Project Report

E-commerce Sales Insights & Strategic Recommendations

Probability and Statistics for Artificial Intelligence (AAI-500-IN1)

University of San Diego

Presentation Link:(Youtube) https://youtu.be/j9ZnU0MSiHE
Github: **https://github.com/aishwaryagulhane05/AAI-500-IN1_PROJECT**

Submitted By:
- Yogesh Sangwilkar
- Meghesh Saini
- Aishwarya Gulhane

Date: 23rd June, 2025

Table of Contents

Introduction

E-commerce has rapidly transformed retail by enabling data-driven customer engagement and performance analysis. This project explores a UK-based B2C transactional dataset with over 500,000 records to uncover key sales insights, customer behavior, and operational challenges like cancellations and pricing variability.

The analysis covers time-based sales trends (monthly, weekly, weekday-wise), regional and product performance, and customer segmentation. Metrics such as Revenue, RFM (Recency, Frequency, Monetary) scores, Cohort Analysis, and cancellation impact are used to assess business performance.

Visual tools like bar charts, line graphs, pie charts, heatmaps, and advanced techniques like RFM segmentation and CLTV support in-depth insights. The project also includes revenue forecasting and strategic recommendations to increase profitability, reduce cancellations, and enhance customer retention.

This end-to-end analysis offers actionable insights and predictive capabilities to support smarter business decisions in the evolving e-commerce landscape.

Dataset

Source: **E-commerce Business Transaction (Kaggle)**

Data Description

This dataset contains sales transactions from a UK-based e-commerce company specializing in gifts and homewares for both adults and children. The shop has been operational online since 2007 and serves a global customer base, including end-users and small businesses that purchase in bulk for retail.

- **Time Period:** December 2018 to December 2019
- **Number of Rows:** ~500,000
- **Number of Columns:** 8

Column Details:

- **TransactionNo** *(Categorical)*: A unique, six-digit identifier for each transaction. Transactions starting with 'C' indicate cancellations.
- **Date** *(Numeric/Datetime)*: The date and time when the transaction occurred.
- **ProductNo** *(Categorical)*: A five or six-digit unique identifier for each product.
- **Product** *(Categorical)*: Name of the item sold.
- **Price** *(Numeric)*: Unit price of the item in British Pounds (£).
- **Quantity** *(Numeric)*: Number of units purchased. Negative values denote cancelled transactions.
- **CustomerNo** *(Categorical)*: A five-digit unique identifier for each customer.
- **Country** *(Categorical)*: The country where the customer is located.

Additional Context:

- The dataset contains a small percentage of cancelled orders, mainly due to out-of-stock products.
- Customers often cancel an order if not all items can be fulfilled, preferring complete deliveries.

Project Structure

```
ecommerce-sales-project/
│
├── 📁 data/
│   ├── raw/                          # Original dataset from Kaggle
│   ├── processed/                          # Cleaned & transformed datasets
│   └── data_dictionary.md                  # Notes about variables and schema
├── 📁 notebooks/
│   ├── 01_data_cleaning.ipynb        # Data wrangling and missing value handling
│   ├── 02_eda_part1.ipynb            # Visualizations, outlier detection, stats
│   ├── 02_eda_part1.ipynb            # Visualizations, outlier detection, stats
│   ├── 03_modeling.ipynb             # Modeling and evaluation
├── 📁 visuals/
   ├── data_cleansing_charts/             # Histogram, boxplot, pie chart, etc.
│   ├── eda_charts/                         # Histogram, boxplot, pie chart, etc.
│   └── model_charts/                       # Forecasts etc.
├── 📁 reports/
│   ├── Final-Project-Report-Group3.pdf     # Final technical report
├── 📁 presentation/
│   ├── Final-Project-Presentation-Group3.mp4
│   └── slides.pptx                          # Slide deck used in video
├── 📁 meta/
│   ├── team_contacts.txt                   # Team contact details
│   ├── meeting_notes.md              # Weekly sync-up meeting notes
│   ├── role_assignment.md            # Roles: prep, EDA, modeling, report
│   └── ai_usage_notes.md             # Notes on ChatGPT or Copilot usage
│
├── requirements.txt                        # Libraries used
├── README.md                               # Project intro & how to run
└── .gitignore                        # Ignore unnecessary files
```

Data Cleaning/Preparation

Contributor: **Yogesh Sangwikar**

The goal of this phase is to transform the raw sales transaction data into a clean and accurate form for analysis.

This involves removing missing, duplicated, and invalid records, and converting columns to their appropriate data types to ensure data quality and integrity.

