Big_Basket_Analysis

November 3, 2024

First lets start with importing the Libraries required.

[7]:
```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
import seaborn as sns
```

Then lets import the data.

[9]:
```
Data = pd.read_csv('/content/BigBasket Products.csv')
```

Now using the head function to display first 12 rows.

[10]:
```
Data.head(12)
```

[10]:

	index	product	category	sub_category	brand
0	1	Garlic Oil - Vegetarian Capsule 500 mg	Beauty & Hygiene	Hair Care	Sri Sri Ayurveda
1	2	Water Bottle - Orange	Kitchen, Garden & Pets	Storage & Accessories	Mastercook
2	3	Brass Angle Deep - Plain, No.2	Cleaning & Household	Pooja Needs	Trm
3	4	Cereal Flip Lid Container/Storage Jar - Assort…	Cleaning & Household	Bins & Bathroom Ware	Nakoda
4	5	Creme Soft Soap - For Hands & Body	Beauty & Hygiene	Bath & Hand Wash	Nivea
5	6	Germ - Removal Multipurpose Wipes	Cleaning & Household	All Purpose Cleaners	Nature Protect
6	7	Multani Mati	Beauty & Hygiene	Skin Care	Satinance
7	8	Hand Sanitizer - 70% Alcohol Base	Beauty & Hygiene	Bath & Hand Wash	Bionova
8	9	Biotin & Collagen Volumizing Hair Shampoo + Bi…	Beauty & Hygiene	Hair Care	StBotanica
9	10	Scrub Pad - Anti- Bacterial, Regular			
10	11	Wheat Grass Powder - Raw			
11	12	Butter Cookies Gold Collection			

```
9     Cleaning & Household  Mops, Brushes & Scrubs       Scotch brite
10    Gourmet & World Food  Cooking & Baking Needs          NUTRASHIL
11    Gourmet & World Food   Chocolates & Biscuits           Sapphire

    sale_price  market_price                          type  rating  \
0        220.0         220.0                Hair Oil & Serum     4.1
1        180.0         180.0          Water & Fridge Bottles     2.3
2        119.0         250.0                 Lamp & Lamp Oil     3.4
3        149.0         176.0       Laundry, Storage Baskets     3.7
4        162.0         162.0            Bathing Bars & Soaps     4.4
5        169.0         199.0  Disinfectant Spray & Cleaners     3.3
6         58.0          58.0                       Face Care     3.6
7        250.0         250.0          Hand Wash & Sanitizers     4.0
8       1098.0        1098.0            Shampoo & Conditioner     3.5
9         20.0          20.0        Utensil Scrub-Pad, Glove     4.3
10       261.0         290.0              Flours & Pre-Mixes     4.0
11       600.0         600.0        Luxury Chocolates, Gifts     2.2

                                          description
0   This Product contains Garlic Oil that is known…
1   Each product is microwave safe (without lid), …
2   A perfect gift for all occasions, be it your m…
3   Multipurpose container with an attractive desi…
4   Nivea Creme Soft Soap gives your skin the best…
5   Stay protected from contamination with Multipu…
6   Satinance multani matti is an excellent skin t…
7   70%Alcohol based is gentle of hand leaves skin…
8   An exclusive blend with Vitamin B7 Biotin, Hyd…
9   Scotch Brite Anti- Bacterial Scrub Pad thoroug…
10  Wheatgrass is a superfood potent health food w…
11  Enjoy a tin full of delicious butter cookies m…
```

Now, Lets get the discription of the dataframe.

[11]:
```
Data.describe()
```

[11]:
```
              index     sale_price  market_price        rating
count  27555.00000   27549.000000  27555.000000  18919.000000
mean   13778.00000     334.648391    382.056664      3.943295
std     7954.58767    1202.102113    581.730717      0.739217
min        1.00000       2.450000      3.000000      1.000000
25%     6889.50000      95.000000    100.000000      3.700000
50%    13778.00000     190.320000    220.000000      4.100000
75%    20666.50000     359.000000    425.000000      4.300000
max    27555.00000  112475.000000  12500.000000      5.000000
```

This code is to see the information of the dataframe.

