Rajalakshmi Engineering College
(An Autonomous Institution)
Rajalakshmi Nagar,
Thandalam- 602105

DEPARTMENT OF ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING

AD23632 - Framework for Data Visualization and Analytics

Mini Project: Zepto Sales Analysis

Report submitted by

REGISTRATION NUMBER : 231501012 & 231501047

STUDENT NAME : Amirtha & Giridhar

YEAR : 2023-2027

SUBJECT CODE : AD23632

Table of Contents

Chapter 1: Abstract

The growing reliance on data analytics in business strategy has underscored the importance of visualizing and understanding sales performance across multiple dimensions. This project, Zepto Sales Analysis, aims to deliver a comprehensive examination of Zepto's sales data using both Power BI and Python-based analytics. The primary objective is to uncover meaningful insights that drive data-informed decisions and optimize business outcomes. Specifically, the study seeks to analyse total and average sales, product performance, and outlet efficiency across various categories such as fat content, outlet size, and location type. By employing data visualization and DAX-based key performance indicators (KPIs), the project highlights sales distribution patterns, identifies top-performing product types, and reveals temporal sales trends. Furthermore, the analysis demonstrates how interactive dashboards can enhance decision-making by providing a clear and intuitive understanding of complex datasets. Ultimately, this project bridges descriptive and diagnostic analytics to empower strategic planning and operational improvement within Zepto's retail framework.

Chapter 2: Introduction

In the modern retail landscape, data analytics plays a vital role in understanding business performance and guiding strategic decisions. The Zepto Sales Analysis project focuses on evaluating sales data to uncover key patterns, trends, and insights that influence revenue generation.

By utilizing Power BI for interactive visualization and Python for analytical validation, this study examines factors such as product category, outlet size, fat content, and location type to assess their impact on overall sales. The integration of dashboards and data modelling techniques helps translate raw data into meaningful insights.

Ultimately, this project highlights how business intelligence tools can enhance sales monitoring, improve operational efficiency, and support data-driven decision-making within Zepto's retail framework.

Chapter 3: Dataset Description

The dataset used for the Zepto Sales Analysis project captures a comprehensive view of sales-related attributes across various outlets, products, and locations. It is a structured tabular dataset, making it suitable for both descriptive and comparative analysis of sales performance across different business dimensions.

Key variables include:

- **Item Details:** Item Identifier, Item Type, Item Fat Content, Item Weight, Item Visibility - These attributes describe product characteristics and are used to analyse performance differences among product categories and nutritional classifications.

- **Outlet Attributes:** Outlet Identifier, Outlet Type, Outlet Size, Outlet Location Type, Outlet Establishment Year - These variables represent the business and operational context of each outlet, helping identify how store type, size, and location influence overall sales.

- **Sales Metrics:** Sales, Rating - These are the primary performance indicators, reflecting revenue generation and customer satisfaction levels across different outlets and product categories.

This dataset is particularly valuable because it integrates product-level details with outlet characteristics, allowing for multidimensional analysis of sales performance. By linking quantitative measures such as total and average sales with qualitative attributes like outlet type and size, the dataset enables a deeper understanding of factors driving business success and customer preferences.

Chapter 4: Objective

The main objective of this project is to analyse Zepto's sales data to uncover key performance insights and identify factors influencing overall business outcomes. To achieve this, the study defines specific analytical goals that provide clarity and direction:

1. **Sales Performance Evaluation:** Examine total and average sales across different product categories, outlet types, outlet sizes, and locations to understand key revenue drivers.

2. **Trend Analysis:** Identify sales trends over time based on outlet establishment year to reveal growth patterns and highlight high-performing periods.

3. **Product and Category Insights:** Compare the performance of various item types and fat content categories to determine which products contribute most to overall sales.

4. **Outlet Efficiency:** Assess how outlet characteristics such as size, type, and location affect sales performance, helping identify operational strengths and weaknesses.

5. **Tool Integration:** Demonstrate the combined power of Power BI for interactive visualization and Python for analytical validation, showcasing how both tools enhance data-driven decision-making.

By achieving these objectives, the project aims to deliver practical business insights that support strategic planning and operational improvement. For Zepto, this analysis offers a data-backed understanding of sales dynamics, enabling informed decisions to boost profitability and efficiency.

