



## Quantium Exploratory Data Analysis

---

**Frantz Alexander**

### Case Study Highlights:

- Perform analysis on consumer transaction data.
- Extrapolate essential business metrics.
- Derive insights from key trends on consumer segments.

## Libraries

---

```
In [ ]: import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import scipy as st

%matplotlib inline
```

## Data Preparation

---

### Import Dataset

```
In [ ]: def wrangle(filepath):
    df = pd.read_csv(filepath)

    # Setting column names to lower case
    df.columns = df.columns.str.lower()
```

```
return df
```

```
In [ ]: df = wrangle("QVI_transaction_data.csv")
```

## Data Preprocessing

Display the number of entries, the names and number of column features, the data type, and the memory space used.

```
In [ ]: df.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
```

This dataset consists of: 264836 records with 8 column features.

### Column Feature Description:

date: refers to the date of the transaction.

store\_nbr: refers to the unique store id number.

lylty\_card\_nbr: refers to the customer loyalty card number id.

txn\_id: refers to the transaction id of a product purchase.

prod\_nbr: refers to the product id.

prod\_name: refers to the product name.

prod\_qty: refers to the quantity of products sold.

tot\_sales: refers to the total sales revenue received from each sales transaction.

## Setting date column to datetime

```
In [ ]: df["date"] = pd.to_datetime(df["date"])
```

Checking the first 5 rows of the transaction dataset

```
In [ ]: df.head()
```

Out[ ]:		date	store_nbr	lylty_card_nbr	txn_id	prod_nbr	prod_name	prod_qty	tot_s
<b>0</b>	2018-10-17		1	1000	1	5	Natural Chip Compny SeaSalt175g	2	
<b>1</b>	2019-05-14		1	1307	348	66	CCs Nacho Cheese 175g	3	
<b>2</b>	2019-05-20		1	1343	383	61	Smiths Crinkle Cut Chips Chicken 170g	2	
<b>3</b>	2018-08-17		2	2373	974	69	Smiths Chip Thinly S/Cream&Onion 175g	5	
<b>4</b>	2018-08-18		2	2426	1038	108	Kettle Tortilla ChpsHny&Jlpno Chili 150g	3	

Checking for Null values

```
In [ ]: df.isnull().sum()
```

```
Out[ ]: 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
```

Assessing for Duplicated Entries

```
In [ ]: df[df.duplicated(subset = None, keep = False)]
```

Out[ ]:		date	store_nbr	lylty_card_nbr	txn_id	prod_nbr	prod_name	prod_qty	to
<b>124843</b>	2018-10-01		107	107024	108462	45	Smiths Thinly Cut Roast Chicken 175g	2	
<b>124845</b>	2018-10-01		107	107024	108462	45	Smiths Thinly Cut Roast Chicken 175g	2	

### Removing Duplicated Entries

```
In [ ]: df = pd.DataFrame.drop_duplicates(df)
```

### Checking the Number of Stores

```
In [ ]: df["store_nbr"].nunique()
```

```
Out[ ]: 272
```

### Checking the Number of Unique Values in the Product Name Column

```
In [ ]: df["prod_name"].nunique()
```

```
Out[ ]: 114
```

### Assessing the Unique Values in the Product Name Column

```
In [ ]: df["prod_name"].unique()
```

```

Out[ ]: 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 Crinkle 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 Crisps 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)

```

There is presence of inconsistent formatting for many of the product names including the names of product manufacturers.

Brand names that need adjusting: Natural Chip Compny, Natural ChipCo and NCC all refer to the Natural Chip Company, GrnWves refers to Grain Waves, Infzns refers to Infuzions, RRD refers to Red Rock Deli, Snbts to Sunbites. They will need to be adjusted before performing the analysis.

This will ensure that the data quality is high.

Additional columns will be created to categorize the company brand and product type.

## Adjusting the formatting for the Product names

```
In [ ]: replace_product_names = {
    "Natural Chip          Compny SeaSalt175g" : " Natural Chip Company| Potato Chips
    "CCs Nacho Cheese      175g" : " CCs| Corn Chips| Nacho Cheese| 175g",
    "Grain Waves           Sweet Chilli 210g" : " Grain Waves| Corn Chips| Sweet Chil
    "Smiths Crinkle Cut    Chips Chicken 170g" : " Smiths| Potato Chips| Crinkle Cut
    "Smiths Chip Thinly    S/Cream&Onion 175g" : " Smiths| Potato Chips| Thinly Cut S
    "Old El Paso Salsa     Dip Tomato Mild 300g" : " Old El Paso| Salsa Dip| Salsa Di
    "Grain Waves Sour      Cream&Chives 210G" : " Grain Waves| Corn Chips| Sour Cream
    "Kettle Sensations     Siracha Lime 150g" : " Kettle| Potato Chips| Sensations Si
    "Twisties Cheese       270g" : " Twisties| Corn Chips| Cheese| 270g",
    "WW Crinkle Cut        Chicken 175g" : " Woolworths| Potato Chips| Crinkle Cut Ch
    "Thins Chips Light&    Tangy 175g" : " Thins Chips| Potato Chips| Light & Tangy|
    "NCC Sour Cream &      Garden Chives 175g" : " Natural Chip Company| Potato Chips
    "Smiths Crinkle        Original 330g" : " Smiths| Potato Chips| Crinkle Original|
    "Kettle Sea Salt       And Vinegar 175g" : " Kettle| Potato Chips| Sea Salt And V
    "Smiths Chip Thinly    Cut Original 175g" : " Smiths| Potato Chips| Chip Thinly C
    "Red Rock Deli SR      Salsa & Mzzrlla 150g" : " Red Rock Deli| Potato Chips| Spe
    "Thins Chips           Originl salted 175g" : " Thins Chips| Potato Chips| Origina
    "Red Rock Deli Sp      Salt & Truffle 150G" : " Red Rock Deli| Potato Chips| Sea
    "Smiths Thinly         Swt Chli&S/Cream175G" : " Smiths| Potato Chips| Thinly Swe
    "Doritos Mexicana      170g" : " Doritos| Corn Chips| Mexicana| 170g",
    "Smiths Crinkle Cut    French OnionDip 150g" : " Smiths| Potato Chips| Crinkle Cu
    "Natural ChipCo        Hony Soy Chckn175g" : " Natural Chip Company| Potato Chips
    "Kettle Tortilla ChpsHny&Jlpno Chili 150g" : " Kettle| Potato Chips| Tortilla C
    "Smiths Crinkle Chips  Salt & Vinegar 330g" : " Smiths| Potato Chips| Crinkle Ch
    "Doritos Corn Chip     Mexican Jalapeno 150g" : " Doritos| Corn Chips| Corn Chip Me
    "CCs Original 175g" : " CCs| Corn Chips| Original| 175g",
    "Burger Rings 220g" : " Burger Rings|Corn Chips| Burger Rings| 220g",
    "Doritos Corn Chip     Southern Chicken 150g" : " Doritos| Corn Chips| Corn Chip So
    "Cheezels Cheese Box   125g" : " Cheezels|Corn Chips| Cheese Box| 125g",
    "Infzns Crn Crnchers   Tangy Gcamole 110g" : " Infuzions| Corn Chips| Corn Crunch
    "Kettle Original 175g" : " Kettle| Potato Chips| Original| 175g",
    "Red Rock Deli Thai    Chilli&Lime 150g" : " Red Rock Deli| Potato Chips| Thai Ch
    "Pringles Sthrn Fried  Chicken 134g" : " Pringles| Potato Chips| Southern Fried C
    "Pringles Sweet&Spcy   BBQ 134g" : " Pringles| Potato Chips| Sweet & Spicy BBQ| 1
    "Kettle Chilli 175g" : " Kettle| Potato Chips| Chilli| 175g",
    "Dorito Corn Chp       Supreme 380g" : " Doritos| Corn Chips| Corn Chips Supreme|
    "Twisties Chicken270g" : " Twisties| Corn Chips| Chicken| 270g",
    "Smiths Thinly Cut     Roast Chicken 175g" : " Smiths| Potato Chips| Thinly Cut R
    "Smiths Crinkle Cut    Tomato Salsa 150g" : " Smiths| Potato Chips| Crinkle Cut T
    "Kettle Mozzarella     Basil & Pesto 175g" : " Kettle| Potato Chips| Mozzarella B
    "Infuzions Thai Sweet  Chili PotatoMix 110g" : " Infuzions| Corn Chips| Thai Swee
    "Kettle Sensations     Camembert & Fig 150g" : " Kettle| Potato Chips| Sensations
    "Smith Crinkle Cut     Mac N Cheese 150g" : " Smiths| Potato Chips| Crinkle Cut M
    "Kettle Honey Soy      Chicken 175g" : " Kettle| Potato Chips| Honey Soy Chicken|
    "Thins Chips Seasoned  chicken 175g" : " Thins| Potato Chips| Chips Seasoned Chic
    "Smiths Crinkle Cut    Salt & Vinegar 170g" : " Smiths| Potato Chips| Crinkle Cut
    "Infuzions BBQ Rib     Prawn Crackers 110g" : " Infuzions| Corn Chips| BBQ Rib Pr
    "GrnWves Plus Btroot    & Chilli Jam 180g" : " Grain Waves| Corn Chips| Plus Beetr
    "Tyrrells Crisps       Lightly Salted 165g" : " Tyrrells| Potato Chips| Crisps Li
    "Kettle Sweet Chilli   And Sour Cream 175g" : " Kettle| Potato Chips| Sweet Chill
    "Doritos Salsa         Medium 300g" : " Doritos| Salsa Dip| Salsa Medium| 300g",
    "Kettle 135g Swt Pot   Sea Salt" : " Kettle| Potato Chips| Sweet Potato Sea Salt|
```

"Pringles SourCream Onion 134g" : " Pringles| Potato Chips| Sour Cream & Onion  
 "Doritos Corn Chips Original 170g" : " Doritos| Corn Chips| Original| 170g",  
 "Twisties Cheese Burger 250g" : " Smiths| Potato Chips| Twisties Cheese Bur  
 "Old El Paso Salsa Dip Chnky Tom Ht300g" : " Old El Paso| Salsa Dip| Hot Chun  
 "Cobs Popd Swt/Chlli &Sr/Cream Chips 110g" : " Cobs Popd| Potato Chips| Sweet C  
 "Woolworths Mild Salsa 300g" : " Woolworths| Salsa Dip| Mild Salsa| 300g",  
 "Natural Chip Co Tmato Hrb&Spce 175g" : " Natural Chip Company| Potato Chip  
 "Smiths Crinkle Cut Chips Original 170g" : " Smiths| Potato Chips| Crinkle Cut  
 "Cobs Popd Sea Salt Chips 110g" : " Cobs Popd| Potato Chips| Sea Salt Chips| 1  
 "Smiths Crinkle Cut Chips Chs&Onion170g" : " Smiths|Potato Chips| Crinkle Cut  
 "French Fries Potato Chips 175g" : " French Fries| Potato Chips| Original Potat  
 "Old El Paso Salsa Dip Tomato Med 300g" : " Old El Paso| Salsa Dip| Salsa Dip  
 "Doritos Corn Chips Cheese Supreme 170g" : " Doritos| Corn Chips| Corn Chips C  
 "Pringles Original Crisps 134g" : " Pringles| Potato Chips| Original Potato C  
 "RRD Chilli& Coconut 150g" : " Red Rock Deli| Potato Chips| Thai Red Ch  
 "WW Original Corn Chips 200g" : " Woolworths| Corn Chips| Original Corn Chip  
 "Thins Potato Chips Hot & Spicy 175g" : " Thins|Potato Chips| Potato Chips Hot  
 "Cobs Popd Sour Crm &Chives Chips 110g" : " Cobs Popd| Potato Chips| Sour Crea  
 "Smiths Crnkle Chip Orgnl Big Bag 380g" : " Smiths| Potato Chips| Crinkle Chip  
 "Doritos Corn Chips Nacho Cheese 170g" : " Doritos| Corn Chips| Nacho Cheese|  
 "Kettle Sensations BBQ&Maple 150g" : " Kettle| Potato Chips| Sensations BBQ &  
 "WW D/Style Chip Sea Salt 200g" : " Woolworths| Chicken Chips| D/Style Chip  
 "Pringles Chicken Salt Crips 134g" : " Pringles| Potato Chips| Chicken Salt  
 "WW Original Stacked Chips 160g" : " Woolworths| Potato Chips| Original Stacked  
 "Smiths Chip Thinly CutSalt/Vinegr175g" : " Smiths| Potato Chips| Thinly Cut S  
 "Cheezels Cheese 330g" : " Cheezels| Corn Chips| Cheese| 330g",  
 "Tostitos Lightly Salted 175g" : " Tostitos| Corn Chips| Lightly Salted| 175  
 "Thins Chips Salt & Vinegar 175g" : " Thins| Potato Chips| Chips Salt & Vinega  
 "Smiths Crinkle Cut Chips Barbecue 170g" : " Smiths|Potato Chips| Crinkle Cut  
 "Cheetos Puffs 165g" : " Cheetos| Corn Chips| Puffs| 165g",  
 "RRD Sweet Chilli & Sour Cream 165g" : " Red Rock Deli| Potato Chips| Sweet Ch  
 "WW Crinkle Cut Original 175g" : " Woolworths| Potato Chips| Crinkle Cut O  
 "Tostitos Splash Of Lime 175g" : " Tostitos| Corn Chips| Splash Of Lime| 175g"  
 "Woolworths Medium Salsa 300g" : " Woolworths|Salsa Dip| Medium Salsa| 300g",  
 "Kettle Tortilla ChpsBtroot&Ricotta 150g" : " Kettle| Potato Chips| Tortilla Ch  
 "CCs Tasty Cheese 175g" : " CCs| Corn Chips| Tasty Cheese| 175g",  
 "Woolworths Cheese Rings 190g" : " Woolworths| Corn Chips| Cheese Rings| 190g  
 "Tostitos Smoked Chipotle 175g" : " Tostitos| Corn Chips| Smoked Chipotle|  
 "Pringles Barbeque 134g" : " Pringles| Potato Chips| Barbeque| 134g",  
 "WW Supreme Cheese Corn Chips 200g" : " Woolworths| Corn Chips| Supreme Chees  
 "Pringles Mystery Flavour 134g" : " Pringles| Potato Chips| Mystery Flavour|  
 "Tyrrells Crisps Ched & Chives 165g" : " Tyrrells| Potato Chips| Crisps Mat  
 "Snbts Whlgrn Crisps Cheddr&Mstrd 90g" : " Sunbites| Corn Chips| Wholegrain Cri  
 "Cheetos Chs & Bacon Balls 190g" : " Cheetos| Corn Chips| Cheese & Bacon Balls|  
 "Pringles Slt Vingar 134g" : " Pringles| Potato Chips| Salt & Vingar| 134g",  
 "Infuzions SourCream&Herbs Veg Strws 110g" : " Infuzions| Potato Chips| Veggie  
 "Kettle Tortilla ChpsFeta&Garlic 150g" : " Kettle| Corn Chips| Tortilla Chips F  
 "Infuzions Mango Chutny Papadums 70g" : " Infuzions| Potato Chips| Papadums  
 "RRD Steak & Chimuchurri 150g" : " Red Rock Deli| Potato Chips| Steak &  
 "RRD Honey Soy Chicken 165g" : " Red Rock Deli| Potato Chips| Honey Soy C  
 "Sunbites Whlegrn Crisps Frch/Onin 90g" : " Sunbites| Corn Chips| Wholegrain  
 "RRD Salt & Vinegar 165g" : " Red Rock Deli| Potato Chips| Salt & Vinegar| 165  
 "Doritos Cheese Supreme 330g" : " Doritos| Corn Chips| Cheese Supreme| 330  
 "Smiths Crinkle Cut Snag&Sauce 150g" : " Smiths| Potato Chips| Crinkle Cut Sna  
 "WW Sour Cream &OnionStacked Chips 160g" : " Woolworths| Potato Chips| Sour Cre  
 "RRD Lime & Pepper 165g" : " Red Rock Deli| Potato Chips| Lime & Pepper| 165g



```

    "Natural ChipCo Sea Salt & Vinegr 175g" : " Natural Chip Company| Potato Chips
    "Red Rock Deli Chikn&Garlic Aioli 150g" : " Red Rock Deli| Potato Chips| Chicke
    "RRD SR Slow Rst      Pork Belly 150g" : " Red Rock Deli| Potato Chips| Special
    "RRD Pc Sea Salt      165g" : " Red Rock Deli| Potato Chips| Potato Chips Sea Sa
    "Smith Crinkle Cut    Bolognese 150g" : " Smiths|Potato Chips| Crinkle Cut Bolog
    "Doritos Salsa Mild   300g" : " Doritos| Salsa Dip| Salsa Mild| 300g"
}