Summary of Cleaning Steps:

1. Import libraries.
2. Load raw data.
3. Check nulls and drop them.
4. Remove duplicate transactions.
5. Convert data types.
6. Handle peculiarities (e.g. canceled transactions).
7. Enrich data with Year, Month, Week, Region.
8. Save the cleaned file.

Initial Data Overview

- **Source:** ~/data/raw/Sales Transaction v.4a.csv
- **Shape of raw data:** 536,350 rows × 8 columns
- **Columns:** Date, CustomerNo, Country, SKU, Product, Price, Quantity, TransactionNo

Steps Undertaken in Data Cleaning

Handling Missing Values

- Checked for nulls across all columns:
- Identified **~55 missing entries** in CustomerNo (~0.01% of the data).
    - Decision: Dropped all null rows

Observation: This confirms that there are null values in customerNo

need to check how many records are having null values

```
null_val_count = sales_data['CustomerNo'].isna().sum()
null_val_count
```

```
55
```

There are 55 records in CustomerNo column which have null values

Since these records are very low (55 / 536350) , i.e just 0.01 % , we easily take a call to delete them as it will not effect our anlaysis and recommendation in any way !

```
# deleting the null values from the main file itself
sales_data.dropna(inplace=True)
# confirm whether 55 records have been deleted
sales_data.shape
```

```
(536295, 8)
```

we see that the number of records are now 536295 which confirms that 55 records have been deleted

- **Rows after removal:** 536,295

Removing Duplicate Records

- Checked for duplicate transactions:
  - Found **~5,200 duplicate records**.
  - Decision: Removed duplicate rows

```
# to check number of duplicate records
sales_data.duplicated().sum()
```

```
5200
```

There are 5200 records having duplicate records , so we need to delete them.!

```
# Drop duplicate rows and keep the first occurrence
sales_data = sales_data.drop_duplicates()
sales_data.shape
```

```
(531095, 8)
```

We can see that now the records have come down to 531095 (after deleting 5200 rows from earlier file which had 536,295 records)

```
# check whether there are still any duplicate values now ?
sales_data.duplicated().sum()
```

```
0
```

  - **Rows after removal:** 531,095

Correcting Data Types

- Converted Date to datetime:

```
[24]: # astype
      # we had to use, format = Mixed as the data was very inconsitent in format
      sales_data['Date'] = pd.to_datetime(sales_data['Date'], format='mixed', dayfirst=False, errors='coerce')
      sales_data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 536295 entries, 0 to 536349
Data columns (total 8 columns):
 #   Column         Non-Null Count   Dtype
---  ------         --------------   -----
 0   TransactionNo  536295 non-null  object
 1   Date           536295 non-null  datetime64[ns]
 2   ProductNo      536295 non-null  object
 3   ProductName    536295 non-null  object
 4   Price          536295 non-null  float64
 5   Quantity       536295 non-null  int64
 6   CustomerNo     536295 non-null  float64
 7   Country        536295 non-null  object
dtypes: datetime64[ns](1), float64(2), int64(1), object(4)
memory usage: 36.8+ MB
```

Handling Peculiarities

- Investigated cancellation transactions indicated by TransactionNo starting with "C".
- Analyzed negative Quantity indicating canceled sales

```
In [5]: sales_data.tail(5)
```

Out[5]:

	TransactionNo	Date	ProductNo	ProductName	Price	Quantity	CustomerNo	Country
536345	C536548	12/1/2018	22168	Organiser Wood Antique White	18.96	-2	12472.0	Germany
536346	C536548	12/1/2018	21218	Red Spotty Biscuit Tin	14.09	-3	12472.0	Germany
536347	C536548	12/1/2018	20957	Porcelain Hanging Bell Small	11.74	-1	12472.0	Germany
536348	C536548	12/1/2018	22580	Advent Calendar Gingham Sack	16.35	-4	12472.0	Germany
536349	C536548	12/1/2018	22767	Triple Photo Frame Cornice	20.45	-2	12472.0	Germany

Observation: few values in Quanity column seem to have negative values !

On reading about this data , we got to know that negative quantity values are corresponding to the sales cancellation !

This can also be identified, as we see a "C" alphabet in front of the transaction no data

These peculiarities of data need to be considered while processing!

- Decision: Retained canceled transactions for cancellation impact analysis.

