eda-project-starbuck-analysis

March 2, 2024

0.1 Project Name : Starbuck Analysis

0.1.1 Project Type : Eksploratory Data Analysis (EDA)

0.1.2 Contribution : Individual

0.2 Project Summary :

- **The purpose of anlysis:** The objective of this project is to understand the factors influencing Starbucks product prices or identify patterns among all variables. Our analysis aims to provide valuable insights for consumers and baristas at Starbucks, as well as offer optimal insights for Starbucks' business.
- This project involves exploring and cleaning the dataset to prepare it for analysis. The data exploration process includes identifying and understanding data characteristics such as data types, missing values, and the distribution of values. Data cleaning involves identifying and addressing problems or inconsistencies in the data, such as errors, missing values, or duplicate records, and removing outliers.
- Through this process, we can identify and rectify any issues with the data, ensuring it is ready for further analysis. This is a crucial step in any data analysis project, as it allows us to work with high-quality data and avoid potential biases or errors that could impact the results. Clean and prepared data can now be used to answer specific research questions.
- Once the data is cleaned and prepared, we start exploring and summarizing it by describing the data and creating visualizations. This involves identifying patterns and trends in the data, developing relationships between different variables, or uncovering underlying causes of certain patterns or trends using various methods.
- We utilized data visualization to explore and understand patterns in Starbucks data. Various graphs and charts were created to visualize the data, and observations and insights were documented beneath each graph to enhance our understanding of the data and identify useful insights and patterns.
- Through this process, we can uncover trends and relationships in the data that are challenging to identify through raw data alone, such as factors influencing price and availability. We found that beverage categories, preparation methods, and nutritional compositions significantly impact price and popularity. Our analysis provides useful information for consumers and sellers at Starbucks.
- The observations and insights identified through this process will be beneficial for future analysis and decision-making regarding Starbucks. Additionally, our analysis offers valuable information for consumers and sellers at Starbucks.

0.3 Problem Statements :

1. Which beverage categories beverage consistently rank as the best-sellers at Starbucks?
2. Which beverage spesific beverage rank as best seller at starbucks?
3. What types of beverage have high calorie content at Starbucks?
4. provide insight into the types of beverage that sell well at Starbucks
5. What sells best in the type of beverage preparation at Starbucks?

there is a lot of problem statements and we have to finds information and insights through different different problem statements so now lets start…

0.4 Importing Library

[ ]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from warnings import filterwarnings
filterwarnings('ignore')
sns.set(style='darkgrid', palette='pastel')
```

0.5 Load the starbuck dataset

[ ]:
```
dataset = '/content/starbucks.csv'

df = pd.read_csv(dataset)
```

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

[ ]:
```
         Beverage_category       Beverage       Beverage_prep  Calories  \
0                   Coffee  Brewed Coffee               Short         3
1                   Coffee  Brewed Coffee                Tall         4
2                   Coffee  Brewed Coffee              Grande         5
3                   Coffee  Brewed Coffee               Venti         5
4  Classic Espresso Drinks    Caffè Latte  Short Nonfat Milk        70

   Total Fat (g)  Trans Fat (g)   Saturated Fat (g)   Sodium (mg)  \
0            0.1            0.0                 0.0             0
1            0.1            0.0                 0.0             0
2            0.1            0.0                 0.0             0
3            0.1            0.0                 0.0             0
4            0.1            0.1                 0.0             5

    Total Carbohydrates (g)   Cholesterol (mg)   Dietary Fibre (g)  \
0                         5                  0                   0
1                        10                  0                   0
```

```
2                         10              0                  0
3                         10              0                  0
4                         75             10                  0

    Sugars (g)   Protein (g)  Vitamin A (% DV)  Vitamin C (% DV)  \
0            0           0.3                0%                0%
1            0           0.5                0%                0%
2            0           1.0                0%                0%
3            0           1.0                0%                0%
4            9           6.0               10%                0%

   Calcium (% DV)  Iron (% DV)  Caffeine (mg)
0              0%           0%            175
1              0%           0%            260
2              0%           0%            330
3              2%           0%            410
4             20%           0%             75
```

0.6 UNDERSTAND THE GIVEN VARIABLES

Beverage_category: The category or type of beverage included in the dataset.

Beverage: The specific name of the Starbucks beverage identified in the dataset.

Beverage_prep: The preparation or serving method of the beverage, such as hot or cold, including various variations like latte or iced.

Calories: The total number of calories contained in one serving of the beverage.

Total Fat (g): The total amount of fat in grams in one serving of the beverage.

Trans Fat (g): The amount of trans fat in grams in one serving of the beverage.

Saturated Fat (g): The amount of saturated fat in grams in one serving of the beverage.

Sodium (mg): The amount of sodium in milligrams in one serving of the beverage.

Total Carbohydrates (g): The total amount of carbohydrates in grams in one serving of the beverage.

Cholesterol (mg): The amount of cholesterol in milligrams in one serving of the beverage.

Dietary Fibre (g): The amount of dietary fiber in grams in one serving of the beverage.

Sugars (g): The amount of sugar in grams in one serving of the beverage.

Protein (g): The amount of protein in grams in one serving of the beverage.

Vitamin A (% DV): The percentage of the daily recommended intake of Vitamin A in one serving of the beverage.

Vitamin C (% DV): The percentage of the daily recommended intake of Vitamin C in one serving of the beverage.

Calcium (% DV): The percentage of the daily recommended intake of calcium in one serving of the beverage.

Iron (% DV): The percentage of the daily recommended intake of iron in one serving of the beverage.

Caffeine (mg): The amount of caffeine in milligrams in one serving of the beverage.

0.7 Data Exploration and Data Cleaning

[ ]:
```
# basic information about the dataset
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 242 entries, 0 to 241
Data columns (total 18 columns):
 #   Column                      Non-Null Count  Dtype
---  ------                      --------------  -----
 0   Beverage_category           242 non-null    object
 1   Beverage                    242 non-null    object
 2   Beverage_prep               242 non-null    object
 3   Calories                    242 non-null    int64
 4    Total Fat (g)              242 non-null    object
 5   Trans Fat (g)               242 non-null    float64
 6   Saturated Fat (g)           242 non-null    float64
 7    Sodium (mg)                242 non-null    int64
 8    Total Carbohydrates (g)    242 non-null    int64
 9   Cholesterol (mg)            242 non-null    int64
 10   Dietary Fibre (g)          242 non-null    int64
 11   Sugars (g)                 242 non-null    int64
 12   Protein (g)                242 non-null    float64
 13  Vitamin A (% DV)            242 non-null    object
 14  Vitamin C (% DV)            242 non-null    object
 15   Calcium (% DV)             242 non-null    object
 16  Iron (% DV)                 242 non-null    object
 17  Caffeine (mg)               241 non-null    object
dtypes: float64(3), int64(6), object(9)
memory usage: 34.2+ KB
```

So, Beverage_category, Beverage, Beverage_prep, Total Fat (g) fall into categorial variable category

While Calories, Trans Fat (g), Saturated Fat (g), Sodium (mg), Total Carbohydrates (g), Cholesterol (mg), Dietary Fibre (g), Sugars (g), Protein (g) are numeric variable category

---

[ ]:
```
# checking column of starbuck dataset
df.columns
```