```

```

In [ ]: def replaced_value(old_value, new_value, dataframe = df, column = "prod_name"):
        """ This function replaces the old value in a dataset with a new value.

        Paramaters:
            old_value: The specified value to replace.
            new_value: The new value to replace with.
            dataframe: The specified dataframe to modify.
            column: The specified column to modify.
        """
        dataframe[column] = dataframe[column].str.replace(old_value, new_value)

```

```

In [ ]: for old, new in replace_product_names.items():
        replaced_value(old, new)

```

Checking that the product name column has the correct number of products.

```

In [ ]: assert len(df["prod_name"].unique()) == 114

```

Checking the product name category is properly formatted.

```

In [ ]: sorted(df["prod_name"].unique())

```

```
Out[ ]: [' Burger Rings|Corn Chips| Burger Rings| 220g',
' CCs| Corn Chips| Nacho Cheese| 175g',
' CCs| Corn Chips| Original| 175g',
' CCs| Corn Chips| Tasty Cheese| 175g',
' Cheetos| Corn Chips| Cheese & Bacon Balls| 190g',
' Cheetos| Corn Chips| Puffs| 165g',
' Cheezels| Corn Chips| Cheese| 330g',
' Cheezels|Corn Chips| Cheese Box| 125g',
' Cobs Popd| Potato Chips| Sea Salt Chips| 110g',
' Cobs Popd| Potato Chips| Sour Cream & Chives Chips| 110g',
' Cobs Popd| Potato Chips| Sweet Chilli & Sour Cream Chips| 110g',
' Doritos| Corn Chips| Cheese Supreme| 330g',
' Doritos| Corn Chips| Corn Chip Mexican Jalapeno| 150g',
' Doritos| Corn Chips| Corn Chip Southern Chicken| 150g',
' Doritos| Corn Chips| Corn Chips Cheese Supreme| 170g',
' Doritos| Corn Chips| Corn Chips Supreme| 380g',
' Doritos| Corn Chips| Mexicana| 170g',
' Doritos| Corn Chips| Nacho Cheese| 170g',
' Doritos| Corn Chips| Original| 170g',
' Doritos| Salsa Dip| Salsa Medium| 300g',
' Doritos| Salsa Dip| Salsa Mild| 300g',
' French Fries| Potato Chips| Original Potato Chips| 175g',
' Grain Waves| Corn Chips| Plus Beetroot & Chilli Jam| 180g',
' Grain Waves| Corn Chips| Sour Cream & Chives| 210g',
' Grain Waves| Corn Chips| Sweet Chilli| 210g',
' Infuzions| Corn Chips| BBQ Rib Prawn Crackers| 110g',
' Infuzions| Corn Chips| Corn Crunchers Tangy Guacamole| 110g',
' Infuzions| Corn Chips| Thai Sweet Chili Potato Mix| 110g',
' Infuzions| Potato Chips| Papadums Mango Chutney| 70g',
' Infuzions| Potato Chips| Veggie Straws Sour Cream & Herbs| 110g',
' Kettle| Corn Chips| Tortilla Chips Feta & Garlic| 150g',
' Kettle| Potato Chips| Chilli| 175g',
' Kettle| Potato Chips| Honey Soy Chicken| 175g',
' Kettle| Potato Chips| Mozzarella Basil & Pesto| 175g',
' Kettle| Potato Chips| Original| 175g',
' Kettle| Potato Chips| Sea Salt And Vinegar| 175g',
' Kettle| Potato Chips| Sensations BBQ & Maple| 150g',
' Kettle| Potato Chips| Sensations Camembert & Fig| 150g',
' Kettle| Potato Chips| Sensations Siracha Lime| 150g',
' Kettle| Potato Chips| Sweet Chilli & Sour Cream| 175g',
' Kettle| Potato Chips| Sweet Potato Sea Salt| 135g',
' Kettle| Potato Chips| Tortilla Chips Beetroot & Ricotta| 150g',
' Kettle| Potato Chips| Tortilla Chips Honey & Jalapeno Chili| 150g',
' Natural Chip Company| Potato Chips| Honey Soy Chicken| 175g',
' Natural Chip Company| Potato Chips| Sea Salt & Vinegar| 175g',
' Natural Chip Company| Potato Chips| Sea Salt| 175g',
' Natural Chip Company| Potato Chips| Sour Cream & Garden Chives| 175g',
' Natural Chip Company| Potato Chips| Tomato Herbs & Spice| 175g',
' Old El Paso| Salsa Dip| Hot Chunky Tomato Salsa Dip| 300g',
' Old El Paso| Salsa Dip| Salsa Dip Tomato Medium| 300g',
' Old El Paso| Salsa Dip| Salsa Dip Tomato Mild| 300g',
' Pringles| Potato Chips| Barbeque| 134g',
' Pringles| Potato Chips| Chicken Salt Crisps| 134g',
' Pringles| Potato Chips| Mystery Flavour| 134g',
' Pringles| Potato Chips| Original Potato Crisps| 134g',
' Pringles| Potato Chips| Salt & Vingar| 134g',
```

' Pringles| Potato Chips| Sour Cream & Onion| 134g',  
' Pringles| Potato Chips| Southern Fried Chicken| 134g',  
' Pringles| Potato Chips| Sweet & Spicy BBQ| 134g',  
' Red Rock Deli| Potato Chips| Chicken & Garlic Aioli| 150g',  
' Red Rock Deli| Potato Chips| Honey Soy Chicken| 165g',  
' Red Rock Deli| Potato Chips| Lime & Pepper| 165g',  
' Red Rock Deli| Potato Chips| Potato Chips Sea Salt| 165g',  
' Red Rock Deli| Potato Chips| Salt & Vinegar| 165g',  
' Red Rock Deli| Potato Chips| Sea Salt & Black Truffle| 150g',  
' Red Rock Deli| Potato Chips| Special Reserve Salsa & Mozzarella| 150g',  
' Red Rock Deli| Potato Chips| Special Reserve Slow Roasted Pork Belly| 150g',  
' Red Rock Deli| Potato Chips| Steak & Chimichurri| 150g',  
' Red Rock Deli| Potato Chips| Sweet Chilli & Sour Cream| 165g',  
' Red Rock Deli| Potato Chips| Thai Chilli & Lime| 150g',  
' Red Rock Deli| Potato Chips| Thai Red Chilli & Creamy Coconut| 150g',  
' Smiths| Potato Chips| Chip Thinly Cut Original| 175g',  
' Smiths| Potato Chips| Crinkle Chip Original Big Bag| 380g',  
' Smiths| Potato Chips| Crinkle Chips Salt & Vinegar| 330g',  
' Smiths| Potato Chips| Crinkle Cut Chips Chicken| 170g',  
' Smiths| Potato Chips| Crinkle Cut Chips Original| 170g',  
' Smiths| Potato Chips| Crinkle Cut French Onion Dip| 150g',  
' Smiths| Potato Chips| Crinkle Cut Mac N Cheese| 150g',  
' Smiths| Potato Chips| Crinkle Cut Salt & Vinegar| 170g',  
' Smiths| Potato Chips| Crinkle Cut Snag & Sauce| 150g',  
' Smiths| Potato Chips| Crinkle Cut Tomato Salsa| 150g',  
' Smiths| Potato Chips| Crinkle Original| 330g',  
' Smiths| Potato Chips| Thinly Cut Roast Chicken| 175g',  
' Smiths| Potato Chips| Thinly Cut Salt & Vinegar| 175g',  
' Smiths| Potato Chips| Thinly Cut Sour Cream & Onion| 175g',  
' Smiths| Potato Chips| Thinly Sweet Chilli & Sour Cream| 175g',  
' Smiths| Potato Chips| Twisties Cheese Burger| 250g',  
' Smiths| Potato Chips| Crinkle Cut Bolognese| 150g',  
' Smiths| Potato Chips| Crinkle Cut Chips Barbecue| 170g',  
' Smiths| Potato Chips| Crinkle Cut Chips Cheess & Onion| 170g',  
' Sunbites| Corn Chips| Wholegrain Crisps Cheddar & Mustard| 90g',  
' Sunbites| Corn Chips| Wholegrain Crisps French Onion| 90g',  
' Thins Chips| Potato Chips| Light & Tangy| 175g',  
' Thins Chips| Potato Chips| Original Salted| 175g',  
' Thins| Potato Chips| Chips Salt & Vinegar| 175g',  
' Thins| Potato Chips| Chips Seasoned Chicken| 175g',  
' Thins| Potato Chips| Potato Chips Hot & Spicy| 175g',  
' Tostitos| Corn Chips| Lightly Salted| 175g',  
' Tostitos| Corn Chips| Smoked Chipotle| 175g',  
' Tostitos| Corn Chips| Splash Of Lime| 175g',  
' Twisties| Corn Chips| Cheese| 270g',  
' Twisties| Corn Chips| Chicken| 270g',  
' Tyrrells| Potato Chips| Crisps Lightly Salted| 165g',  
' Tyrrells| Potato Chips| Crisps Mature Cheddar & Chives| 165g',  
' Woolworths| Chicken Chips| D/Style Chips Sea Salt| 200g',  
' Woolworths| Corn Chips| Cheese Rings| 190g',  
' Woolworths| Corn Chips| Original Corn Chips| 200g',  
' Woolworths| Corn Chips| Supreme Cheese Corn Chips| 200g',  
' Woolworths| Potato Chips| Crinkle Cut Chicken| 175g',  
' Woolworths| Potato Chips| Crinkle Cut Original| 175g',  
' Woolworths| Potato Chips| Original Stacked Chips| 160g',  
' Woolworths| Potato Chips| Sour Cream & Onion Stacked Chips| 160g',

```
' Woolworths| Salsa Dip| Mild Salsa| 300g',  
' Woolworths|Salsa Dip| Medium Salsa| 300g']
```

## Creating New Product Category Columns

```
In [ ]: df[["brand_name", "product_type", "product_name", "pack_size"]] = df["prod_name"].s
```

Assessing the formatting of the brand name column.

```
In [ ]: df["brand_name"].unique()
```

```
Out[ ]: array([' Natural Chip Company', ' CCs', ' Smiths', ' Kettle',  
              ' Old El Paso', ' Grain Waves', ' Doritos', ' Twisties',  
              ' Woolworths', ' Thins Chips', ' Burger Rings', ' Cheezels',  
              ' Infuzions', ' Red Rock Deli', ' Pringles', ' Thins', ' Tyrrells',  
              ' Cobs Popd', ' French Fries', ' Tostitos', ' Cheetos',  
              ' Sunbites'], dtype=object)
```

Removing extra spaces in product, pack\_size and product\_type columns

```
In [ ]: stripped_columns = ["brand_name", "product_name", "pack_size", "product_type"]  
  
for i in stripped_columns:  
    df[i] = df[i].str.strip()
```

```
In [ ]: df["brand_name"].unique()
```

```
Out[ ]: array(['Natural Chip Company', 'CCs', 'Smiths', 'Kettle', 'Old El Paso',  
              'Grain Waves', 'Doritos', 'Twisties', 'Woolworths', 'Thins Chips',  
              'Burger Rings', 'Cheezels', 'Infuzions', 'Red Rock Deli',  
              'Pringles', 'Thins', 'Tyrrells', 'Cobs Popd', 'French Fries',  
              'Tostitos', 'Cheetos', 'Sunbites'], dtype=object)
```

```
In [ ]: assert len(df["brand_name"].unique()) == 22
```

Inspecting the pack size column

```
In [ ]: df["pack_size"].unique()
```

```
Out[ ]: array(['175g', '170g', '150g', '300g', '330g', '210g', '270g', '220g',  
              '125g', '110g', '134g', '380g', '180g', '165g', '135g', '250g',  
              '200g', '160g', '190g', '90g', '70g'], dtype=object)
```

Removing the grams from the pack size column and recasting as type int

```
In [ ]: df["pack_size"] = (  
    df["pack_size"]  
    .str[:-1]  
    .astype(int)  
)
```

## Adding the Product Price Column

```
In [ ]: df["product_price"] = df["tot_sales"]/df["prod_qty"]
```

Drop Rows Containing Salsa Dip

```
In [ ]: df = df[df["product_type"].str.contains("Salsa Dip") == False]
```

---

## Import: Purchase Behaviour Dataset



```
In [ ]: df2 = wrangle("QVI_purchase_behaviour.csv")
```

```
In [ ]: df2.head()
```

```
Out [ ]:
```

	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 [ ]: len(df2)
```

```
Out [ ]: 72637
```

```
In [ ]: df2["premium_customer"].unique()
```

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

```
In [ ]: df2["lifestage"].unique()
```

```
Out [ ]: array(['YOUNG SINGLES/COUPLES', 'YOUNG FAMILIES', 'OLDER SINGLES/COUPLES',  
              'MIDAGE SINGLES/COUPLES', 'NEW FAMILIES', 'OLDER FAMILIES',  
              'RETIRES'], dtype=object)
```

Checking for duplicates in the loyalty card column

```
In [ ]: df2[df2.duplicated(subset = None, keep = False)]
```

```
Out [ ]:
```


	lylty_card_nbr	lifestage	premium_customer
--	----------------	-----------	------------------

```
In [ ]: df2["lylty_card_nbr"].nunique()
```

```
Out[ ]: 72637
```

```
In [ ]: df2.isnull().sum()
```

```
Out[ ]: lylty_card_nbr      0  
        lifestage          0  
        premium_customer   0  
        dtype: int64
```



## Data Integration: Merging the Transaction Dataset with the Customer Segments Dataset

```
In [ ]: df = pd.merge(df, df2, how = "left", on = "lylty_card_nbr")
```

Displaying 10 random rows

```
In [ ]: df.sample(10)
```

Out[ ]:

	date	store_nbr	lylty_card_nbr	txn_id	prod_nbr	prod_name	prod_qty	tc
<b>210873</b>	2019-05-04	154	154244	154472	60	Kettle  Corn Chips  Tortilla Chips Feta & Gar...	2	
<b>127513</b>	2019-05-20	47	47242	42667	15	Twisties  Corn Chips  Cheese  270g	1	
<b>40917</b>	2019-03-03	119	119260	123141	112	Tyrrells  Potato Chips  Crisps Mature Cheddar...	2	
<b>29537</b>	2018-07-08	137	137067	139274	49	Infuzions  Potato Chips  Veggie Straws Sour C...	2	
<b>116639</b>	2019-05-31	90	90196	88908	107	Smiths  Potato Chips  Crinkle Cut French Onio...	1	
<b>90338</b>	2018-07-08	218	218116	217944	44	Thins Chips  Potato Chips  Light & Tangy  175g	2	
<b>97244</b>	2018-12-17	55	55101	49057	87	Infuzions  Corn Chips  BBQ Rib Prawn Crackers...	2	
<b>221124</b>	2018-10-03	175	175311	176656	30	Doritos  Corn Chips  Corn Chips Cheese Suprem...	2	
<b>40088</b>	2018-09-16	110	110065	112074	9	Kettle  Potato Chips  Tortilla Chips	2	

	date	store_nbr	lylty_card_nbr	txn_id	prod_nbr	prod_name	prod_qty	tc
						Beetroot...		
						Woolworths		
						Potato		
50787	2019-04-26	249	249363	251280	96	Chips	2	
						Original		
						Stacked Ch...		

Removing the Old Product Name Column

```
In [ ]: df.drop(columns = ["prod_name"], inplace = True)
```

## Explore



## Outlier Detection