[12]:
```
Data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 27555 entries, 0 to 27554
Data columns (total 10 columns):
 #   Column        Non-Null Count  Dtype  
---  ------        --------------  -----  
 0   index         27555 non-null  int64  
 1   product       27554 non-null  object 
 2   category      27555 non-null  object 
 3   sub_category  27555 non-null  object 
 4   brand         27554 non-null  object 
 5   sale_price    27549 non-null  float64
 6   market_price  27555 non-null  float64
 7   type          27555 non-null  object 
 8   rating        18919 non-null  float64
 9   description   27440 non-null  object 
dtypes: float64(3), int64(1), object(6)
memory usage: 2.1+ MB
```

Since we have described the dataframe lets see if there are any missing values

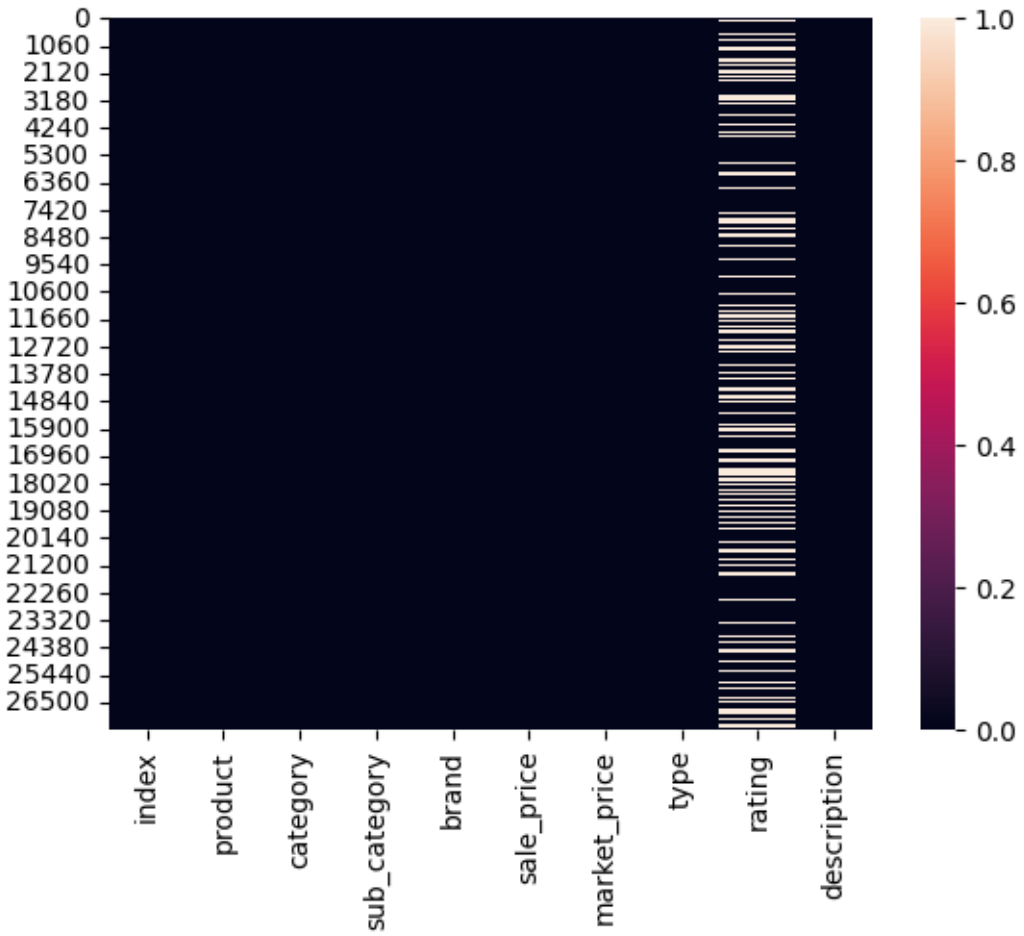
[13]:
```
Data.isnull().sum()
```

[13]:
```
index             0
product           1
category          0
sub_category      0
brand             1
sale_price        6
market_price      0
type              0
rating         8636
description     115
dtype: int64
```

This code is to draw the heatmap of the missing values in the data set.