```
[ ]: Index(['Beverage_category', 'Beverage', 'Beverage_prep', 'Calories',
            ' Total Fat (g)', 'Trans Fat (g) ', 'Saturated Fat (g)', ' Sodium (mg)',
            ' Total Carbohydrates (g) ', 'Cholesterol (mg)', ' Dietary Fibre (g)',
            ' Sugars (g)', ' Protein (g) ', 'Vitamin A (% DV) ', 'Vitamin C (% DV)',
            ' Calcium (% DV) ', 'Iron (% DV) ', 'Caffeine (mg)'],
           dtype='object')
```

```
[ ]: # checking shape of the dataset
     df.shape
```

```
[ ]: (242, 18)
```

```
[ ]: # Checking the null value
     null_value = df.isnull().sum() * 100 / len(df)

     # Displaying the results
     print("Percentage of Null Values in Each Column:")
     print(null_value)
```

```
Percentage of Null Values in Each Column:
Beverage_category            0.000000
Beverage                     0.000000
Beverage_prep                0.000000
Calories                     0.000000
 Total Fat (g)               0.000000
Trans Fat (g)                0.000000
Saturated Fat (g)            0.000000
 Sodium (mg)                 0.000000
 Total Carbohydrates (g)     0.000000
Cholesterol (mg)             0.000000
 Dietary Fibre (g)           0.000000
 Sugars (g)                  0.000000
 Protein (g)                 0.000000
Vitamin A (% DV)             0.000000
Vitamin C (% DV)             0.000000
 Calcium (% DV)              0.000000
Iron (% DV)                  0.000000
Caffeine (mg)                0.413223
dtype: float64
```

column of **Caffein (mg)** have null value, so first we are good to fill those with some substitutes in both the columns first

```
[ ]: df['Caffeine (mg)'].fillna('unknown', inplace=True)
```

```
[ ]: # so the null values are removed
     df['Caffeine (mg)'].isnull().sum()
```

[ ]: 0

[ ]:
```
# checking the duplicated value
duplicated_val = df.drop_duplicates()
duplicated_val.count()
```

[ ]:
```
Beverage_category            242
Beverage                     242
Beverage_prep                242
Calories                     242
 Total Fat (g)               242
Trans Fat (g)                242
Saturated Fat (g)            242
 Sodium (mg)                 242
 Total Carbohydrates (g)     242
Cholesterol (mg)             242
 Dietary Fibre (g)           242
 Sugars (g)                  242
 Protein (g)                 242
Vitamin A (% DV)             242
Vitamin C (% DV)             242
 Calcium (% DV)              242
Iron (% DV)                  242
Caffeine (mg)                242
dtype: int64
```

so, there is no any duplicate rows in Dataset

[ ]:
```
df.sample(5)
```

[ ]:
```
               Beverage_category  \
240   Frappuccino® Blended Crème
144             Tazo® Tea Drinks
33       Classic Espresso Drinks
194  Frappuccino® Blended Coffee
86     Signature Espresso Drinks

                                             Beverage      Beverage_prep  \
240             Vanilla Bean (Without Whipped Cream)            Soymilk
144  Tazo® Full-Leaf Red Tea Latte (Vanilla Rooibos)            Soymilk
33           Vanilla Latte (Or Other Flavoured Latte)            Soymilk
194                    Mocha (Without Whipped Cream)         Whole Milk
86            Hot Chocolate (Without Whipped Cream)   Short Nonfat Milk

     Calories   Total Fat (g)  Trans Fat (g)   Saturated Fat (g)   Sodium (mg)  \
240       180             1.5            0.2                 0.0             0
```

```
144      80    1.5     0.2      0.0     0
33      160      4     0.5      0.0     0
194     290      4     2.5      0.1    10
86      130    1.5     1.0      0.0     5

      Total Carbohydrates (g)  Cholesterol (mg)  Dietary Fibre (g)  \
240                       160                37                  1
144                        40                14                  0
33                         95                23                  1
194                       220                61                  1
86                         70                26                  1

      Sugars (g)   Protein (g)  Vitamin A (% DV)  Vitamin C (% DV)  \
240           35           3.0                4%                0%
144           13           3.0                4%                0%
33            20           7.0               10%                0%
194           58           4.0                4%                0%
86            23           7.0               10%                0%

     Calcium (% DV)  Iron (% DV)  Caffeine (mg)
240             10%           6%              0
144             10%           6%              0
33              30%          15%             75
194             10%           8%            110
86              20%          10%             10
```

[ ]:
```
# Replace all space with underscore
df.columns = df.columns.str.replace(' ', '_')
```

[ ]:
```
# Removed some columns that were not needed for analysis
dropped_col = ['Trans_Fat_(g)_', 'Vitamin_A_(%_DV)_', 'Vitamin_C_(%_DV)',
 ↪'_Calcium_(%_DV)_', 'Iron_(%_DV)_']

