QLik Project Documentation

Name - Rahul Singh
Email - Rahulsingh.primary@gmail.com

The Business Problem

This project aims to revolutionize supply chain management through data-driven insights using Qlik. Leveraging advanced analytics, it seeks to optimize logistics, forecasting, and inventory management, enhancing operational efficiency and responsiveness.
This transformative project endeavors to reshape the landscape of supply chain management by harnessing the power of Qlik's data-driven insights. Employing cutting-edge analytics, it strives to revolutionize key facets such as logistics, forecasting, and inventory management, with the overarching goal of elevating operational efficiency and responsiveness to new heights.

Business Requirements

Implement a robust data integration strategy to aggregate and centralize relevant data from diverse supply chain sources. Utilize Qlik's advanced visualization capabilities to create intuitive and dynamic dashboards, providing stakeholders with clear insights into the entire supply chain ecosystem. Leverage Qlik's advanced analytics features to analyse historical logistics data, identify patterns, and optimize transportation routes. Implement real-time tracking and monitoring solutions to enhance visibility into the movement of goods, reducing lead times and minimizing transportation costs. Implement real-time analytics to facilitate quick decision-making in response to unforeseen events or changes in demand, ensuring a proactive and responsive supply chain.

Literature Survey

1. Introduction

The digital transformation of supply chain management (SCM) has become increasingly crucial in recent years. Companies are leveraging advanced data analytics to optimize their operations, improve decision-making, and enhance overall efficiency. Qlik, a leading business intelligence (BI) and data visualization platform, plays a pivotal role in this transformation by providing powerful tools for data-driven insights.

2. Role of Data Analytics in Supply Chain Management

Data analytics has revolutionized SCM by enabling companies to analyze large volumes of data from various sources. Key areas impacted by data analytics include:

- **Logistics Optimization**: Advanced analytics help in route optimization, reducing transportation costs, and improving delivery times.
- **Demand Forecasting**: Predictive analytics enhance demand forecasting accuracy, enabling better inventory management and reducing stockouts.
- **Inventory Management**: Data-driven insights assist in maintaining optimal inventory levels, reducing holding costs, and improving turnover rates.
- **Supplier Relationship Management**: Analytics facilitate better supplier performance evaluation and risk management.

3. Qlik's Contribution to Supply Chain Analytics

Qlik provides a comprehensive suite of tools for data integration, visualization, and advanced analytics, making it an ideal platform for SCM. Key features include:

- **Associative Data Model**: Qlik's associative model allows users to explore data freely, uncovering hidden relationships and insights.
- **Real-time Data Integration**: Qlik supports real-time data integration from various sources, ensuring up-to-date and accurate analytics.
- **Interactive Dashboards**: Qlik's interactive dashboards provide intuitive visualizations, enabling users to make informed decisions quickly.
- **Advanced Analytics Integration**: Qlik integrates with advanced analytics tools such as R and Python, enhancing its analytical capabilities.

4. Case Studies and Applications

Several organizations have successfully implemented Qlik for SCM, demonstrating significant improvements in efficiency and responsiveness.

- **Logistics Optimization at XYZ Corp**: By leveraging Qlik's real-time data integration and interactive dashboards, XYZ Corp achieved a 15% reduction in transportation costs and improved delivery times by 20%.
- **Inventory Management at ABC Inc**: ABC Inc used Qlik's predictive analytics to optimize inventory levels, resulting in a 25% reduction in holding costs and a 30% improvement in inventory turnover.
- **Demand Forecasting at DEF Ltd**: DEF Ltd integrated Qlik with advanced analytics tools to enhance demand forecasting accuracy, reducing stockouts by 40% and improving customer satisfaction.

5. Challenges and Future Directions

Despite the benefits, implementing data-driven innovations in SCM presents several challenges:

- **Data Quality and Integration**: Ensuring high-quality data from multiple sources is critical for accurate analytics.

- **Change Management**: Organizations must adapt to new processes and technologies, which can be a significant hurdle.
- **Scalability**: As companies grow, their data analytics infrastructure must scale accordingly to handle increasing data volumes.

Future directions in data-driven SCM include the integration of artificial intelligence (AI) and machine learning (ML) for enhanced predictive analytics, the use of blockchain for improved transparency and security, and the adoption of Internet of Things (IoT) devices for real-time data collection.

6. Conclusion

The integration of Qlik into SCM has the potential to transform operations by providing data-driven insights that enhance logistics, forecasting, and inventory management. As companies continue to adopt advanced analytics, the landscape of SCM will evolve, leading to greater efficiency and responsiveness in the supply chain.

Social Or Business Impact.

Business Impact

1. **Operational Efficiency**: Qlik's analytics optimize logistics, forecasting, and inventory management, leading to cost savings and streamlined operations.
2. **Enhanced Decision-Making**: Clear, actionable insights improve decision-making at all levels, from strategic to operational.
3. **Competitive Advantage**: Data-driven supply chains respond quickly to market changes, increasing agility and resilience.
4. **Customer Satisfaction**: Better forecasting and inventory management ensure product availability, boosting customer satisfaction and loyalty.
5. **Cost Reduction**: Optimized logistics and inventory lower transportation and holding costs, improving profitability.
6. **Risk Management**: Qlik helps identify and mitigate supply chain risks, enhancing resilience.

Social Impact

1. **Sustainability**: Efficient resource use reduces waste and environmental impact.
2. **Economic Development**: Effective supply chains drive economic growth and job creation.
3. **Supply Chain Transparency**: Increased visibility promotes trust and accountability.
4. **Social Responsibility**: Improved monitoring ensures ethical sourcing and labor practices.
5. **Community Impact**: Reliable delivery of goods benefits local communities.
6. **Innovation and Education**: Adoption of advanced technologies fosters innovation and skill development.

Code and Screenshots

Libraries used - Numpy - pandas, matlplotlib, seaborn.

1. Loading Libraries and Data

- Import necessary libraries for data manipulation, visualization, and interactive plotting.
- Load the dataset from the input directory.

python
Copy code

```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
import plotly.graph_objs as go
from plotly.subplots import make_subplots
import plotly.express as px

%matplotlib inline

# Initialize Plotly
from plotly.offline import init_notebook_mode
init_notebook_mode(connected=True)

# Load dataset
df =
pd.read_csv('/kaggle/input/dataco-smart-supply-chain-for-big-data-analysis/Data
CoSupplyChainDataset.csv')
```

2. Data Preprocessing

- Display column names and initial data inspection.
- Select relevant features and create a copy for analysis.
- Remove unnecessary or duplicate columns.

python
Copy code

```
# Display columns
print(df.columns)

# Create a copy of the data with selected features
feature_list = [
    'Type', 'Benefit per order', 'Sales per customer', 'Delivery Status',
'Late_delivery_risk',
    'Category Name', 'Customer City', 'Customer Country', 'Customer Id',
'Customer Segment',
    'Customer State', 'Customer Zipcode', 'Department Name', 'Latitude',
'Longitude',
```

```
    'Market', 'Order City', 'Order Country', 'Order Customer Id', 'order date
(DateOrders)',
    'Order Id', 'Order Item Cardprod Id', 'Order Item Discount', 'Order Item
Discount Rate',
    'Order Item Id', 'Order Item Product Price', 'Order Item Profit Ratio',
'Order Item Quantity',
    'Sales', 'Order Item Total', 'Order Profit Per Order', 'Order Region',
'Order State',
    'Order Status', 'Order Zipcode', 'Product Card Id', 'Product Category Id',
'Product Description',
    'Product Image', 'Product Name', 'Product Price', 'Product Status',
'shipping date (DateOrders)',
    'Shipping Mode'
]

df1 = df[feature_list]
```

3. Data Visualization

- **Delivery Status Visualization**: Bar plot showing the number of orders by delivery status.

python
Copy code

```
data_delivery_status = df1.groupby(['Delivery Status'])['Order
Id'].count().reset_index(name='Number of Orders').sort_values(by='Number of
Orders', ascending=False)
fig = px.bar(data_delivery_status, x='Delivery Status', y='Number of Orders',
color='Number of Orders')
fig.show()
```

- **Delivery Status by Region**: Bar plot showing delivery status distribution by region.

python
Copy code

```
data_delivery_status_region = df1.groupby(['Delivery Status', 'Order
Region'])['Order Id'].count().reset_index(name='Number of
Orders').sort_values(by='Number of Orders', ascending=False)
fig = px.bar(data_delivery_status_region, x='Delivery Status', y='Number of
Orders', color='Order Region')
fig.show()
```

- **Top 20 Customers by Order Quantity**: Bar plot showing the top 20 customers based on the number of orders.

python
Copy code

```
df1['Customer_ID_STR'] = df1['Customer Id'].astype(str)
data_customers = df1.groupby(['Customer_ID_STR'])['Order
Id'].count().reset_index(name='Number of Orders').sort_values(by='Number of
Orders', ascending=False)
fig = px.bar(data_customers.head(20), x='Number of Orders',
y='Customer_ID_STR', color='Number of Orders')
fig.show()
```

- **Top 20 Customers by Profit**: Bar plot showing the top 20 customers based on the profit of orders.

python
Copy code

```
data_customers_profit = df1.groupby(['Customer_ID_STR'])['Order Profit Per
Order'].sum().reset_index(name='Profit of Orders').sort_values(by='Profit of
Orders', ascending=False)
fig = px.bar(data_customers_profit.head(20), x='Profit of Orders',
y='Customer_ID_STR', color='Profit of Orders')
fig.show()
```

- **Customer Segments**: Pie chart showing the distribution of orders across different customer segments.

python
Copy code

```
data_customer_segment = df1.groupby(['Customer Segment'])['Order
Id'].count().reset_index(name='Number of Orders').sort_values(by='Number of
Orders', ascending=False)
fig = px.pie(data_customer_segment, values='Number of Orders', names='Customer
Segment', title='Number of Orders by Customer Segment',
color_discrete_sequence=px.colors.sequential.RdBu)
fig.show()
```

- **Category Analysis**: Bar plot showing the number of orders across different categories.

python
Copy code

```
data_category_name = df1.groupby(['Category Name'])['Order
Id'].count().reset_index(name='Number of Orders').sort_values(by='Number of
Orders', ascending=True)
fig = px.bar(data_category_name, x='Number of Orders', y='Category Name',
color='Number of Orders')
fig.show()
```

- **Geographical Analysis**: Bar plots and choropleth maps showing the number of orders and profits by region and country.

python
Copy code
```
data_region = df1.groupby(['Order Region'])['Order
Id'].count().reset_index(name='Number of Orders').sort_values(by='Number of
Orders', ascending=True)
fig = px.bar(data_region, x='Number of Orders', y='Order Region', color='Number
of Orders')
fig.show()

data_countries = df1.groupby(['Order Country'])['Order
Id'].count().reset_index(name='Number of Orders').sort_values(by='Number of
Orders', ascending=True)
fig = px.bar(data_countries.head(20), x='Number of Orders', y='Order Country',
color='Number of Orders')
fig.show()

df_geo = df1.groupby(['Order Country', 'Order City'])['Order Profit Per
Order'].sum().reset_index(name='Profit of Orders').sort_values(by='Profit of
Orders', ascending=False)
fig = px.choropleth(df_geo, locationmode='country names', locations='Order
Country', color='Profit of Orders', hover_name='Order Country',
color_continuous_scale=px.colors.sequential.Plasma)
fig.show()
```

4. Sales Analysis

- **Sales by Country**: Bar plot showing sales by country.

python
Copy code
```
df_sales_country = df1.groupby(['Order
Country'])['Sales'].sum().reset_index(name='Sales of
Orders').sort_values(by='Sales of Orders', ascending=False)
fig = px.bar(df_sales_country.head(10), x='Sales of Orders', y='Order Country',
color='Sales of Orders')
fig.show()
```

- **Sales by Product Name**: Bar plot showing sales by product name.

python
Copy code
```
df_sales_product = df1.groupby(['Product
Name'])['Sales'].sum().reset_index(name='Sales of
Orders').sort_values(by='Sales of Orders', ascending=False)
fig = px.bar(df_sales_product.head(10), x='Sales of Orders', y='Product Name',
color='Sales of Orders')
fig.show()
```

- **Sales by Product and Delivery Status**: Bar plot showing sales by product name and delivery status.

python
Copy code
```
df_sales_pd = df1.groupby(['Product Name', 'Delivery
Status'])['Sales'].sum().reset_index(name='Sales of
Orders').sort_values(by='Sales of Orders', ascending=False)
fig = px.bar(df_sales_pd.head(10), x='Sales of Orders', y='Product Name',
color='Delivery Status')
fig.show()
```

- **Sales by Product and Order Region**: Bar plot showing sales by product name and order region.

python
Copy code
```
df_sales_pr = df1.groupby(['Product Name', 'Order
Region'])['Sales'].sum().reset_index(name='Sales of
Orders').sort_values(by='Sales of Orders', ascending=False)
fig = px.bar(df_sales_pr.head(10), x='Sales of Orders', y='Product Name',
color='Order Region')
fig.show()
```

- **Sales by Category Name**: Bar plot showing sales by category name.

python
Copy code
```
df_sales_category = df1.groupby(['Category
Name'])['Sales'].sum().reset_index(name='Sales of
Orders').sort_values(by='Sales of Orders', ascending=False)
fig = px.bar(df_sales_category.head(10), x='Sales of Orders', y='Category
Name', color='Sales of Orders')
fig.show()
```

- **Sales by Type of Payment**: Bar plot showing sales by type of payment.

python
Copy code
```
df_sales_type = df1.groupby(['Type'])['Sales'].sum().reset_index(name='Sales of
Orders').sort_values(by='Sales of Orders', ascending=False)
fig = px.bar(df_sales_type.head(10), x='Sales of Orders', y='Type',
color='Sales of Orders')
fig.show()

df_sales_tp = df1.groupby(['Type', 'Product
Name'])['Sales'].sum().reset_index(name='Sales of
Orders').sort_values(by='Sales of Orders', ascending=False)
```

```
fig = px.bar(df_sales_tp.head(10), x='Sales of Orders', y='Type',
color='Product Name')
fig.show()
```

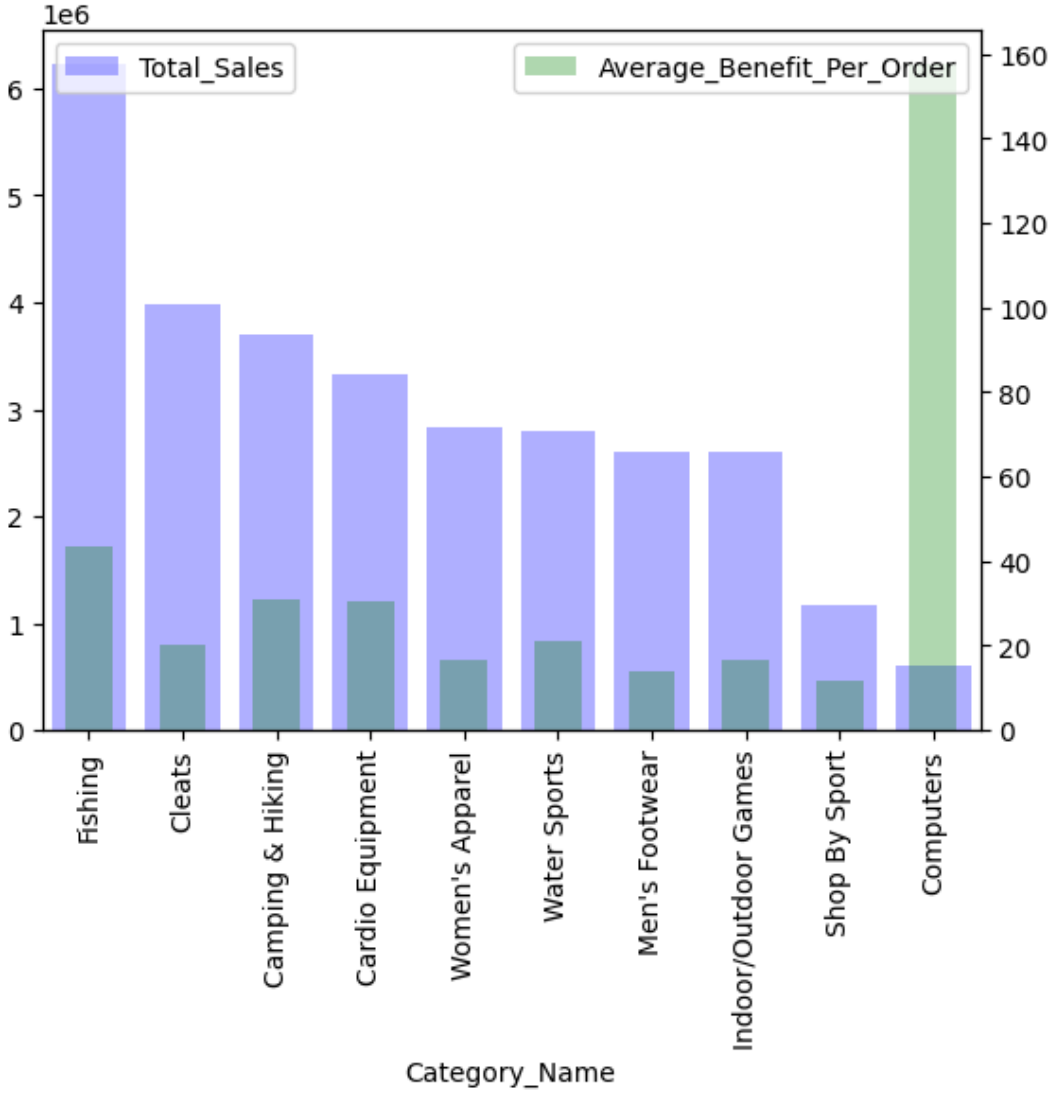
5. Date and Sales Analysis

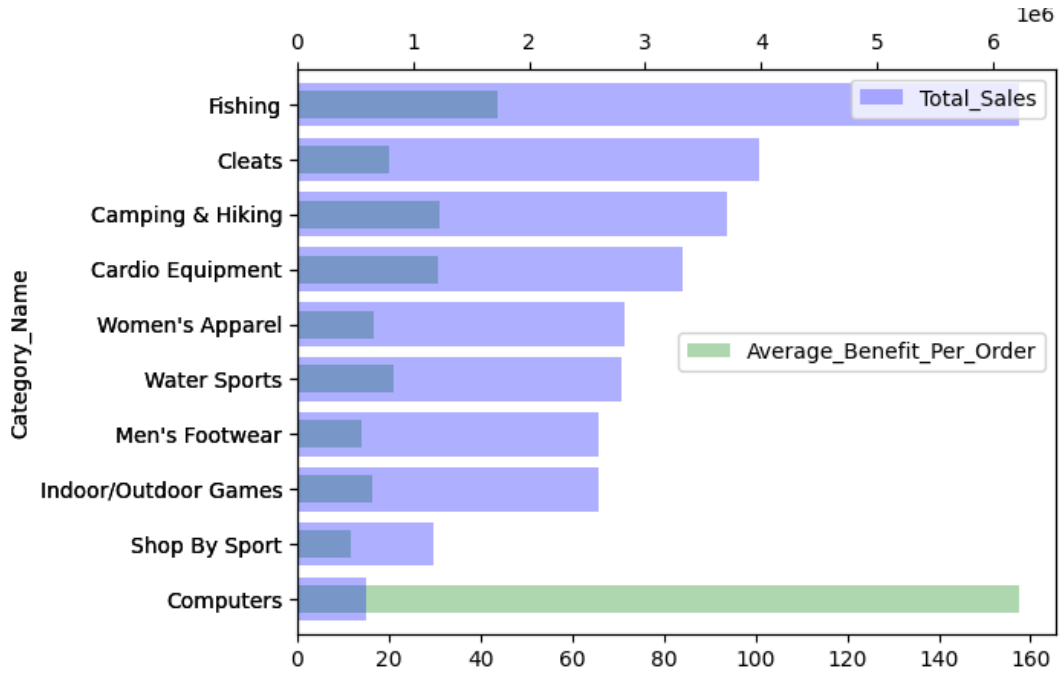
- **Sales by Date**: Time series analysis of sales data.

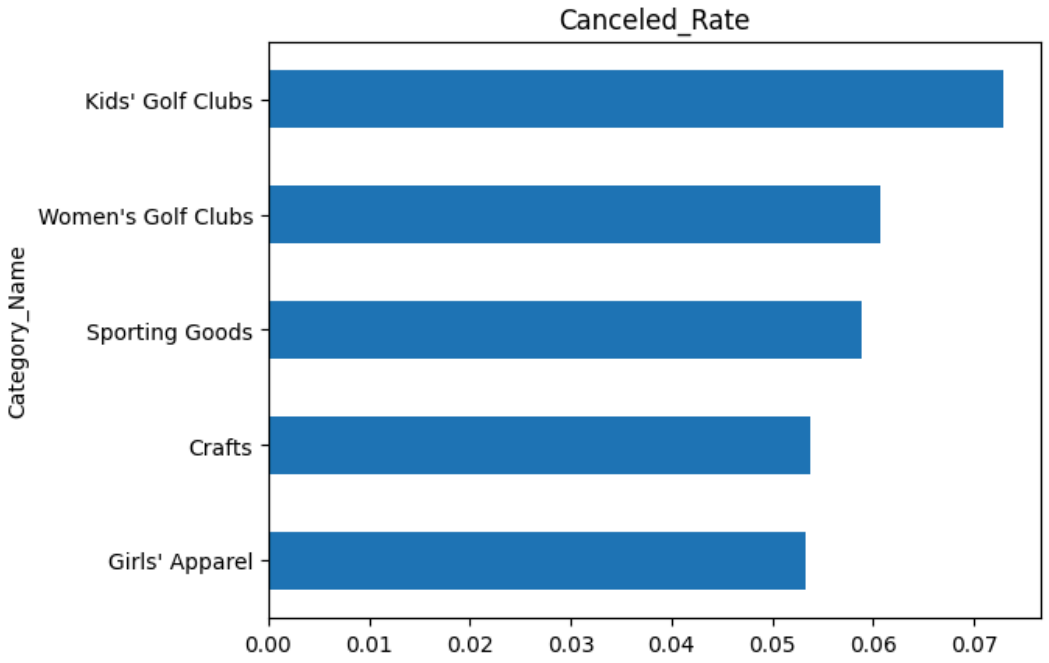
python
Copy code

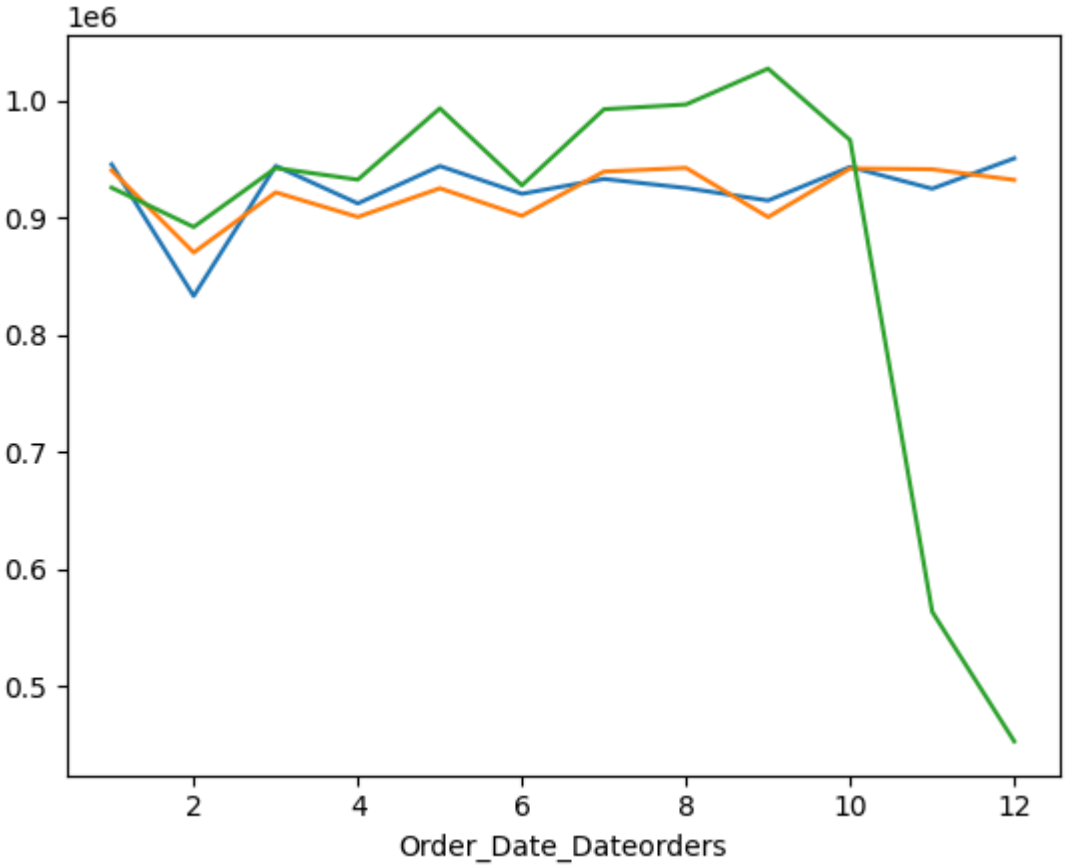
```
import datetime as dt

data_orderdate = df[['order date (DateOrders)', 'Sales']]
data_orderdate['order_date'] = pd.to_datetime(data_orderdate['order date
(DateOrders)'])
data_orderdate["Quarter"] = data_orderdate['order_date'].dt.quarter
data_orderdate["Month"] = data_orderdate['order_date'].dt.month
data_orderdate['order_year'] = data_orderdate['order_date'].dt.year
data_orderdate['order_dayofweek'] = data_orderdate['order_date'].dt.dayofweek
data_orderdate['order_day'] = data_orderdate['order_date'].dt.day
data_orderdate['order_month'] = data_orderdate['order_date'].dt.month