[14]:
```
sns.heatmap(Data.isnull())
```

[14]:
```
<Axes: >
```



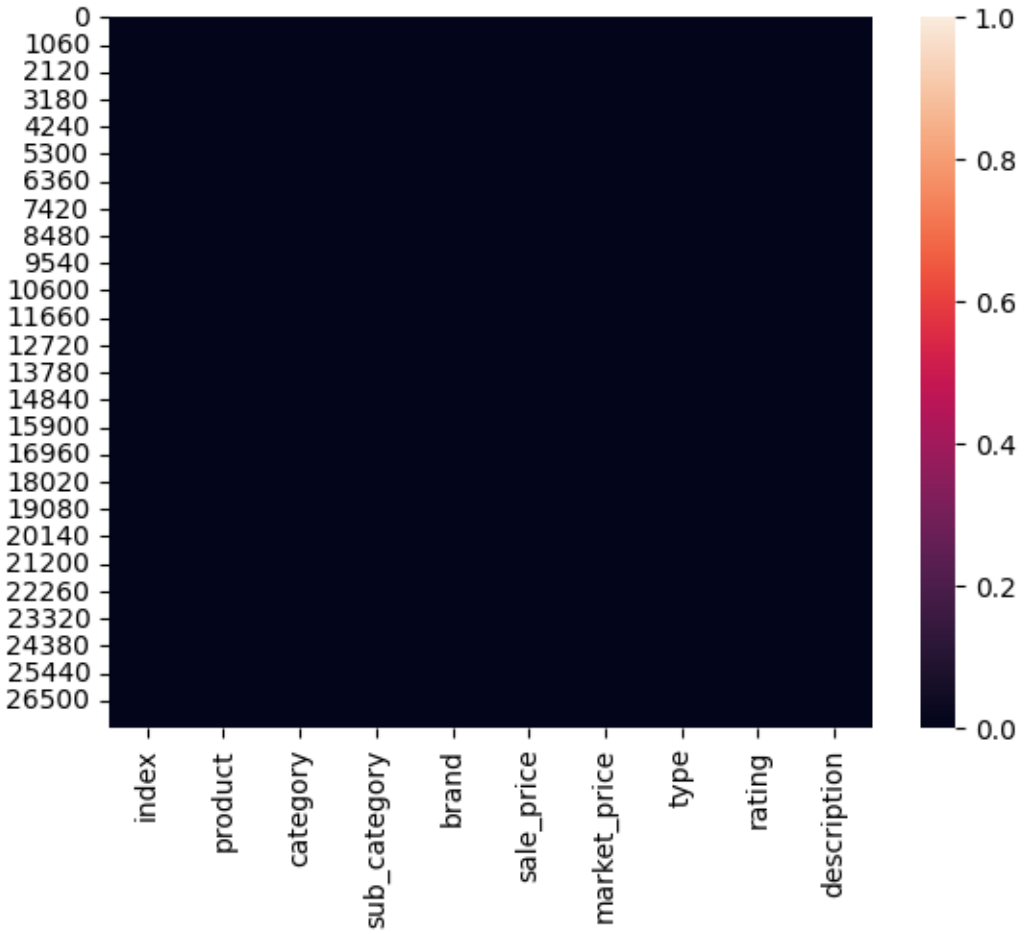
Now lets replace the missing values

[ ]:
```
# Fill missing 'sale_price' with the mean
Data['sale_price'].fillna(Data['sale_price'].mean(), inplace=True)

# Fill missing 'rating' with the mean
Data['rating'].fillna(Data['rating'].mean(), inplace=True)

# Fill missing 'brand' with 'unknown'
Data['brand'].fillna('unknown', inplace=True)

# Fill missing 'description' with 'unknown'
Data['description'].fillna('unknown', inplace=True)
```

Now lets see the heat map after replacing the missing values.

[16]:
```
sns.heatmap(Data.isnull())
```

[16]: <Axes: >



This code is to find out the top and least sold products.