Chapter 5: Methodology

The Zepto Sales Analysis follows a structured Python-based workflow:

1. **Import Libraries & Load Dataset**
    - Used pandas, matplotlib, and seaborn to load and inspect the dataset.

2. **Data Preprocessing**
    - Filled missing values in Item Weight with the mean.
    - Standardized Item Fat Content categories (LF → Low Fat, reg → Regular).
    - Verified data types for numerical and categorical columns.

3. **Summary Statistics**
    - Calculated descriptive statistics and frequency counts for key categorical columns.

4. **Visual Analysis**
    - Bar charts for sales by fat content, item type, outlet type, and location.
    - Pie chart for outlet size contribution.
    - Line chart for sales trends over outlet establishment years.
    - Scatter plot for visibility vs sales and correlation heatmap for numeric features.

5. **Insight Extraction**
    - Derived revenue patterns, top products, outlet efficiency, and temporal trends to guide business decisions.

Chapter 6: Python Implementation

Python serves as the primary environment for data preprocessing and exploratory data analysis in the Zepto Sales Analysis project. Key libraries such as pandas, NumPy, matplotlib, and seaborn are used for cleaning, summarizing, and visualizing sales data.

The workflow begins with importing the dataset, standardizing column names, and converting relevant variables (such as sales, order date, and quantity) into numeric or datetime types. Missing or invalid entries are handled systematically, either through imputation or removal, to ensure a clean and reliable dataset.

Visualizations form the core of the Python implementation. Line plots and area plots are used to examine daily, weekly, and monthly sales trends, while bar charts display top-performing product categories and individual items. Boxplots reveal variations in sales across regions or customer types, and heatmaps illustrate correlations between variables such as sales, order quantity, and peak order hours. Comparisons across categories such as city, product category, and customer segment allow for insights into group-level performance differences.

The implementation also includes feature engineering, such as creating new fields for revenue per order, order frequency per customer, and peak hour indicators, which help understand customer purchasing behaviour and optimize operational strategies. Visualizations are saved for inclusion in reports and for integration into Power BI or Tableau dashboards, enabling interactive and business-oriented analysis.

In essence, Python provides a reproducible and transparent foundation for Zepto sales analysis, ensuring that insights are data-driven, verifiable, and actionable for business decision-making.

Chapter 7: Power BI Dashboard

Power BI was used to create interactive dashboards, enabling stakeholders to explore Zepto's sales data in a business-friendly way. The data imported into Power BI was preprocessed in Python to ensure cleanliness and consistency.

Key features of the dashboard include:

- **Line and Bar Charts:** Compare total and average sales across product categories, outlet types, and fat content.
- **Scatter Plots:** Examine relationships between variables such as item visibility and sales.
- **Stacked Bar Charts:** Show differences in sales by outlet type, size, and location.
- **Slicers and Filters:** Allow interactive exploration of the dataset by outlet, product type, fat content, and location for deeper insights.

These visualizations provide a clear, interactive view of sales patterns, product performance, and outlet efficiency, supporting data-driven decision-making.

Fig 7.1: Power BI Dashboard

Chapter 8: Tableau Dashboard

Tableau complements Power BI by creating visually engaging dashboards ideal for presentations and storytelling. The cleaned dataset was imported, and calculated fields were added to derive metrics such as the ratio of social media time to work hours and the perceived productivity gap.

Key features of the Tableau dashboard include:

- **Combined Dashboards:** Multiple sheets integrated into a cohesive story to illustrate trends and insights.
- **Visual Analytics:** Bar charts, line charts, and tables to analyse sales performance, outlet efficiency, and product popularity.
- **Storytelling Focus:** Enables clear communication of patterns and insights to stakeholders through interactive and visually appealing graphics.

Tableau's visual storytelling approach enhances understanding of sales dynamics and supports strategic decision-making.