```
In [ ]: df.describe()
```

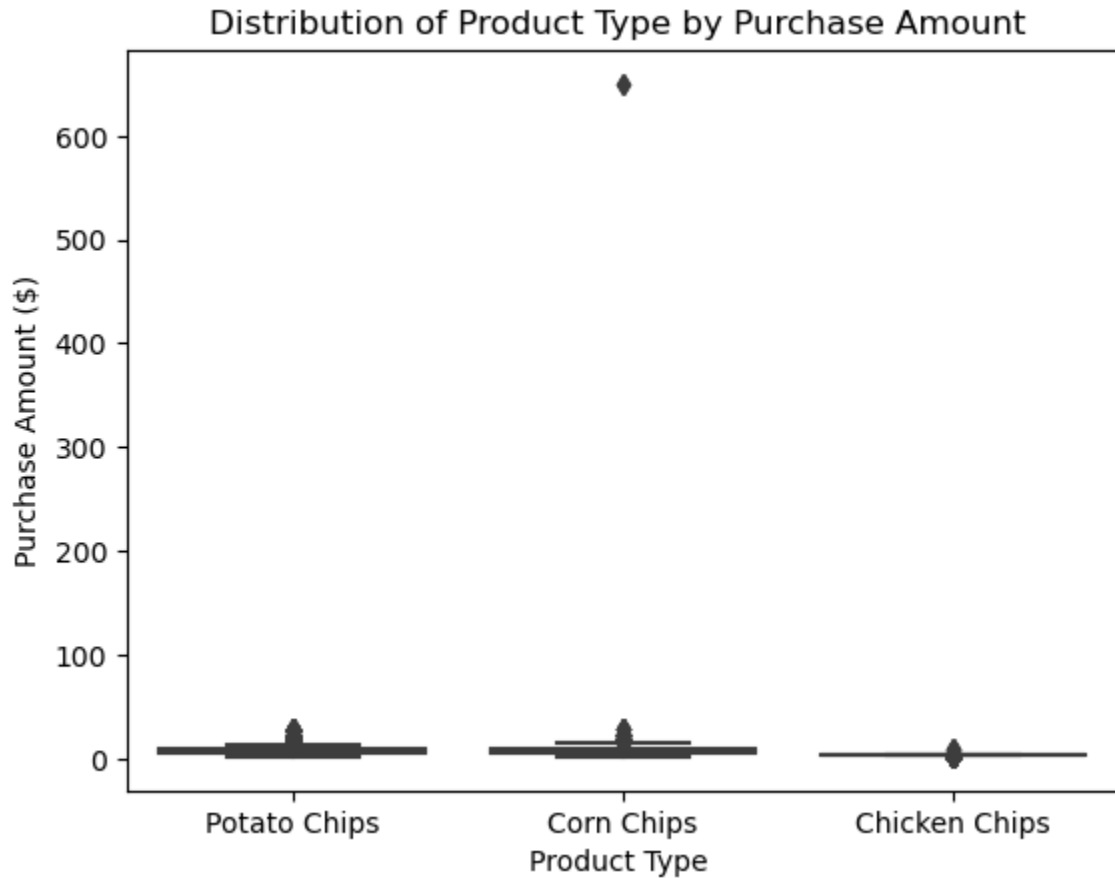
	date	store_nbr	lylty_card_nbr	txn_id	prod_nbr	
count	249669	249669.000000	2.496690e+05	2.496690e+05	249669.000000	249669
mean	2018-12-30 02:26:16.381529600	135.044391	1.355204e+05	1.351235e+05	56.294334	
min	2018-07-01 00:00:00	1.000000	1.000000e+03	1.000000e+00	1.000000	
25%	2018-09-30 00:00:00	70.000000	7.001600e+04	6.757400e+04	27.000000	
50%	2018-12-30 00:00:00	130.000000	1.303600e+05	1.351480e+05	53.000000	
75%	2019-03-31 00:00:00	203.000000	2.030800e+05	2.026340e+05	86.000000	
max	2019-06-30 00:00:00	272.000000	2.373711e+06	2.415841e+06	114.000000	2
std	NaN	76.773724	8.065760e+04	7.813169e+04	33.528818	

Box Plot Visualization of Total Sales Distribution by Product Type before Removal of Outliers

```
In [ ]: sns.boxplot(
    data = df,
    x = "product_type",
    y = "tot_sales"
```



```
)
plt.xlabel("Product Type ")
plt.ylabel("Purchase Amount ($)")
plt.title("Distribution of Product Type by Purchase Amount ");
```



## Outlier Removal

```
In [ ]: df[df["prod_qty"] == 200]
```

```
Out [ ]:
```

	date	store_nbr	lylty_card_nbr	txn_id	prod_nbr	prod_qty	tot_sales	brand
<b>65786</b>	2018-08-19	226	226000	226201	4	200	650.0	I
<b>65787</b>	2019-05-20	226	226000	226210	4	200	650.0	I

```
In [ ]: quantity_mask = df["prod_qty"] < 200
df = df[quantity_mask]
```

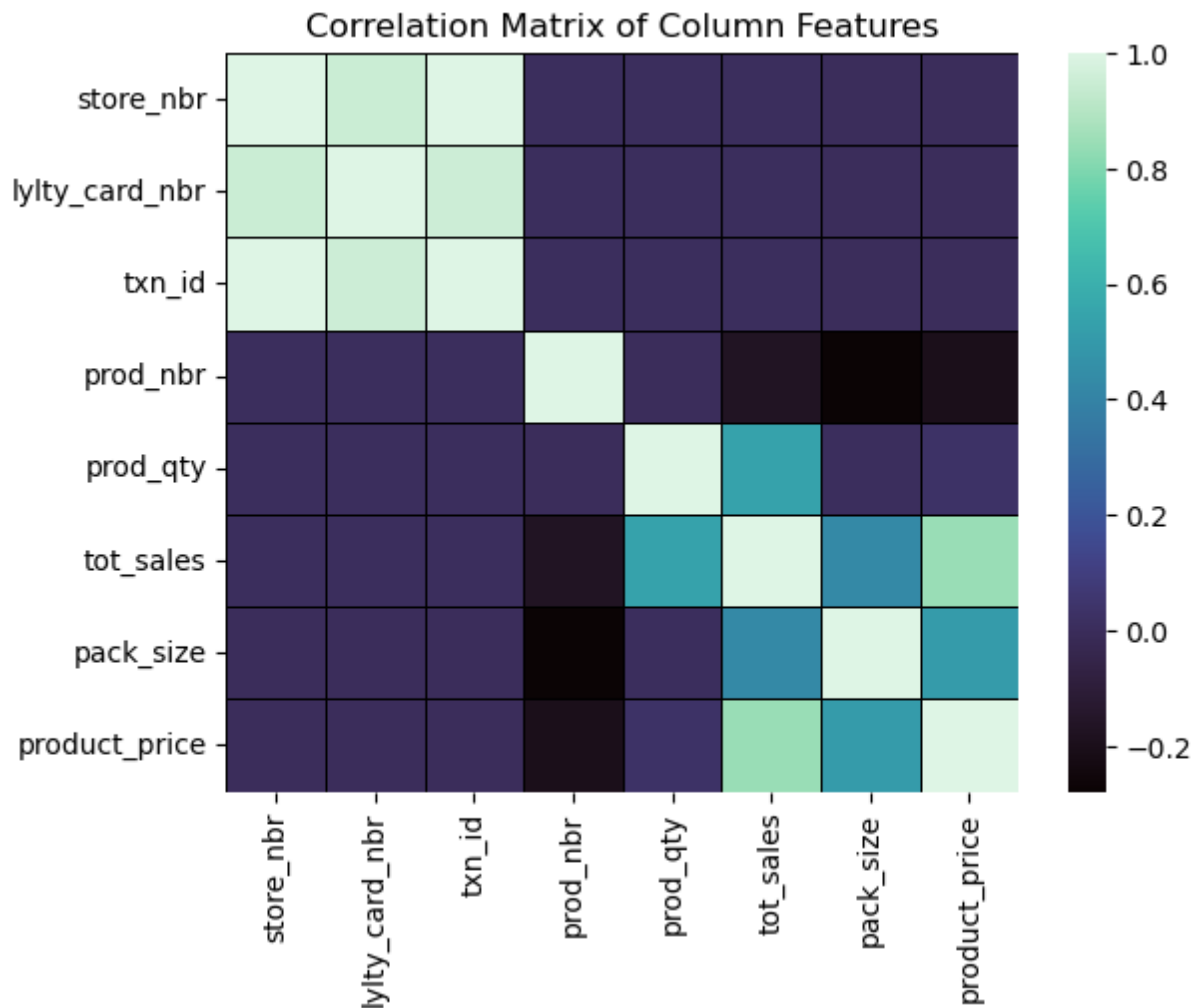
The Box plot showed 2 extreme outliers with purchase quantities of 200. These extreme outliers were removed in order to normalize the mean.

Then removed all outlier data points beyond 3 standard deviations from the mean.

---

## Heatmap Visualization for Correlation Matrix of Features

```
In [ ]: df_corr = df.select_dtypes("number").corr()
sns.heatmap(
    data = df_corr,
    linecolor = "black",
    linewidths = 0.5,
    cmap = "mako"
)
plt.title("Correlation Matrix of Column Features");
```



The heatmap shows us that the column features are not highly correlated.

---

### Key Objectives Overview:

- Identify consumer segments that drives sales and factors that influence the sales of higher priced products.
- Revenue metrics for brands and products.

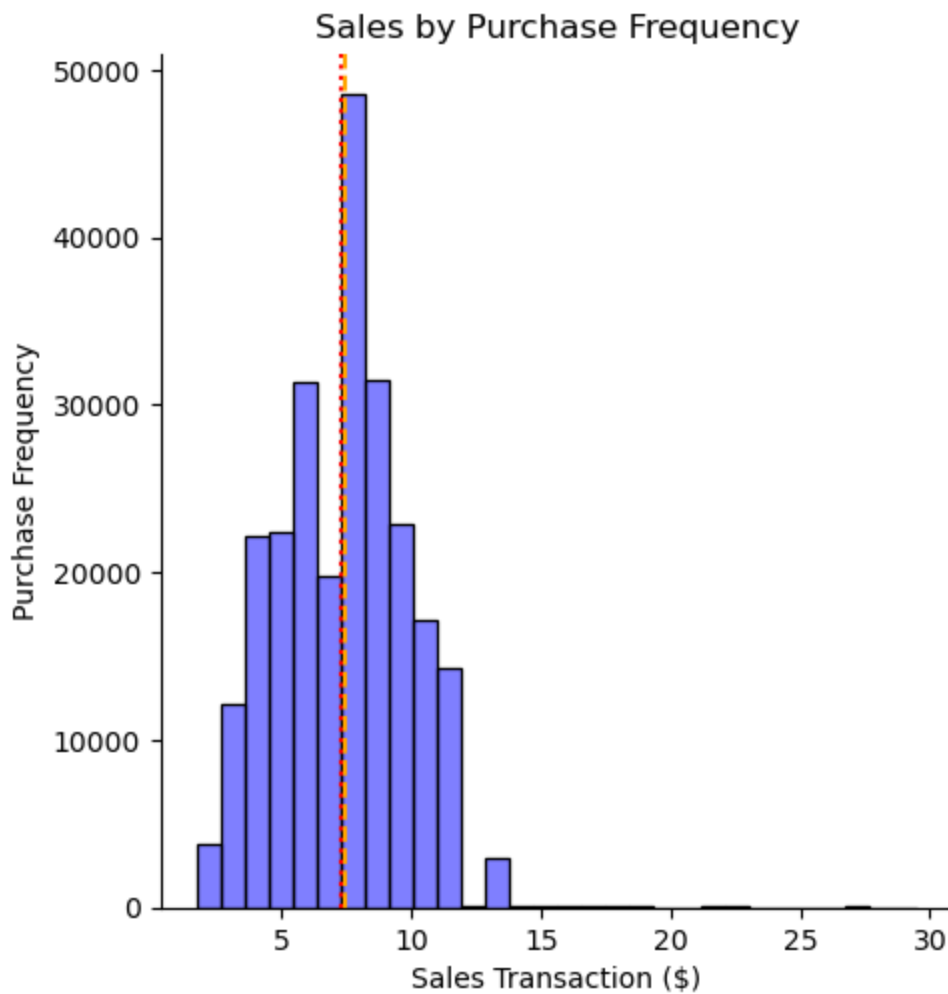
Evaluate time periods to identify sales trends.

## Sales Metrics

### Sales by Purchase Frequency

```
In [ ]: sns.displot(
        data = df["tot_sales"],
        color = "blue",
        bins = 30,
        alpha = 0.5
    )
    plt.axvline(
        np.mean(df["tot_sales"]),
        color = "red",
        linestyle = ":"
    )
    plt.axvline(
        np.median(df["tot_sales"]),
        color = "orange",
        linestyle = "--"
    )
    plt.xlabel("Sales Transaction ($)")
    plt.ylabel("Purchase Frequency")
    plt.title("Sales by Purchase Frequency");
```

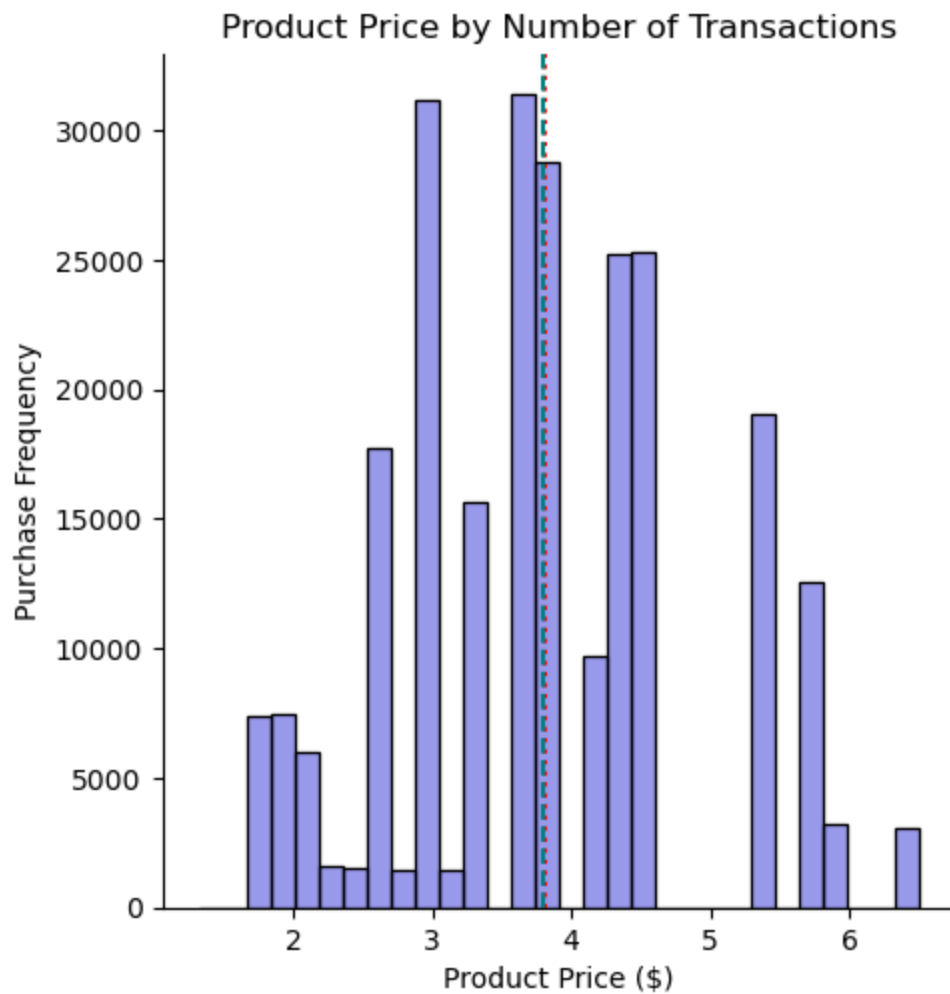
```
c:\ProgramData\Anaconda3\envs\skorch-ml\Lib\site-packages\seaborn\axisgrid.py:118: UserWarning: The figure layout has changed to tight
    self._figure.tight_layout(*args, **kwargs)
```



Product Price by Number of Transactions

```
In [ ]: sns.displot(
    data = df["product_price"],
    color = "mediumblue",
    bins = 30,
    alpha = 0.4
)
plt.axvline(
    np.mean(df["product_price"]),
    color = "red",
    linestyle = ":"
)
plt.axvline(
    np.median(df["product_price"]),
    color = "teal",
    linestyle = "--"
)
plt.xlabel("Product Price ($)")
plt.ylabel("Purchase Frequency")
plt.title("Product Price by Number of Transactions");
```

c:\ProgramData\Anaconda3\envs\skorch-ml\Lib\site-packages\seaborn\axisgrid.py:118: UserWarning: The figure layout has changed to tight  
self.\_figure.tight\_layout(\*args, \*\*kwargs)

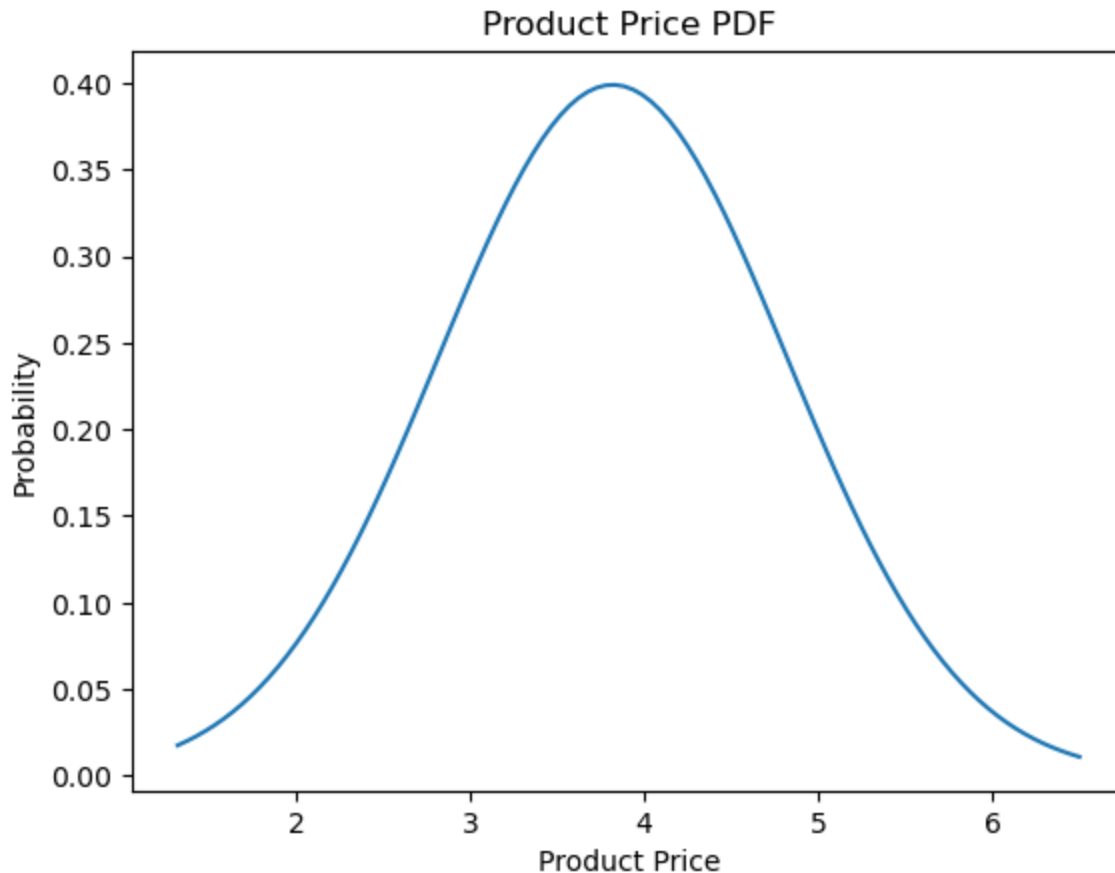


Probability Density Function of Product Price

```
In [ ]: xs = np.linspace(df["product_price"].min(), df["product_price"].max(), 100)
        ys1 = st.stats.norm.pdf(xs, loc = df["product_price"].mean(), scale = 1)

        plt.plot(xs, ys1, label = "normal approximation")
        plt.ylabel('Probability')
        plt.xlabel("Product Price")
        plt.title("Product Price PDF")
        ;
```

Out[ ]: ''



```
In [ ]: total_sales_revenue = df["tot_sales"].sum().round(2)
print(f"Total sales revenue amounted up to: ${total_sales_revenue}.")
```

Total sales revenue amounted up to: \$1819778.4.