[20]:
```
# Group by product and count the occurrences
product_counts = Data['product'].value_counts()

# Find the top 5 and bottom 5 products
top_products = product_counts.nlargest(5)
bottom_products = product_counts.nsmallest(5)

# Print the results
print("Top 5 Products by Count:")
print(top_products)
print("\nBottom 5 Products by Count:")
bottom_products
```

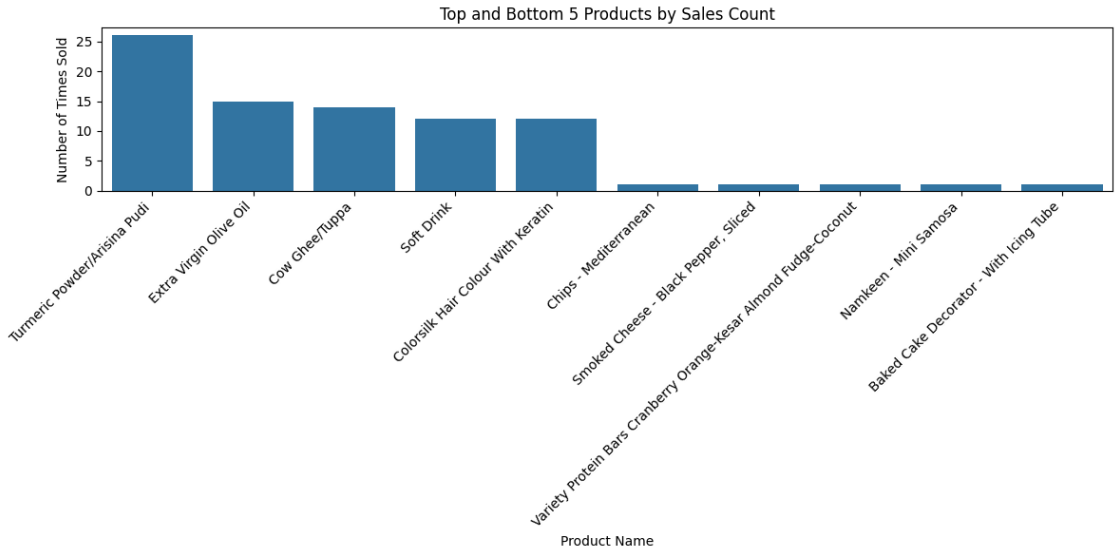
```
Top 5 Products by Count:
```

```
product
Turmeric Powder/Arisina Pudi          26
Extra Virgin Olive Oil                15
Cow Ghee/Tuppa                        14
Soft Drink                            12
Colorsilk Hair Colour With Keratin    12
Name: count, dtype: int64

Bottom 5 Products by Count:
```

[20]:
```
product
Chips - Mediterranean                                                  1
Smoked Cheese - Black Pepper, Sliced                                   1
Variety Protein Bars Cranberry Orange-Kesar Almond Fudge-Coconut       1
Namkeen - Mini Samosa                                                  1
Baked Cake Decorator - With Icing Tube                                 1
Name: count, dtype: int64
```

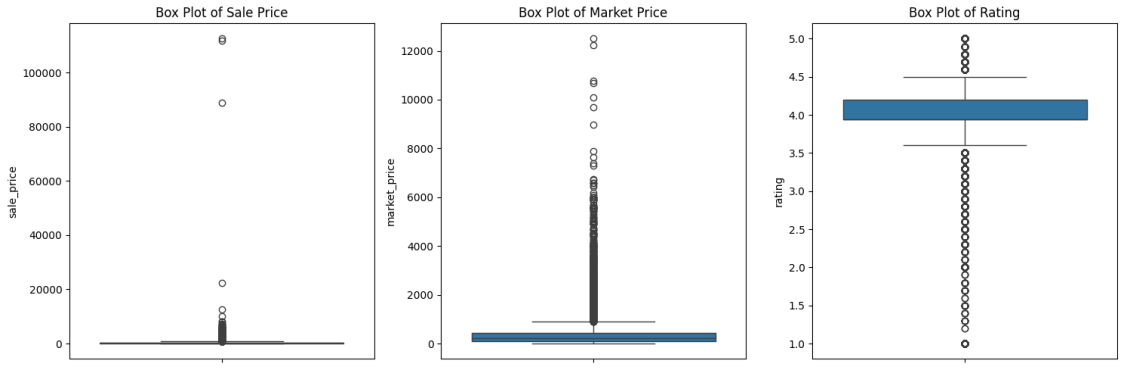
This is the Visual representation of the top and lest sold products.

[21]:
```
# Combine top and bottom products for visualization
top_bottom_products = pd.concat([top_products, bottom_products])

# Create the bar plot
plt.figure(figsize=(12, 6))
sns.barplot(x=top_bottom_products.index, y=top_bottom_products.values)
plt.xticks(rotation=45, ha='right')  # Rotate x-axis labels for readability
plt.xlabel("Product Name")
plt.ylabel("Number of Times Sold")
plt.title("Top and Bottom 5 Products by Sales Count")
plt.tight_layout()
plt.show()
```



Now lets find out the outlies in the dataframe.