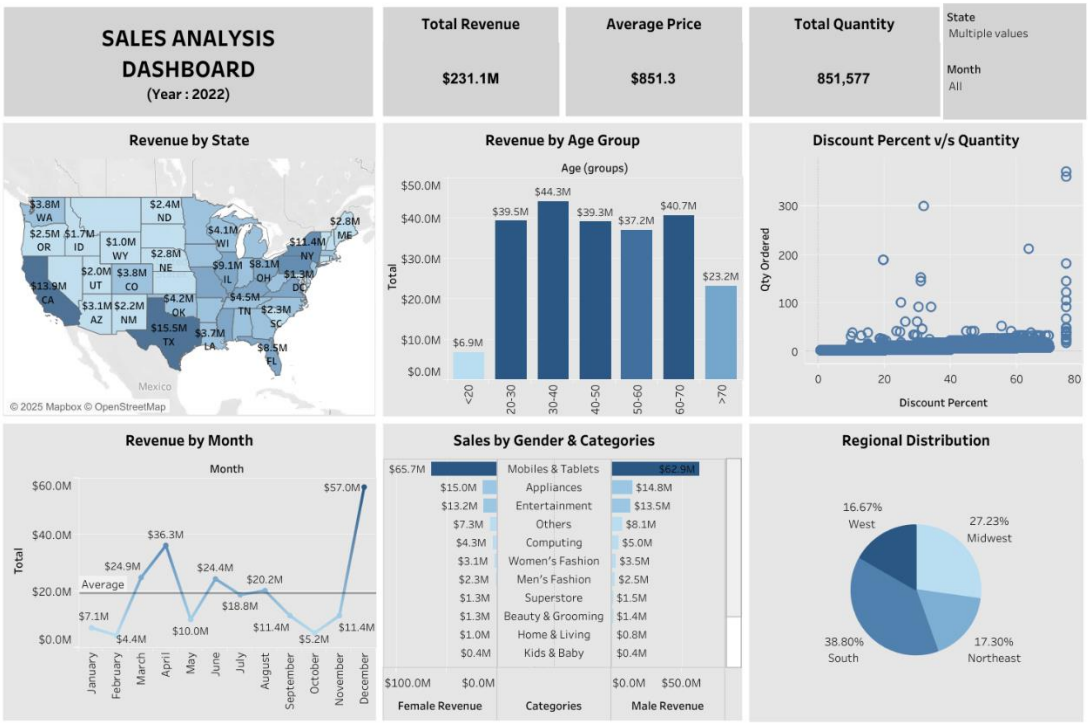


Fig 8.1: Tableau Dashboard

Chapter 9: Analysis

The analysis of Zepto's sales data highlights several important patterns across products, outlets, and time:

1. **Product Performance Patterns**
    - Items with Low Fat content generated the highest sales, indicating customer preference for healthier options.
    - Fruits and Vegetables emerged as the top-selling category, while Seafood had the lowest sales, reflecting demand differences across product types.
2. **Outlet Efficiency and Size**
    - Medium-sized outlets contributed the most to total revenue, whereas high-sized outlets underperformed, suggesting potential operational inefficiencies.
    - Outlet type and location also influenced sales, with urban outlets and supermarkets generating higher revenue compared to smaller or rural outlets.
3. **Temporal Trends**
    - Sales peaked in 2018, revealing the highest revenue-generating period.
    - Understanding these trends can help in inventory planning, promotions, and expansion strategies.
4. **Correlation and Relationship Insights**
    - A moderate correlation exists between item visibility and sales, indicating that better product placement or visibility positively affects performance.
    - Numerical and categorical analyses from dashboards revealed patterns between outlet characteristics, product categories, and revenue generation.

5. **Dashboard Insights**

    - Power BI and Tableau dashboards provided interactive visualizations that made trends, top-performing products, and outlet performance easily interpretable.

    - Slicers and filters enabled granular analysis by outlet, product type, fat content, and location, empowering strategic decision-making.

Overall, the findings underscore the multidimensional nature of sales performance: product characteristics, outlet attributes, and temporal trends collectively influence revenue. By leveraging both descriptive and visual analytics, Zepto can make informed decisions to optimize product offerings, improve outlet efficiency, and maximize revenue.

Chapter 10 : Conclusion

The Zepto Sales Analysis demonstrates that sales performance is influenced by multiple factors, including product characteristics, outlet attributes, and temporal trends. Key findings indicate that low-fat products and fruits & vegetables are the top performers, medium-sized outlets generate the most revenue, and 2018 was the peak sales year. Outlet type, size, and location also play a crucial role in revenue generation, highlighting operational strengths and areas for improvement.