```
In [ ]: total_quantity_sold = df["prod_qty"].sum()
print(f"Total sales quantity sold: {total_quantity_sold} bags.")
```

Total sales quantity sold: 475909 bags.

```
In [ ]: print(f"The average amount of revenue received per bag: ${(total_sales_revenue / to
```

The average amount of revenue received per bag: \$3.82

```
In [ ]: product_price = (df["product_price"] >= 3) & (df["product_price"] <= 4.5)
df_product_price = df[product_price]["tot_sales"].sum().round(2)
print(f"{((df_product_price / total_sales_revenue).round(4))*100}% of revenue was g
```

52.93% of revenue was generated by products between \$3 and \$4.50.

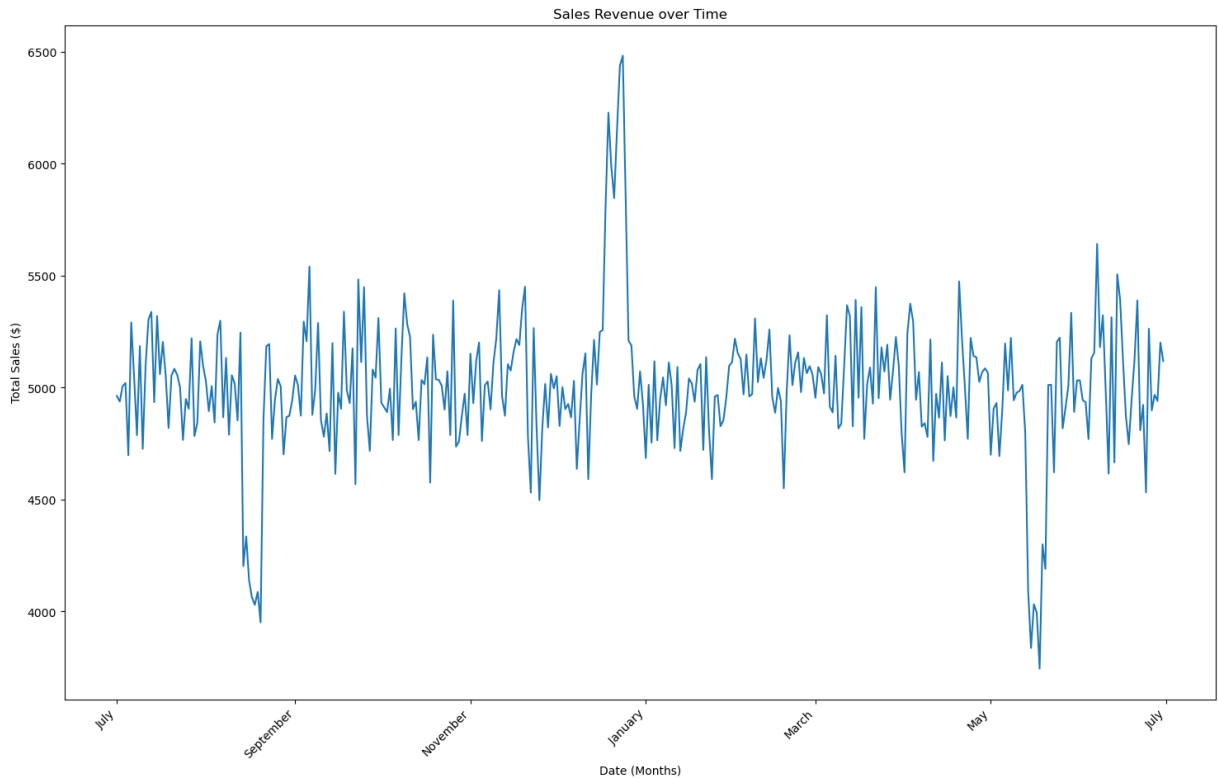
```
In [ ]: high_product_price = df["product_price"] > 4.5
df_high_product_price = df[high_product_price]["tot_sales"].sum().round(2)
print(f"{((df_high_product_price / total_sales_revenue).round(5))*100}% of revenue
```

34.68% of revenue was generated by high priced brands and bags.

Sales Revenue Over Time

```
In [ ]: plt.figure(figsize = (18,12))
df.groupby("date")["tot_sales"].sum().plot(kind = "line")
```

```
loc, labels = plt.xticks()
plot_labels = ["July", "September", "November", "January", "March", "May", "July"]
plt.xticks(ticks = loc, labels = plot_labels, rotation = 45)
plt.xlabel("Date (Months)")
plt.ylabel("Total Sales ($)")
plt.title("Sales Revenue over Time");
```



## Sales Metrics Insights:

### Key Metrics:

Total Revenue was 1,808,654.55.

Total Sales Quantity was 473,350 bags.

Average Revenue per bag: 3.82.

### Sales Trends:

Sales significantly decreased in August just before September and again in early May.

Sales significantly increase over the Christmas Holiday season in December.

52.93% of purchase sales occurred between the product prices of 3 and 4.5.

34.6% of purchase sales occurred for products priced above 4.5.

### Conclusion:

Sales promotions need to take place in August, December and May.

A marketing strategy that targets consumers that purchase higher priced products would be effective.

Marketing for the larger bag sizes would also likely generate more revenue.

## Customer Segmentation Analysis

### Sales by Premium Customer

```
In [ ]: pd.pivot_table(  
    data = df,  
    index = ["premium_customer"],  
    values = ["prod_qty", "tot_sales"],  
    aggfunc = ["sum", "mean"]  
).style.background_gradient(cmap = "mako")
```

Out[ ]:

	sum		mean	
	prod_qty	tot_sales	prod_qty	tot_sales
premium_customer				
Budget	167747	636632.850000	1.910297	7.249953
Mainstream	182813	706252.500000	1.901885	7.347459
Premium	125349	476893.050000	1.906942	7.255002

### Sales By Lifestage and Purchasing Habits

```
In [ ]: pd.crosstab(  
    index = df["lifestage"],  
    columns = df["premium_customer"],  
    values = df["tot_sales"],  
    aggfunc = "sum"  
).style.background_gradient(cmap = "mako")
```

Out [ ]:

	premium_customer		
	Budget	Mainstream	Premium
lifestage			
MIDAGE SINGLES/COUPLES	33705.400000	85262.750000	55042.350000
NEW FAMILIES	20716.050000	16078.000000	10861.700000
OLDER FAMILIES	158379.950000	97280.850000	75983.000000
OLDER SINGLES/COUPLES	128683.800000	125737.100000	124457.050000
RETIREEES	106606.200000	146328.750000	91951.950000
YOUNG FAMILIES	130919.050000	87227.850000	79249.100000
YOUNG SINGLES/COUPLES	57622.400000	148337.200000	39347.900000

### Sales By Lifestage and Purchasing Habits in Percentage



```
In [ ]: pd.crosstab(
    index = df["lifestage"],
    columns = df["premium_customer"],
    values = df["tot_sales"],
    aggfunc = "sum",
    normalize = True
).style.background_gradient(cmap = "mako")
```

```
Out[ ]:
```

	premium_customer	Budget	Mainstream	Premium
lifestage				
MIDAGE SINGLES/COUPLES		0.018522	0.046853	0.030247
NEW FAMILIES		0.011384	0.008835	0.005969
OLDER FAMILIES		0.087033	0.053458	0.041754
OLDER SINGLES/COUPLES		0.070714	0.069095	0.068391
RETIREEES		0.058582	0.080410	0.050529
YOUNG FAMILIES		0.071942	0.047933	0.043549
YOUNG SINGLES/COUPLES		0.031665	0.081514	0.021622

Sales by Customer Lifestage Comparing Average Quantity and Total Sales

```
In [ ]: pd.pivot_table(
    data = df,
    index = ["lifestage"],
    values = ["prod_qty", "tot_sales"],
    aggfunc = ["sum", "mean"],
    sort = True
).style.background_gradient(cmap = "mako")
```

```
Out[ ]:
```

	sum		mean	
	prod_qty	tot_sales	prod_qty	tot_sales
lifestage				
MIDAGE SINGLES/COUPLES	45058	174010.500000	1.901181	7.342215
NEW FAMILIES	12186	47655.750000	1.857056	7.262382
OLDER FAMILIES	89075	331643.800000	1.946356	7.246669
OLDER SINGLES/COUPLES	98264	378877.950000	1.913165	7.376620
RETIREEES	88819	344886.900000	1.892423	7.348338
YOUNG FAMILIES	79622	297396.000000	1.940344	7.247374
YOUNG SINGLES/COUPLES	62885	245307.500000	1.832901	7.149946

Sales by Lifestage & Premium Customer

```
In [ ]: pd.pivot_table(  
    data = df,  
    index = ["premium_customer", "lifestage"],  
    values = ["tot_sales", "prod_qty"],  
    aggfunc = ["sum", "mean"]  
).style.background_gradient(cmap = "mako")
```

Out[ ]:

		sum		mean	
		prod_qty	tot_sales	prod_qty	tot_sales
premium_customer	lifestage				
Budget	MIDAGE SINGLES/COUPLES	9019	33705.400000	1.892363	7.072052
	NEW FAMILIES	5282	20716.050000	1.855286	7.276449
	OLDER FAMILIES	42425	158379.950000	1.945387	7.262470
	OLDER SINGLES/COUPLES	33204	128683.800000	1.914327	7.419072
	RETIREEES	27192	106606.200000	1.893066	7.421763
	YOUNG FAMILIES	34936	130919.050000	1.941428	7.275301
	YOUNG SINGLES/COUPLES	15689	57622.400000	1.806032	6.633176
Mainstream	MIDAGE SINGLES/COUPLES	21413	85262.750000	1.911875	7.612746
	NEW FAMILIES	4097	16078.000000	1.856366	7.285002
	OLDER FAMILIES	26132	97280.850000	1.948550	7.253810
	OLDER SINGLES/COUPLES	33018	125737.100000	1.911206	7.278137
	RETIREEES	38114	146328.750000	1.886645	7.243280
	YOUNG FAMILIES	23530	87227.850000	1.941259	7.196424
	YOUNG SINGLES/COUPLES	36509	148337.200000	1.852778	7.527896
Premium	MIDAGE SINGLES/COUPLES	14626	55042.350000	1.891130	7.116932
	NEW FAMILIES	2807	10861.700000	1.861406	7.202719
	OLDER FAMILIES	20518	75983.000000	1.945572	7.204912
	OLDER SINGLES/COUPLES	32042	124457.050000	1.913984	7.434266
	RETIREEES	23513	91951.950000	1.901116	7.434666
	YOUNG FAMILIES	21156	79249.100000	1.937540	7.257908
	YOUNG SINGLES/COUPLES	10687	39347.900000	1.806152	6.649975

Customer Segmentation Insights:

Consumer Segment: Premium

Older singles/couples segment had the highest sales with 31,869 packs and revenue of 123,668.95.

Consumer Segment: Budget

The Older families lifestage segment had the highest sales at 42,425 packs and revenue of 158,379.95.

Consumer Segment: Mainstream

The Mainstream Customer segment had the highest sales and quantity.

The Young Singles/Couples lifestage segment had the highest sales at 36,350 packs and revenue of 147,634.30.

Retirees also had sales results that were very close at 37,968 packs sold with revenue of 146,328.75.

The older singles/couples lifestage segment was the customer segment with the strongest sales overall with total revenue of 376,366.65.

Conclusion:

Target marketing towards older families, retirees, older singles/couples and young singles/couples would likely yield stronger sales results.

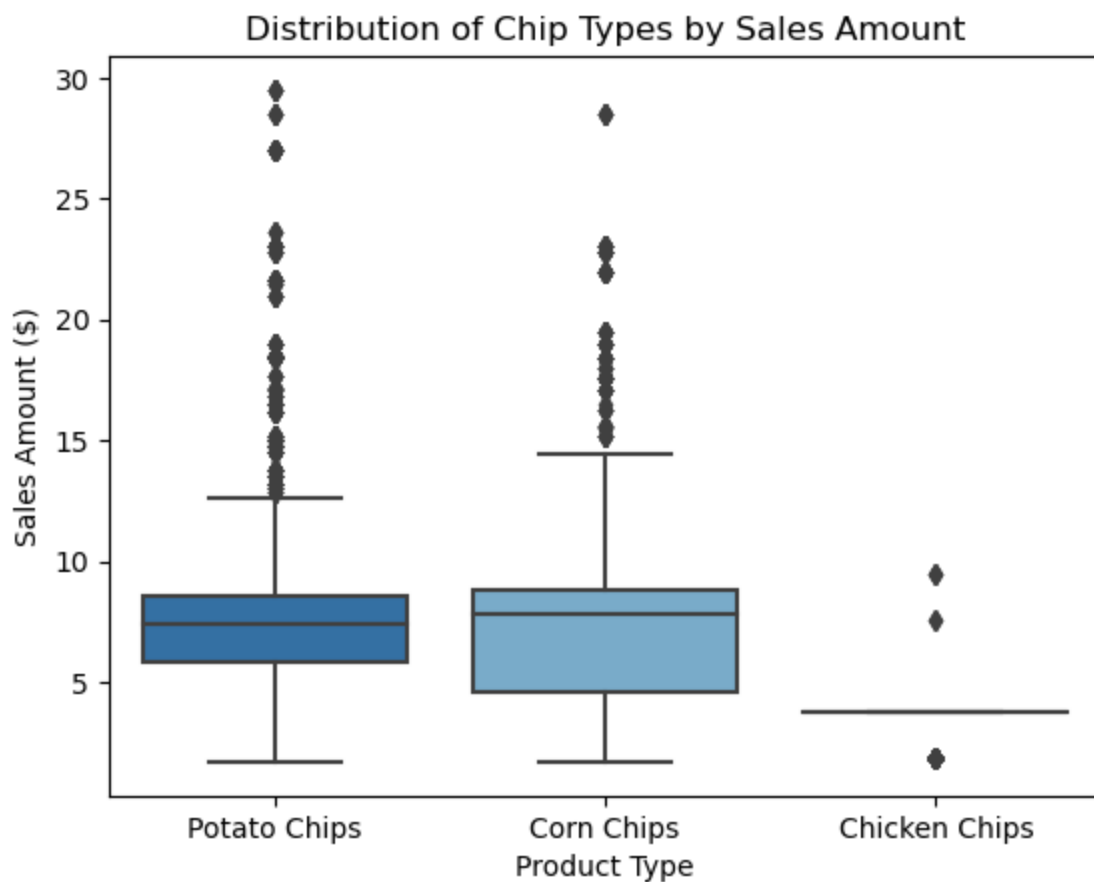
---

## Product Type Sales Analysis



### Distribution of Chip Types by Sales Amount

