AussieFoods

Section 1

First and the most, importing necessary library for data manipulation analysis, creating data visualization and to create aesthetically pleasing and informative data visualization. As following:

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

2. 1 Import food supplier in Python

The data name: foodsupplier is importing as following:

```
In [2]:
   ...:
   ...: file_path = "/Users/sujanchokhal/Desktop/Food_dataset.xlsx"
   ...: data = pd.read_excel(file_path)
   ...: print(data.head())
   Item_id      Item  ...  product feedback              other feedback
0     1122   Oranges  ...             agree        I like your product
1     1123    Apples  ...          disagree  I don't like this product
2     1124   Bananas  ...    strongly agree  prodict not receive ontime
3     1125   Lettuce  ...  strongly diagree             receive ontime
4     1126  Tomatoes  ...             agree        I like your product

[5 rows x 18 columns]

In [3]:
```

The data has been imported into python using the pandas library. And it displayed the first few rows.

2.2 Perform basic analysis

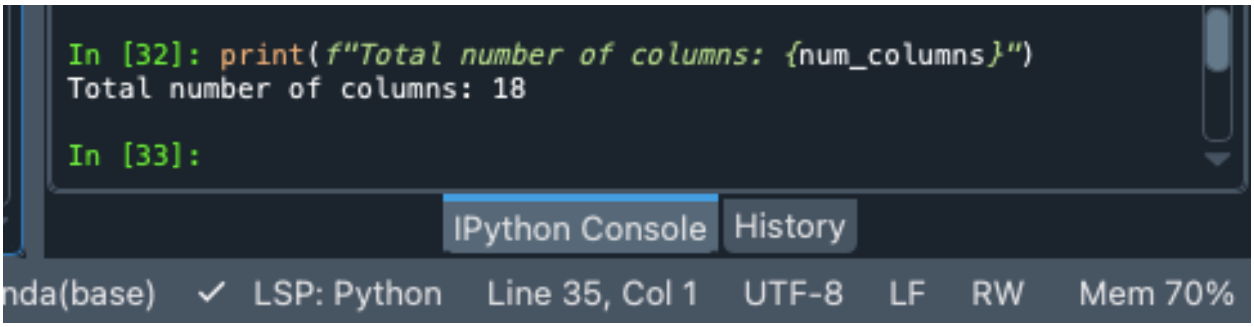
After imported the data, data looks correct with no missing value, duplicate records and unexpected characters. here, need to cleaning the data for clean column names and strip spaces which converts all column names to lowercase with .str.lower().

```
data.columns = data.columns.str.strip().str.lower()
```

The following performances will be done on the basis of this data.

2.2.1 Total number of columns in the data set

The total number of columns in the data set is 18. this also shown in while importing the dataset. The following coding gives the result.



And the columns names are:

```
In [32]: print(f"Total number of columns: {num_columns}")
Total number of columns: 18

In [33]: print(data.columns)
Index(['Item_id', 'Item', 'Store ', 'suplier', 'suplier location ', 'quantity',
       'Item sold', 'purchase price', 'total cost', 'sold price ', 'Discounts',
       'total sales ', 'Date', 'Month Number', 'Month Name', 'Year',
       'product feedback', 'other feedback'],
      dtype='object')
```

2.2.2 Total profit company make each year

The total profit company make each year, it begins by extracting the year from the 'date' column and then computes the profit for each entry by subtracting the 'total cost' from 'total sales'.