For Zepto, these insights suggest actionable strategies: optimizing inventory based on top-selling items, focusing on medium-performing outlets for efficiency improvements, and tailoring marketing campaigns according to outlet location and size. Interactive dashboards in Power BI and Tableau further enhance decision-making by enabling stakeholders to explore data dynamically and identify trends quickly.

Future work could expand the analysis by incorporating larger or more granular datasets, exploring customer demographics, and using predictive analytics to forecast sales trends. Additionally, integrating external factors such as promotions, seasonal effects, or competitor data could provide deeper insights for strategic planning.

Overall, this study underscores the value of combining descriptive analytics, visual dashboards, and data-driven insights to optimize sales strategy and operational performance within Zepto’s retail framework.

Chapter 11: Appendix

11.1 Python Code

Importing required libraries

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

Reading and inspecting the excel file

```
df = pd.read_excel("/content/Zepto Grocery Data.xlsx")
print(df.shape)
print(df.info())
print(df.head())
```

```
(8523, 12)
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 8523 entries, 0 to 8522
Data columns (total 12 columns):
 #   Column                     Non-Null Count  Dtype
---  ------                     --------------  -----
 0   Item Fat Content           8523 non-null   object
 1   Item Identifier            8523 non-null   object
 2   Item Type                  8523 non-null   object
 3   Outlet Establishment Year  8523 non-null   int64
 4   Outlet Identifier          8523 non-null   object
 5   Outlet Location Type       8523 non-null   object
 6   Outlet Size                8523 non-null   object
 7   Outlet Type                8523 non-null   object
 8   Item Visibility            8523 non-null   float64
 9   Item Weight                7060 non-null   float64
 10  Sales                      8523 non-null   float64
 11  Rating                     8523 non-null   float64
dtypes: float64(4), int64(1), object(7)
memory usage: 799.2+ KB
None
  Item Fat Content Item Identifier              Item Type  \
0          Regular           FDX32  Fruits and Vegetables
1          Low Fat           NCB42     Health and Hygiene
2          Regular           FDR28           Frozen Foods
3          Regular           FDL50                Canned
4          Low Fat           DRI25           Soft Drinks
```

```
   Outlet Establishment Year Outlet Identifier Outlet Location Type  \
0                       2012            OUT049                Tier 1
1                       2022            OUT018                Tier 3
2                       2016            OUT046                Tier 1
3                       2014            OUT013                Tier 3
4                       2015            OUT045                Tier 2

  Outlet Size        Outlet Type  Item Visibility  Item Weight     Sales  \
0      Medium  Supermarket Type1         0.100014        15.10  145.4786
1      Medium  Supermarket Type2         0.008596        11.80  115.3492
2       Small  Supermarket Type1         0.025896        13.85  165.0210
3        High  Supermarket Type1         0.042278        12.15  126.5046
4       Small  Supermarket Type1         0.033970        19.60   55.1614

   Rating
0     5.0
1     5.0
2     5.0
3     5.0
4     5.0
```

Data Cleaning and Preprocessing

1. Checking and Handling Missing Values

```
df.isnull().sum()
df['Item Weight'].fillna(df['Item Weight'].mean(), inplace=True)
df.dropna(subset=['Outlet Size'], inplace=True)
```

```
/tmp/ipython-input-4213215548.py:2: FutureWarning: A value is trying to be set on a copy of a DataFrame
The behavior will change in pandas 3.0. This inplace method will never work because the intermediate obj

For example, when doing 'df[col].method(value, inplace=True)', try using 'df.method({col: value}, inplac


  df['Item Weight'].fillna(df['Item Weight'].mean(), inplace=True)
```

2. Standardizing Text Columns

```
df['Item Fat Content'].replace({'LF': 'Low Fat','low fat': 'Low Fat','reg': 'Regular'}, inplace=True)
```

```
/tmp/ipython-input-3332972348.py:1: FutureWarning: A value is trying to be set on a copy of a DataFrame or Series
The behavior will change in pandas 3.0. This inplace method will never work because the intermediate object on wh

For example, when doing 'df[col].method(value, inplace=True)', try using 'df.method({col: value}, inplace=True)'


  df['Item Fat Content'].replace({'LF': 'Low Fat','low fat': 'Low Fat','reg': 'Regular'}, inplace=True)
```

