Big Bag 380g	3233
Thins Potato Chips Hot & Spicy 175g	3229

"PROD_QTY" = the most frequently occurring product qty is 2 at 82.1% frequency

The second most frequently occurring product qty is 1 at 10.4% frequency.

The largest PROD_QTY is 200 with a frequency of <0.1% occurring 2x.

PROD_QTY of 4 occurs 450 times in the dataset with a frequency of 0.2%.

"LIFESTAGE" = there are 7 distinct life stages in the dataset.

OLDER SINGLES/COUPLES 54478 RETIREES 49763 OLDER FAMILIES 48596 YOUNG FAMILIES 43592 YOUNG SINGLES/COUPLES 36377 Other values (2) 32029

PREMIUM_CUSTOMERS = 195,145 of the rows were not associated with a premium customer

While 69,690 were associated with a premium customer. There were 125,455 more premium customers in this dataset than non

premium customers.

Relationships:

PROD_NAME has a high cardinality: 114 distinct values High cardinality

STORE_NBR is highly correlated with LYLTY_CARD_NBR and 1 other fields High correlation

LYLTY_CARD_NBR is highly correlated with STORE_NBR and 1 other fields High correlation

TXN_ID is highly correlated with STORE_NBR and 1 other fields High correlation

STORE_NBR is highly correlated with LYLTY_CARD_NBR and 1 other fields High correlation

LYLTY_CARD_NBR is highly correlated with STORE_NBR and 1 other fields High correlation

TXN_ID is highly correlated with STORE_NBR and 1 other fields High correlation

PROD_QTY is highly correlated with TOT_SALES High correlation

TOT_SALES is highly correlated with PROD_QTY High correlation

STORE_NBR is highly correlated with LYLTY_CARD_NBR and 1 other fields High correlation

LYLTY_CARD_NBR is highly correlated with STORE_NBR and 1 other fields High correlation

TXN_ID is highly correlated with STORE_NBR and 1 other fields High correlation

STORE_NBR is highly correlated with LYLTY_CARD_NBR and 1 other fields High correlation

LYLTY_CARD_NBR is highly correlated with STORE_NBR and 1 other fields High correlation

TXN_ID is highly correlated with STORE_NBR and 1 other fields High correlation

PROD_QTY is highly correlated with TOT_SALES High correlation

TOT_SALES is highly correlated with PROD_QTY High correlation

PROD_QTY is highly skewed (γ1 = 220.1026848) Skewed

TOT_SALES is highly skewed (γ1 = 68.56956685) Skewed

Customers tend to shop at the same stores often as there is high correlation between store number and loyalty card number.

Total sales and product quantity are highly correlated.

In [ ]: