Data Science Project using Python

on

SUPPLY CHAIN DATA

Submitted by :

Name : Addetla Sai Chethan

Reg No : 12109834

Email : chethanaddetla@gmail.com

Cell No : 6302746267

Discipline of CSE/IT

Lovely School of Computer Science and Engineering

Lovely Professional University, Phagwara

Table of Content

INTRODUCTION

Supply chains play a crucial role in modern business operations, encompassing the intricate network of activities involved in the production and distribution of goods to consumers. Effective supply chain management not only ensures the smooth flow of products but also enhances customer satisfaction and drives business profitability.

This report delves into the analysis of a Fashion and Beauty startup's supply chain using Python, leveraging real-world data to uncover insights that can optimize operational efficiencies and maximize value creation for customers. The dataset used in this analysis includes comprehensive information across various stages of the supply chain, such as sourcing, manufacturing, transportation, inventory management, sales, and customer demographics.

Throughout this report, we will explore methodologies and techniques in Python to dissect and analyze key components of the supply chain. By applying data-driven insights, we aim to identify areas of improvement and strategic opportunities that can enhance the overall effectiveness of the supply chain.

Through this journey of Supply Chain Analysis, we will demonstrate how Python's analytical capabilities can be harnessed to extract actionable insights, ultimately paving the way for informed decision-making and sustainable business growth in the dynamic landscape of fashion and beauty industries.

OBJECTIVE

1) Analyzing the relationship between the price of products and the revenue generated by them.
2) Examine sales performance categorized by product type.
3) Calculate the total revenue generated from shipping carriers.
4) Evaluate the average lead time and average manufacturing costs for all products.
5) Analyzing the revenue generated by each SKU (Stock Keeping Unit).
6) Investigate the order quantity of each SKU to understand demand patterns.
7) Assess the shipping costs associated with different carriers.
8) Explore the distribution of costs by transportation mode.
9) Calculate the average defect rate across all product types.
10) Analyzing defect rates categorized by mode of transportation.

Source of Data set

The dataset is taken from Kaggle. Kaggle is a platform that hosts a variety of datasets from different domains such as healthcare, finance, sports, and more. The datasets on Kaggle are contributed by users and organizations from all over the world.

To access datasets on Kaggle, you first need to create an account on the platform. Once you have an account, you can search for datasets using the search bar on the Kaggle homepage or browse through the datasets by category.

Some of the popular datasets on Kaggle include the Titanic dataset, the IMDB movie review dataset, the New York City Airbnb dataset, E-commerce sales dataset.

About: Supply chain analytics is a valuable part of data-driven decision-making in various industries such as manufacturing, retail, healthcare, and logistics. It is the process of collecting, analyzing and interpreting data related to the movement of products and services from suppliers to customers.

Here are the details of my chosen data set.

- Name: Supply Chain Dataset
- Link: https://statso.io/supply-chain-analysis-case-study/
- Format: CSV
- No. of data sets: 1
- Number of Rows: 100 • Number of columns: 24
    i. String: 9
    ii. Decimal: 15
    iii. Time Date:
    iv. Other: 0 • Size: 21.4 MB

Date Fields:

1) Product Type
2) SKU
3) Price
4) Availability
5) Number of products sold
6) Revenue generated
7) Customer demographics
8) Stock levels
9) Lead times
10) Order quantities
11) Shipping times
12) Shipping carriers
13) Shipping costs
14) Supplier name
15) Location
16) Lead time
17) Production volumes
18) Manufacturing lead time
19) Manufacturing costs
20) Inspection results
21) Defect rates
22) Transportation modes
23) Routes
24) Costs

Packages or Library Used

1. Pandas

Purpose: Data Manipulation and Analysis

Description: Pandas is an essential library for data manipulation and analysis in Python. It provides two primary data structures: Series (one-dimensional) and DataFrame (two-dimensional), which are capable of handling and manipulating large datasets efficiently. Pandas offers a wide range of functionalities including data cleaning, merging, reshaping, and aggregation. It simplifies complex data operations with intuitive syntax and powerful data handling capabilities.