3. Verifying Data Types

```
df.dtypes
```

	0
Item Fat Content	object
Item Identifier	object
Item Type	object
Outlet Establishment Year	int64
Outlet Identifier	object
Outlet Location Type	object
Outlet Size	object
Outlet Type	object
Item Visibility	float64
Item Weight	float64
Sales	float64
Rating	float64

dtype: object

Summarizing Statistics

```
print(df.describe())
print(df['Outlet Type'].value_counts())
print(df['Outlet Location Type'].value_counts())
```

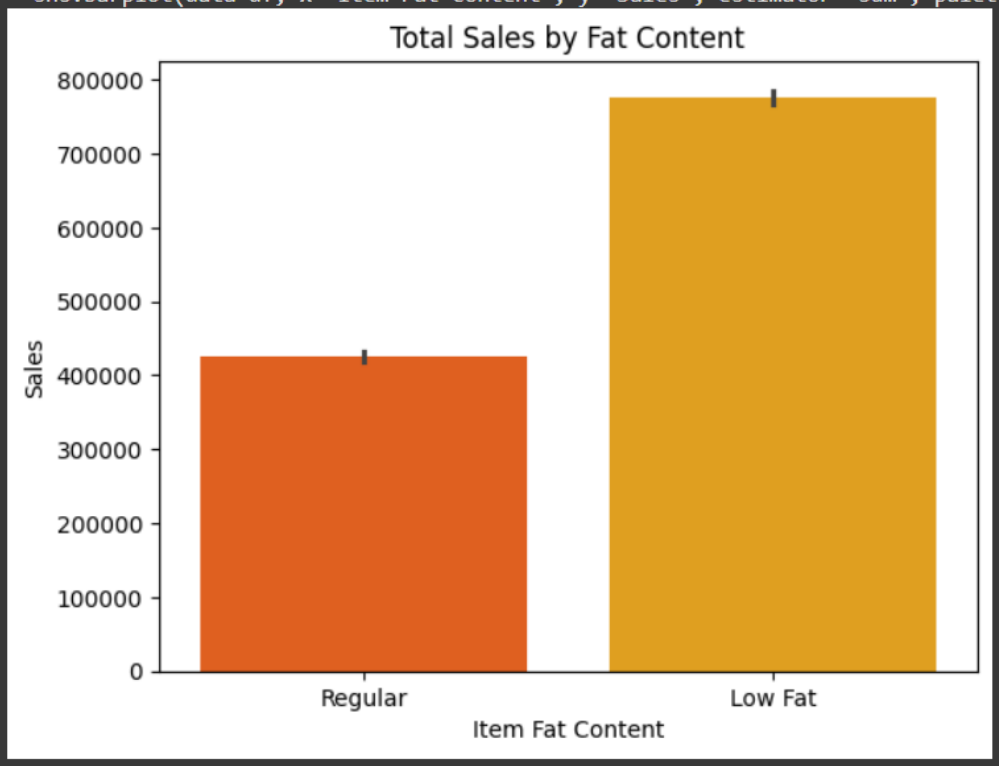
```
       Outlet Establishment Year  Item Visibility  Item Weight        Sales  \
count                8523.000000      8523.000000  8523.000000  8523.000000
mean                 2016.450546         0.066132    12.857645   140.992783
std                     3.189396         0.051598     4.226124    62.275067
min                  2011.000000         0.000000     4.555000    31.290000
25%                  2014.000000         0.026989     9.310000    93.826500
50%                  2016.000000         0.053931    12.857645   143.012800
75%                  2018.000000         0.094585    16.000000   185.643700
max                  2022.000000         0.328391    21.350000   266.888400

            Rating
count  8523.000000
mean      3.965857
std       0.605651
min       1.000000
25%       4.000000
50%       4.000000
75%       4.200000
max       5.000000
Outlet Type
Supermarket Type1    5577
Grocery Store        1083
Supermarket Type3     935
Supermarket Type2     928
Name: count, dtype: int64
Outlet Location Type
Tier 3    3350
Tier 2    2785
Tier 1    2388
Name: count, dtype: int64
```