df.drop(dropped_col, axis=1, inplace=True)
```

0.8 Check Unique Value for variables and doing some experiments

[ ]:
```
df.select_dtypes(include='object').nunique()
```

[ ]:
```
Beverage_category     9
Beverage             33
Beverage_prep        13
_Total_Fat_(g)       24
Caffeine_(mg)        37
dtype: int64
```

[ ]:
```
df['Beverage'].unique()
```

[ ]:
```
array(['Brewed Coffee', 'Caffè Latte',
       'Caffè Mocha (Without Whipped Cream)',
       'Vanilla Latte (Or Other Flavoured Latte)', 'Caffè Americano',
       'Cappuccino', 'Espresso', 'Skinny Latte (Any Flavour)',
       'Caramel Macchiato',
       'White Chocolate Mocha (Without Whipped Cream)',
       'Hot Chocolate (Without Whipped Cream)',
       'Caramel Apple Spice (Without Whipped Cream)', 'Tazo® Tea',
       'Tazo® Chai Tea Latte', 'Tazo® Green Tea Latte',
       'Tazo® Full-Leaf Tea Latte',
       'Tazo® Full-Leaf Red Tea Latte (Vanilla Rooibos)',
       'Iced Brewed Coffee (With Classic Syrup)',
       'Iced Brewed Coffee (With Milk & Classic Syrup)',
       'Shaken Iced Tazo® Tea (With Classic Syrup)',
       'Shaken Iced Tazo® Tea Lemonade (With Classic Syrup)',
       'Banana Chocolate Smoothie', 'Orange Mango Banana Smoothie',
       'Strawberry Banana Smoothie', 'Coffee',
       'Mocha (Without Whipped Cream)', 'Caramel (Without Whipped Cream)',
       'Java Chip (Without Whipped Cream)', 'Mocha', 'Caramel',
       'Java Chip', 'Strawberries & Crème (Without Whipped Cream)',
       'Vanilla Bean (Without Whipped Cream)'], dtype=object)
```

[ ]:
```
df['Beverage_category'].unique()
```

[ ]:
```
array(['Coffee', 'Classic Espresso Drinks', 'Signature Espresso Drinks',
       'Tazo® Tea Drinks', 'Shaken Iced Beverages', 'Smoothies',
       'Frappuccino® Blended Coffee', 'Frappuccino® Light Blended Coffee',
       'Frappuccino® Blended Crème'], dtype=object)
```

[ ]:
```
df['Beverage_prep'].unique()
```

[ ]:
```
array(['Short', 'Tall', 'Grande', 'Venti', 'Short Nonfat Milk', '2% Milk',
       'Soymilk', 'Tall Nonfat Milk', 'Grande Nonfat Milk',
       'Venti Nonfat Milk', 'Solo', 'Doppio', 'Whole Milk'], dtype=object)
```

0.9 Describe the Dataset and removing outliers

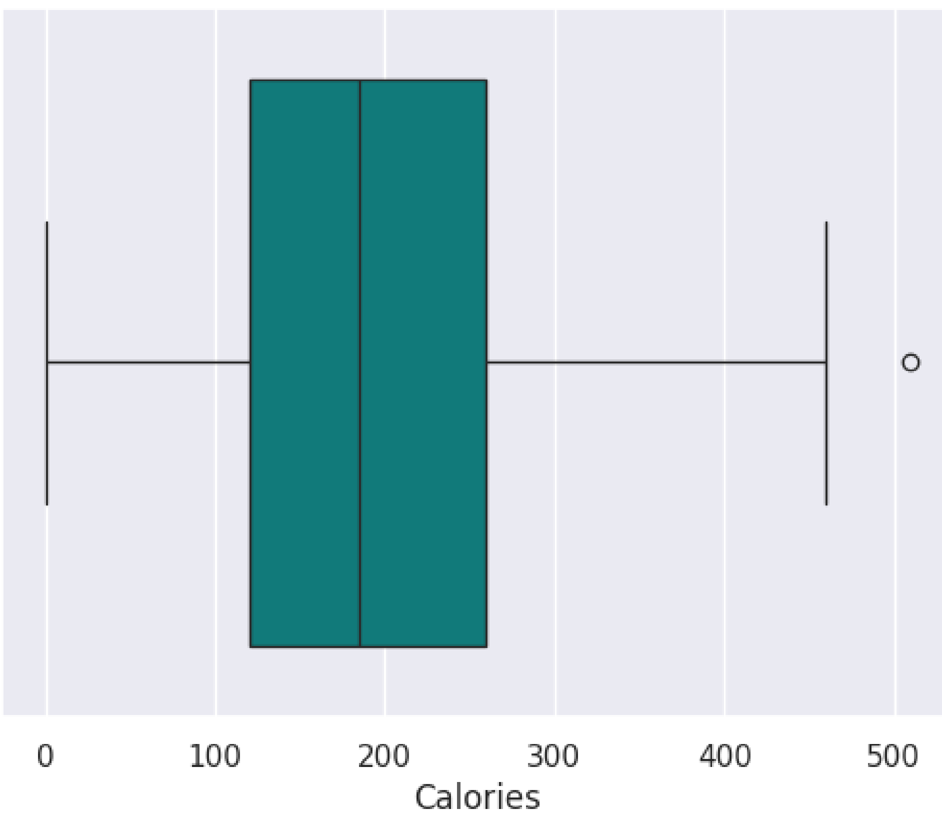
[ ]:
```
# Describe the dataset
df.describe()
```

[ ]:
```
          Calories  Saturated_Fat_(g)  _Sodium_(mg)  _Total_Carbohydrates_(g)_  \
count  242.000000         242.000000    242.000000                 242.000000
mean   193.871901           0.037603      6.363636                 128.884298
std    102.863303           0.071377      8.630257                  82.303223
min      0.000000           0.000000      0.000000                   0.000000
```

```
25%     120.000000           0.000000       0.000000                  70.000000
50%     185.000000           0.000000       5.000000                 125.000000
75%     260.000000           0.100000      10.000000                 170.000000
max     510.000000           0.300000      40.000000                 340.000000

       Cholesterol_(mg)  _Dietary_Fibre_(g)  _Sugars_(g)  _Protein_(g)_
count        242.000000          242.000000   242.000000     242.000000
mean          35.991736            0.805785    32.962810       6.978512
std           20.795186            1.445944    19.730199       4.871659
min            0.000000            0.000000     0.000000       0.000000
25%           21.000000            0.000000    18.000000       3.000000
50%           34.000000            0.000000    32.000000       6.000000
75%           50.750000            1.000000    43.750000      10.000000
max           90.000000            8.000000    84.000000      20.000000
```

Note - calories column is very important so we have to find big outliers in important columns first.

[ ]:
```
sns.boxplot(x=df['Calories'], orient='v', color='darkcyan')
plt.show()
```



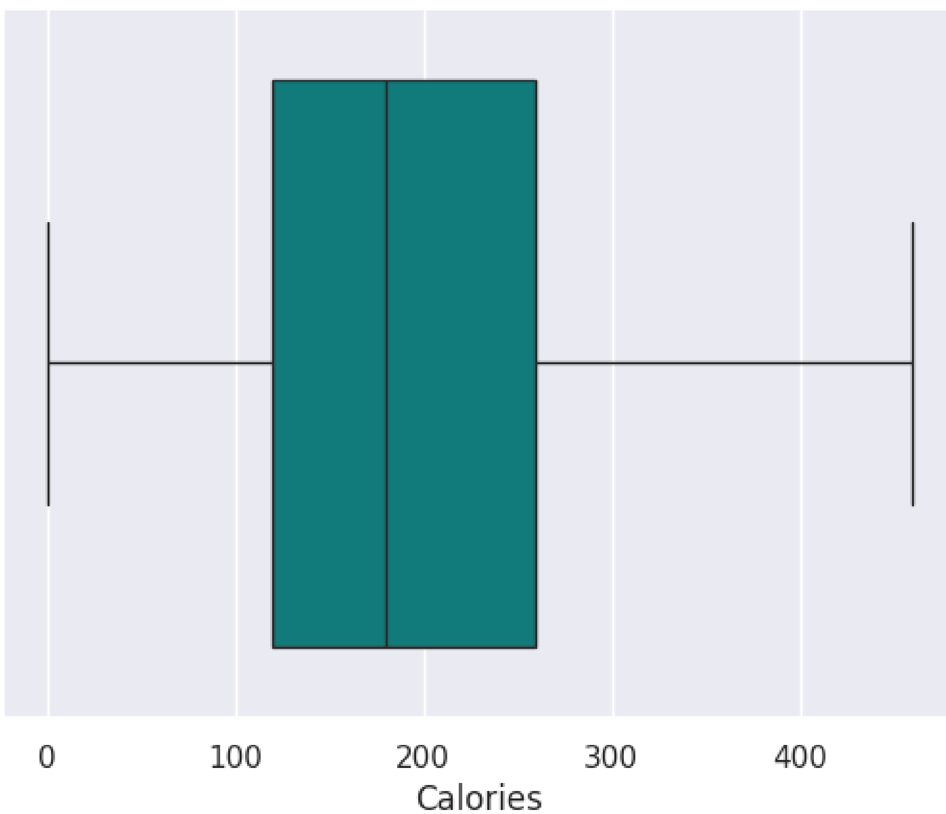
0.9.1 Using IQR Technique

[ ]:
```
# writing a outlier function for removing outliers in important columns.
def iqr_technique(df_col):
  Q1 = np.percentile(df_col, 25)
  Q3 = np.percentile(df_col, 75)
  IQR = Q3 - Q1
  lower_range = Q1 - (1.5 * IQR)
  upper_range = Q3 + (1.5 * IQR)                        # interquantile range

