Supermarket_Sales_Analysis

December 24, 2024

1 Supermarket sales data Analysis

[29]:

```
#import required libraries
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
```

[30]:

```
df=pd.read_csv('Supermart Grocery Sales - Retail Analytics Dataset (1).csv')
```

[31]:

```
df.head()
```

[31]:

```
  Order ID Customer Name          Category      Sub Category         City  \
0      OD1        Harish      Oil & Masala           Masalas      Vellore
1      OD2         Sudha         Beverages     Health Drinks  Krishnagiri
2      OD3       Hussain       Food Grains      Atta & Flour   Perambalur
3      OD4       Jackson  Fruits & Veggies  Fresh Vegetables   Dharmapuri
4      OD5       Ridhesh       Food Grains   Organic Staples         Ooty

   Order Date Region  Sales  Discount  Profit       State
0  11-08-2017  North   1254      0.12  401.28  Tamil Nadu
1  11-08-2017  South    749      0.18  149.80  Tamil Nadu
2  06-12-2017   West   2360      0.21  165.20  Tamil Nadu
3  10-11-2016  South    896      0.25   89.60  Tamil Nadu
4  10-11-2016  South   2355      0.26  918.45  Tamil Nadu
```

[32]:

```
df.shape
```

[32]:

```
(9994, 11)
```

[33]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 9994 entries, 0 to 9993
Data columns (total 11 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   Order ID       9994 non-null   object
 1   Customer Name  9994 non-null   object
 2   Category       9994 non-null   object
 3   Sub Category   9994 non-null   object
 4   City           9994 non-null   object
 5   Order Date     9994 non-null   object
 6   Region         9994 non-null   object
 7   Sales          9994 non-null   int64
 8   Discount       9994 non-null   float64
 9   Profit         9994 non-null   float64
 10  State          9994 non-null   object
dtypes: float64(2), int64(1), object(8)
memory usage: 859.0+ KB
```

[34]:

```
df.describe()
```

[34]:

```
             Sales     Discount       Profit
count  9994.000000  9994.000000  9994.000000
mean   1496.596158     0.226817   374.937082
std     577.559036     0.074636   239.932881
min     500.000000     0.100000    25.250000
25%    1000.000000     0.160000   180.022500
50%    1498.000000     0.230000   320.780000
75%    1994.750000     0.290000   525.627500
max    2500.000000     0.350000  1120.950000
```

[35]:

```
df.isnull().sum()
```

[35]:

```
Order ID         0
Customer Name    0
Category         0
Sub Category     0
City             0
Order Date       0
Region           0
Sales            0
Discount         0
Profit           0
State            0
dtype: int64
```

[36]:

```
df.duplicated()
```

[36]:

```
0       False
1       False
2       False
3       False
4       False
        …
```

```
9989    False
9990    False
9991    False
9992    False
9993    False
Length: 9994, dtype: bool
```

```
[37]: df.drop_duplicates(inplace=True)
```

```
[38]: df.dropna()
```

```
[38]:      Order ID Customer Name          Category       Sub Category         City  \
      0         OD1        Harish      Oil & Masala            Masalas      Vellore
      1         OD2         Sudha         Beverages      Health Drinks  Krishnagiri
      2         OD3       Hussain       Food Grains       Atta & Flour   Perambalur
      3         OD4       Jackson   Fruits & Veggies  Fresh Vegetables   Dharmapuri
      4         OD5       Ridhesh       Food Grains    Organic Staples         Ooty
      …       …           …                …               …          …
      9989   OD9990        Sudeep  Eggs, Meat & Fish              Eggs      Madurai
      9990   OD9991          Alan            Bakery          Biscuits  Kanyakumari
      9991   OD9992          Ravi       Food Grains              Rice         Bodi
      9992   OD9993          Peer      Oil & Masala            Spices   Pudukottai
      9993   OD9994        Ganesh       Food Grains      Atta & Flour  Tirunelveli

            Order Date Region  Sales  Discount  Profit       State
      0     11-08-2017  North   1254      0.12  401.28  Tamil Nadu
      1     11-08-2017  South    749      0.18  149.80  Tamil Nadu
      2     06-12-2017   West   2360      0.21  165.20  Tamil Nadu
      3     10-11-2016  South    896      0.25   89.60  Tamil Nadu
      4     10-11-2016  South   2355      0.26  918.45  Tamil Nadu
      …           …    …    …      …     …       …
      9989  12/24/2015   West    945      0.16  359.10  Tamil Nadu
      9990  07-12-2015   West   1195      0.26   71.70  Tamil Nadu
      9991  06-06-2017   West   1567      0.16  501.44  Tamil Nadu
      9992  10/16/2018   West   1659      0.15  597.24  Tamil Nadu
      9993   4/17/2018   West   1034      0.28  165.44  Tamil Nadu