Visualizations

1. Total Sales by Item Fat Content

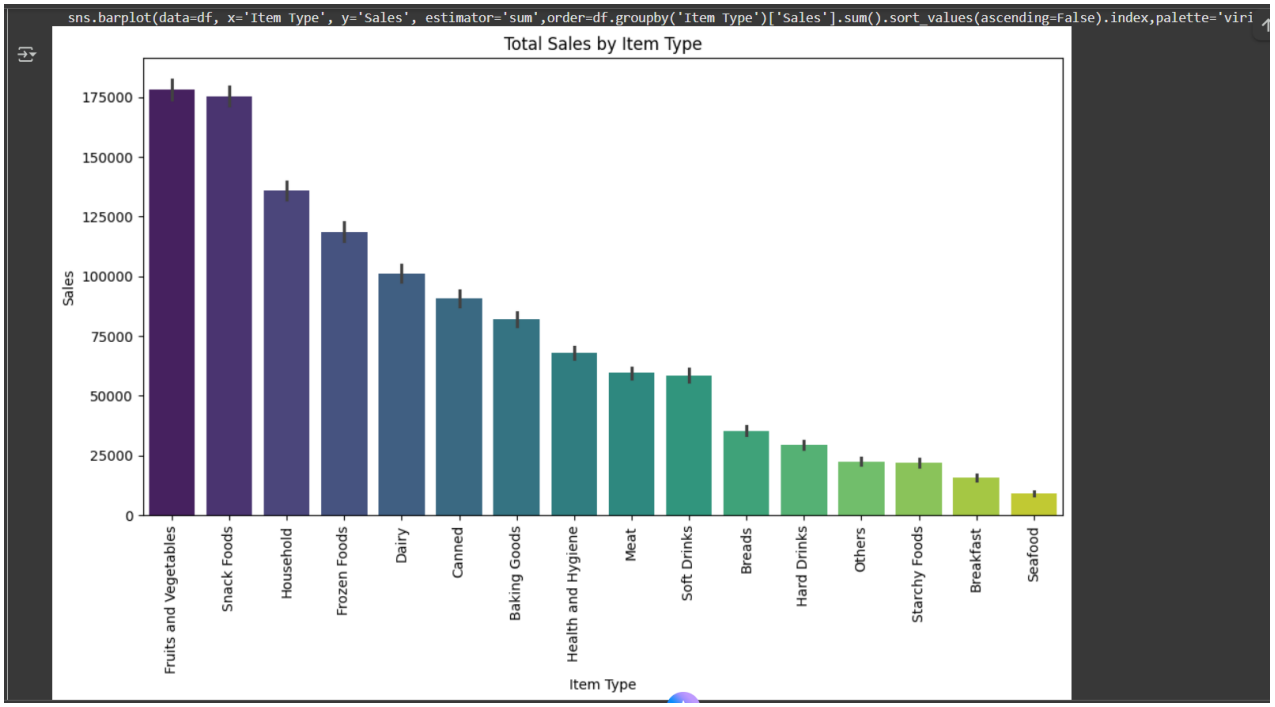
```
sns.barplot(data=df, x='Item Fat Content', y='Sales', estimator='sum', palette='autumn')
plt.title('Total Sales by Fat Content')
plt.show()
```

```
/tmp/ipython-input-2184514814.py:1: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign

  sns.barplot(data=df, x='Item Fat Content', y='Sales', estimator='sum', palette='autumn')
```