[22]:
```
# Box plot for Sale price, Market price and ratings
plt.figure(figsize=(15, 5))

plt.subplot(1, 3, 1)
sns.boxplot(y=Data['sale_price'])
plt.title('Box Plot of Sale Price')

plt.subplot(1, 3, 2)
sns.boxplot(y=Data['market_price'])
plt.title('Box Plot of Market Price')

plt.subplot(1, 3, 3)
sns.boxplot(y=Data['rating'])
plt.title('Box Plot of Rating')

plt.tight_layout()
plt.show()
```



As we can see there are outliers in the sale price, market price and the ratings. So, lets replace it with the mean.

[23]:
```
# Calculate the IQR for 'sale_price'
Q1_sale = Data['sale_price'].quantile(0.25)
Q3_sale = Data['sale_price'].quantile(0.75)
IQR_sale = Q3_sale - Q1_sale

# Define bounds for outliers
lower_bound_sale = Q1_sale - 1.5 * IQR_sale
upper_bound_sale = Q3_sale + 1.5 * IQR_sale
```

```
# Replace outliers with the mean
Data['sale_price'] = np.where((Data['sale_price'] < lower_bound_sale) |␣
 ↪(Data['sale_price'] > upper_bound_sale),
                               Data['sale_price'].mean(), Data['sale_price'])
```

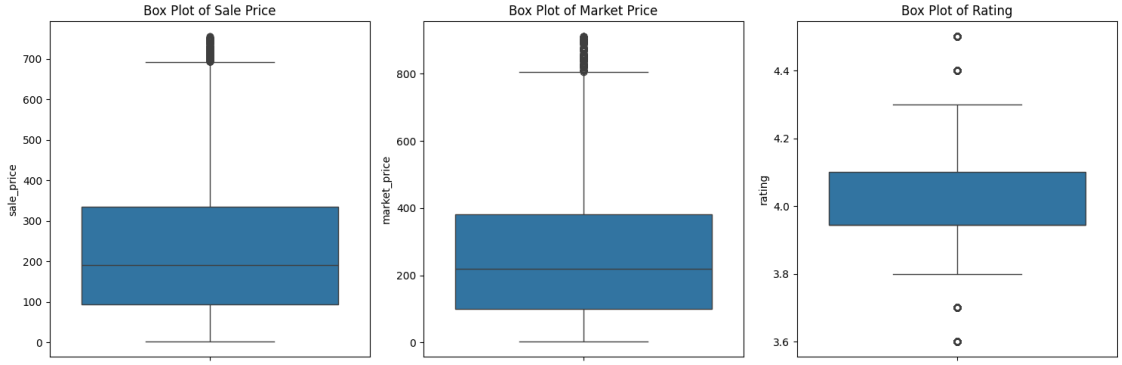
[24]:
```
# Calculate the IQR for 'market_price'
Q1_market = Data['market_price'].quantile(0.25)
Q3_market = Data['market_price'].quantile(0.75)
IQR_market = Q3_market - Q1_market

# Define bounds for outliers
lower_bound_market = Q1_market - 1.5 * IQR_market
upper_bound_market = Q3_market + 1.5 * IQR_market

# Replace outliers with the mean
Data['market_price'] = np.where((Data['market_price'] < lower_bound_market) |␣
 ↪(Data['market_price'] > upper_bound_market),
                               Data['market_price'].mean(), Data['market_price'])
```

[25]:
```
# Calculate the IQR for 'rating'
Q1_rating = Data['rating'].quantile(0.25)
Q3_rating = Data['rating'].quantile(0.75)
IQR_rating = Q3_rating - Q1_rating

# Define bounds for outliers
lower_bound_rating = Q1_rating - 1.5 * IQR_rating
upper_bound_rating = Q3_rating + 1.5 * IQR_rating

# Replace outliers with the mean
Data['rating'] = np.where((Data['rating'] < lower_bound_rating) |␣
 ↪(Data['rating'] > upper_bound_rating),
                               Data['rating'].mean(), Data['rating'])
```

Since we have removed the outliers lets see the box plot now.