      [9994 rows x 11 columns]
```

```
[39]: df['Order Date']=pd.to_datetime(df['Order Date'],infer_datetime_format=True,␣
        ↪errors='coerce')
```

```
C:\Users\ARCHANA CHOUGALE\AppData\Local\Temp\ipykernel_15440\2595410413.py:1:
UserWarning: The argument 'infer_datetime_format' is deprecated and will be
removed in a future version. A strict version of it is now the default, see
https://pandas.pydata.org/pdeps/0004-consistent-to-datetime-parsing.html. You
can safely remove this argument.
  df['Order Date']=pd.to_datetime(df['Order Date'],infer_datetime_format=True,
errors='coerce')
```

[40]:
```
df.info()
```

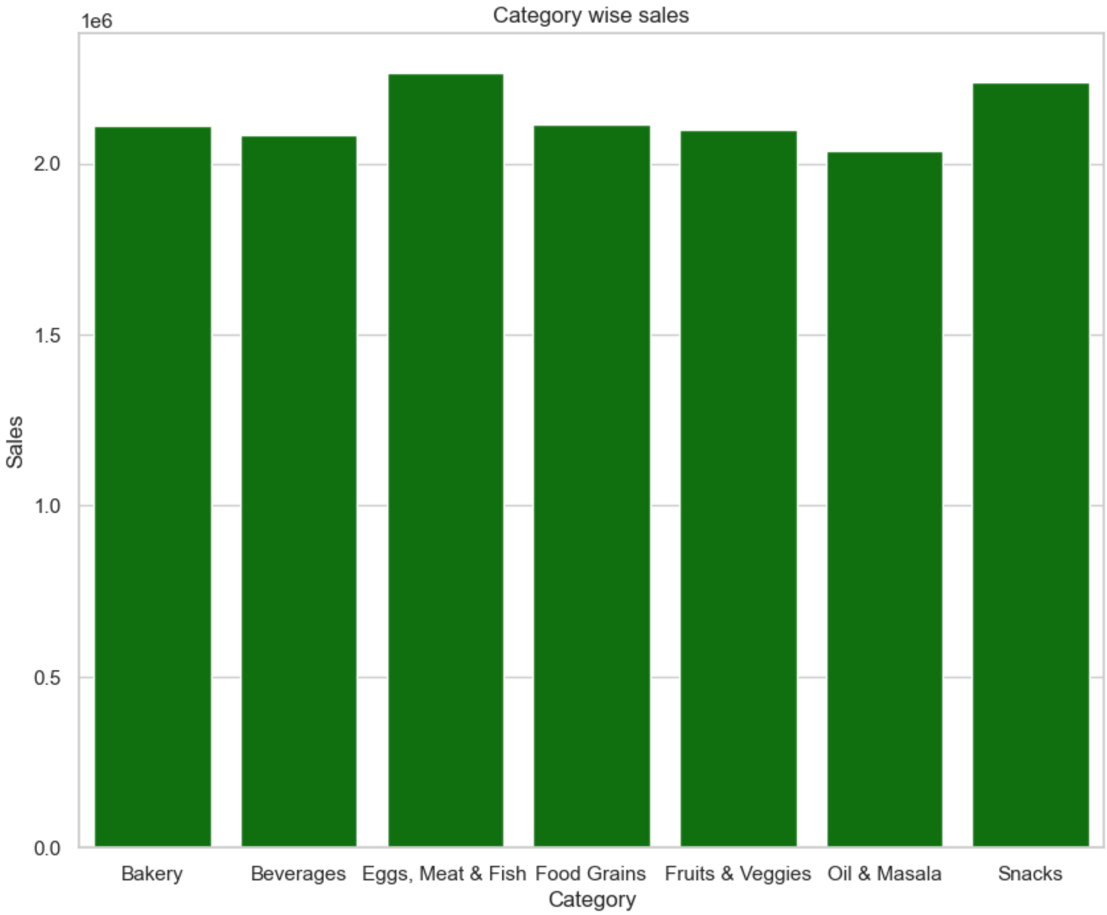
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 9994 entries, 0 to 9993
Data columns (total 11 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   Order ID       9994 non-null   object
 1   Customer Name  9994 non-null   object
 2   Category       9994 non-null   object
 3   Sub Category   9994 non-null   object
 4   City           9994 non-null   object
 5   Order Date     4042 non-null   datetime64[ns]
 6   Region         9994 non-null   object
 7   Sales          9994 non-null   int64
 8   Discount       9994 non-null   float64
 9   Profit         9994 non-null   float64
 10  State          9994 non-null   object
dtypes: datetime64[ns](1), float64(2), int64(1), object(7)
memory usage: 859.0+ KB
```