Key Features:

- **Data Cleaning**: Handle missing data, duplicate values, and perform data type conversions.
- **Data Transformation**: Apply functions to data, pivot tables, and group by operations.
- **Data Aggregation**: Summarize data using various aggregation functions (sum, mean, count, etc.).
- **Data Input/Output**: Read and write data to various file formats (CSV, Excel, SQL databases, etc.).

2. NumPy

Purpose: Numerical Computing

Description: NumPy is a fundamental package for numerical computing in Python. It provides support for arrays and matrices, along with a collection of mathematical functions to operate on these data structures. NumPy arrays are more efficient than Python lists for numerical operations due to their fixed size and homogeneous data type. It is widely used as the underlying data structure for other scientific computing libraries.

Key Features:

- **Array Operations**: Perform element-wise operations, mathematical functions, and linear algebra.
- **Efficiency**: NumPy arrays are more efficient in terms of memory and performance compared to Python lists.
- **Integration**: Serves as the foundation for other scientific libraries like Pandas and SciPy.

3. Matplotlib

Purpose: Data Visualization

Description: Matplotlib is a comprehensive library for creating static, animated, and interactive visualizations in Python. It offers a variety of plotting functions to visualize data, including line plots, bar charts, scatter plots, histograms, and more. Matplotlib is highly customizable, allowing users to tweak the appearance of plots to suit their needs. It serves as a base for other visualization libraries like Seaborn.

Key Features:

- **Plotting Capabilities**: Create a wide range of plots to visualize data.
- **Customization**: Customize plots with labels, titles, legends, and more.
- **Integration**: Works well with NumPy and Pandas for seamless data visualization.

4. Seaborn

Purpose: Statistical Data Visualization

Description: Seaborn is built on top of Matplotlib and provides a high-level interface for creating attractive and informative statistical graphics. It simplifies the process of creating complex visualizations and includes themes and color palettes to make plots more visually appealing. Seaborn works seamlessly with Pandas DataFrames, making it a valuable tool for exploratory data analysis.

Key Features:

- **Enhanced Visualizations**: Create more informative and visually appealing plots.
- **Statistical Plots**: Includes built-in functions for creating complex statistical visualizations.
- **Integration**: Works well with Pandas for easy plotting of DataFrames.

Platform: Jupyter Notebook

Purpose: Interactive Computing Environment

Description: Jupyter Notebook is an open-source web application that allows you to create and share documents containing live code, equations, visualizations, and narrative text. It provides an interactive environment for data analysis and visualization, making it an ideal platform for data science projects. Jupyter supports numerous programming languages, including Python, and integrates well with data science libraries.

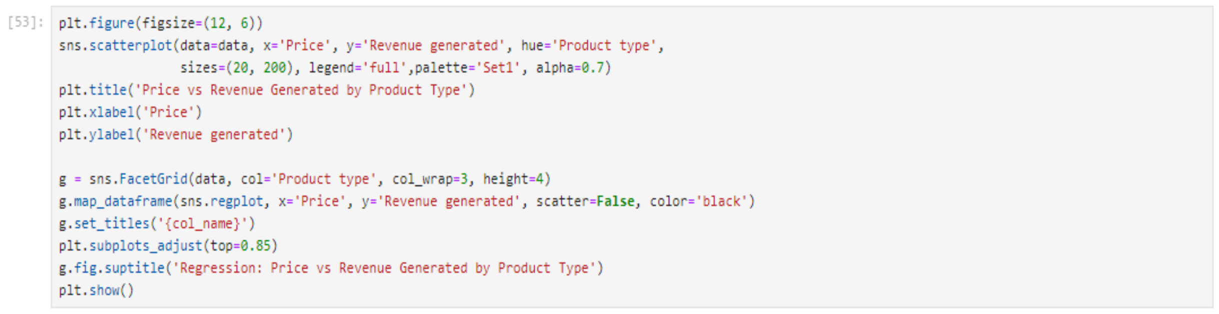
Key Features:

- **Interactive Code Execution**: Run code in cells and see results immediately.
- **Rich Media Output**: Display plots, tables, and rich text alongside code.
- **Documentation**: Combine code with narrative text using Markdown.