The output of total profit for each year are as follows:

2020 94814.400735

2021 483390.220588

2022 491879.725000

```
...:
...:
...: data.columns = data.columns.str.strip()
...: data['Profit'] = data['total sales'] - data['total cost']
...:
...: # Calculate Profit
...: data['Profit'] = data['total sales'] - data['total cost']
...:
...: # Create a 'Year' column if it doesn't exist
...: data['Year'] = data['Date'].dt.year
...:
...: # Group by 'Year' and calculate the total profit each year
...: yearly_profit = data.groupby('Year')['Profit'].sum()
...:
...: # Print the total profit for each year
...: print(yearly_profit)
...:
...:
...:
...: print(data.columns)
Year
2020     94814.400735
2021    483390.220588
2022    491879.725000
Name: Profit, dtype: float64
Index(['Item_id', 'Item', 'Store', 'suplier', 'suplier location', 'quantity',
```

IPython Console History

2.2.3 Calculate total profit based one each state/location

```
In [100]:
     ...:
     ...:
     ...: location_profit = data.groupby('suplier location')['Profit'].sum()
     ...:
     ...: # Printing the total profit for each supplier location
     ...: print("Total profit based on each supplier location:")
     ...: print(location_profit)
Total profit based on each supplier location:
suplier location
NSW                              130720.777941
Northern Territory               105652.372059
Queensland                       217075.363971
South Australia                  135775.691176
Tasmania                         176907.819485
The Australian Capital Territory 128669.389706
VIC                                1501.550000
Western Australia                173781.381985
Name: Profit, dtype: float64
```

The code groups the data by the 'suplier location' column and then calculates the sum of profits for each location.the output as follows:

suplier location	Profit
NSW	130721
Northern Territory	105652
Queensland	217075
South Australia	135776
Tasmania	176908
The Australian Capital Territory	128669
VIC	1501.55
Western Australia	173781

This above output shows the total profit calculated for each supplier location, providing a summary of profits grouped by location. The highest profit states is Queensland with 217075 means while lowest total profit state is VIC with 1501.55.

2.2.4 Identify monthly sales based on state

```
In [9]: state_monthly = data.groupby("Month Name")["total sales"].sum()
   ...: print(state_monthly)
Month Name
April         7.545379e+05
August        4.802047e+05
December      1.287499e+06
February      5.039217e+05
January       4.693395e+05
July          5.277811e+05
June          8.323886e+05
March         3.547378e+05
May           3.938378e+05
November      7.690192e+05
October       1.513277e+06
September     8.529769e+05
Name: total sales, dtype: float64

In [10]:
```

code efficiently showing monthly sales, aggregating the data based on month and supplier location, resulting in a more concise representation of monthly sales for each location.