[41]:
```
df.head()
```

[41]:
```
  Order ID Customer Name          Category      Sub Category         City  \
0      OD1        Harish      Oil & Masala           Masalas      Vellore
1      OD2         Sudha         Beverages     Health Drinks  Krishnagiri
2      OD3       Hussain       Food Grains      Atta & Flour   Perambalur
3      OD4       Jackson  Fruits & Veggies  Fresh Vegetables   Dharmapuri
4      OD5       Ridhesh       Food Grains   Organic Staples         Ooty

  Order Date Region  Sales  Discount  Profit       State
0 2017-11-08  North   1254      0.12  401.28  Tamil Nadu
1 2017-11-08  South    749      0.18  149.80  Tamil Nadu
2 2017-06-12   West   2360      0.21  165.20  Tamil Nadu
3 2016-10-11  South    896      0.25   89.60  Tamil Nadu
4 2016-10-11  South   2355      0.26  918.45  Tamil Nadu
```

2 category wise sales

[42]:
```
category_sales=df.groupby(['Category'])['Sales'].sum().reset_index()
category_sales.sort_values(by='Sales')
plt.figure(figsize=(10,8))
sns.barplot(data=category_sales,x='Category',y='Sales',color='green')
plt.xlabel('Category')
plt.ylabel('Sales')
plt.title('Category wise sales')
plt.show()
```



Observations:

1.Eggs,meat and fish category has the highest sales over all categories.