Analysis of Data set

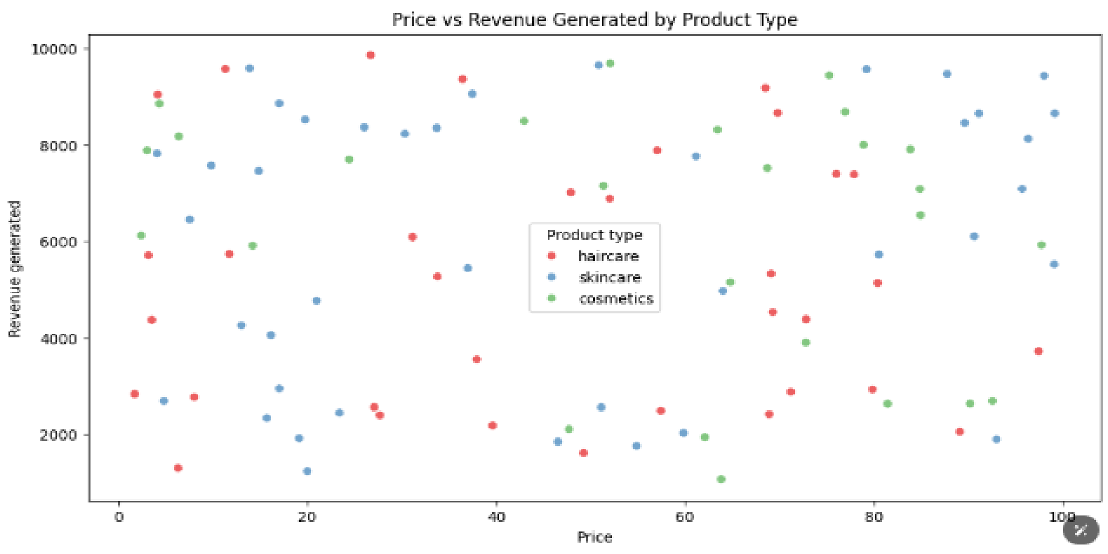
Objective 1: Analyze the relationship between the price of products and the revenue generated by them.
Description: The company derives more revenue from skincare products, and the higher the price of skincare products, the more revenue they generate

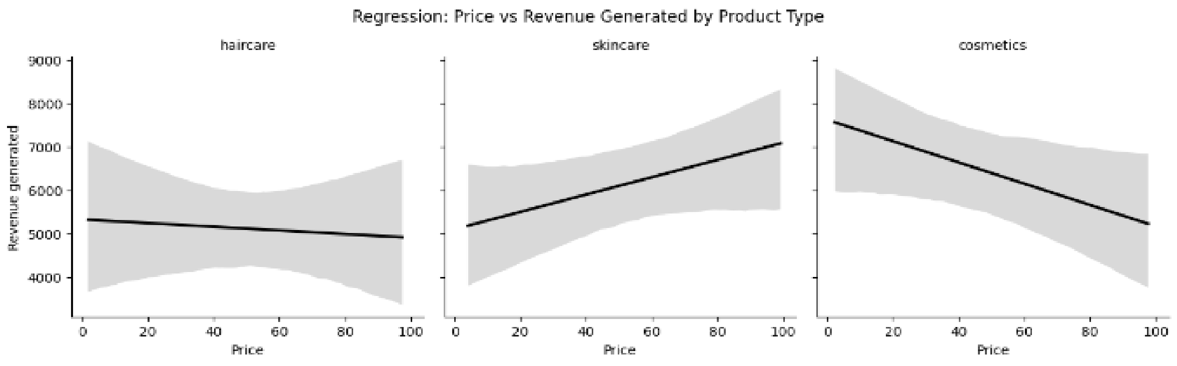
Specification:



```
[53]: plt.figure(figsize=(12, 6))
      sns.scatterplot(data=data, x='Price', y='Revenue generated', hue='Product type',
                      sizes=(20, 200), legend='full',palette='Set1', alpha=0.7)
      plt.title('Price vs Revenue Generated by Product Type')
      plt.xlabel('Price')
      plt.ylabel('Revenue generated')

      g = sns.FacetGrid(data, col='Product type', col_wrap=3, height=4)
      g.map_dataframe(sns.regplot, x='Price', y='Revenue generated', scatter=False, color='black')
      g.set_titles('{col_name}')
      plt.subplots_adjust(top=0.85)
      g.fig.suptitle('Regression: Price vs Revenue Generated by Product Type')
      plt.show()
```