data_orderdate['Month_Year'] = data_orderdate['order_date'].dt.to_period('M')

df_sales_date =
data_orderdate.groupby(['Month_Year'])['Sales'].sum().reset_index(name='Sales
of Orders').sort_values(by='Month_Year', ascending=True)

fig = px.line(df_sales_date, x='Month_Year', y='Sales of Orders')
fig.show()
```

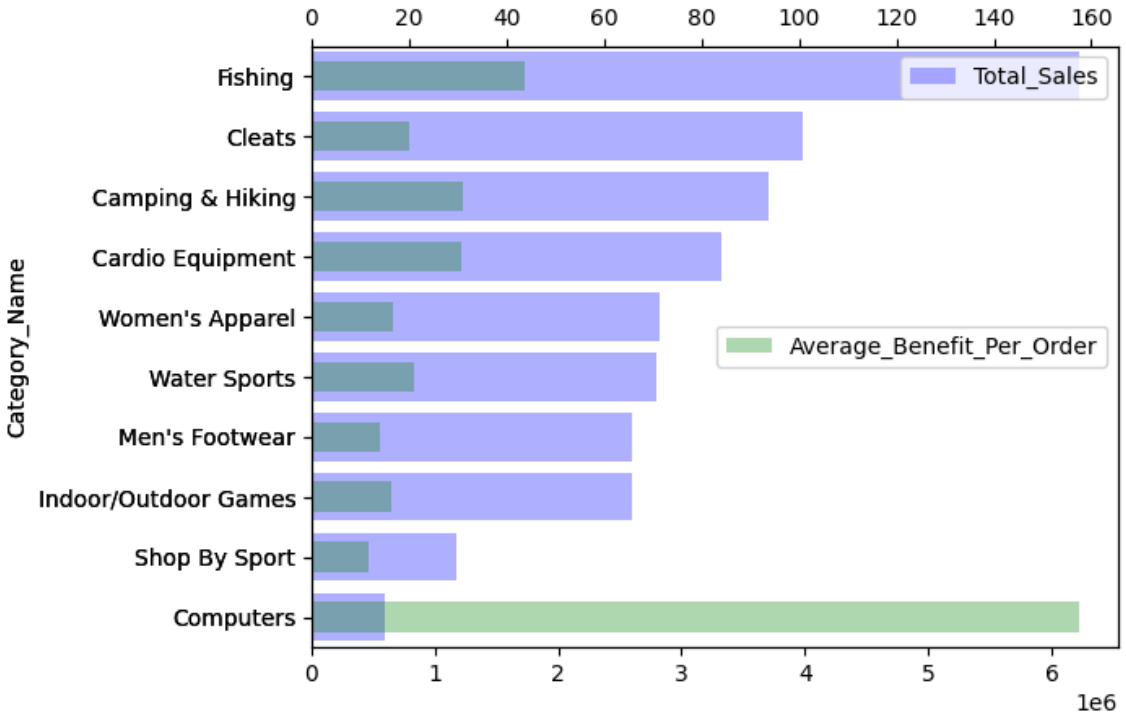
	Type	Days_For_Shipping_Real	Days_For_Shipment_Scheduled	Benefit_Per_Order	Sales_Per_Customer	Delivery_Status	Late
0	DEBIT	3	4	91.250000	314.640015	Advance shipping	0
1	TRANSFER	5	4	-249.089996	311.359985	Late delivery	1
2	CASH	4	4	-247.779999	309.720001	Shipping on time	0
3	DEBIT	3	4	22.860001	304.809998	Advance shipping	0
4	PAYMENT	2	4	134.210007	298.250000	Advance shipping	0