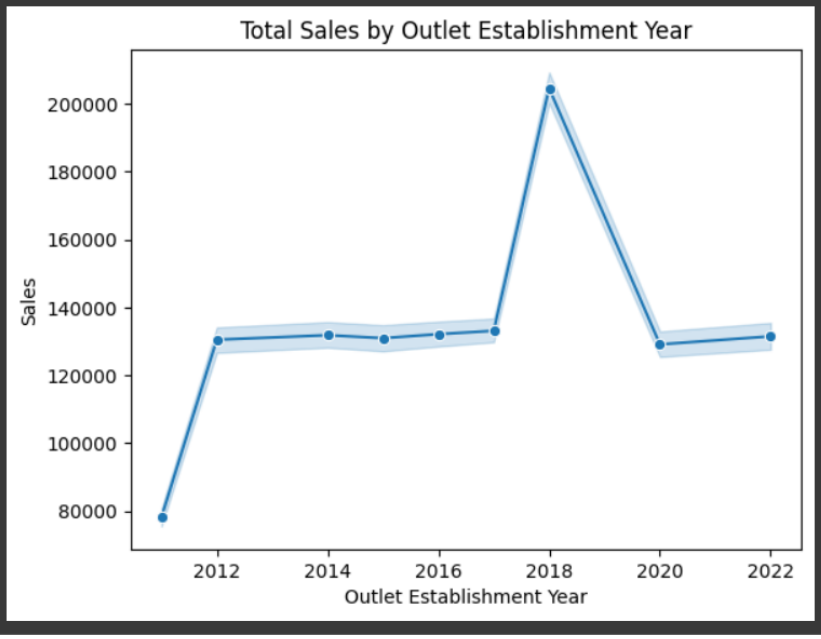
2. Sales by Item Type

```
plt.figure(figsize=(12,6))
sns.barplot(data=df, x='Item Type', y='Sales', estimator='sum',order=df.groupby('Item Type')['Sales'].sum().sort_values(ascending=False).index,palette='viridis')
plt.xticks(rotation=90)
plt.title('Total Sales by Item Type')
plt.show()
```



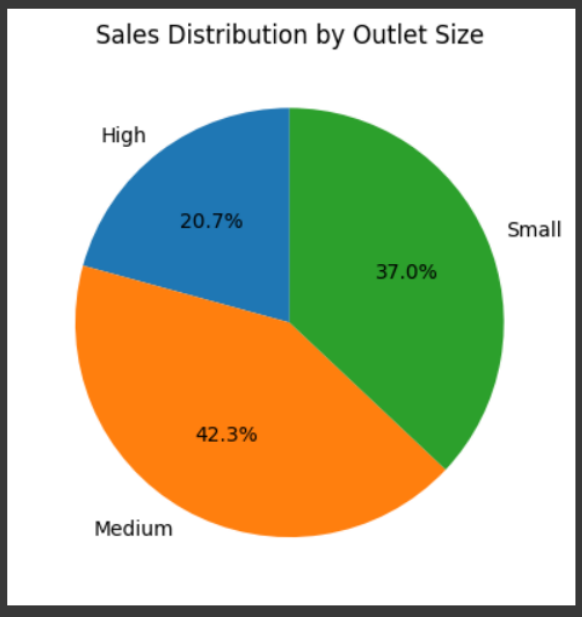
3. Outlet Establishment Year vs Total Sales

```
sns.lineplot(data=df, x='Outlet Establishment Year', y='Sales', estimator='sum', marker='o')
plt.title('Total Sales by Outlet Establishment Year')
plt.show()
```



4. Sales Distribution by Outlet Size

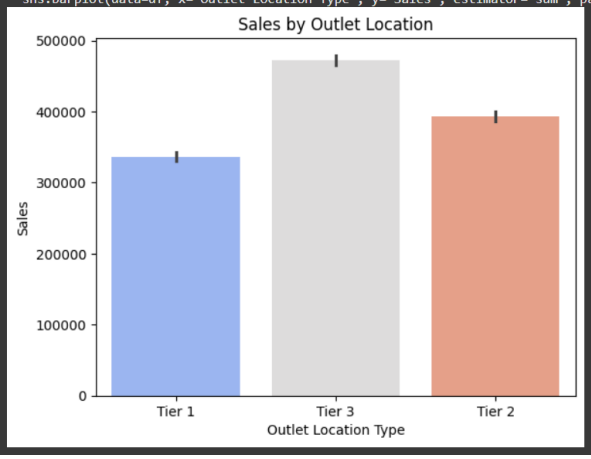
```
sales_by_size = df.groupby('Outlet Size')['Sales'].sum().reset_index()
plt.pie(sales_by_size['Sales'], labels=sales_by_size['Outlet Size'], autopct='%1.1f%%', startangle=90)
plt.title('Sales Distribution by Outlet Size')
plt.show()
```



5. Sales by Outlet Location

```
sns.barplot(data=df, x='Outlet Location Type', y='Sales', estimator='sum', palette='coolwarm')
plt.title('Sales by Outlet Location')
plt.show()
```