```
In [ ]: sns.boxplot(  
    data = df,  
    x = "product_type",  
    y = "tot_sales",  
    palette = "Blues_r"  
)  
plt.xlabel("Product Type")  
plt.ylabel("Sales Amount ($)")  
plt.title("Distribution of Chip Types by Sales Amount");
```



Product Type by Quantity and Total Revenue

```
In [ ]: pd.pivot_table(
    data = df,
    index = "product_type",
    values = ["prod_qty", "tot_sales"],
    aggfunc = ["sum"],
    sort = False
).style.background_gradient(cmap = "mako")
```

Out[ ]:

product_type	sum	
	prod_qty	tot_sales
Potato Chips	315526	1196816.500000
Corn Chips	157620	617712.200000
Chicken Chips	2763	5249.700000

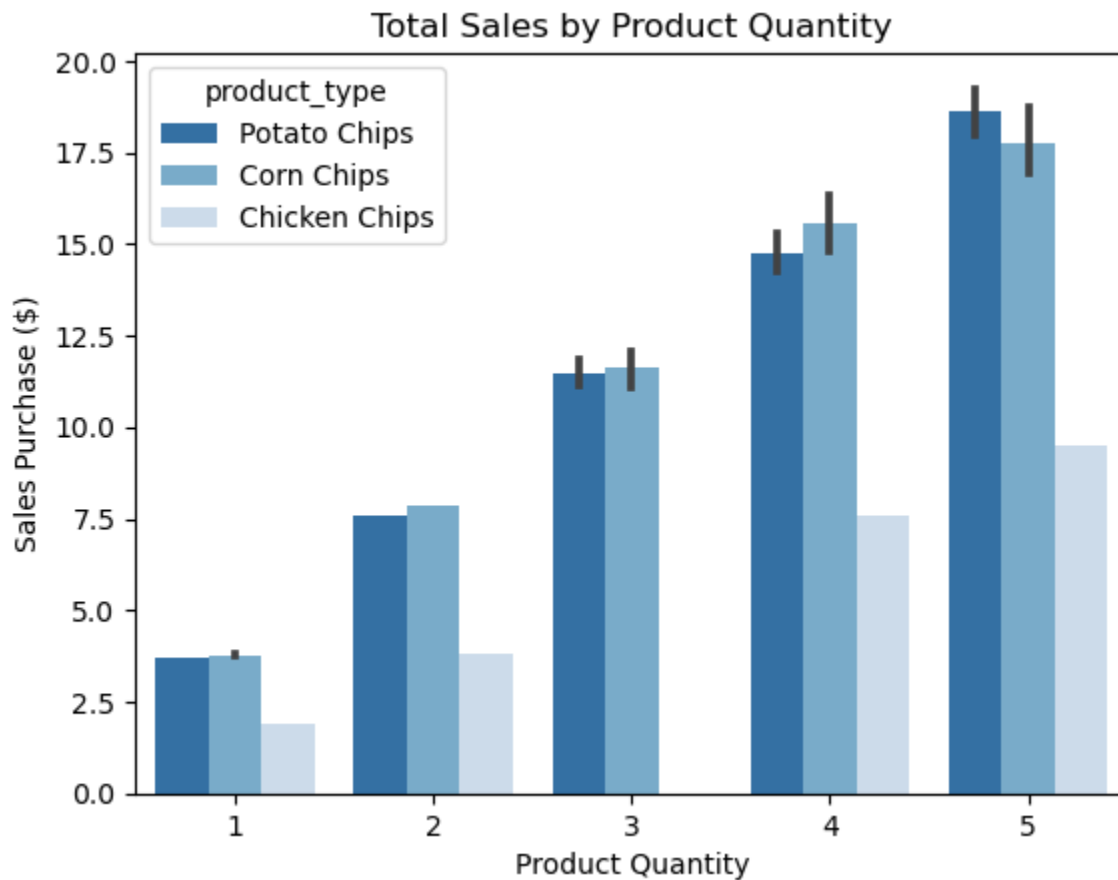
Total Sales by Product Quantity

```
In [ ]: sns.barplot(
    data = df,
    x = "prod_qty",
    y = "tot_sales",
```

```

    hue = "product_type",
    palette = "Blues_r"
)
plt.xlabel("Product Quantity")
plt.ylabel("Sales Purchase ($)")
plt.title("Total Sales by Product Quantity");

```



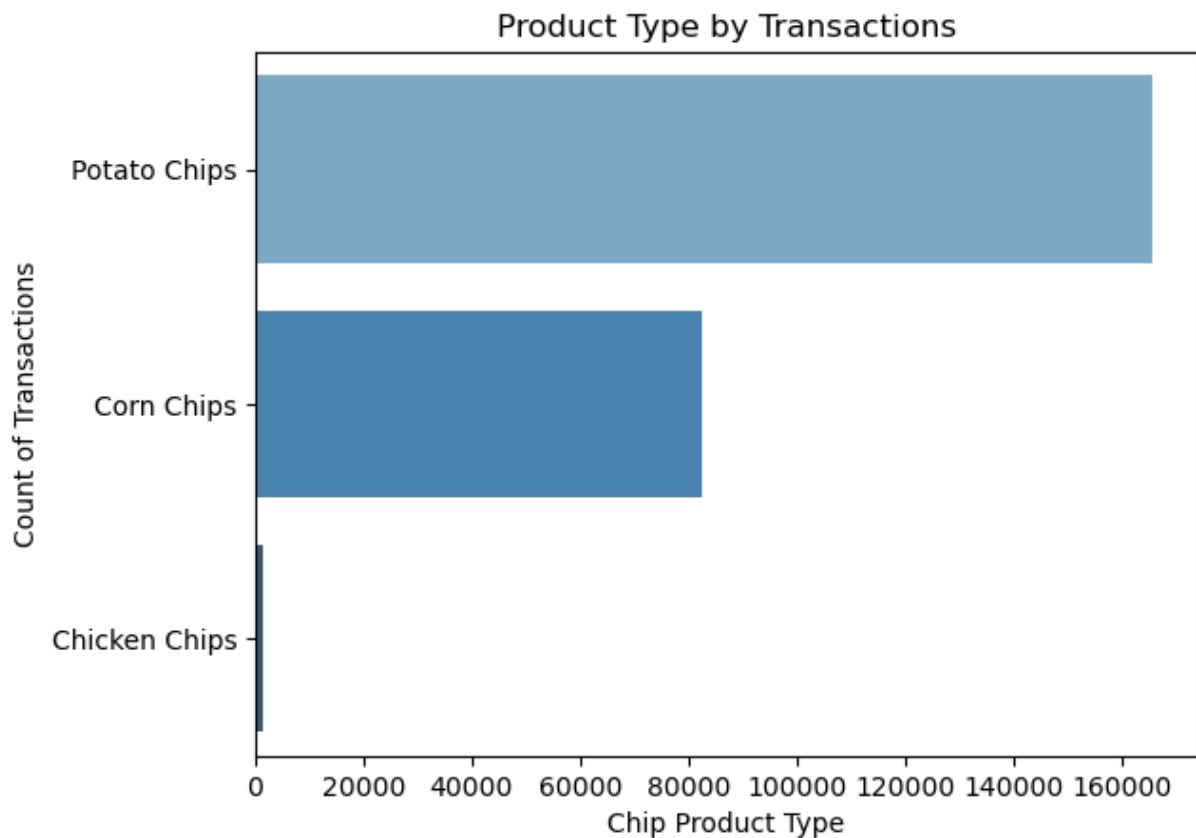
Chip Product Type by Transactions

```

In [ ]: barplot_df = df["product_type"].value_counts().reset_index()

sns.barplot(
    data = barplot_df,
    x = "count",
    y = "product_type",
    palette = "Blues_d"
)
plt.xlabel("Chip Product Type")
plt.ylabel("Count of Transactions")
plt.title("Product Type by Transactions");

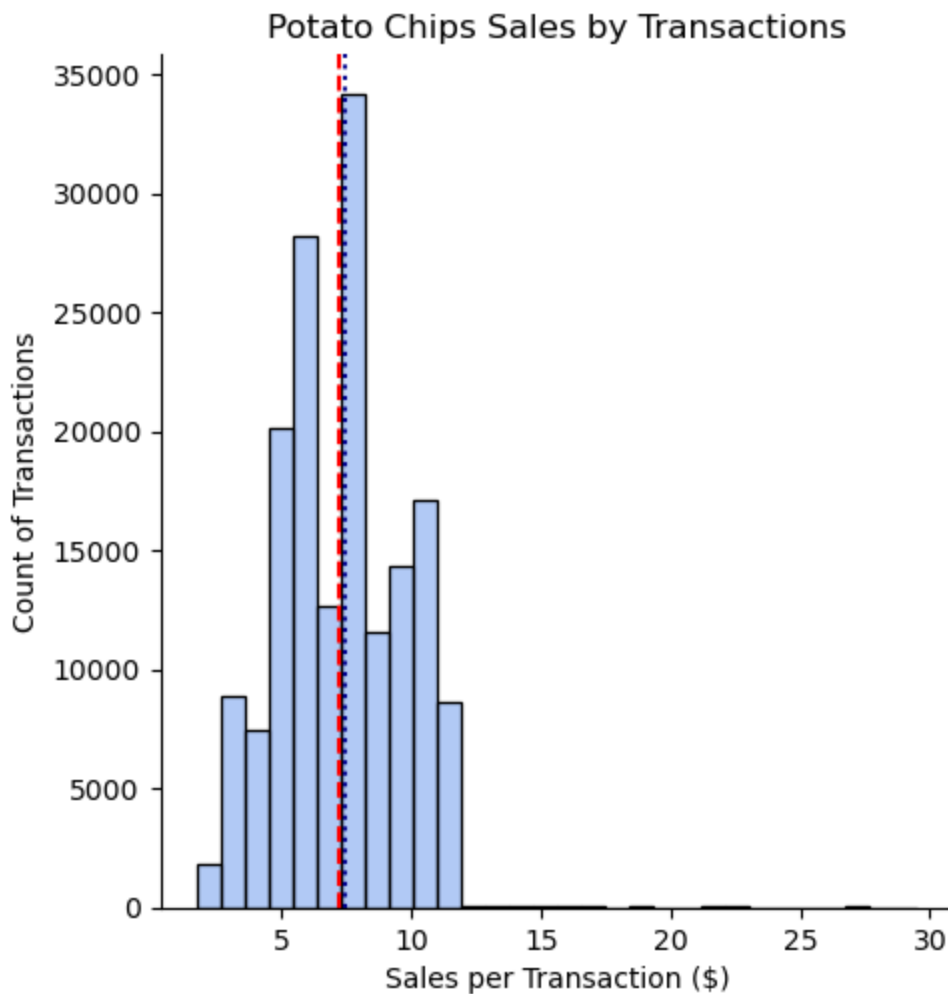
```



Potato Chips Sales by Transactions

```
In [ ]: sns.displot(
    data = df.query('product_type == "Potato Chips"')['tot_sales'],
    color = "cornflowerblue",
    alpha = 0.5,
    bins = 30
)
plt.axvline(
    np.mean(df.query('product_type == "Potato Chips"')['tot_sales']),
    color = "red",
    ls = "--",
)
plt.axvline(
    np.median(df.query('product_type == "Potato Chips"')['tot_sales']),
    color = "darkblue",
    linestyle = ":"
)
plt.xlabel("Sales per Transaction ($)")
plt.ylabel("Count of Transactions")
plt.title("Potato Chips Sales by Transactions");
```

c:\ProgramData\Anaconda3\envs\skorch-ml\Lib\site-packages\seaborn\axisgrid.py:118: UserWarning: The figure layout has changed to tight  
self.\_figure.tight\_layout(\*args, \*\*kwargs)

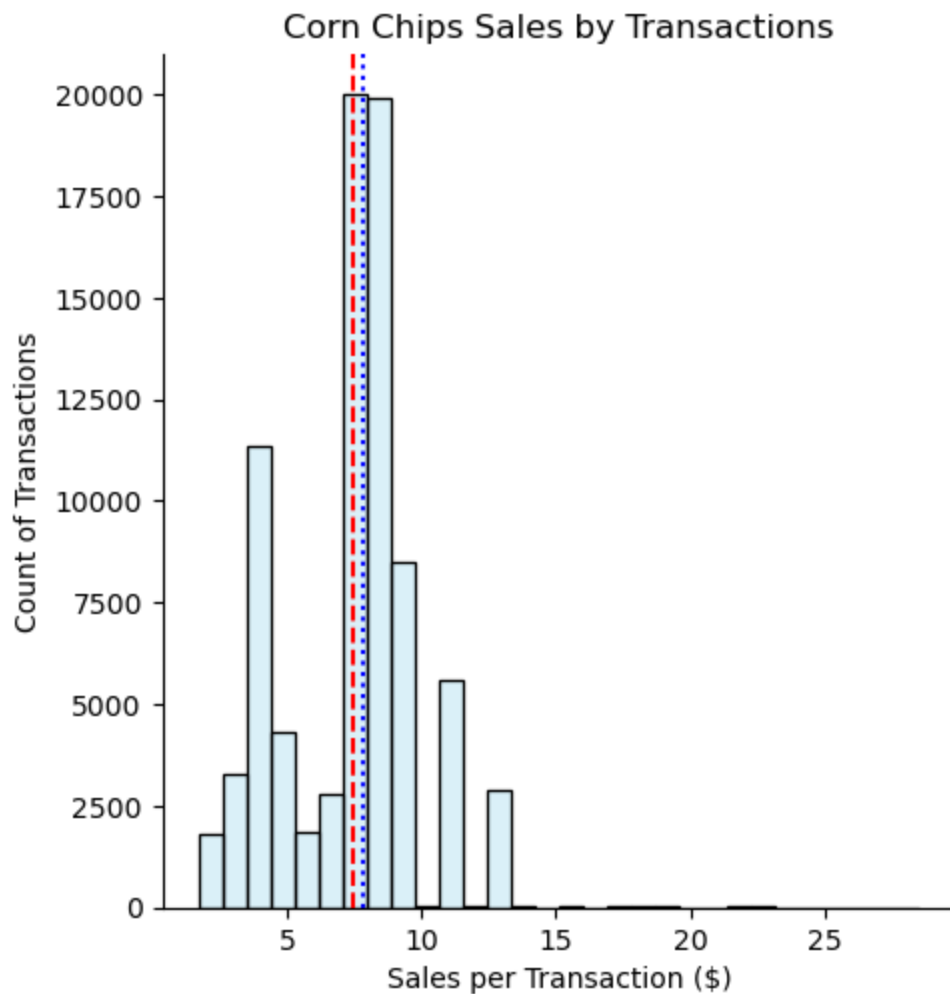


Corn Chips Sales by Transactions

```
In [ ]: sns.displot(
    data = df.query('product_type == "Corn Chips"')['tot_sales'],
    color = "skyblue",
    alpha = 0.3,
    bins = 30,
)
plt.axvline(
    np.mean(df.query('product_type == "Corn Chips"')['tot_sales']),
    color = "red",
    ls = "--"
)
plt.axvline(
    np.median(df.query('product_type == "Corn Chips"')['tot_sales']),
    color = "blue",
    ls = ":"
)
plt.xlabel("Sales per Transaction ($)")
plt.ylabel("Count of Transactions")
plt.title("Corn Chips Sales by Transactions");
```

c:\ProgramData\Anaconda3\envs\skorch-ml\Lib\site-packages\seaborn\axisgrid.py:118: UserWarning: The figure layout has changed to tight  
self.\_figure.tight\_layout(\*args, \*\*kwargs)

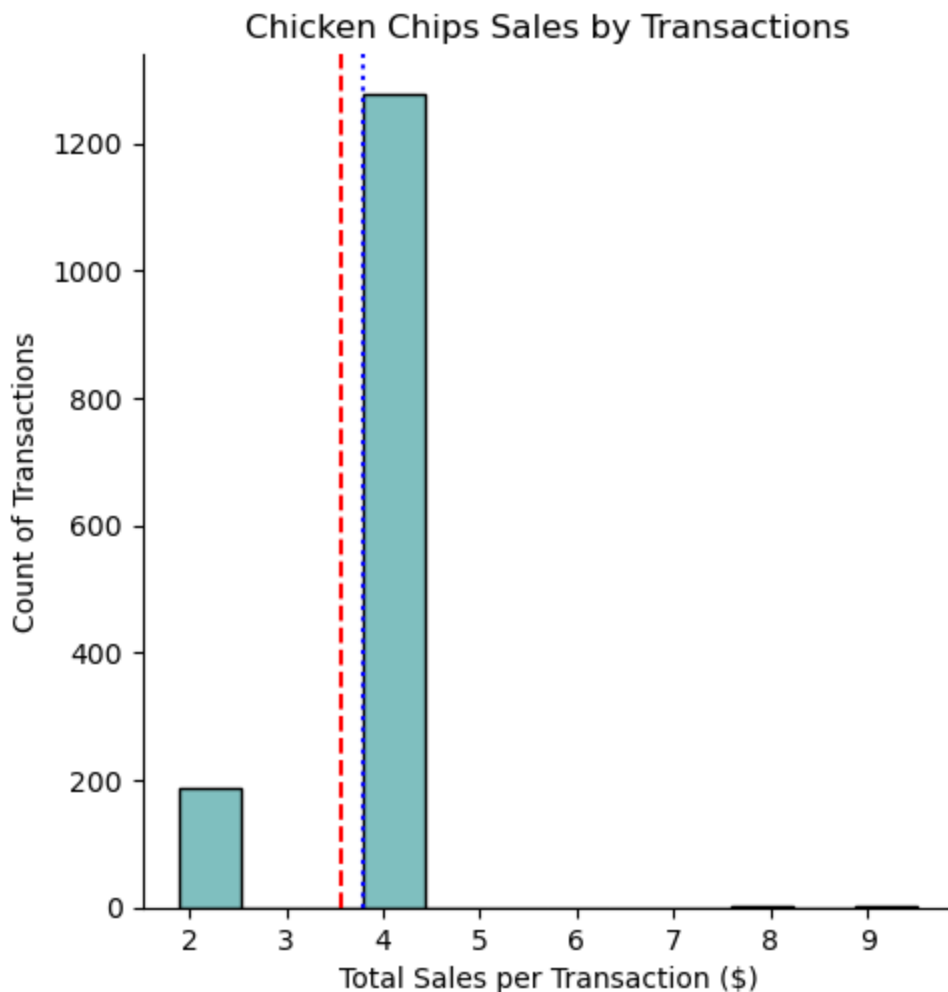




Chicken Chips Sales by Transactions