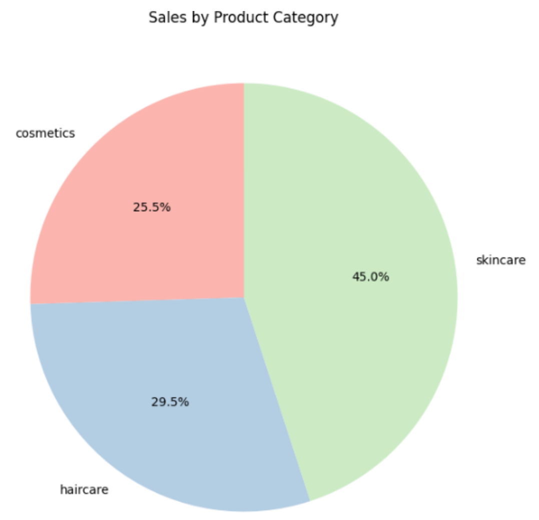


Objective 2: Examine sales performance categorized by product type.

Description: 45% of the business comes from skincare products, 29.5% from haircare, and 25.5% from cosmetics

Specification:

```
plt.figure(figsize=(8, 8))
grouped_data = data.groupby('Product type')['Number of products sold'].sum().reset_index()
plt.pie(grouped_data['Number of products sold'], labels=grouped_data['Product type'], autopct='%1.1f%%', startangle=90,colors=plt.cm.Pastel1.colors)
plt.title('Sales by Product Category')
plt.show()
```

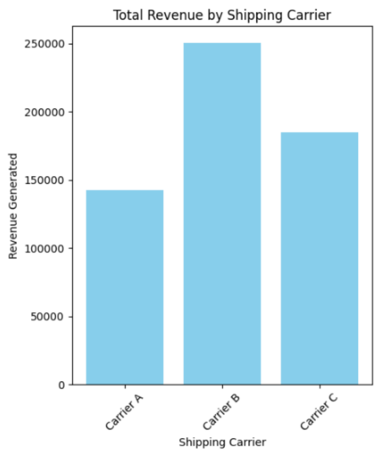


Objective 3: Calculate the total revenue generated from shipping carriers.

Description: The company is using three carriers for transportation, and Carrier B helps the company in generating more revenue.