Adding Derived Time Columns

- Derived columns Year, Month, Week to facilitate granular analysis

```
# Assuming 'Date' is already a datetime column, add year month and week columns
sales_data['Year'] = sales_data['Date'].dt.year
sales_data['Month'] = sales_data['Date'].dt.month
sales_data['Week'] = sales_data['Date'].dt.isocalendar().week
sales_data['weekday'] = sales_data['Date'].dt.day_name()
sales_data.head()
```

	TransactionNo	Date	ProductNo	ProductName	Price	Quantity	CustomerNo	Country	Year	Month	Week	weekday
0	581482	2019-12-09	22485	Set Of 2 Wooden Market Crates	21.47	12	17490.0	United Kingdom	2019	12	50	Monday
1	581475	2019-12-09	22596	Christmas Star Wish List Chalkboard	10.65	36	13069.0	United Kingdom	2019	12	50	Monday
2	581475	2019-12-09	23235	Storage Tin Vintage Leaf	11.53	12	13069.0	United Kingdom	2019	12	50	Monday
3	581475	2019-12-09	23272	Tree T-Light Holder Willie Winkie	10.65	12	13069.0	United Kingdom	2019	12	50	Monday
4	581475	2019-12-09	23239	Set Of 4 Knick Knack Tins Poppies	11.94	6	13069.0	United Kingdom	2019	12	50	Monday

Feature Engineering

- Added a new column TotalSales by multiplying Price and Quantity to capture the total transaction value per row, enabling deeper revenue and profitability analyses.

Exporting Cleaned Data

- **Saved cleaned file: as** ~data/processed/01_data_cleaning.csv
- **Final Shape:** 531,095 rows × 14 columns

Summary of Changes

Cleaning Step	Rows Before	Rows After	Columns Impacted	Outcome
Check for Missing Values	536,350	536,295	CustomerNo	Dropped all NaN rows
Remove Duplicates	536,295	531,095	All columns	Removed exact duplicate transaction records
Convert Date to Datetime	—	—	Date	Enabled time-based operations
Convert Numerical Columns	—	—	CustomerNo, Price, Quantity	Converted columns to numeric types
Handle Cancellations	—	—	TransactionNo, Quantity	Flagged cancellation transactions

Add Derived Columns	—	—	Year, Month, Week	Derived new time-based columns
Save Clean Data	—	—	All columns	Saved cleaned dataset as CSV file

The data cleaning process successfully prepared the raw sales data for analysis by removing incomplete and duplicate entries, ensuring proper data types, and enriching the dataset with time-based features. The cleaned data is now suitable for downstream analyses such as sales trend exploration, product performance, and customer behavior analysis.

Exploratory Data Analysis - Part 1

Contributor: **Meghesh Saini**

Loading Cleaned Data

The cleaned data after the data cleaning process is taken and further taken for processing and performing exploratory analysis.

- **Source:** ~/data/processed/01_data_cleaning.csv
- df and df_cancelled are loaded from the same file, separating cancellation transaction
    - df = data frame having all the active transactions
    - df_positive= Sales values that are positive (not canceled)
    - df_cancelled= data frame having all cancelled transactions.

Total Sales per Country

```
df_positive = df[df["TotalSales"] > 0] #creating a new dataset of positive sales value for further calculations

# Group by Country and sum the TotalSales
sales_by_country = df_positive.groupby("Country")["TotalSales"].sum().reset_index()

# Sort by TotalSales in descending order (optional)
sales_by_country = sales_by_country.sort_values(by="TotalSales", ascending=False).reset_index(drop=True)

# Show the result
print('The total sales in Dollars grouped by countries \n')
print(sales_by_country)
print('The total number of values in this dataset is: ' , df_positive.shape[0])
```