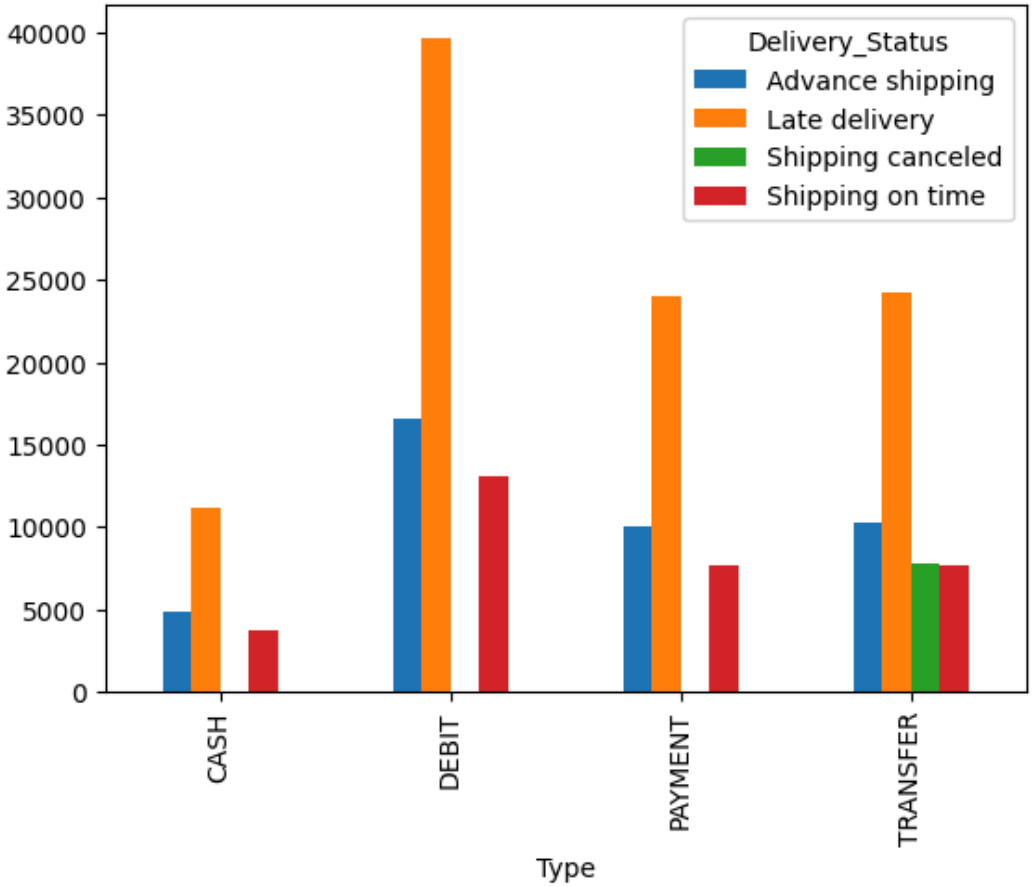


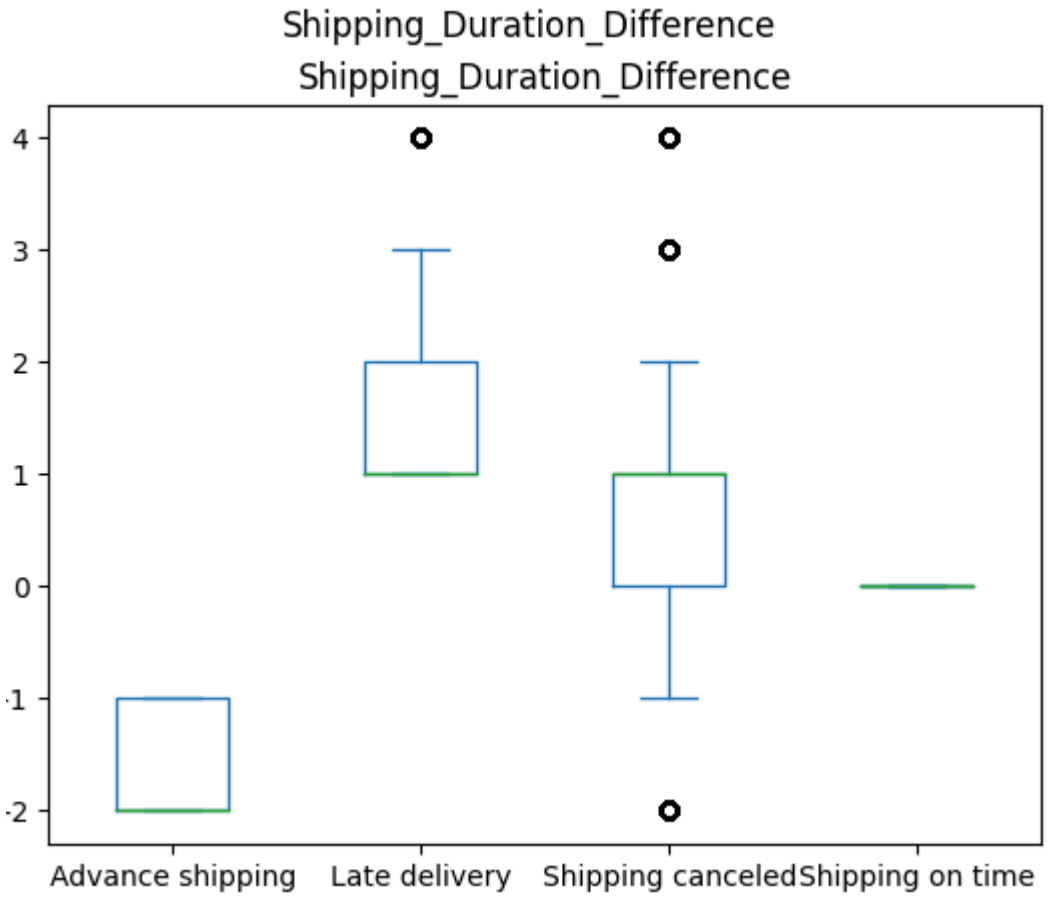

1e6
0
1
2
3
4
5
6
Total_Sales
Average_Benefit_Per_Order
Category_Name
Fishing
Cleats
Camping & Hiking
Cardio Equipment
Women's Apparel
Water Sports
Men's Footwear
Indoor/Outdoor Games
Shop By Sport
Computers
0
20
40
60
80
100
120
140
160

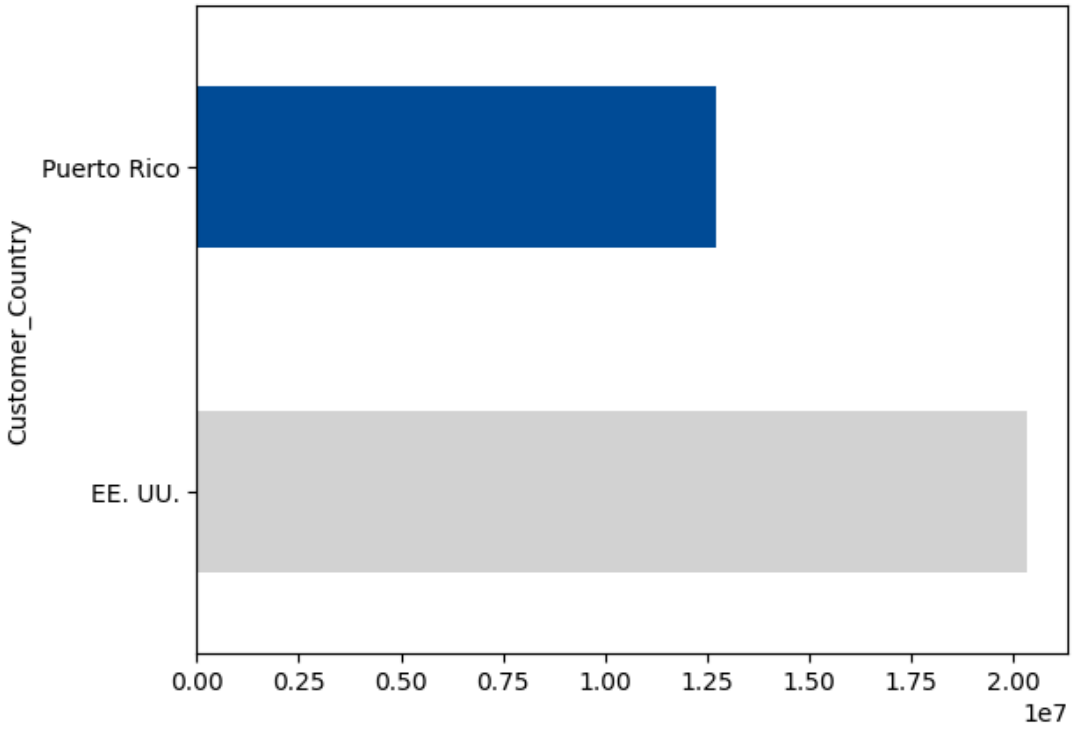

Canceled_Rate
Category_Name
Kids' Golf Clubs
Women's Golf Clubs
Sporting Goods
Crafts
Girls' Apparel
0.00
0.01
0.02
0.03
0.04
0.05
0.06
0.07