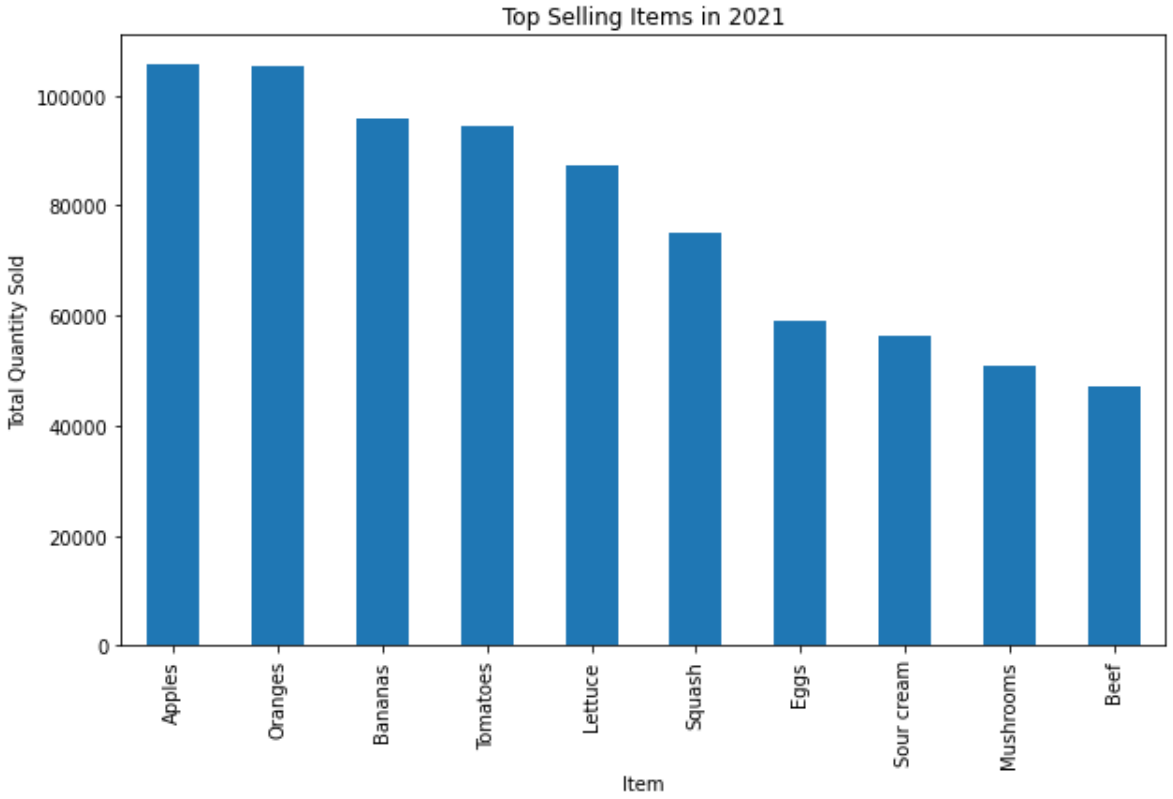
```
sales'].sum()

In [12]:
    ...: print("Monthly sales based on state:")
    ...: print(monthly_sales_by_state)
Monthly sales based on state:
Month Name  suplier location
April       NSW                                201917.957353
            Northern Territory                  97804.139706
            Queensland                         100116.100000
            South Australia                     90633.692647
            Tasmania                            32447.038235
                                                       ...
September   Queensland                         203665.130515
            South Australia                     40163.062868
            Tasmania                           170221.341544
            The Australian Capital Territory    79187.547426
            Western Australia                  156252.857353
Name: total sales, Length: 85, dtype: float64

In [13]:
```

The above showcases representing the total sales for each month, offering a clear overview of sales trends throughout the year.

1.3 identify and visualise items sold in the year 2021.

The code groups the filtered data by 'Item' and calculates the total quantity sold for each item in 2021 and Group by 'Item' and calculate the total quantity sold for each item.

```
In [38]:
    ...:
    ...: data_2021 = data[data['Year'] == 2021]
    ...:
    ...: #the code groups the filtered data by 'Item' and calculate
the total quantity sold for each item in 2021,
    ...: # Group by 'Item' and calculate the total quantity sold fo
each item
    ...: item_sales_2021 = data_2021.groupby('Item')['Item
sold'].sum().sort_values(ascending=False)
    ...:
    ...: # Visualize the top-selling items
    ...: #the top-selling items in 2021 are visualized in a bar
chart, displaying the top 10 items
    ...: plt.figure(figsize=(10, 6))
    ...: item_sales_2021.head(10).plot(kind='bar', title='Top
Selling Items in 2021')
    ...: plt.xlabel('Item')
    ...: plt.ylabel('Total Quantity Sold')
    ...: plt.show()
```



The above graph represents items sold in the year 2021. It created a bar chart that visually represent the top 10 selling items in 2021. Whereas, the x-axis shows the item such as apple, oranges, Bananas, eggs many more items and y-axis shows the total quantity sold from 0 -100000. The result is series of item names and their total quantity sold. The highest top selling which are more than 100000 are apples, oranges, bananas. Eggs, sour cream, mushroom and beer are in the point of 60000 quantity sold. Overall, this graph helps to identify trends and preferences within the dataset.

The codes came up with the filter the dataset to 2021 only. And showing the output. From the sorting results, it gives to identify the top-selling items with the highest quantities sold at the top of the list.

To sum up, the code and output effectively filter, summarizes and visualizes the sales quantity and item for the year 2021 which helps to enabling quick insight into the best-performing items during 2021.