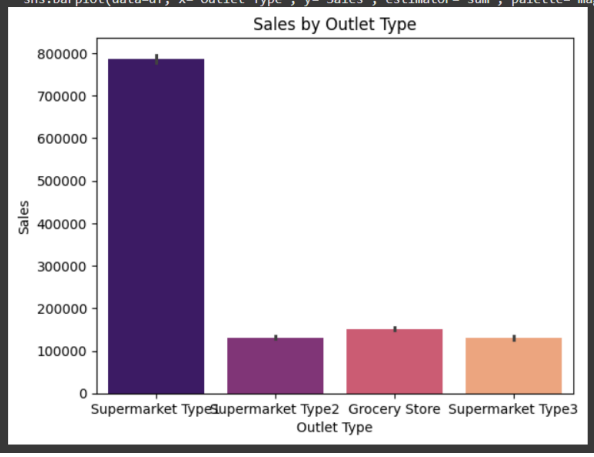
```
/tmp/ipython-input-175242674.py:1: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the same effect.

  sns.barplot(data=df, x='Outlet Location Type', y='Sales', estimator='sum', palette='coolwarm')
```



6. Sales by Outlet Type

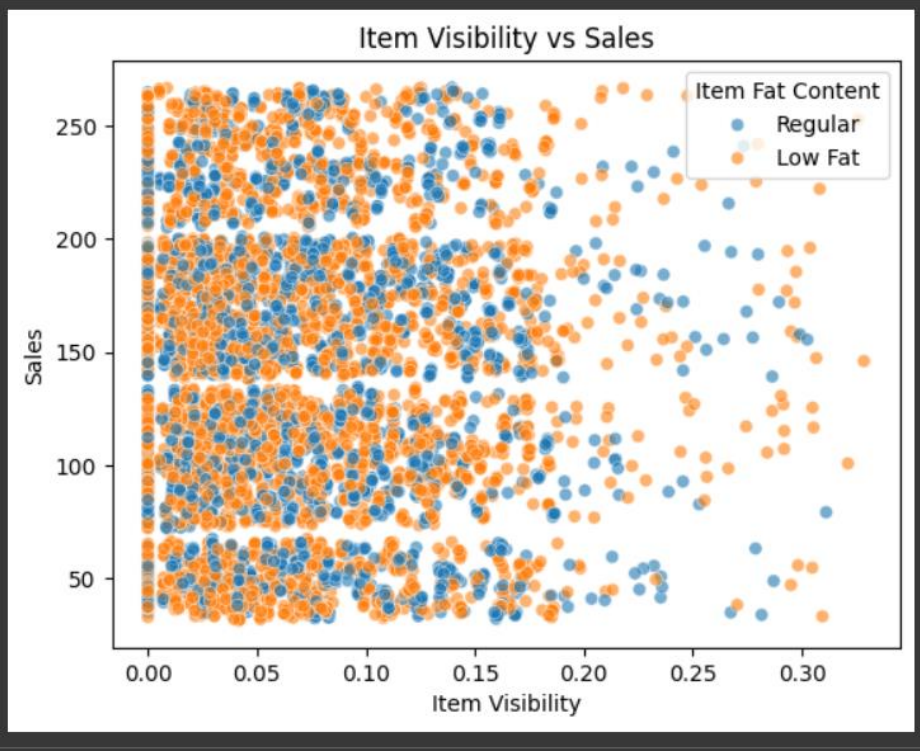
```
sns.barplot(data=df, x='Outlet Type', y='Sales', estimator='sum', palette='magma')
plt.title('Sales by Outlet Type')
plt.show()
```

```
/tmp/ipython-input-1240346257.py:1: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the same effect.

  sns.barplot(data=df, x='Outlet Type', y='Sales', estimator='sum', palette='magma')
```



7. Relationship Between Visibility and Sales

```
sns.scatterplot(data=df, x='Item Visibility', y='Sales', hue='Item Fat Content', alpha=0.6)
plt.title('Item Visibility vs Sales')
plt.show()
```



8. Correlation Heatmap

```
plt.figure(figsize=(8,6))
sns.heatmap(df.select_dtypes(include=['float64', 'int64']).corr(), annot=True, cmap='YlGnBu', fmt='.2f')
plt.title('Correlation Heatmap')
plt.show()
```