2.Oil and Masala has lowest sales among all categories

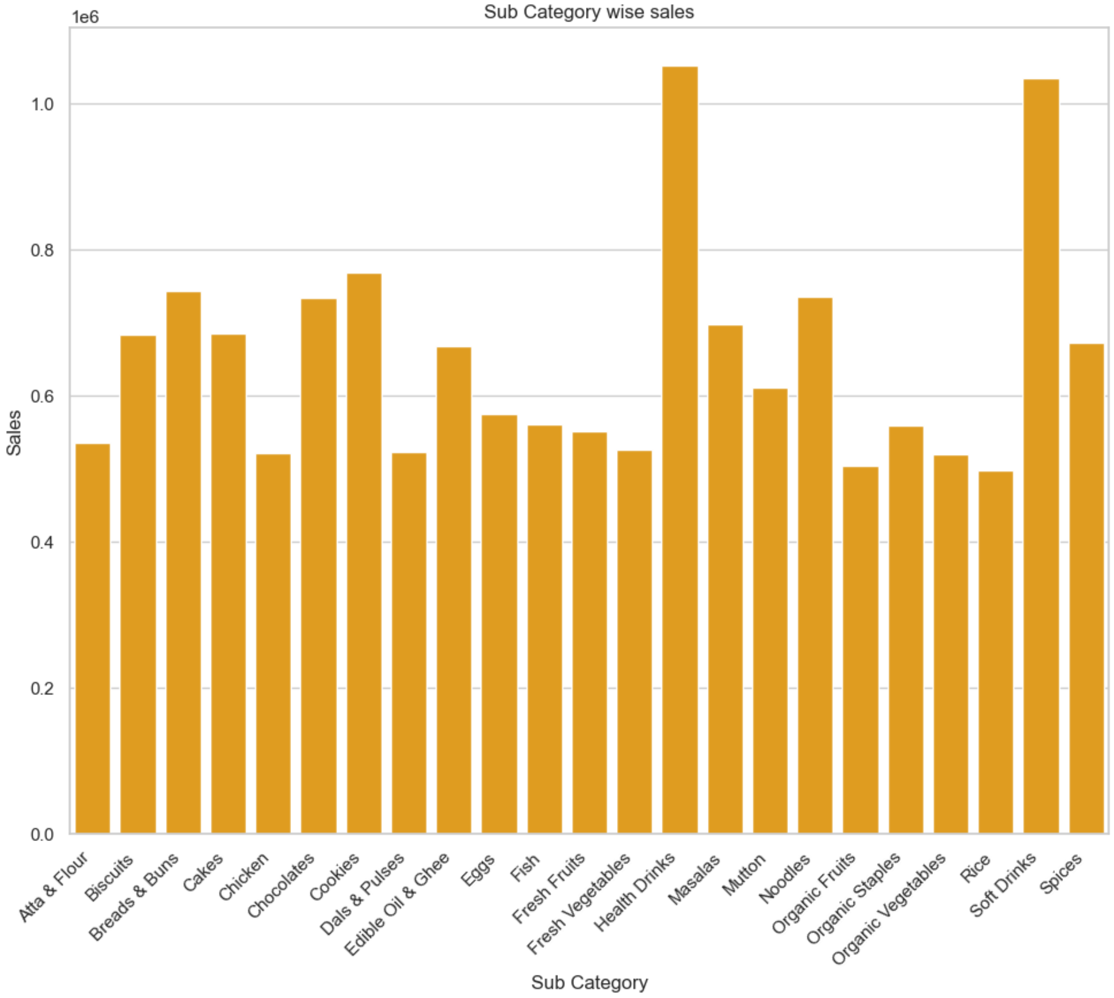
[ ]:

```

```

3 Sales by Sub-Category

[43]:

```
Sub_category_sales=df.groupby(['Sub Category'])['Sales'].sum().reset_index()
Sub_category_sales.sort_values(by='Sales')
plt.figure(figsize=(10,8))
sns.barplot(data=Sub_category_sales,x='Sub Category',y='Sales',color='orange')
plt.tight_layout()
plt.xlabel('Sub Category')
plt.ylabel('Sales')
plt.title('Sub Category wise sales')
plt.xticks(rotation=45, ha='right')

plt.show()
```



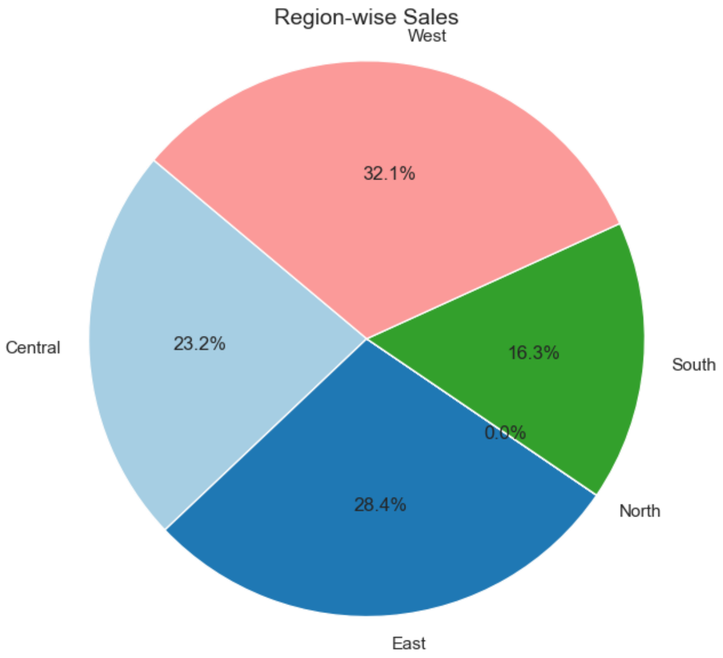
Observations:

- `Health Drinks and Soft Drinks has the highest number of total sales in the entire sub catego`
- `While, Rice and Organic Fruits has the lowest number of total sales in the entire sub catego`

4 Region wise sales

[44]:

```python
region_sales = df.groupby(['Region'])['Sales'].sum()
plt.figure(figsize=(10, 6))
plt.pie(region_sales, labels=region_sales.index, autopct='%1.1f%%', startangle=140, colors=plt.cm.Paired.colors)
plt.title('Region-wise Sales', fontsize=14)
plt.axis('equal')
plt.tight_layout()
plt.show()
```



Observations:

```
1.West region has most(32.1%) sales as compared to all regions,Then east(28.4%) region followe

2.However North is not included in the chart as it has only one record in the dataset.
```