```
In [ ]: sns.displot(
    data = df.query('product_type == "Chicken Chips")["tot_sales"],
    color = "teal",
    alpha = 0.5
)
plt.axvline(
    np.mean(df.query('product_type == "Chicken Chips")["tot_sales"]),
    color = "red",
    ls = "--"
)
plt.axvline(
    np.median(df.query('product_type == "Chicken Chips")["tot_sales"]),
    color = "blue",
    ls = ":"
)
plt.xlabel("Total Sales per Transaction ($)")
plt.ylabel("Count of Transactions")
plt.title("Chicken Chips Sales by Transactions");
```

c:\ProgramData\Anaconda3\envs\skorch-ml\Lib\site-packages\seaborn\axisgrid.py:118: UserWarning: The figure layout has changed to tight  
 self.\_figure.tight\_layout(\*args, \*\*kwargs)



```
In [ ]: potato_chip_sales = df.query('product_type == "Potato Chips")["tot_sales"].sum().round(2)
potato_chip_qty = df.query('product_type == "Potato Chips")["prod_qty"].sum()

print(f"Potato Chip Sales represent {(potato_chip_sales / total_sales_revenue).round(2)}% of the total sales revenue
and {(potato_chip_qty / total_quantity_sold).round(3) * 100}% of the total quantity sold.")
```

Potato Chip Sales represent 66.0% of the total sales revenue and 66.3% of the total quantity of chips sold.

```
In [ ]: print(f"The average sales revenue recieved per pack of chips\
made from potatoes was: ${ (potato_chip_sales / potato_chip_qty).round(2)}.")
```

The average sales revenue recieved per pack of chips made from potatoes was: \$3.79.

```
In [ ]: corn_chip_sales = df.query('product_type == "Corn Chips")["tot_sales"].sum().round(2)
corn_chip_qty = df.query('product_type == "Corn Chips")["prod_qty"].sum()

print(f"Chips made from corn represent {(corn_chip_sales / total_sales_revenue).round(2)}% of the total sales revenue
and make up {(corn_chip_qty / total_quantity_sold).round(3) * 100}% of the total quantity sold.")
```

Chips made from corn represent 34.0% of the total sales revenue and make up 33.1% of the total quantity of chips sold.

```
In [ ]: print(f"The average sales revenue recieved from each pack of chips made \
from corn was: ${ (corn_chip_sales / corn_chip_qty).round(2)}.")
```

The average sales revenue recieved from each pack of chips made from corn was: \$3.92.

```
In [ ]: chicken_chip_sales = df.query('product_type == "Chicken Chips")["tot_sales"].sum()
chicken_chip_qty = df.query('product_type == "Chicken Chips")["prod_qty"].sum()

print(f"Chips made from chicken breast meat represent {(chicken_chip_qty / total_sales).round(4)}% of total sales revenue and make up {(chicken_chip_qty / total_quantity_sold).round(4)}% of total quantity sold.")
```

Chips made from chicken breast meat represent 0.152% of the total sales revenue and make up 0.58% of the total quantity of chips sold.

```
In [ ]: print(f"The average sales revenue recieved from each pack of chips made from \
chicken breast meat was: ${((chicken_chip_sales / chicken_chip_qty).round(2))}.")
```

The average sales revenue recieved from each pack of chips made from chicken breast meat was: \$1.9.

```
In [ ]: print(f"Corn chips recieved ${((corn_chip_sales / corn_chip_qty).round(2))} - (potato_
```

Corn chips recieved \$0.12999999999999999 more per bag.

## Product Type Sales Insights:

Potato Chips:

Potato Chip sales generated 66% of total sales revenue and 66.3% of the total quantity of bags sold.

Potato chip brands sold almost 2x as many bags compared to brands that produced chips made from corn.

The majority of Potato chip sales transactions were between 5 and 8.

Corn Chips:

Corn chip products generated the most revenue per bag sold at \$3.92.

Corn chip products recieved 13 cents more revenue per bag on average than potato chips.

The majority of Corn chip sales transactions were between 7 and 10.

Conclusion:

Sales promotions for corn chip products would increase sales.

---

## Product Size & Quantity Analysis



### Chip Quantity Sales by Transactions

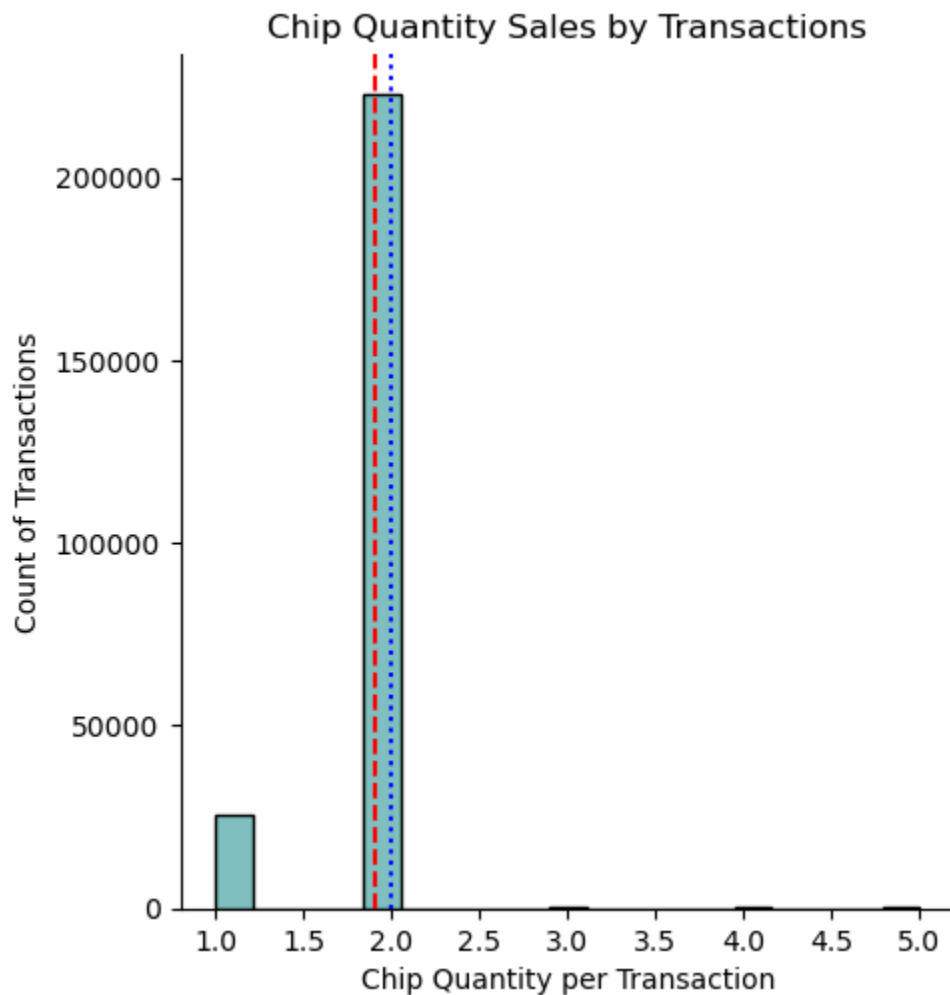
```
In [ ]: sns.displot(
    df["prod_qty"],
    color = "teal",
```

```

    alpha = 0.5,
)
plt.axvline(
    np.mean(df["prod_qty"]),
    color = "red",
    ls = "--"
)
plt.axvline(
    np.median(df["prod_qty"]),
    color = "blue",
    ls = ":"
)
plt.xlabel("Chip Quantity per Transaction")
plt.ylabel("Count of Transactions")
plt.title("Chip Quantity Sales by Transactions");

```

c:\ProgramData\Anaconda3\envs\skorch-ml\Lib\site-packages\seaborn\axisgrid.py:118: UserWarning: The figure layout has changed to tight  
self.\_figure.tight\_layout(\*args, \*\*kwargs)



Total Sales and Average Sales by Product Quantity

```

In [ ]: pd.pivot_table(
    data = df,
    index = "prod_qty",
    values = "tot_sales",

```

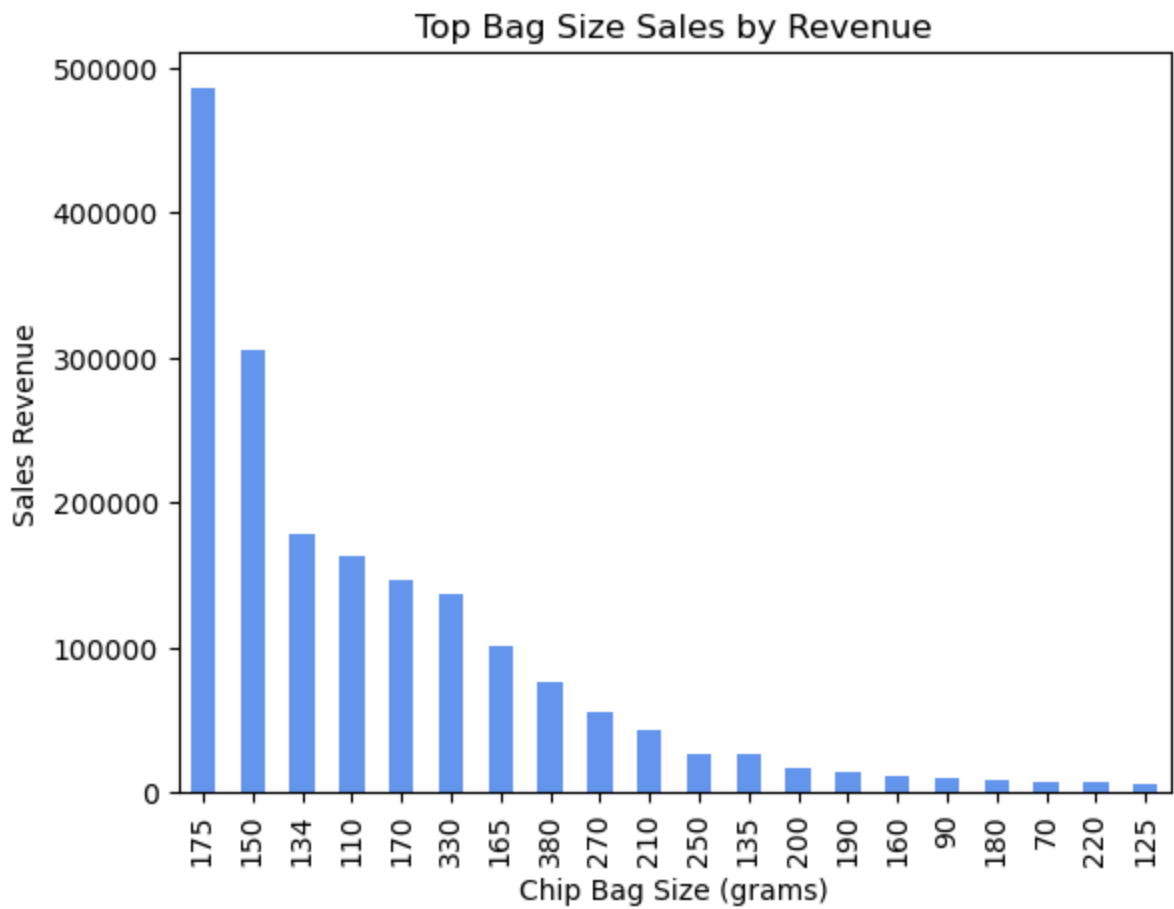
```
aggfunc = ["sum", "mean"]
).style.background_gradient(cmap = "mako")
```

Out[ ]:

	sum	mean
	tot_sales	tot_sales
<b>prod_qty</b>		
1	95937.000000	3.712158
2	1705824.500000	7.662598
3	4742.850000	11.511772
4	5616.600000	14.977600
5	7657.450000	18.275537

Sales Revenue by Pack Size

```
In [ ]: (
    df.groupby("pack_size")["tot_sales"]
      .sum()
      .sort_values(
        ascending = False
      )
      .plot(
        kind = "bar",
        color = "cornflowerblue"
      )
    )
plt.xlabel("Chip Bag Size (grams)")
plt.ylabel("Sales Revenue")
plt.title("Top Bag Size Sales by Revenue");
```



Bag Size(g) and Sales Quantity by Total Sales Revenue

```
In [ ]: pd.pivot_table(  
    data = df,  
    index = "pack_size",  
    columns = ["prod_qty"],  
    values = "tot_sales",  
    aggfunc = "sum"  
) .style.background_gradient(cmap = "mako")
```

Out [ ]:	prod_qty	1	2	3	4	5
	pack_size					
	70	436.800000	6307.200000	14.400000	57.600000	36.000000
	90	625.600000	8904.600000	40.800000	20.400000	85.000000
	110	8002.800000	153504.800000	342.000000	326.800000	589.000000
	125	399.000000	5283.600000	12.600000	16.800000	21.000000
	134	9094.600000	166592.500000	444.000000	592.000000	932.400000
	135	1356.600000	24553.200000	63.000000	33.600000	84.000000
	150	16122.000000	285216.800000	813.900000	913.600000	1222.200000
	160	720.100000	9766.000000	34.200000	60.800000	66.500000
	165	5842.800000	94401.200000	337.800000	372.800000	406.000000
	170	7736.500000	137606.700000	320.700000	359.600000	649.500000
	175	25974.200000	454456.900000	1290.300000	1573.600000	2136.400000
	180	582.800000	7886.400000	27.900000	24.800000	46.500000
	190	902.400000	13303.200000	70.200000	93.600000	43.500000
	200	1046.900000	14854.200000	11.400000	38.000000	57.000000
	210	2264.400000	40456.800000	93.600000	100.800000	133.200000
	220	427.800000	6283.600000	20.700000	18.400000	80.500000
	250	1298.600000	24510.000000	77.400000	103.200000	107.500000
	270	2709.400000	52108.800000	124.200000	184.000000	299.000000
	330	6788.700000	128694.600000	376.200000	592.800000	342.000000
	380	3605.000000	71133.400000	227.550000	133.400000	320.250000

Average Sales Revenue for each Bag Size

```
In [ ]: pack_size_pivot = pd.pivot_table(
    data = df,
    index = "pack_size",
    values = ["tot_sales"],
    aggfunc = ["mean", "median"],
)
(
    pack_size_pivot
    .sort_values(by = "pack_size", ascending = False)
    .style.background_gradient(cmap = "mako")
)
```

Out[ ]:

	mean	median
	tot_sales	tot_sales
pack_size		
380	11.754925	11.800000
330	10.908636	11.400000
270	8.818679	9.200000
250	8.234995	8.600000
220	4.367647	4.600000
210	6.863648	7.200000
200	3.578694	3.800000
190	4.812321	3.600000
180	5.836785	6.200000
175	7.311925	6.600000
170	7.339889	8.800000
165	6.626175	6.000000
160	3.585051	3.800000
150	7.054984	7.800000
135	8.010562	8.400000
134	7.077344	7.400000
125	3.942916	4.200000
110	7.270532	7.600000
90	3.216888	3.400000
70	4.546782	4.800000

Product Quantity and Total Sales Grouped by Pack Size