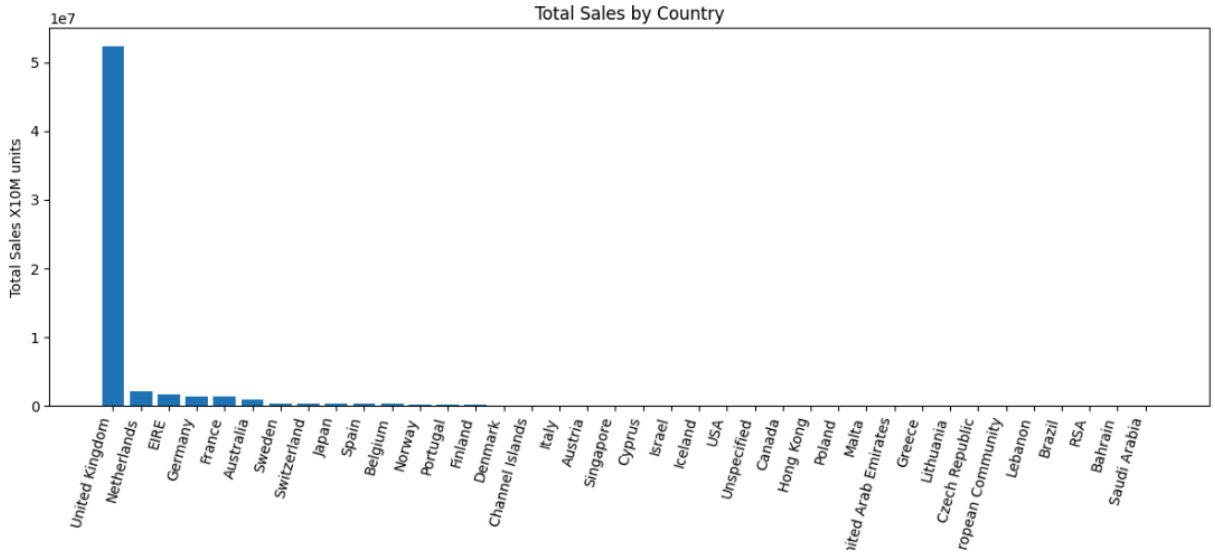
```
The total sales in Dollars grouped by countries

                Country   TotalSales
0        United Kingdom  52346795.60
1           Netherlands   2151553.59
2                  EIRE   1711819.39
3               Germany   1369839.62
4                France   1329903.39
5             Australia    995414.01
6                Sweden    401879.89
7           Switzerland    361691.96
8                 Japan    293155.44
9                 Spain    280843.80
10              Belgium    272131.88
11               Norway    188612.52
12             Portugal    175959.28
13              Finland    120972.15
14              Denmark    101083.99
```

Observation: Shows total sales across all countries is given in the data above.
Finding: Countries like the United Kingdom dominate sales as the company is based in the UK.
Recommendation: Focus marketing and expansion efforts in top-performing countries.

Bar Graph of Country and Total Sales in the country

```
#Plotting the bar graph of Country and Sale to get an Idea of global reach
print('Plotting Total sales Country wise')
plt.figure(figsize=(12, 6))
plt.bar(sales_by_country["Country"], sales_by_country["TotalSales"])
plt.xticks(rotation=75, ha='right')
plt.xlabel("Country")
plt.ylabel("Total Sales X10M units")
plt.title("Total Sales by Country")
plt.tight_layout()
plt.show() #plotting the results of sale value by country
```

```
Plotting Total sales Country wise
```



Observation: Visualizes total sales across countries.
Finding: Clear trend of higher sales from certain countries.
Recommendation: Allocate resources for high-performing markets.

It is clear that since the company is based in the UK, it has the highest sale value in that country. It is also clear that the next 5 countries, Netherlands, Ireland, Germany, France and Australia can also perform better if campaigns are run in these countries.