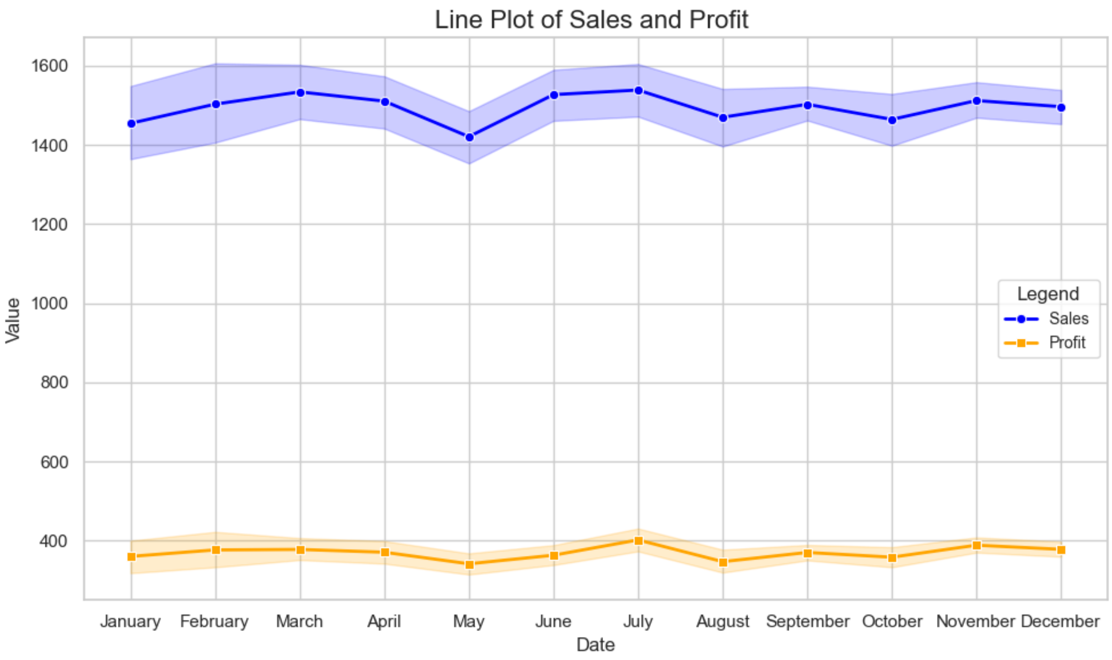
[45]:

```python
#df['Order Month'] = df['Order Date'].dt.month
df['Order Month Name'] = df['Order Date'].dt.month_name()
```

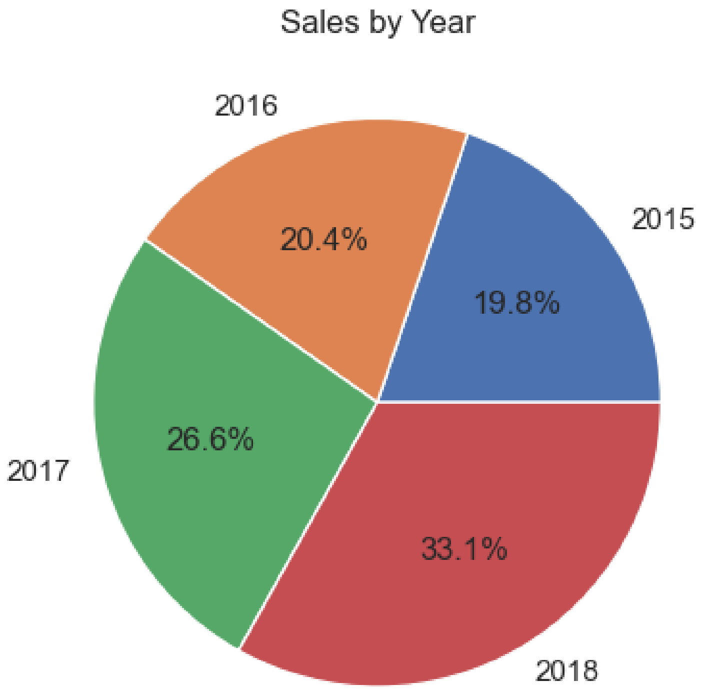
[46]:

```python
month_order = [
    "January", "February", "March", "April", "May", "June",
    "July", "August", "September", "October", "November", "December"]

# Convert 'Order Month Name' into a categorical variable with the correct order
```

```
df['Order Month Name'] = pd.Categorical(df['Order Month Name'],␣
 ↪categories=month_order, ordered=True)

# Sort the DataFrame by the ordered 'Order Month Name' column
df = df.sort_values(by='Order Month Name')
# Set Seaborn theme
sns.set_theme(style="whitegrid")

# Create figure and axes
plt.figure(figsize=(10, 6))


# Line plot for Sales vs. Discount
sns.lineplot(data=df, x='Order Month Name', y='Sales', color='blue',␣
 ↪label='Sales ', marker='o', linewidth=2)

# Line plot for Profit vs. Discount
sns.lineplot(data=df, x='Order Month Name', y='Profit', color='orange',␣
 ↪label='Profit', marker='s', linewidth=2)

# Add title and labels
plt.title('Line Plot of Sales and Profit ', fontsize=16)
plt.xlabel('Date', fontsize=12)
plt.ylabel('Value', fontsize=12)

# Add legend
plt.legend(title='Legend', fontsize=10)

# Show plot
plt.tight_layout()
plt.show()
```



```
[47]: df['year'] = df['Order Date'].dt.year
      Yearly_Sales=df.groupby('year')['Sales'].sum()

      year_labels = Yearly_Sales.index.astype(int)
      plt.pie(Yearly_Sales, labels=year_labels, autopct='%1.1f%%')
      plt.title('Sales by Year')
      plt.show()
```