3. Section 2

3.1 identify state which has more positive feedback

```
In [18]:
    ...:
    ...:
    ...: feedback_mapping = {
    ...:     'Extremely Poor': 1,
    ...:     'Disagree': 2,
    ...:     'Agree': 3,
    ...:     'Strongly Disagree': 4,
    ...:     'Strongly Agree': 5
    ...: }
    ...:
    ...: # Map the feedback categories to numerical scores
    ...: data['feedback_score'] = data['product feedback'].map(feedback_mapping)
    ...:
    ...: # Group by 'suplier location' and calculate the mean feedback score for each location
    ...: location_feedback = data.groupby('suplier location')['feedback_score'].mean()
    ...:
    ...: # Finding the location with the highest average feedback score
    ...: state_with_highest_feedback = location_feedback.idxmax()
    ...: highest_feedback_score = location_feedback.max()
    ...:
    ...: print(f"The state with the most positive product feedback is {state_with_highest_feedback} with an average feedback score of
{highest_feedback_score:.2f}.")
The state with the most positive product feedback is NSW  with an average feedback score of 1.00.

In [19]:
```

The above coding and output show the state with the most positive product feedback is NSW with an average feedback score of 1.00. from the dataset, grouping by 'supplier location' and calculate the mean product feedback for each location and it define a mapping of feedback categories to numerical scores. This code maps qualitative feedback categories to numerical scores using a predefined mapping. The mapping textual feedback categories in 'Extremely Poor': 1, 'Disagree': 2, 'Agree': 3, 'Strongly Disagree': 4 'Strongly Agree': 5 with transformation of qualitative feedback quantitatively. The coding included the supplier location with the highest average feedback score. As a result. NSW is the most positive product feedback which means it indicates the level of customer satisfaction associated with products or services provided by suppliers in New South Wales.

Overall, this output helps to understand the customer satisfaction according to location and provides better data-driver decision with improvement and highlight of location.

3.2 Does product price varies city to city

```
In [40]:
    ...: city_item_prices = data.groupby(['suplier location', 'Item'])['sold price'].mean().unstack()
    ...:
    ...: # Visualize the price variations using a heatmap
    ...: plt.figure(figsize=(12, 8))
    ...: plt.imshow(city_item_prices, cmap='viridis', aspect='auto')
    ...: plt.colorbar(label='Mean Price')
    ...: plt.title('Price Variations by City and Item')
    ...: plt.xlabel('Item')
    ...: plt.ylabel('City')
    ...: plt.xticks(range(len(city_item_prices.columns)), city_item_prices.columns, rotation=90)
    ...: plt.yticks(range(len(city_item_prices.index)), city_item_prices.index)
    ...:
    ...: plt.show()
    ...:
    ...: # Discuss the results
    ...: print("Price variations by city and item:")
    ...: print(city_item_prices)
Price variations by city and item:
Item                              Apples   Bananas  ...  Wild Salmon     Yogurt
suplier location                                    ...
NSW                            10.004157  6.117892  ...     6.200000   9.271429
Northern Territory              5.075000  9.150000  ...    10.918750  12.127206
Queensland                      4.842857  7.839372  ...    10.398836   3.616667
South Australia                 8.895466  9.206801  ...     3.950000   4.950000
Tasmania                        5.793061  4.075000  ...     8.414286  16.904044
The Australian Capital Territory  2.575000  8.450000  ...     6.700000  10.003002
VIC                                  NaN       NaN  ...          NaN        NaN
Western Australia               9.777206  6.645466  ...     4.200000   4.575000

[8 rows x 17 columns]

In [41]:
```

The above coding and output show the variation in product prices by city (supplier location). Here, Group by 'supplier location' and 'Item' to calculate the mean price for each item in each city. The major two factors and supplier location and item. It also calculates the mean of sold price for each item. The above table includes the item of apple, bananas. Beef, celery, cheese, cottage cheese, cucumber, eggs, lettuce, milk, mushrooms, oranges, sour cream, squash, tomatoes, wild salmon and yogurt. And supplier location includes the NSW, Northern Territory, Queensland, South Australia, Tasmania, The Australian Capital Territory, VIC and Western Australia.