Specification:

```
total_revenue = data.groupby('Shipping carriers')['Revenue generated'].sum().reset_index()
plt.figure(figsize=(5, 6))
plt.bar(total_revenue['Shipping carriers'], total_revenue['Revenue generated'], color='skyblue')
plt.title('Total Revenue by Shipping Carrier')
plt.xlabel('Shipping Carrier')
plt.ylabel('Revenue Generated')
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```



Objective 4: Evaluate the average lead time and average manufacturing costs for all products.

Description: Average lead time and Average Manufacturing Costs for all products of the company:

Specification:

```
avg_lead_time = data.groupby('Product type')['Lead time'].mean().reset_index()
avg_manufacturing_costs = data.groupby('Product type')['Manufacturing costs'].mean().reset_index()
result = pd.merge(avg_lead_time, avg_manufacturing_costs, on='Product type')
result.rename(columns={'Lead time': 'Average Lead Time', 'Manufacturing costs': 'Average Manufacturing Costs'}, inplace=True)
print(result)
```

```
  Product type  Average Lead Time  Average Manufacturing Costs
0    cosmetics          13.538462                    43.052740
1     haircare          18.705882                    48.457993
2     skincare          18.000000                    48.993157
```

Objective 5: Analyze the revenue generated by each SKU (Stock Keeping Unit).

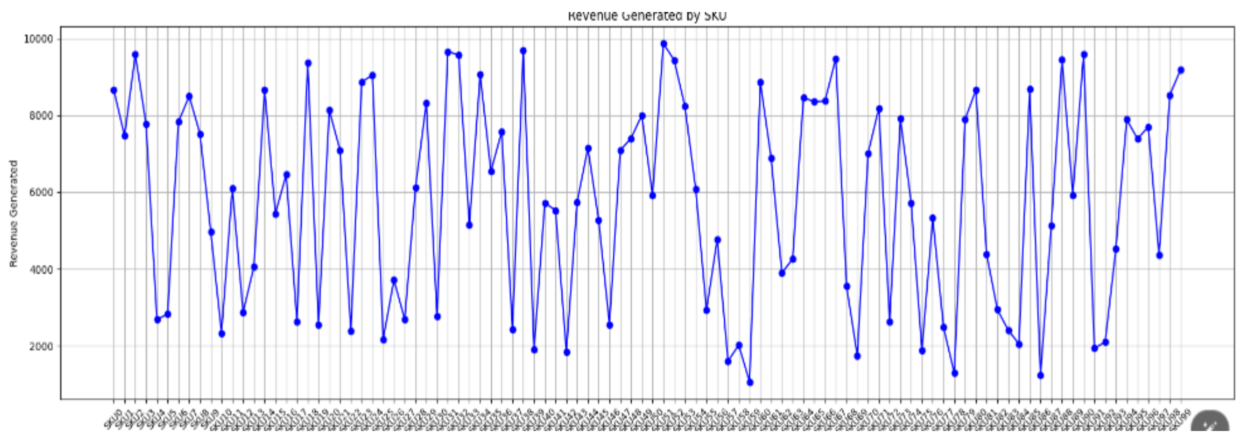
Description: Revenue generated by each SKU is known from the graph.

Specification:

```
plt.figure(figsize=(18, 6))
plt.plot(data['SKU'], data['Revenue generated'], marker='o', linestyle='-', color='b', label='Revenue generated')

plt.title('Revenue Generated by SKU')
plt.xlabel('SKU')
plt.ylabel('Revenue Generated')
plt.xticks(rotation=45,fontsize=9)
plt.grid(True)