```
In [ ]: (
    df.groupby("pack_size")[["prod_qty", "tot_sales"]]
      .sum()
      .sort_values(
        by = "tot_sales",
        ascending = False
      )
    ).style.background_gradient(cmap = "mako")
```



Out[ ]:

	prod_qty	tot_sales
pack_size		
175	126465	485431.400000
150	82174	304288.500000
134	48019	177655.500000
110	42835	162765.400000
170	38088	146673.000000
330	23999	136794.300000
165	29051	101360.600000
380	12273	75419.600000
270	12049	55425.400000
210	11962	43048.800000
250	6069	26096.700000
135	6212	26090.400000
200	8425	16007.500000
190	5673	14412.900000
160	5604	10647.600000
90	5692	9676.400000
180	2764	8568.400000
70	2855	6852.000000
220	2970	6831.000000
125	2730	5733.000000

### Pack Size Analysis Insights:

#### Bag Size:

The top 5 Bag sizes(g): 175, 150, 134, 110, 170.

Bag size of 175g generated revenue of 482,094.40.

Bag size of 150g generated revenue of 302,922.40.

Bag size of 134g generated revenue of 176138.50.

The bag sizes 175 and 150 have significantly more transaction purchases than the 3rd bag size of 134g.

The bag size 330 generates almost as much revenue as the 170.

#### Product Quantity:

Consumers typically favor buying 2 bags in a single purchase transaction. This holds especially true for the larger bag sizes.

Conclusion:

Developing a promotion marketing strategy to increase purchases of 2 bags with the 175g and 150g bags sizes would likely yield positive results.

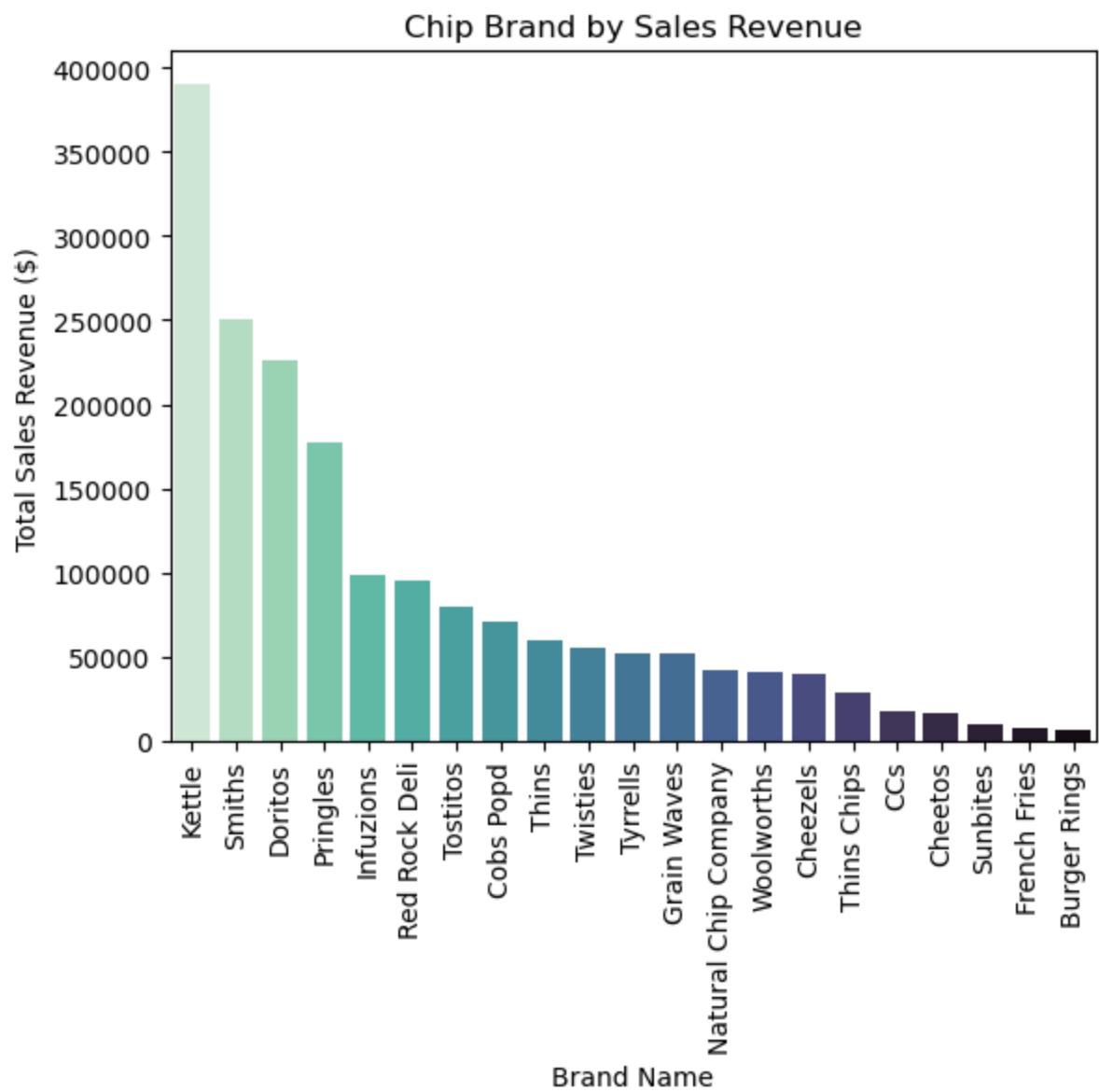
---

## Product Brand Analysis



### Chip Brand by Sales Revenue

```
In [ ]: brand_sales = (
    df.groupby("brand_name")["tot_sales"]
    .sum()
    .sort_values(
        ascending = False
    )
    .reset_index()
)
ax = sns.barplot(
    data = brand_sales,
    x = "brand_name",
    y = "tot_sales",
    palette = "mako_r"
)
ax.set_xticklabels(ax.get_xticklabels(), rotation = 90)
plt.xlabel("Brand Name")
plt.ylabel("Total Sales Revenue ($)")
plt.title("Chip Brand by Sales Revenue");
```



Brands by Total Sales Revenue and Total Quantity Sold

```
In [ ]: (
    df.groupby("brand_name")[["prod_qty", "tot_sales"]]
      .sum()
      .sort_values(
        "tot_sales",
        ascending = False
      )
    ).style.background_gradient(cmap = "mako")
```

Out[ ]:

	prod_qty	tot_sales
brand_name		
Kettle	79051	390239.800000
Smiths	66406	250750.900000
Doritos	48331	226329.900000
Pringles	48019	177655.500000
Infuzions	27119	99047.600000
Red Rock Deli	33646	95046.000000
Tostitos	18134	79789.600000
Cobs Popd	18571	70569.800000
Thins	18107	59739.900000
Twisties	12049	55425.400000
Tyrrells	12298	51647.400000
Grain Waves	14726	51617.200000
Natural Chip Company	14106	42318.000000
Woolworths	22333	41059.100000
Cheezels	8747	40029.900000
Thins Chips	8822	29112.600000
CCs	8609	18078.900000
Cheetos	5530	16884.500000
Sunbites	5692	9676.400000
French Fries	2643	7929.000000
Burger Rings	2970	6831.000000

Top 10 Products by Total Sales Revenue

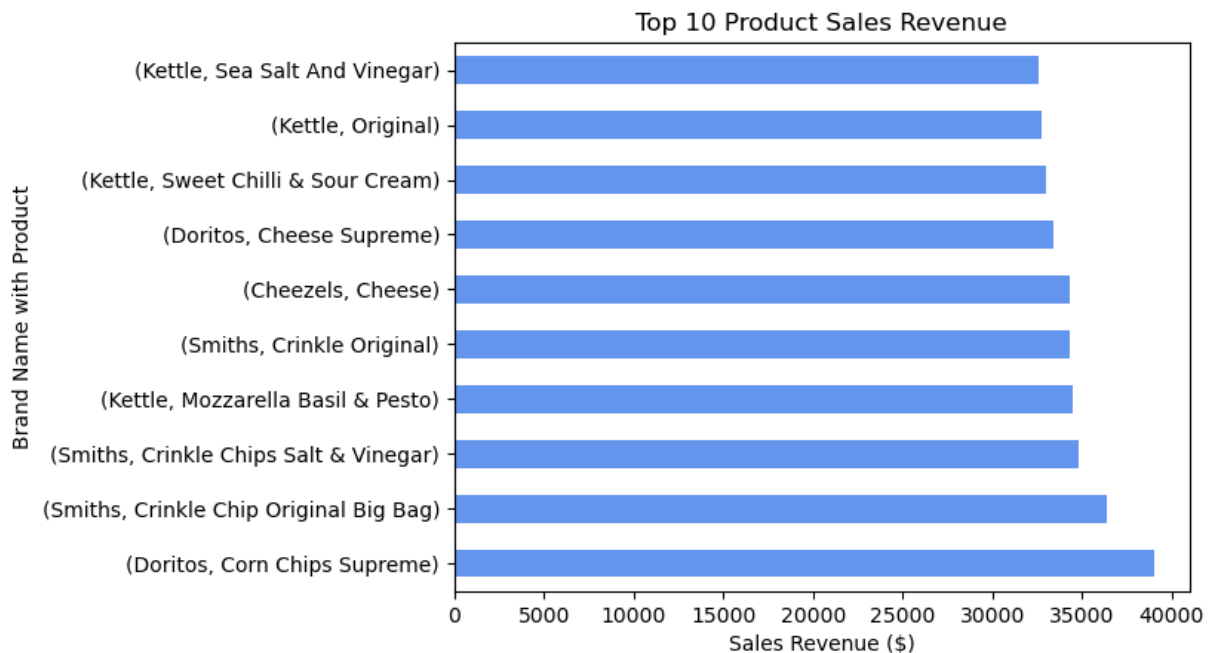
```
In [ ]: (
    df.groupby(["brand_name", "product_name", "pack_size"])[["prod_qty", "tot_sales"]]
    .sum()
    .sort_values(by = "tot_sales", ascending = False)
    .head(10)
).style.background_gradient(cmap = "mako")
```

Out[ ]:

			prod_qty	tot_sales
brand_name	product_name	pack_size		
Doritos	Corn Chips Supreme	380	6109	39052.000000
Smiths	Crinkle Chip Original Big Bag	380	6164	36367.600000
	Crinkle Chips Salt & Vinegar	330	6106	34804.200000
Kettle	Mozzarella Basil & Pesto	175	6381	34457.400000
Smiths	Crinkle Original	330	6018	34302.600000
Cheezels	Cheese	330	6017	34296.900000
Doritos	Cheese Supreme	330	5858	33390.600000
Kettle	Sweet Chilli & Sour Cream	175	6120	33031.800000
	Original	175	6064	32740.200000
	Sea Salt And Vinegar	175	6035	32589.000000

Top 10 Product Sales Revenue

```
In [ ]: (
    df.groupby(["brand_name", "product_name"])["tot_sales"]
      .sum()
      .sort_values(ascending = False)
      .head(10)
      .plot(kind = "barh", color = "cornflowerblue")
)
plt.xlabel("Sales Revenue ($)")
plt.ylabel("Brand Name with Product")
plt.title("Top 10 Product Sales Revenue");
```



## Top 10 products by Product Quantity

```
In [ ]: (
    df.groupby(["brand_name", "product_name", "pack_size"])["prod_qty", "tot_sales"]
      .sum()
      .sort_values(by = "prod_qty", ascending = False)
      .head(10)
    ).style.background_gradient(cmap = "mako")
```

Out[ ]:

			prod_qty	tot_sales
brand_name	product_name	pack_size		
Kettle	Mozzarella Basil & Pesto	175	6381	34457.400000
	Tortilla Chips Honey & Jalapeno Chili	150	6309	29021.400000
Cobs Popd	Sea Salt Chips	110	6277	23852.600000
	Sweet Chlli & Sour Cream Chips	110	6256	23772.800000
Tostitos	Splash Of Lime	175	6234	27429.600000
Tyrrells	Crisps Mature Cheddar & Chives	165	6227	26149.200000
Kettle	Sweet Potato Sea Salt	135	6212	26090.400000
Infuzions	Thai Sweet Chili Potato Mix	110	6206	23582.800000
Thins	Potato Chips Hot & Spicy	175	6185	20410.500000
Doritos	Corn Chips Cheese Supreme	170	6180	27183.200000

## Product Brand Analysis Insights:

Doritos Supreme is the best selling corn based product and the best overall selling product with 6,134 packs sold and generating revenue of 38,918.75.

The Mozzarella Basil and Pesto product sold 6,284 packs and Tortilla Chips Honey and Jalapeno Chili sold 6,258 packs.

Kettle is the brand with the most sales with 78,423 packs sold and generated revenue of 387,066.20.

Smiths sold the second most bags with 66,152 packs and generated revenue of 249,464.10.

The Kettle brand produced the top 3 products with the greatest quantity sold.

Doritos and Pringles had similar sales quantity numbers with Doritos selling 48,007 packs and Pringles with 47609 packs.

Doritos produced 224,891.95 and Pringles produced 176,138.50 revenue.

Due to Doritos products having a higher price point and larger bag sizes caused the brand to generate more revenue.

---

## High Ticket Items Analysis



### Consumer Segment by Total Product Quantity and Total Sales Revenue

```
In [ ]: df_high = df.copy()
high_mask = df_high["product_price"] > 4.5
high_ticket_products = df_high[high_mask]

pd.pivot_table(
    data = high_ticket_products,
    index = "premium_customer",
    values = ["prod_qty", "tot_sales"],
    aggfunc = ["sum", "mean"]
).style.background_gradient(cmap = "mako")
```

Out[ ]:

	sum		mean	
	prod_qty	tot_sales	prod_qty	tot_sales
premium_customer				
Budget	41697	217701.000000	1.923826	10.044339
Mainstream	47919	249780.400000	1.907755	9.944279
Premium	31324	163610.800000	1.916427	10.009838

Lifestage Segment by Product Quantity and Total Sales Revenue

```
In [ ]: pd.pivot_table(
    data = high_ticket_products,
    index = "lifestage",
    values = ["prod_qty", "tot_sales"],
    aggfunc = ["sum", "mean"]
).style.background_gradient(cmap = "mako")
```

Out[ ]:

	sum		mean	
	prod_qty	tot_sales	prod_qty	tot_sales
lifestage				
MIDAGE SINGLES/COUPLES	11907	62114.400000	1.911850	9.973410
NEW FAMILIES	3257	17010.000000	1.878316	9.809689
OLDER FAMILIES	20808	108647.100000	1.952336	10.193948
OLDER SINGLES/COUPLES	25617	133557.200000	1.924209	10.032089
RETIREEES	23648	123354.000000	1.901117	9.916714
YOUNG FAMILIES	18708	97735.400000	1.941268	10.141683
YOUNG SINGLES/COUPLES	16995	88674.100000	1.861854	9.714516

Consumer and Lifestage Segements by Product Quantity and Total Sales Revenue