The output illustrates the different items have varying average prices in different cities. For examples apple has high prices in NSW and Western Australia. For Bananas Tasmania is the lowest one. For wild salmon higher prices in Northern Territory and most of the item prices has higher in NSW. This information can be helpful for policymakers, consumer and can impact on supply, demand, local market and other economic factors.

Overall, this visualization shows the how product price varies from city to city and can be informative for stakeholder regarding price differences.

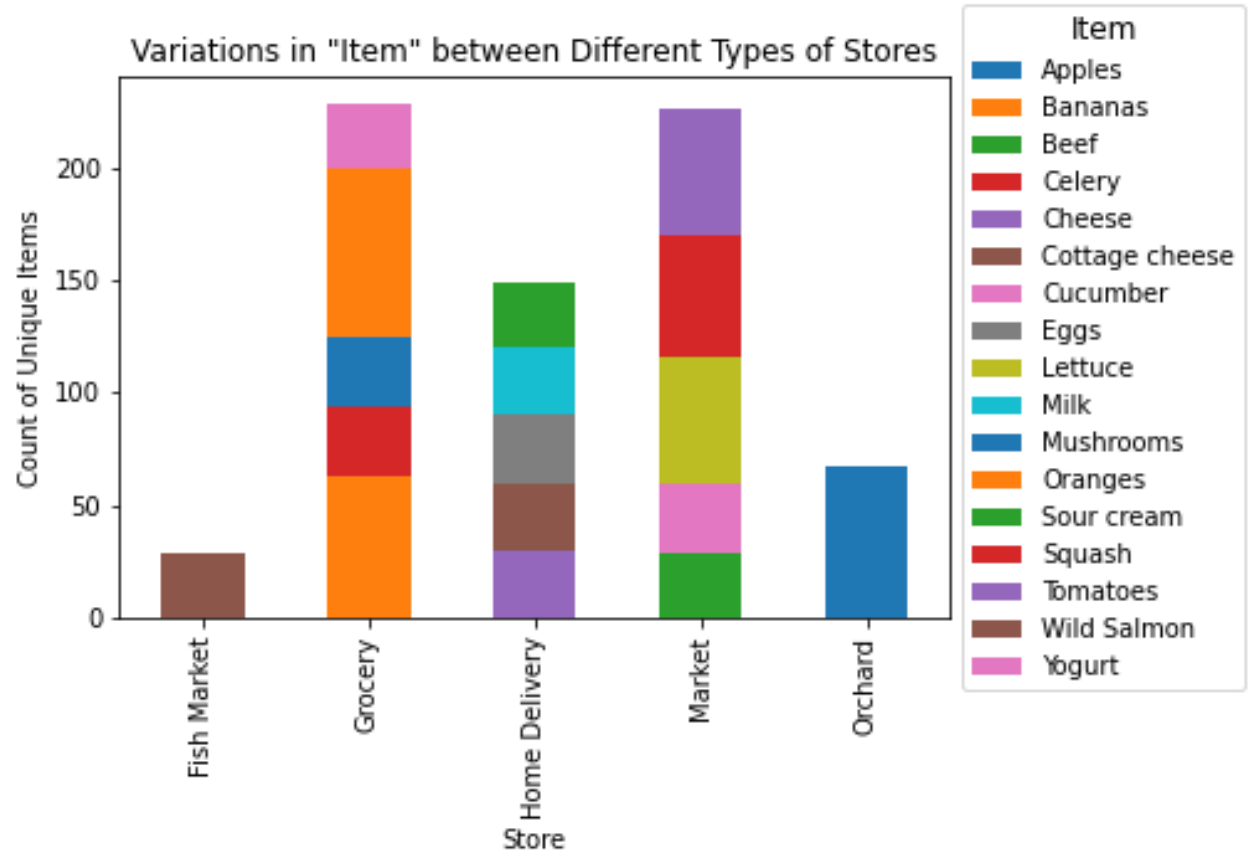
4. Section3

4.1 Does variable "Item" vary between different types of Stores

```
...:
...:
...:
...: store_item_counts = data.groupby(['store', 'item']).size().unstack(fill_value=0)
...:
...: # Visualize the variations in 'item' between different types of stores
...: plt.figure(figsize=(12, 8))
...: store_item_counts.plot(kind='bar', stacked=True)
...: plt.title('Variations in "Item" between Different Types of Stores')
...: plt.xlabel('Store')
...: plt.ylabel('Count of Unique Items')