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



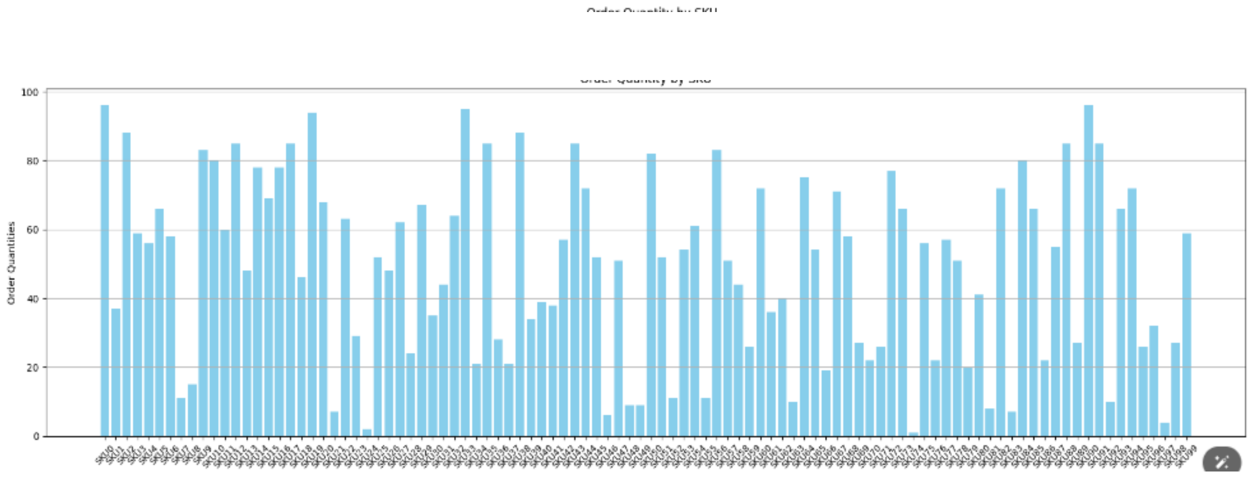
Objective 6: Investigate the order quantity of each SKU to understand demand patterns.

Description: Order quantity of each SKU was known from the graph.

Specification:

```
plt.figure(figsize=(18, 6))
plt.bar(data['SKU'], data['Order quantities'], color='skyblue')
plt.title('Order Quantity by SKU')

plt.xlabel('SKU')
plt.ylabel('Order Quantities')
plt.xticks(rotation=45,fontsize=9)
plt.grid(True, axis='y')
plt.tight_layout()
plt.show()
```

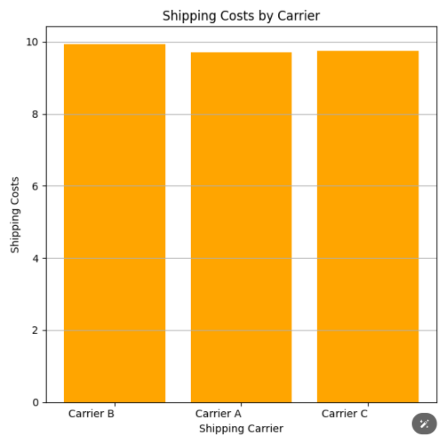


Objective 6: Assess the shipping costs associated with different carriers.

Description: In one of the above visualizations, we discovered that Carrier B helps the company in more revenue. It is also the most costly Carrier among the three.

Specification:

```
plt.figure(figsize=(6, 6))
plt.bar(data['Shipping carriers'], data['Shipping costs'], color='orange')
plt.title('Shipping Costs by Carrier')
plt.xlabel('Shipping Carrier')
plt.ylabel('Shipping Costs')
plt.xticks(rotation=0, ha='right')
plt.grid(axis='y')
plt.tight_layout()
plt.show()
```

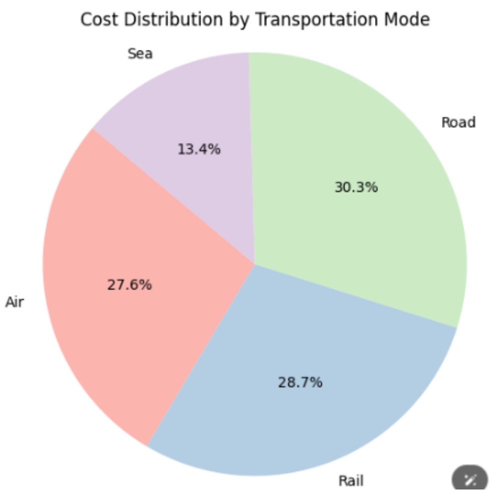


Objective 8: Explore the distribution of costs by transportation mode.