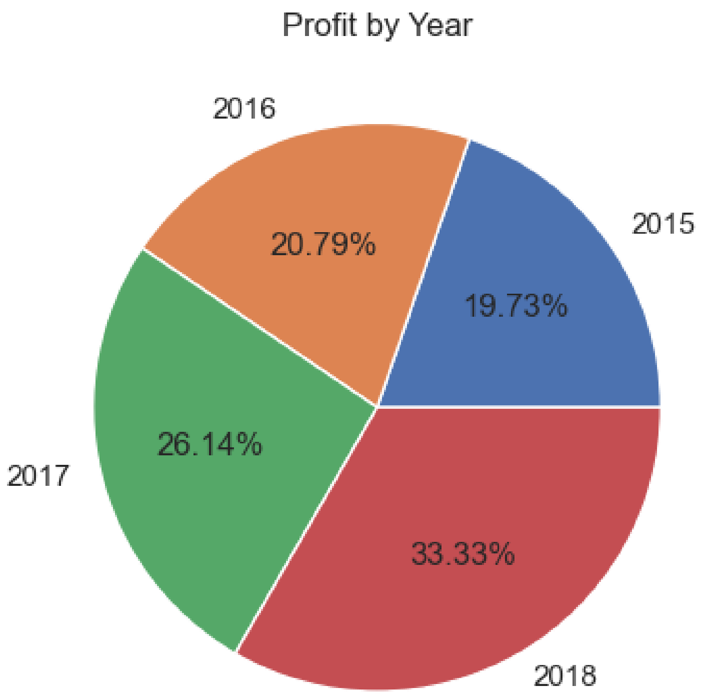
Observations-

1.Year 2018 shows highest Sales amongst all years.

2.2015 shows less sales compared to all years.

[48]:

```
df['year'] = df['Order Date'].dt.year
Yearly_Profit=df.groupby('year')['Profit'].sum()

year_labels = Yearly_Sales.index.astype(int)
plt.pie(Yearly_Profit, labels=year_labels, autopct='%1.2f%%')
plt.title('Profit by Year')
plt.show()
```