...: plt.xticks(rotation=90)
...: plt.legend(title='Item', title_fontsize='12', loc='center left', bbox_to_anchor=(1, 0.5))
...:
...: plt.show()
...:
...: # Discuss the results
...: print("Variations in 'Item' between Different Types of Stores:")
...: print(store_item_counts)
<Figure size 864x576 with 0 Axes>
Variations in 'Item' between Different Types of Stores:
item           Apples  Bananas  Beef  ...  Tomatoes  Wild Salmon  Yogurt
store                                 ...
Fish Market         0        0     0  ...         0           29       0
Grocery             0       63     0  ...         0            0      29
Home Delivery       0        0     0  ...         0            0       0
Market              0        0    29  ...        56            0       0
Orchard            67        0     0  ...         0            0       0

[5 rows x 17 columns]

In [44]:
```

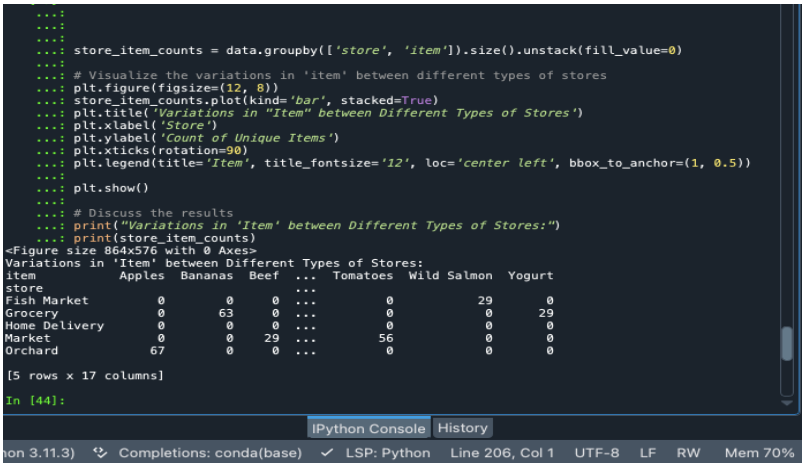
IPython Console | History

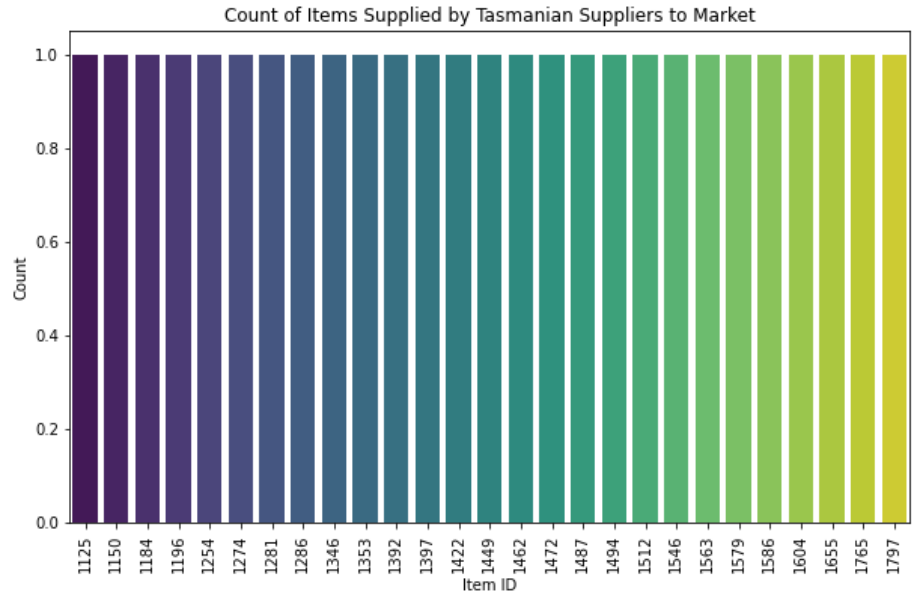
ıon 3.11.3) | Completions: conda(base) | LSP: Python | Line 206, Col 1 | UTF-8 | LF | RW | Mem 70%