Description: The company spends more on Road and Rail modes of transportation for the transportation of Goods.

Specification:

```
plt.figure(figsize=(5, 5))
t_category = data.groupby('Transportation modes')['Costs'].sum().reset_index()
plt.pie(t_category['Costs'], labels=t_category['Transportation modes'], autopct='%1.1f%%', startangle=140 ,colors=plt.cm.Pastel1.colors)
plt.title('Cost Distribution by Transportation Mode')
plt.axis('equal')
plt.tight_layout()
plt.show
```



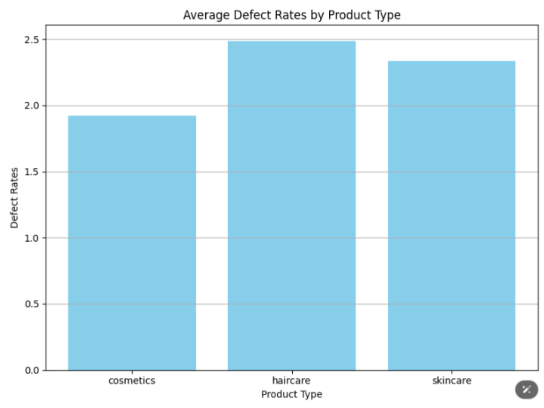
Objective 9: Calculate the average defect rate across all product types.

Description: The defect rate of haircare products is higher. Now let's have a look at the defect rates by mode of transportation.

Specification:

```
defect_rates_by_product = data.groupby('Product type')['Defect rates'].mean().reset_index()

plt.figure(figsize=(8, 6))
plt.bar(defect_rates_by_product['Product type'], defect_rates_by_product['Defect rates'], color='skyblue')
plt.title('Average Defect Rates by Product Type')
plt.xlabel('Product Type')
plt.ylabel('Defect Rates')
plt.xticks(rotation=0)
plt.grid(axis='y')
plt.tight_layout()
plt.show()
```

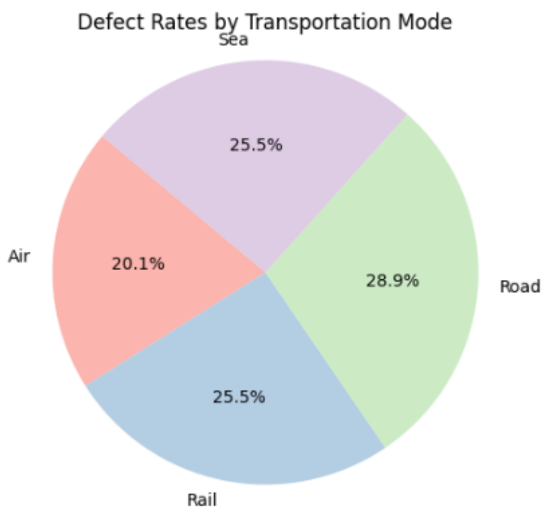


Objective 10: Analyze defect rates categorized by mode of transportation.

Description: Road transportation results in a higher defect rate, and Air transportation has the lowest defect rate.

Specification:

```
grouped_data = data.groupby('Transportation modes')['Defect rates'].mean().reset_index()
plt.figure(figsize=(6, 5))
plt.pie(grouped_data['Defect rates'], labels=grouped_data['Transportation modes'], autopct='%1.1f%%', startangle=140, colors=plt.cm.Pastel1.colors)
plt.title('Defect Rates by Transportation Mode')
plt.axis('equal')
plt.show()
```



Summary:

By integrating these libraries and tools, we successfully conducted a detailed supply chain analysis, examining various components such as product pricing, revenue generation, sales by product type, shipping carriers, lead times, manufacturing costs, SKU revenue, order quantities, shipping costs, transportation mode costs, and defect rates. This comprehensive approach allowed us to gain valuable insights into the supply chain operations of the company and identify areas for improvement.