```
In [ ]: pd.pivot_table(
    data = high_ticket_products,
    index = ["premium_customer", "lifestage"],
    values = ["tot_sales", "prod_qty"],
    aggfunc = ["sum", "mean"]
).style.background_gradient(cmap = "mako")
```



Out[ ]:

		sum		mean	
		prod_qty	tot_sales	prod_qty	tot_sales
premium_customer	lifestage				
Budget	MIDAGE SINGLES/COUPLES	2122	11035.900000	1.915162	9.960199
	NEW FAMILIES	1417	7410.300000	1.876821	9.814967
	OLDER FAMILIES	10128	52901.200000	1.955590	10.214559
	OLDER SINGLES/COUPLES	8847	46083.800000	1.923679	10.020396
	RETIREEES	7482	39104.700000	1.902365	9.942715
	YOUNG FAMILIES	8278	43335.000000	1.946391	10.189278
	YOUNG SINGLES/COUPLES	3423	17830.100000	1.853276	9.653546
Mainstream	MIDAGE SINGLES/COUPLES	6286	32793.300000	1.911800	9.973631
	NEW FAMILIES	1128	5870.000000	1.880000	9.783333
	OLDER FAMILIES	6091	31754.700000	1.951618	10.174527
	OLDER SINGLES/COUPLES	8114	42327.100000	1.925030	10.042017
	RETIREEES	9746	50710.100000	1.898695	9.879232
	YOUNG FAMILIES	5391	28053.900000	1.942003	10.105872
	YOUNG SINGLES/COUPLES	11163	58271.300000	1.865163	9.736224
Premium	MIDAGE SINGLES/COUPLES	3499	18285.200000	1.909934	9.981004
	NEW FAMILIES	712	3729.700000	1.878628	9.840897
	OLDER FAMILIES	4589	23991.200000	1.946141	10.174385
	OLDER SINGLES/COUPLES	8656	45146.300000	1.923983	10.034741
	RETIREEES	6420	33539.200000	1.903350	9.943433
	YOUNG FAMILIES	5039	26346.500000	1.932132	10.102186
	YOUNG SINGLES/COUPLES	2409	12572.700000	1.858796	9.701157

Brands by Sales Revenue

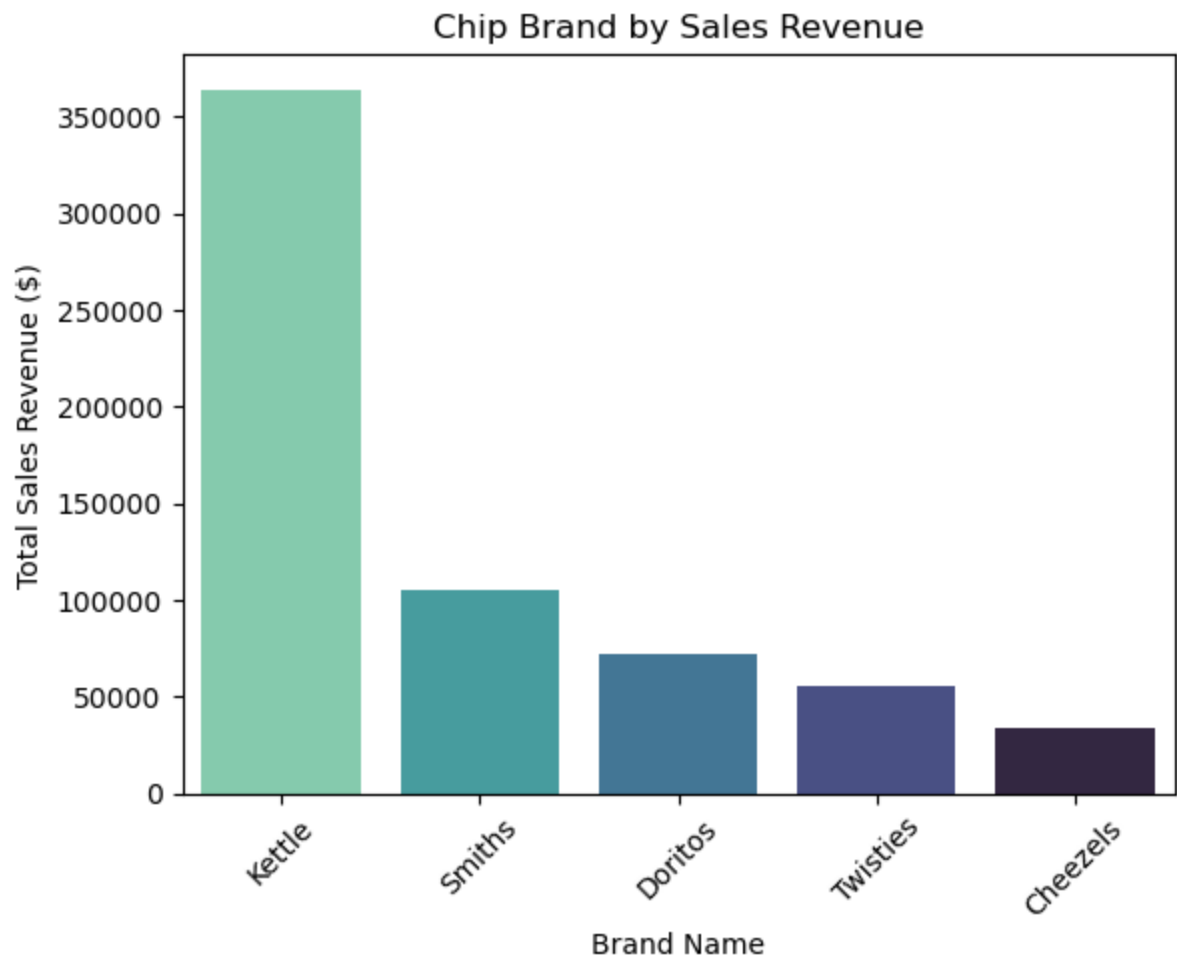
```
In [ ]: (
    high_ticket_products
    .groupby("brand_name")["prod_qty", "tot_sales"]
    .sum()
    .sort_values(
        "tot_sales",
        ascending = False
    )
).style.background_gradient(cmap = "mako")
```

```
Out[ ]:      prod_qty      tot_sales
```

brand_name		
Kettle	72821	364109.400000
Smiths	18288	105474.400000
Doritos	11765	71786.100000
Twisties	12049	55425.400000
Cheezels	6017	34296.900000

Chip Brand by Sales Revenue

```
In [ ]: brand_sales_high = (
    high_ticket_products
    .groupby("brand_name")["tot_sales"]
    .sum()
    .sort_values(
        ascending = False
    )
    .reset_index()
)
ax = sns.barplot(
    data = brand_sales_high,
    x = "brand_name",
    y = "tot_sales",
    palette = "mako_r"
)
ax.set_xticklabels(ax.get_xticklabels(), rotation = 45)
plt.xlabel("Brand Name")
plt.ylabel("Total Sales Revenue ($)")
plt.title("Chip Brand by Sales Revenue");
```



Top 10 Product Sales by Total Sales Revenue

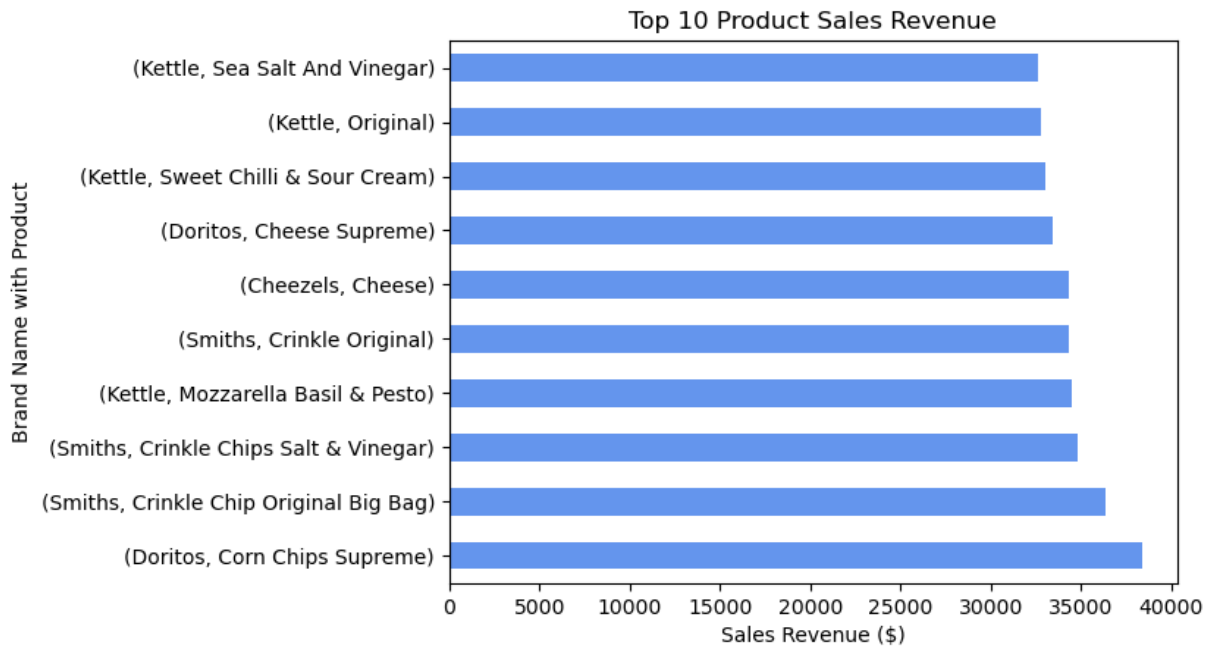
```
In [ ]: (
    high_ticket_products
    .groupby(["brand_name", "product_name", "pack_size"])[["prod_qty", "tot_sales"]]
    .sum()
    .sort_values("tot_sales", ascending = False)
    .head(10)
).style.background_gradient(cmap = "mako")
```

Out[ ]:

			prod_qty	tot_sales
brand_name	product_name	pack_size		
Doritos	Corn Chips Supreme	380	5907	38395.500000
Smiths	Crinkle Chip Original Big Bag	380	6164	36367.600000
	Crinkle Chips Salt & Vinegar	330	6106	34804.200000
Kettle	Mozzarella Basil & Pesto	175	6381	34457.400000
Smiths	Crinkle Original	330	6018	34302.600000
Cheezels	Cheese	330	6017	34296.900000
Doritos	Cheese Supreme	330	5858	33390.600000
Kettle	Sweet Chilli & Sour Cream	175	6115	33021.000000
	Original	175	6062	32734.800000
	Sea Salt And Vinegar	175	6035	32589.000000

Top 10 Product Sales Revenue

```
In [ ]: (
    high_ticket_products
    .groupby(["brand_name", "product_name"])["tot_sales"]
    .sum()
    .sort_values(ascending = False)
    .head(10)
    .plot(kind = "barh", color = "cornflowerblue")
)
plt.xlabel("Sales Revenue ($)")
plt.ylabel("Brand Name with Product")
plt.title("Top 10 Product Sales Revenue");
```



Top 10 Product Sales by Total Quantity

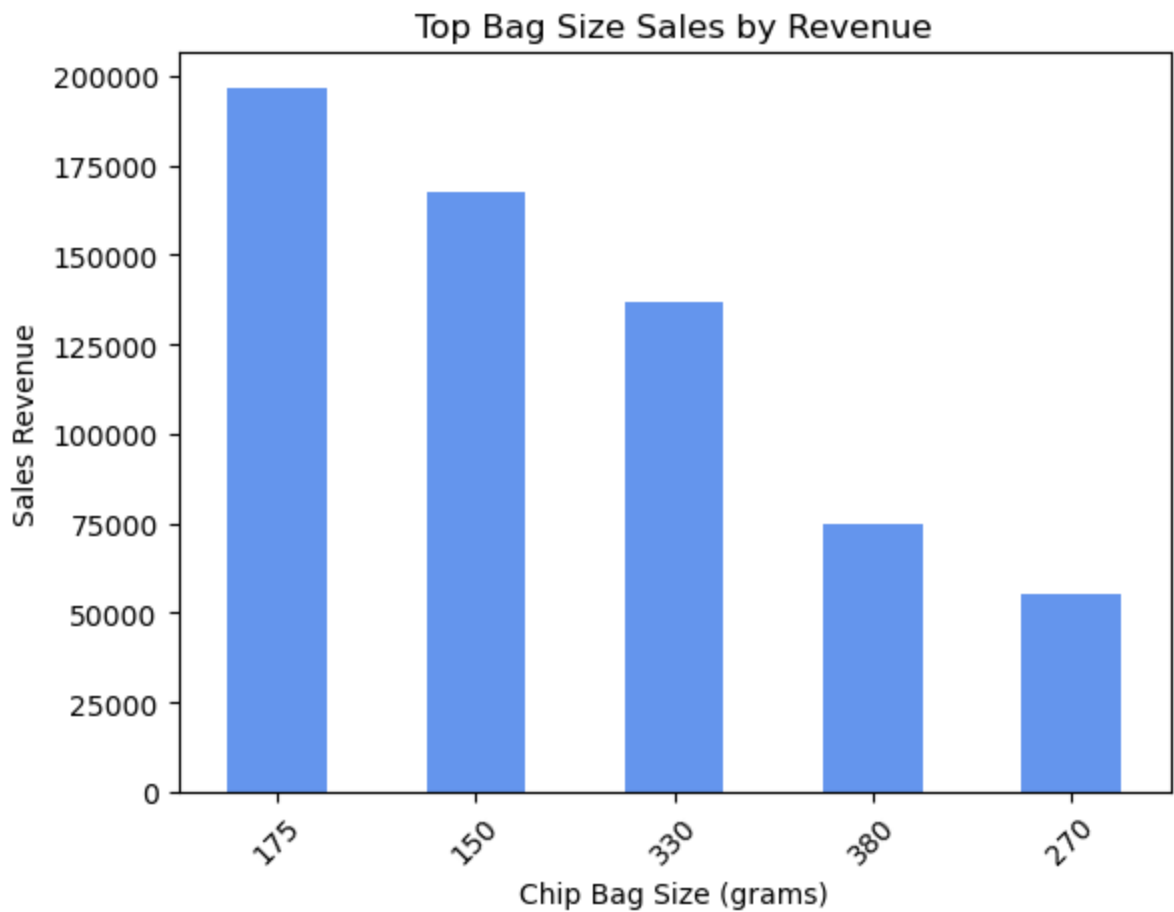
```
In [ ]: (
    high_ticket_products
    .groupby(["brand_name", "product_name", "pack_size"])["prod_qty", "tot_sales"]
    .sum()
    .sort_values("prod_qty", ascending = False)
    .head(10)
    ).style.background_gradient(cmap = "mako")
```

Out[ ]:

			prod_qty	tot_sales
brand_name	product_name	pack_size		
Kettle	Mozzarella Basil & Pesto	175	6381	34457.400000
	Tortilla Chips Honey & Jalapeno Chili	150	6309	29021.400000
Smiths	Crinkle Chip Original Big Bag	380	6164	36367.600000
Kettle	Sensations Camembert & Fig	150	6152	28299.200000
	Sweet Chilli & Sour Cream	175	6115	33021.000000
Smiths	Crinkle Chips Salt & Vinegar	330	6106	34804.200000
Kettle	Original	175	6062	32734.800000
Twisties	Chicken	270	6055	27853.000000
Kettle	Tortilla Chips Beetroot & Ricotta	150	6037	27770.200000
	Sea Salt And Vinegar	175	6035	32589.000000

Top Bag Size Sales by Revenue

```
In [ ]: (
    high_ticket_products
    .groupby("pack_size")["tot_sales"]
    .sum()
    .sort_values(
        ascending = False
    )
    .plot(
        kind = "bar",
        color = "cornflowerblue"
    )
)
loc, labels = plt.xticks()
plot_labels = ["175", "150", "330", "380", "270"]
plt.xticks(ticks = loc, labels = plot_labels, rotation = 45)
plt.xlabel("Chip Bag Size (grams)")
plt.ylabel("Sales Revenue")
plt.title("Top Bag Size Sales by Revenue");
```



Product Quantity and Total Sales by Pack Size

```
In [ ]: (
    high_ticket_products
    .groupby("pack_size")["prod_qty", "tot_sales"]
    .sum()
    .sort_values(
        by = "tot_sales",
        ascending = False
    )
).style.background_gradient(cmap = "mako")
```

Out [ ]:

	prod_qty	tot_sales
pack_size		
175	36416	196646.400000
150	36405	167463.000000
330	23999	136794.300000
380	12071	74763.100000
270	12049	55425.400000

High Ticket Product Sales Metrics

```
In [ ]: print("High priced products produced revenue of ${}.".format(
        high_ticket_products["tot_sales"].sum()
    )
)
```

High priced products produced revenue of \$631092.2.

```
In [ ]: print("High priced products sold {} bags.".format(
        high_ticket_products["prod_qty"].sum()
    )
)
```

High priced products sold 120940 bags.

```
In [ ]: print("High priced products generated {}% of total revenue.".format(
        (high_ticket_products["tot_sales"].sum()
        / total_sales_revenue).round(5)
        * 100)
    )
)
```

High priced products generated 34.68% of total revenue.

```
In [ ]: print("High priced products sold {}% of the total sales quantity.".format(
        (high_ticket_products["prod_qty"].sum()
        / total_quantity_sold).round(5)
        * 100)
    )
)
```

High priced products sold 25.41200000000003% of the total sales quantity.

## High Ticket Products Insights:

### Sales Metrics:

High priced products generated 34.598% of total revenue and sold 25.335% of the total sales quantity.

This means that a significant portion of revenue comes from high priced products.

### Customer Segmentation:

Premium Customer Segment: The Mainstream segment had generated the most revenue at 247,658.80 and sales of 47,511 packs.

### Lifestage Segment:

The Older Singles/Couples segment generated the most revenue at 132,204.10

and sales of 25,364 packs.

The Retirees segment also had very high sales numbers at 122,471.80 and sales of 23,485 packs.

The Older Families segment had produced revenues of 107,556.50 and 20,598 packs.

Mainstream Segment: Out of all the individual lifestage segments the Young Singles/Couples segment had generated the most revenue at 57,878.60 with 11,086 packs sold.



Product Segment:

The Kettle brand vastly outsold every other brand in this product segment with 72,221 packs sold.

Smiths was second with 18,123 and Twisties with 11,944.

Bag sizes

The bag sizes: 175, 150, 330, 380 and 270 were the key sizes for this product segment.

Conclusion:

Focusing on the Older Singles/Couples, Older Families and Retirees lifestage segments should increase revenue and sales quantities.

---

## Strategy Recommendations:

Lifestage Segments:

Target marketing on older singles/couples, older families, retirees and young singles/couples segments:

Consumer Segments:

Attention should be on the Mainstream segment.

This segment drives the most sales revenue.

Bag sizes:

Focus marketing on the popular bag sizes which were 175, 150, 330, 380 and 270g.

Promotions for 2 bag purchases per transaction, especially for larger bag sizes.

Brands:

Marketing centered on Kettle, Doritos, Smith's, Pringles and Twisties brands.

Sales Trends:

Sales promotions needs to take place in August, December and May.

---