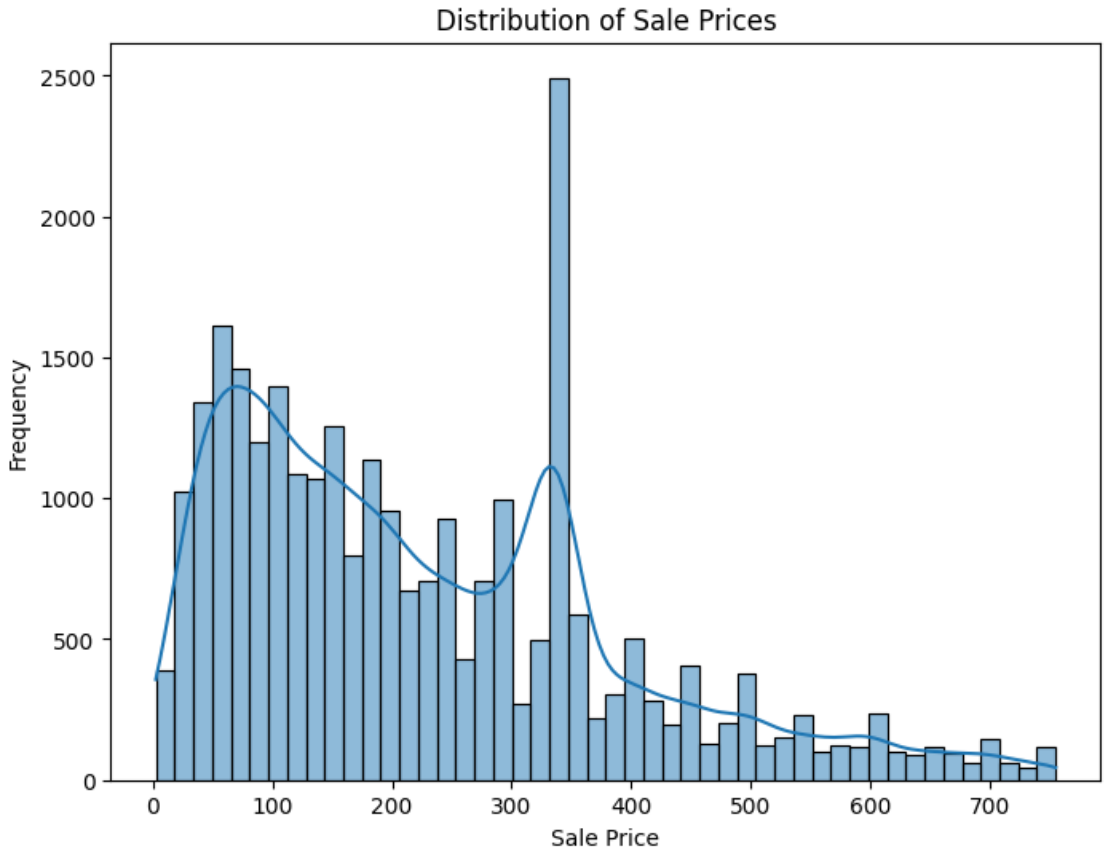
[26]:
```
plt.figure(figsize=(15, 5))

plt.subplot(1, 3, 1)
sns.boxplot(y=Data['sale_price'])
plt.title('Box Plot of Sale Price')

plt.subplot(1, 3, 2)
sns.boxplot(y=Data['market_price'])
plt.title('Box Plot of Market Price')

plt.subplot(1, 3, 3)
sns.boxplot(y=Data['rating'])
```

```
plt.title('Box Plot of Rating')

plt.tight_layout()
plt.show()
```