  return lower_range,upper_range
```

[ ]:
```
lower_bound,upper_bound = iqr_technique(df['Calories'])

df = df[(df.Calories>lower_bound) & (df.Calories<upper_bound)]
```

[ ]:
```
# so the outliers are removed from price column now check with boxplot and also␣
↪check shape of new Dataframe!
sns.boxplot(x = df['Calories'], orient='v', color='darkcyan')
print(df.shape)
```

```
(241, 13)
```



0.10 Segment Category Into Smaller Unique Value

```
[ ]: df['Beverage'].unique()
```

```
[ ]: array(['Brewed Coffee', 'Caffè Latte',
            'Caffè Mocha (Without Whipped Cream)',
            'Vanilla Latte (Or Other Flavoured Latte)', 'Caffè Americano',
            'Cappuccino', 'Espresso', 'Skinny Latte (Any Flavour)',
            'Caramel Macchiato',
            'White Chocolate Mocha (Without Whipped Cream)',
            'Hot Chocolate (Without Whipped Cream)',
            'Caramel Apple Spice (Without Whipped Cream)', 'Tazo® Tea',
            'Tazo® Chai Tea Latte', 'Tazo® Green Tea Latte',
            'Tazo® Full-Leaf Tea Latte',
            'Tazo® Full-Leaf Red Tea Latte (Vanilla Rooibos)',
            'Iced Brewed Coffee (With Classic Syrup)',
            'Iced Brewed Coffee (With Milk & Classic Syrup)',
            'Shaken Iced Tazo® Tea (With Classic Syrup)',
            'Shaken Iced Tazo® Tea Lemonade (With Classic Syrup)',
            'Banana Chocolate Smoothie', 'Orange Mango Banana Smoothie',
            'Strawberry Banana Smoothie', 'Coffee',
            'Mocha (Without Whipped Cream)', 'Caramel (Without Whipped Cream)',
            'Java Chip (Without Whipped Cream)', 'Mocha', 'Caramel',
            'Java Chip', 'Strawberries & Crème (Without Whipped Cream)',
            'Vanilla Bean (Without Whipped Cream)'], dtype=object)
```

(1) Total Beverage count in Beverage category using Bar Plot

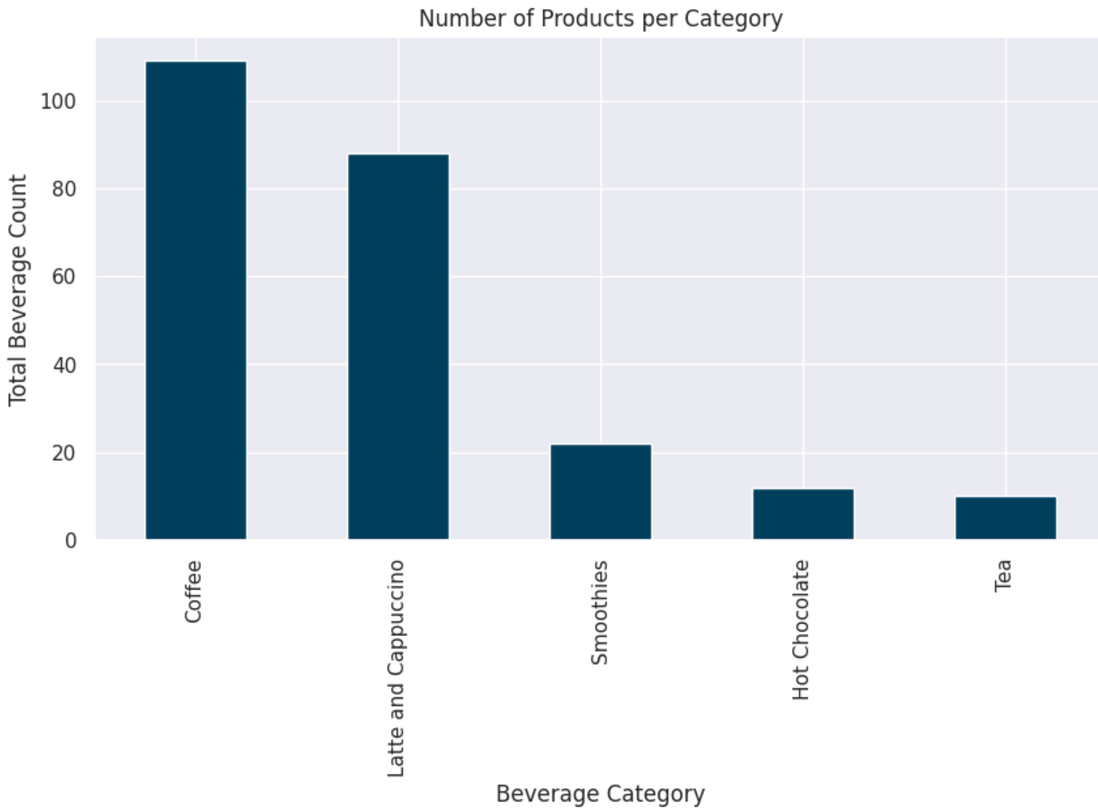
```
[ ]: def segment_cat_drink(beverage_category):
         if any(category in beverage_category for category in ['Brewed Coffee',␣
      ↪'Caffè Americano', 'Espresso', 'Iced Brewed Coffee (With Classic Syrup)',␣
      ↪'Iced Brewed Coffee (With Milk & Classic Syrup)', 'Coffee', 'Mocha (Without␣
      ↪Whipped Cream)', 'Caramel (Without Whipped Cream)', 'Java Chip (Without␣
      ↪Whipped Cream)', 'Mocha', 'Caramel', 'Java Chip']):
             return 'Coffee'
         elif any(category in beverage_category for category in ['Caffè Latte',␣
      ↪'Vanilla Latte (Or Other Flavoured Latte)', 'Skinny Latte (Any Flavour)',␣
      ↪'Cappuccino', 'Tazo® Chai Tea Latte', 'Tazo® Green Tea Latte', 'Tazo®␣
      ↪Full-Leaf Tea Latte', 'Tazo® Full-Leaf Red Tea Latte (Vanilla Rooibos)']):
             return 'Latte and Cappuccino'
         elif any(category in beverage_category for category in ['Caffè Mocha␣
      ↪(Without Whipped Cream)', 'Caramel Macchiato', 'White Chocolate Mocha␣
      ↪(Without Whipped Cream)', 'Mocha (Without Whipped Cream)', 'Caramel (Without␣
      ↪Whipped Cream)', 'Java Chip (Without Whipped Cream)', 'Mocha', 'Caramel',␣
      ↪'Java Chip']):
```