The above graph shows the variation in item between different types of stores. The code first organizes the data by grouping it based on two factors: store and item. The x-axis represents the store types and height of the stacked bars shows the count of unique item available. The grocery store offers more 200 unique items mean while fish market and orchard has less unique items with wild salmon and mushrooms respectively. Overall, this analysis helps to understand how product offerings vary among different store types with revealing patterns and differences in product availability.

4.2 Identify item ids where the supplier is in Tasmania supply to Market

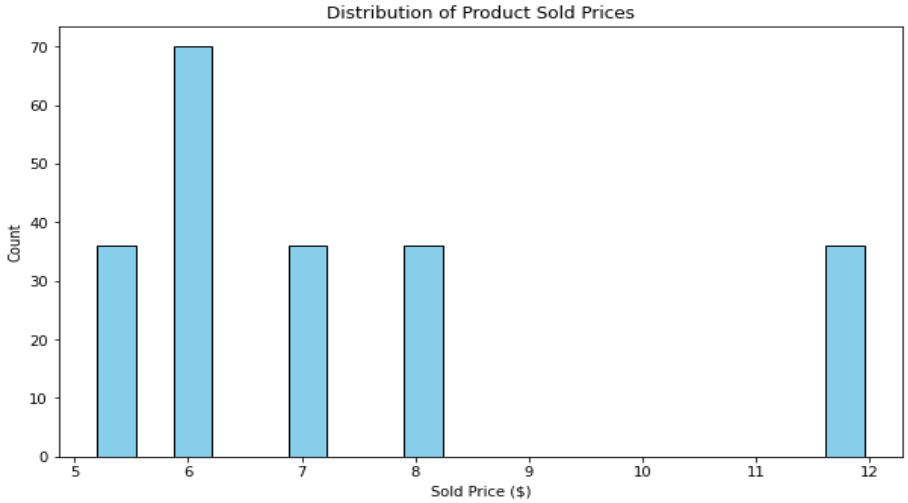




The above coding and chart represent the supplier is in Tasmania to the market. The plots show the distribution of items showing how many of each unique item has been supplied which has the approx. same supply. This will help for the inventory manager and store operations.

4.3 Identify items' product sold prices between 5.00$ and 12.00$

```
histogram,
    ...: plt.figure(figsize=(10, 6))
    ...: plt.hist(filtered_data['sold price'], bins=20, color='skyblue', edgecolor='black')
    ...: plt.title('Distribution of Product Sold Prices')
    ...: plt.xlabel('Sold Price ($)')
    ...: plt.ylabel('Count')
    ...: plt.show()
    ...:
    ...: # Discuss the results
    ...: print("Summary statistics of product sold prices:")
    ...: print(filtered_data['sold price'].describe())
Summary statistics of product sold prices:
count    214.000000
mean       7.337850
std        2.255987
min        5.200000
25%        5.950000
50%        6.950000
75%        7.950000
max       11.950000
Name: sold price, dtype: float64

In [46]:
```



The above coding and graph represent the sales prices of the items with a price range of $5 to $12. 25% of the products were sold at a price less than or equal to $4.95. 50% sold at a price less than $6.95. and 75% of the products sold at a price less than $7.95. On the graph, sold price $6 is the most popular one whereas, remaining are in the same price ratio. Overall, this output helps to understand the central tendency and key percentiles of the data.