Observations-

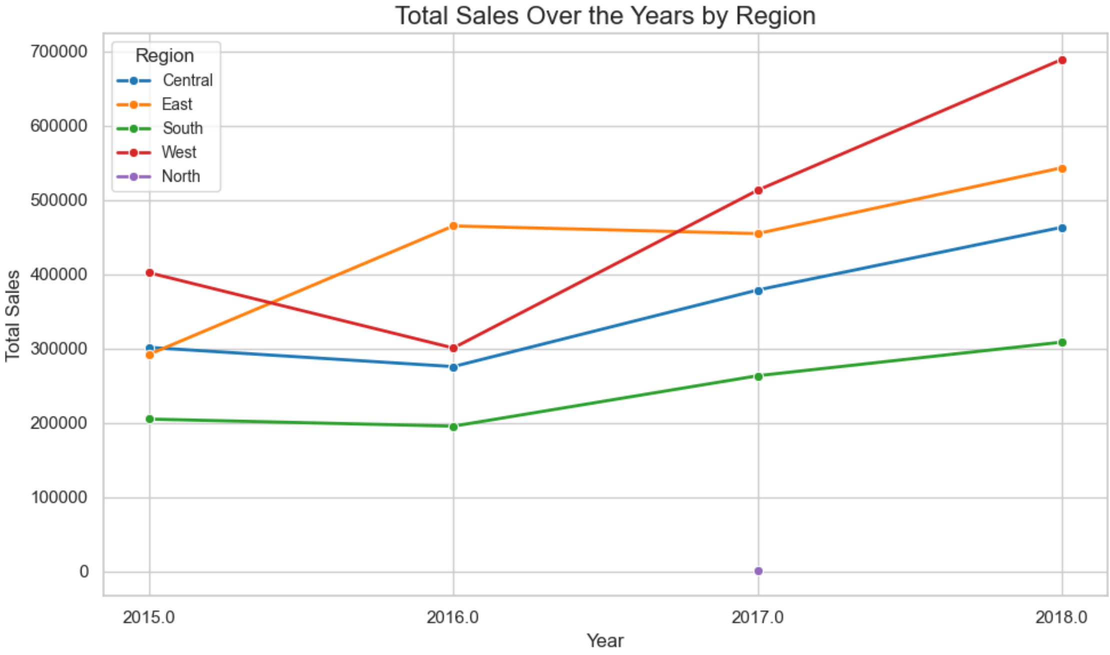
1.Year 2018 shows highest profit amongst all years.

2.2015 shows less profit compared to all years.

[49]:

```
df['year']=df['Order Date'].dt.year
# Group sales by year and region
yearly_sales = df.groupby(['year', 'Region'])['Sales'].sum(numeric_only=True).
  ↪reset_index()

# Ensure year is a string for plotting
yearly_sales['year'] = yearly_sales['year'].astype(str)
#year_labels = yearly_sales.index.astype(int)

# Set Seaborn theme
sns.set_theme(style="whitegrid")

# Create figure
plt.figure(figsize=(10, 6))

# Line plot with Seaborn
```

```
sns.lineplot(data=yearly_sales,␣
↪x='year',y='Sales',hue='Region',marker='o',linewidth=2,palette='tab10')

# Add title and labels
plt.title('Total Sales Over the Years by Region', fontsize=16)
plt.xlabel('Year', fontsize=12)
plt.ylabel('Total Sales', fontsize=12)

# Adjust legend
plt.legend(title='Region', fontsize=10, loc='upper left')

# Show the plot
plt.tight_layout()
plt.show()
```



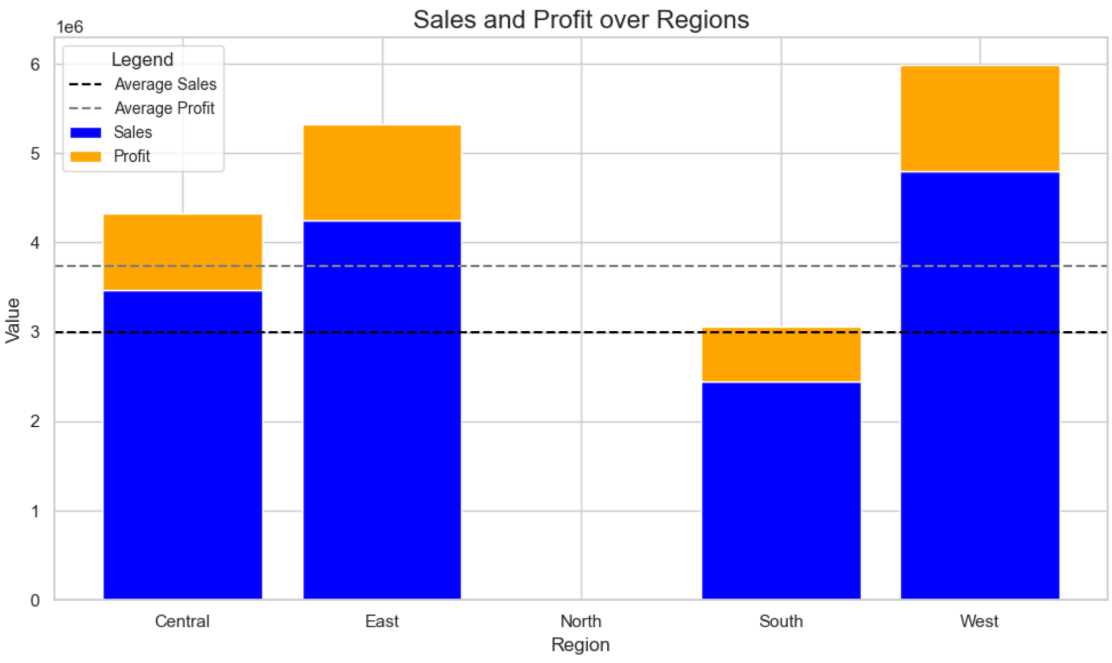
Observation:

In all regions there has been steady increase in sales over the years.

[50]:
```
#Create pivot table for sales and profit
salesprofit = df.pivot_table(index='Region', values=['Sales', 'Profit'],␣
↪aggfunc='sum').reset_index()

# Calculate average sales and profit
averagesales = df.groupby('Region')['Sales'].sum().mean()
averageprofit = df.groupby('Region')['Profit'].sum().mean()
```

```python
print(averagesales)
print(averageprofit)
print(salesprofit)

# Set the figure size
plt.figure(figsize=(10, 6))

# Create a stacked bar chart
regions = salesprofit['Region']
sales = salesprofit['Sales']
profit = salesprofit['Profit']

plt.bar(regions, sales, label='Sales', color='blue')
plt.bar(regions, profit, bottom=sales, label='Profit', color='orange')

# Add horizontal lines for average sales and profit
plt.axhline(y=averagesales, color='black', linestyle='--',label='Average Sales')
plt.axhline(y=averageprofit + averagesales, color='gray', linestyle='--', label='Average Profit')

# Customize the chart
plt.title('Sales and Profit over Regions', fontsize=16)
plt.xlabel('Region', fontsize=12)
plt.ylabel('Value', fontsize=12)
plt.legend(title='Legend', fontsize=10, loc='upper left')

# Display the chart
plt.tight_layout()
plt.show()
```

```
2991396.4
749424.24
    Region      Profit    Sales
0  Central   856806.84  3468156
1     East  1074345.58  4248368
2    North      401.28     1254
3    South   623562.89  2440461
4     West  1192004.61  4798743
```



Observations:

1.Average sales amount is 3738932

2. Average profit amount is 936679.98
3. All regions except South has crossed the average sales and profit mark

5 Insights

[ ]:

```
1. The Eggs, Meat, and Fish category is the top-performing category, indicating␣
 ↪a high demand and consistent consumer preference for these products.
Oil and Masala has the lowest sales, suggesting either a niche demand or␣
 ↪underperformance in this segment, potentially due to competition or pricing.

2.Health Drinks and Soft Drinks dominate the sub-category sales, highlighting␣
 ↪their popularity and potential for growth in the beverage market.
 On the other hand, Rice and Organic Fruits exhibit the lowest sales, signaling␣
 ↪a need to revisit marketing, availability, or pricing strategies for these␣
 ↪sub-catego.

3.The West region leads in sales, accounting for 32.1% of the total, closely␣
 ↪followed by the East region with 28.4%. These regions can be targeted for␣
 ↪further expansion and promotional activities.
```

```
4.The Central region performs moderately, while the North region is underrepresented, having only one record in the dataset. Further data collection and analysis for the North region could provide better insights.

5.2018 stands out with the highest sales, indicating a peak in business performance during that year, possibly due to market trends or strategic initiatives.
2015 shows the lowest sales, warranting a deeper analysis of factors contributing to this underperformance.

6.Similar to sales, 2018 records the highest profit, showcasing effective cost management or pricing strategies.
 2015 has the lowest profit, reinforcing the need to analyze and address the underlying causes.

7.Across all regions, there has been a steady increase in sales over the years, reflecting a positive growth trajectory. This trend indicates increasing market penetration and consumer acceptance.

8.The average sales amount is 3,738,932, while the average profit is 936,679.98.
Regions such as West, East, and Central have surpassed these averages, demonstrating their strong market performance.

9.The South region lags behind, providing an opportunity to explore new strategies to enhance its sales and profitability.
```

[ ]:

```

```