```
        return 'Mocha and Caramel Macchiato'
    elif any(category in beverage_category for category in ['Tazo® Tea',␣
↪'Shaken Iced Tazo® Tea (With Classic Syrup)', 'Shaken Iced Tazo® Tea␣
↪Lemonade (With Classic Syrup)']):
        return 'Tea'
    elif 'Hot Chocolate (Without Whipped Cream)' in beverage_category:
        return 'Hot Chocolate'
    elif any(category in beverage_category for category in ['Banana Chocolate␣
↪Smoothie', 'Orange Mango Banana Smoothie', 'Strawberry Banana Smoothie',␣
↪'Strawberries & Crème (Without Whipped Cream)', 'Vanilla Bean (Without␣
↪Whipped Cream)']):
        return 'Smoothies'
    elif 'Caramel Apple Spice (Without Whipped Cream)' in beverage_category:
        return 'Apple Spice'
    else:
        return 'Other'

# Apply the segmentation function to beverage category
df['Segment_beverage_list'] = df['Beverage'].apply(segment_cat_drink)
```

[ ]:
```
# check the segment_beverage col
df['Segment_beverage_list'].nunique()
```

[ ]: 5

[ ]:
```
plt.figure(figsize=(10,5))
df['Segment_beverage_list'].value_counts().plot(kind='bar', color='#003f5c')
plt.title('Number of Products per Category',fontsize=12)
plt.xlabel('Beverage Category',fontsize=12)
plt.ylabel('Total Beverage Count',fontsize=12)

plt.show()
```



observation –>

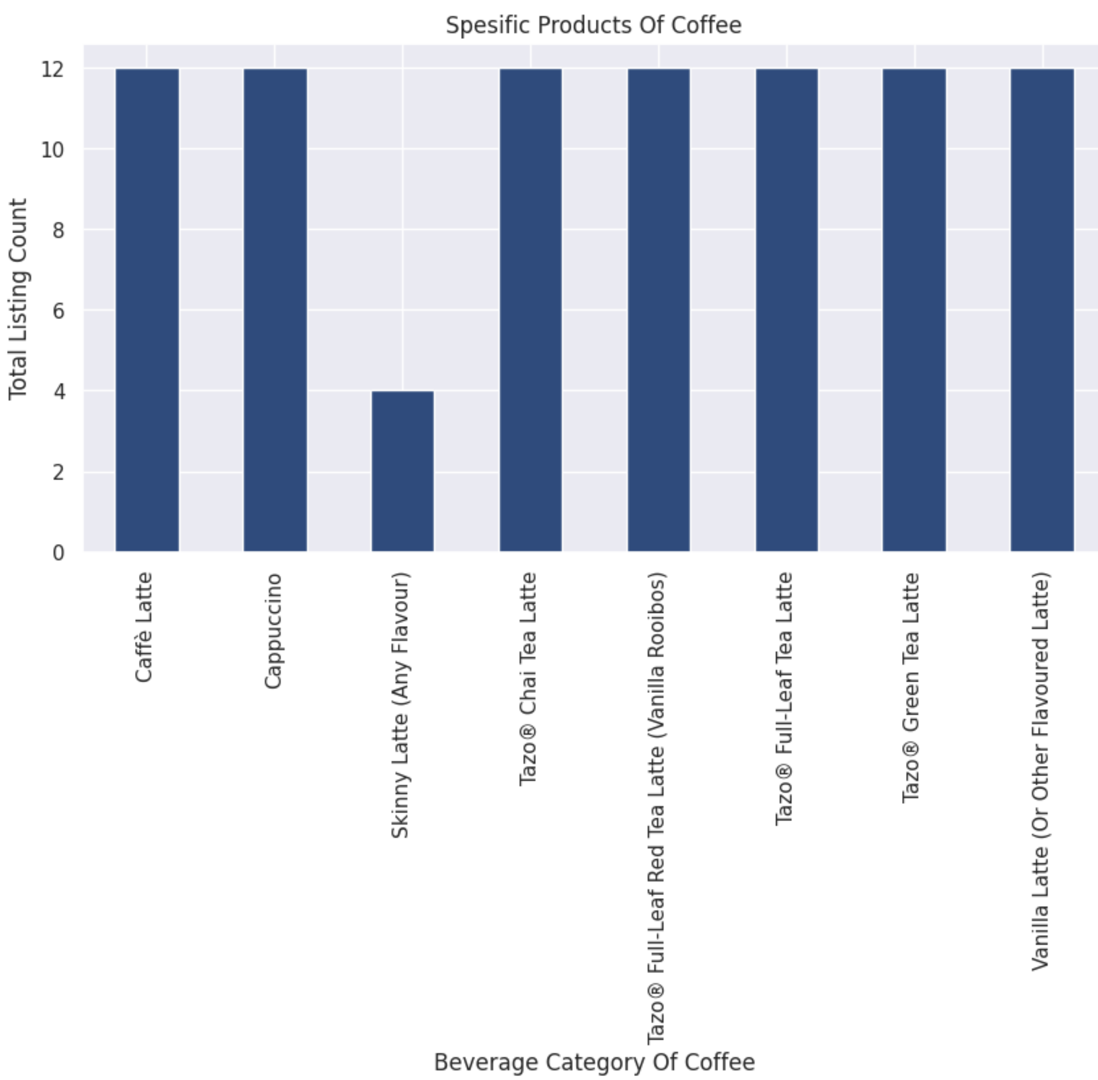
- **Coffee** have the highest number of listing on starbucks baverage, with over 100 each.
- **lattes and cappuccinos** have the second largest total after **coffee** with over 80 each
- **Smoothies, Hot Chocolate and Tea** have significantly fewer listing compared to **Coffee, Latte and Cappucino** with less than 25 each
- **Tea** has the fewest number of the listing, with only 10 each
- The distribution of listings in various beverage categories is uneven, with a concentration of listings in Coffee and Latte Cappuccino
- This can show that the types of coffee and latte cappuccino drinks at Star Buck are higher than other types of drinks, thus causing a higher concentration of properties in this coffee and latte cappuccino.

---

(2) Specific menu that becomes top ranks

[ ]:
```
# Do a visualization for beverage of spesific coffee
```