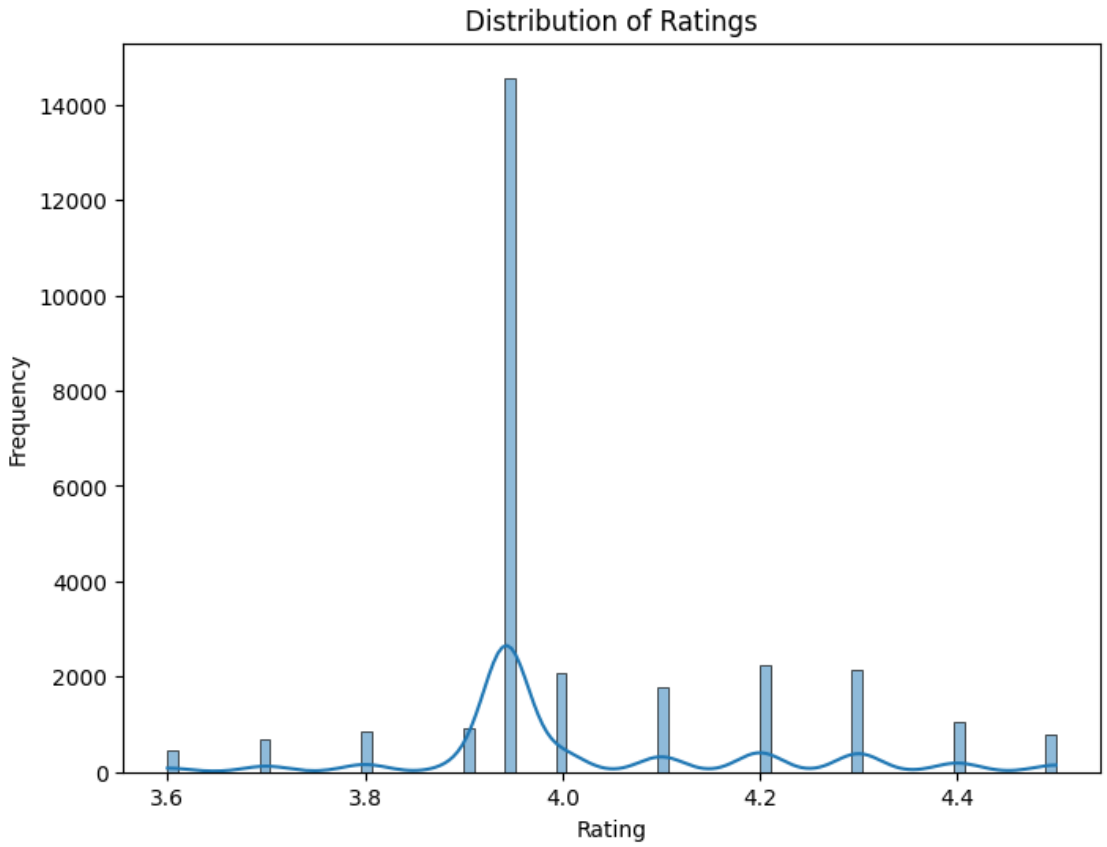
Now lets Draw some plots and visualizations for more insights in the data.

[31]:
```
# Sale Price Distribution: The majority of products have sale prices within a␣
 ↪certain range, with some outliers observed.
# The histogram provides insights into the price points at which most products␣
 ↪are concentrated.

# Distribution of Sale Prices
plt.figure(figsize=(8, 6))
sns.histplot(Data['sale_price'], kde=True)
plt.title('Distribution of Sale Prices')
plt.xlabel('Sale Price')
plt.ylabel('Frequency')
plt.show()
```