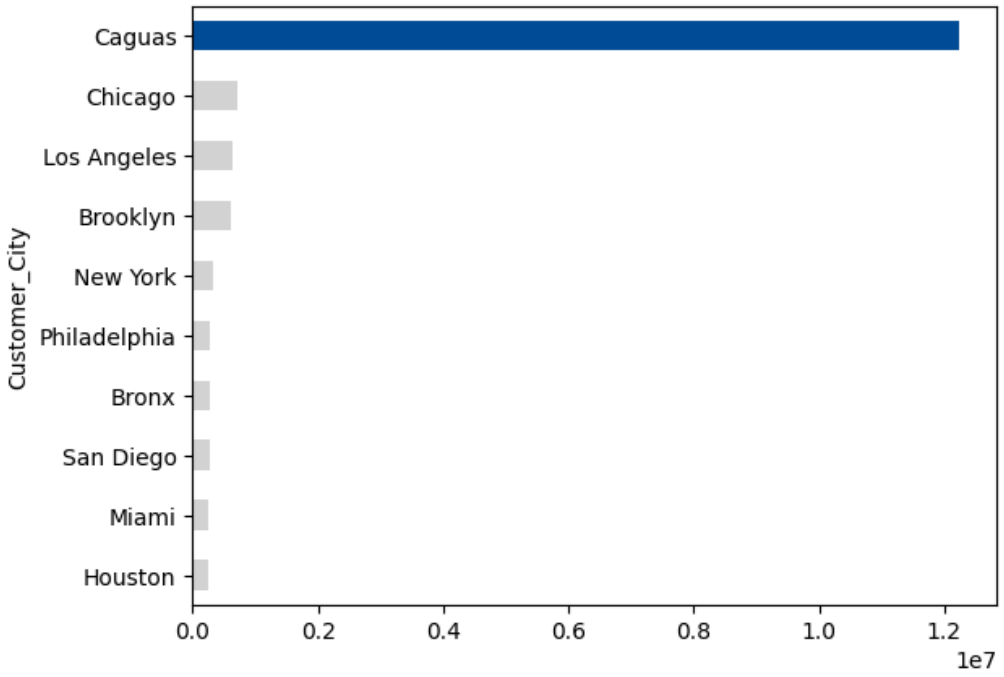

1e6
1.0
0.9
0.8
0.7
0.6
0.5
2
4
6
8
10
12
Order_Date_Dateorders

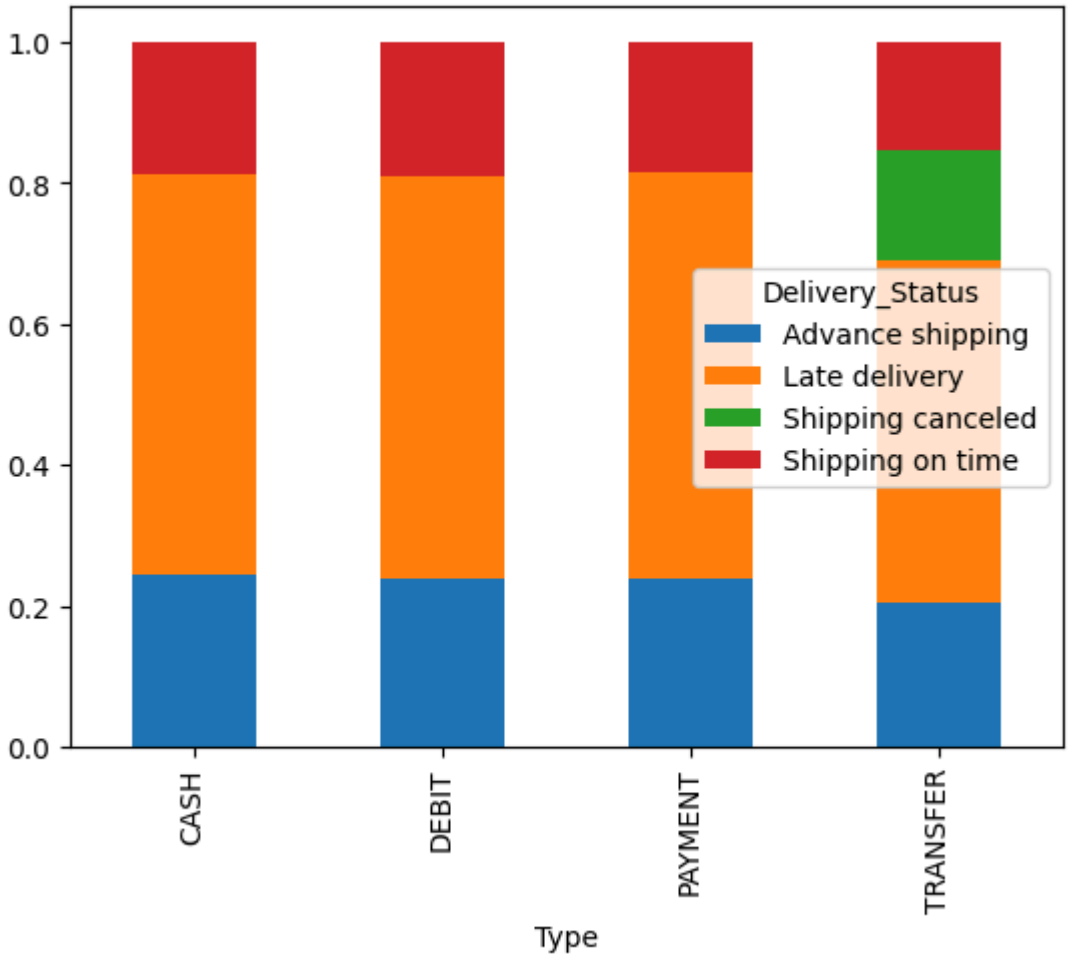

0
20
40
60
80
100
120
140
160
Fishing
Cleats
Camping & Hiking
Cardio Equipment
Women's Apparel
Water Sports
Men's Footwear
Indoor/Outdoor Games
Shop By Sport
Computers
Category_Name
Total_Sales
Average_Benefit_Per_Order
0
1
2
3
4
5
6
1e6


Delivery_Status
Advance shipping
Late delivery
Shipping canceled
Shipping on time
40000
35000
30000
25000
20000
15000
10000
5000
0
CASH
DEBIT
PAYMENT
TRANSFER
Type


Shipping_Duration_Difference
Shipping_Duration_Difference
4
3
2
1
0
-1
-2
Advance shipping
Late delivery
Shipping canceled
Shipping on time


Customer_Country
Puerto Rico
EE. UU.
0.00
0.25
0.50
0.75
1.00
1.25
1.50
1.75
2.00
1e7


Caguas
Chicago
Los Angeles
Brooklyn
New York
Philadelphia
Bronx
San Diego
Miami
Houston
Customer_City
0.0
0.2
0.4
0.6
0.8
1.0
1.2
1e7


1.0
0.8
0.6
0.4
0.2
0.0
Delivery_Status
Advance shipping
Late delivery
Shipping canceled
Shipping on time
CASH
DEBIT
PAYMENT
TRANSFER
Type