```
coffee_spesific = ['Caffè Latte', 'Vanilla Latte (Or Other Flavoured Latte)',␣
 ↪'Skinny Latte (Any Flavour)', 'Cappuccino', 'Tazo® Chai Tea Latte', 'Tazo®␣
 ↪Green Tea Latte', 'Tazo® Full-Leaf Tea Latte', 'Tazo® Full-Leaf Red Tea␣
 ↪Latte (Vanilla Rooibos)']

# Filter DataFrame based on coffee_spesific
coffee_df = df[df['Beverage'].isin(coffee_spesific)]

plt.figure(figsize=(10, 5))
coffee_df['Beverage'].value_counts().sort_index().plot(kind='bar',␣
 ↪color='#2f4b7c')
plt.title('Spesific Products Of Coffee',fontsize=12)
plt.xlabel('Beverage Category Of Coffee',fontsize=12)
plt.ylabel('Total Listing Count',fontsize=12)

plt.show()
```



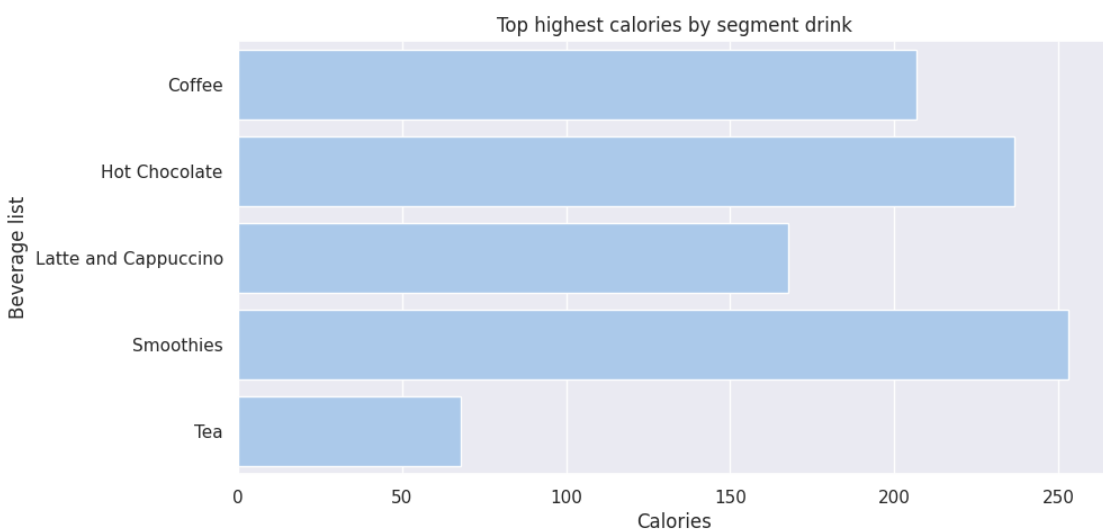
observation –>

- Of the several types of coffee drinks owned by Starbucks, it shows that 7 out of 8 coffees have the same rank level with 12 each which shows uniformity in distribution
- Meanwhile, **skinni latte** has the fewest number of listings, with only 4 each
- This suggests there is significant variation in the popularity of the menu, and further analysis is necessary to understand the factors that may influence this variation in Starbucks product distribution.

---

(3) Total calories in Starbucks beverage

[ ]:
```
drink_by_calori = df.groupby('Segment_beverage_list',␣
↪as_index=False)['Calories'].mean()

plt.figure(figsize=(10, 5))
sns.barplot(data=drink_by_calori, x='Calories', y='Segment_beverage_list')
plt.ylabel('Beverage list', fontsize=12)
plt.xlabel('Calories', fontsize=12)
plt.title('Top highest calories by segment drink', fontsize=12)
plt.show()
```



observation –>

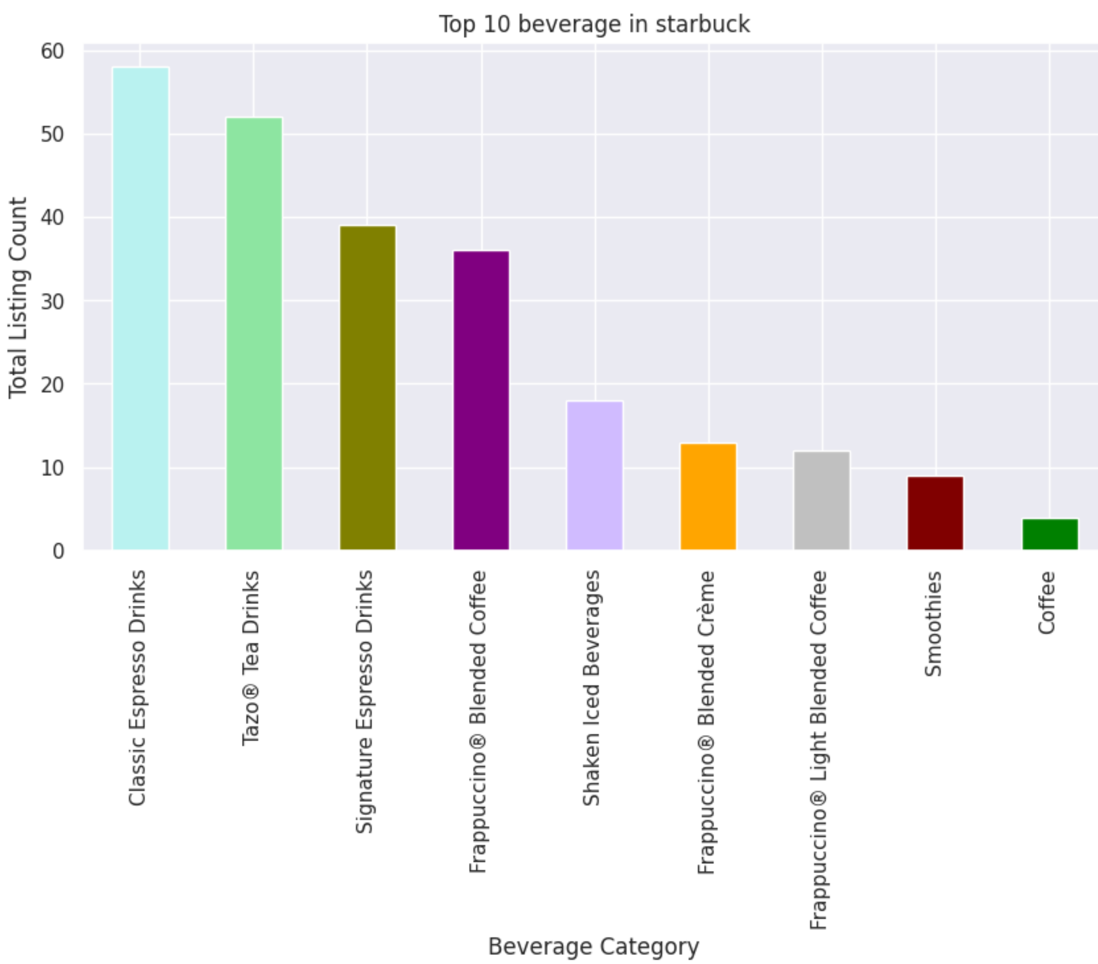
- The highest calorie type of drink at Starbucks in terms of number of listings is a **smoothies** around 250 kcal

- The type of low-calorie drink at Starbucks in terms of number of listings is **tea** with less than 70 kcal
- It can be concluded that the average drink at Starbucks has more than 50 kcal

---

(4) Top beverage menu in starbuck using bar plot

[ ]:
```
# Create a list of colors to use for the bars
colors = ['c', 'g', 'olive', 'purple', 'm', 'orange', '#C0C0C0', '#800000',␣
↪'#008000', '#000080']