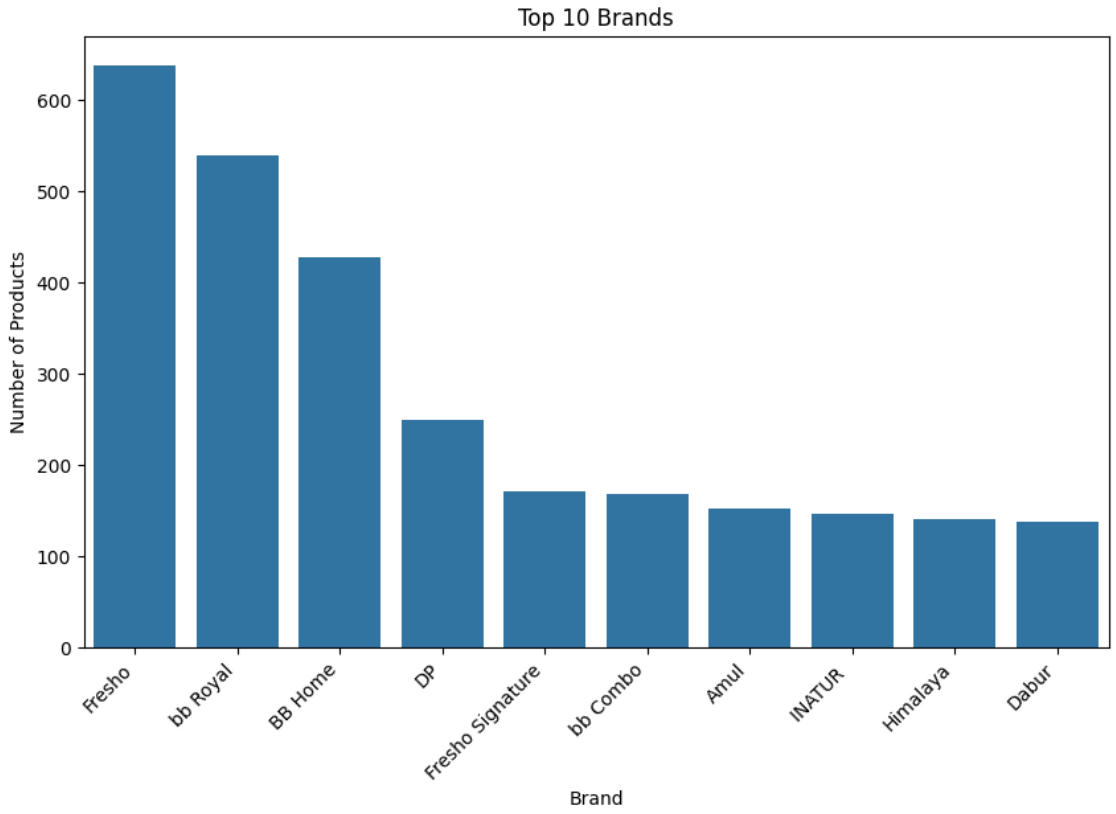


[35]:
```
# Sale Price vs. Market Price: The scatter plot reveals the relationship␣
 ↪between the original market price and the discounted sale price.
# A clear positive correlation is expected, although discounts might cause␣
 ↪variation. The plot can help identify pricing strategies.

# Relationship between Sale Price and Market Price
plt.figure(figsize=(8, 6))
sns.scatterplot(x='market_price', y='sale_price', data=Data)
plt.title('Sale Price vs. Market Price')
plt.xlabel('Market Price')
plt.ylabel('Sale Price')
plt.show()
```

[33]:
```
# Rating Distribution: The distribution of ratings gives an understanding of␣
 ↪customer satisfaction with the products.
# A higher concentration towards the higher rating signifies better customer␣
 ↪satisfaction.

# Distribution of Ratings
plt.figure(figsize=(8, 6))
sns.histplot(Data['rating'], kde=True)
plt.title('Distribution of Ratings')
plt.xlabel('Rating')
plt.ylabel('Frequency')
plt.show()
```



[34]:
```
# Top Brands: The top 10 brands visualization helps in understanding which␣
 ↪brands are more prevalent within the dataset.
# This can be used to assess popularity and market presence.

# Top 10 Brands by Number of Products
top_10_brands = Data['brand'].value_counts().nlargest(10)
plt.figure(figsize=(10, 6))
sns.barplot(x=top_10_brands.index, y=top_10_brands.values)
plt.xticks(rotation=45, ha='right')
plt.title('Top 10 Brands')
plt.xlabel('Brand')
plt.ylabel('Number of Products')
plt.show()
```



1 Final Summary

This analysis focuses on a dataset of BigBasket products. It starts by loading the data and looking at its structure with descriptive statistics and data type information. To handle missing values, the mean is used for numerical columns like 'sale_price' and 'rating', while 'unknown' is used for missing categorical values like 'brand' and 'description'. Next, the analysis identifies the top and bottom 5 most frequently sold products, using a bar chart to visualize these findings. Outliers in 'sale_price', 'market_price', and 'rating' are found using box plots and replaced with the mean of those columns. Finally, the analysis includes various visualizations: histograms to show the distribution of sale prices and ratings, a scatter plot to compare sale prices with market prices, and a bar chart to highlight the top 10 most frequent brands in the dataset. These visualizations provide valuable insights into pricing strategies, customer satisfaction, and brand popularity within BigBasket's product catalog. This comprehensive approach helps in understanding the data better and making informed decisions.