plt.figure(figsize=(10, 5))
df['Beverage_category'].value_counts().plot(kind='bar', color=colors)
plt.ylabel('Total Listing Count', fontsize=12)
plt.xlabel('Beverage Category', fontsize=12)
plt.title('Top 10 beverage in starbuck', fontsize=12)
plt.show()
```



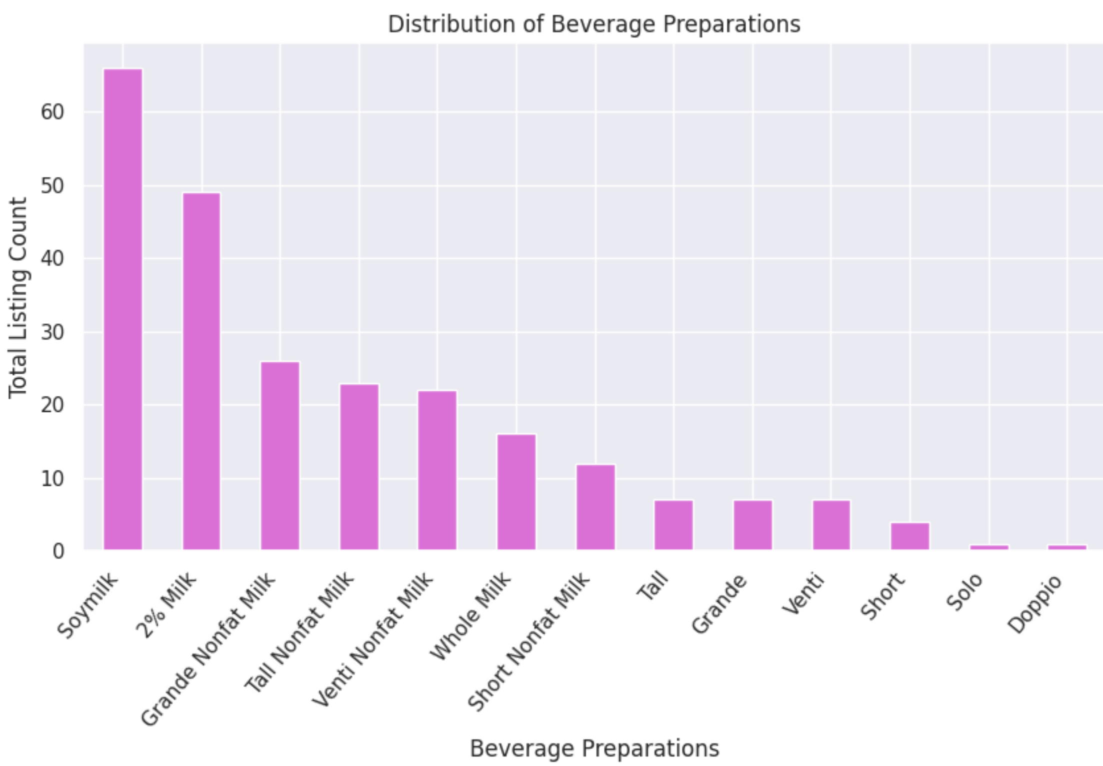
observation –>

- **Classic Espresso Drinks** This category has the largest number of products or drinks, indicating that this may be a broad or popular category with 58 each
- **Tazo® Tea Drinks** got the second highest drink category with a 52 each
- And **coffee** had the lowest number of products, perhaps reflecting that these categories may be more general and less specific in beverage variety with 4 each

---

(5) Distribution of Beverage Preparations Across Starbucks Menu

[ ]:
```
plt.figure(figsize=(10, 5))
df['Beverage_prep'].value_counts().plot(kind='bar', color='orchid')
plt.ylabel('Total Listing Count', fontsize=12)
plt.xlabel('Beverage Preparations', fontsize=12)
plt.title('Distribution of Beverage Preparations', fontsize=12)
plt.xticks(rotation=50, ha='right')
plt.show()
```



observation –>

- **Soymilk** This is the most frequently used milk alternative, with the highest count among the listed milk options with over 60 each

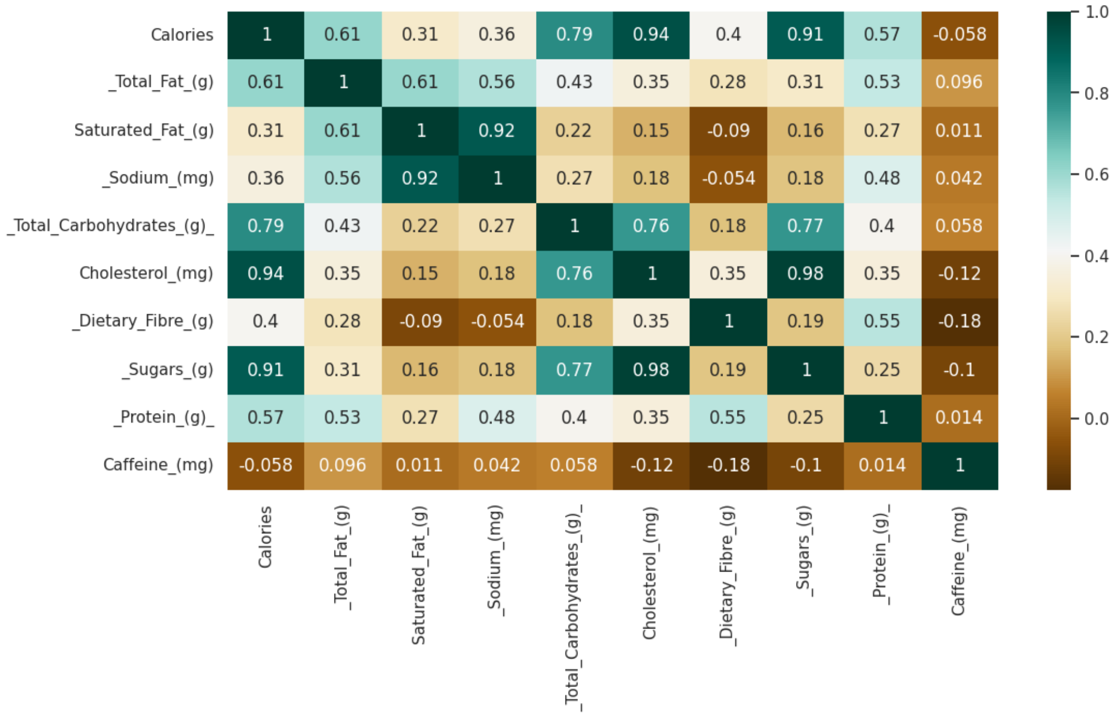
- **2% milk** is also a popular choice, with a significant count, indicating that it is commonly used in Starbucks beverages with around 48 each
- **Tall nonfat milk, venti nonfat milk, whole milk** three preparations have a balanced total listing from the other preparations with around 20 each
- Meanwhile **Solo** and **Doppio** have the lowest number of listings among the others with under 5 each

---

(6) Correlation Heatmap Visualization

[ ]:
```python
# Convert Str Value to NaN
df['Caffeine_(mg)'] = pd.to_numeric(df['Caffeine_(mg)'], errors='coerce')
df['_Total_Fat_(g)'] = pd.to_numeric(df['_Total_Fat_(g)'], errors='coerce')

# Calculate the mean of valid values
mean_caffeine = df['Caffeine_(mg)'].mean()
mean_fat = df['_Total_Fat_(g)'].mean()

# Replace NaN values with the mean
df['Caffeine_(mg)'].fillna(mean_caffeine, inplace=True)
df['_Total_Fat_(g)'].fillna(mean_fat, inplace=True)
```

[ ]:
```python
nums = df.select_dtypes(include=['number'])

plt.figure(figsize=(12, 6))
sns.heatmap(nums.corr(), annot=True, cmap='BrBG')
plt.show()
```



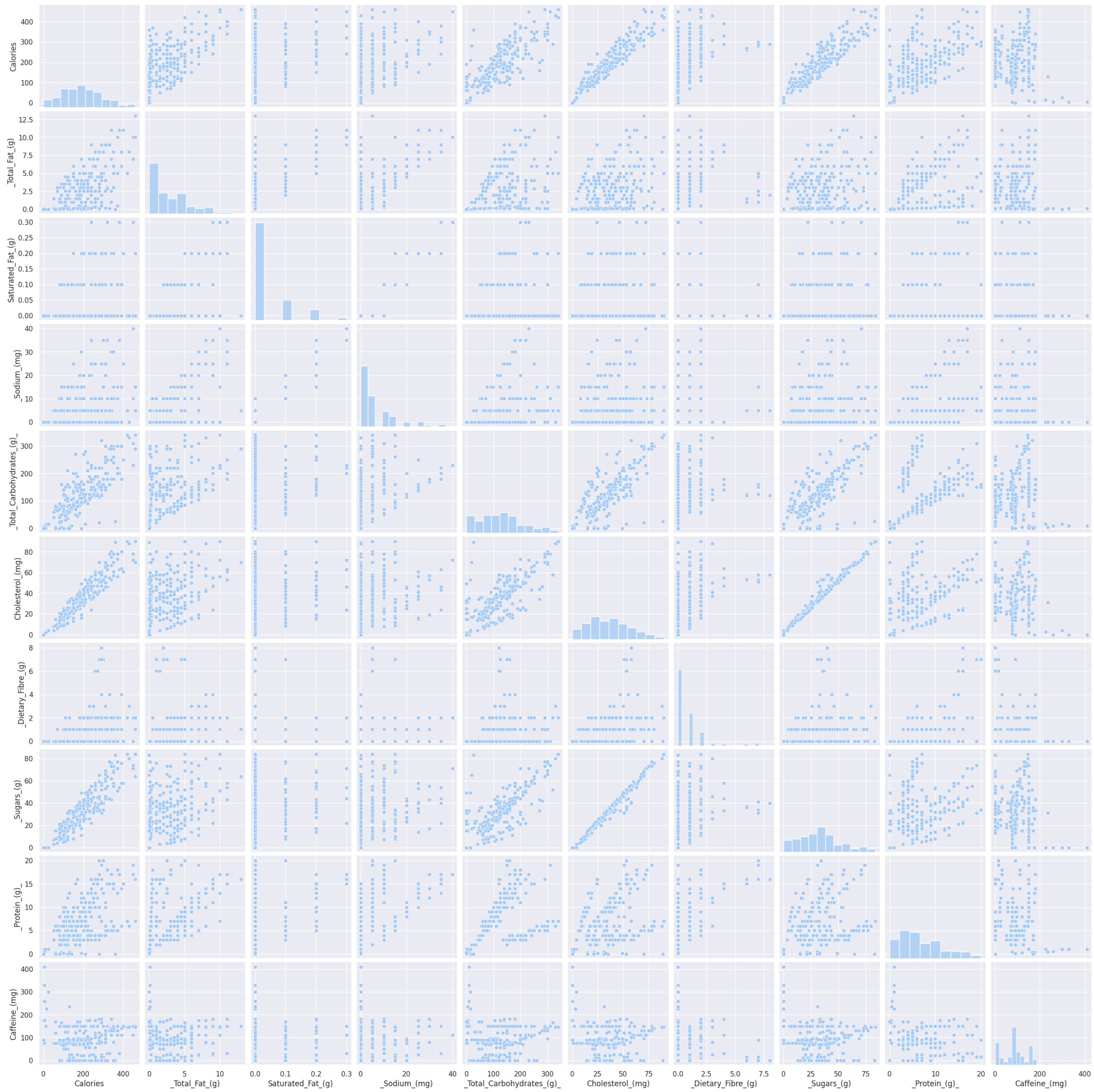
observation –>

- There is a moderate positive correlation (0.61) between the total_fat_(g) and calories columns, Total_fat_(g) also has a good correlation between column satured_fat and sodium
- We can see that the correlation between the caffeine column and the others has a very bad correlation
- There is a strong positive correlation (0.98) between the cholesterol column and the sugar column, this indicates that the two columns have the same relationship
- and there are still many columns that have a good correlation (0.94), such as between the cholesterol column and calories

---

(7) Pair Plot Visualization

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



observation –>

- A pair plot consists of multiple scatterplots arranged in a grid, with each scatterplot showing the relationship between two variables
- It can be used to visualize relationships between multiple variables and to identify patterns in the data.

0.11 BUSINESS CONCLUSION :

- Coffee drinks are still the top menu best seller and tea is a drink that is less popular with Starbucks customers

- but it can be concluded that the details of the drinks purchased have a balanced distribution
- The type of drink that has high calories is smoothies and the drink with the lowest calories is tea
- for the distribution of soymilk preparation, it is a top best seller which is often ordered by Starbucks customers

---

1 Thank You