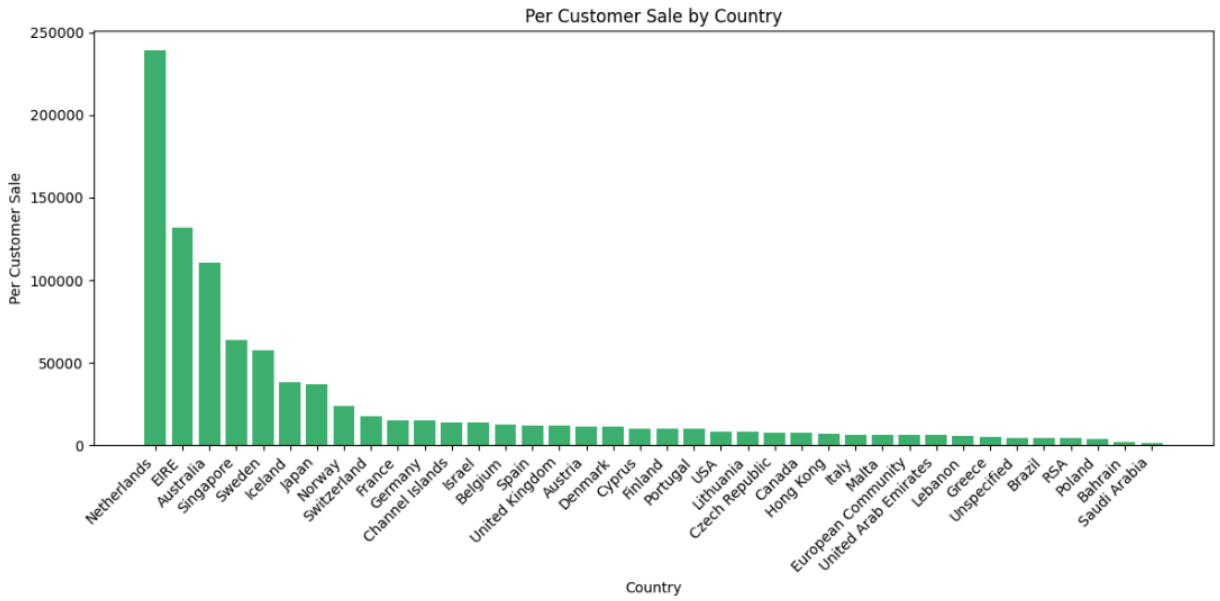
Per-Capita Sales (Total Sale ÷ Number of Customers)

```
#Total sale per Capita = Total sale value/No of customers in that country

# Total sales by country
total_salesc = df_positive.groupby("Country")["TotalSales"].sum()

# Number of Unique customers by country
unique_customers = df_positive.groupby("Country")["CustomerNo"].nunique()

# Per Customer sales = total sales / unique customers
per_capita_sales = (total_salesc / unique_customers).astype(int).reset_index()
per_capita_sales.columns = ["Country", "Per Customer Sale"]

# Sorting in descending order and resetting index
per_capita_sales = per_capita_sales.sort_values(by="Per Customer Sale", ascending=False).reset_index(drop=True)

# Show the values
print('The value of per customer sale based on (total sale in the country / no on customers in the country)\n')
print(per_capita_sales)
```

```
The value of per customer sale based on (total sale in the country / no on customers in the country)

                Country  Per Customer Sale
0           Netherlands             239061
1                  EIRE             131678
2             Australia             110601
3             Singapore              63480
4                Sweden              57411
5               Iceland              38307
6                 Japan              36644
7                Norway              23576
8           Switzerland              17223
9                France              15286
10              Germany              15053
11      Channel Islands              13704
12               Israel              13680
13              Belgium              12369
14                Spain              12210
15       United Kingdom              12159
16              Austria              11524
17              Denmark              11231
```

Per Customer sale is the sale value obtained by dividing total sales in the country by number of customers in the country. It gives a picture of how well the company is performing in the country based on this value.

The graph shows per customer sales by country



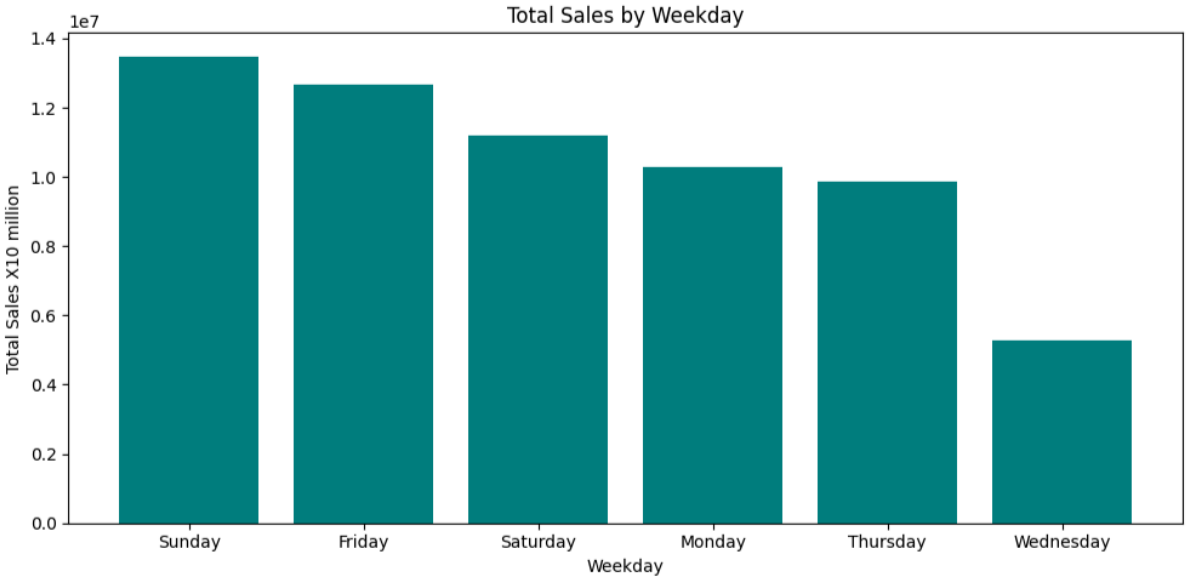
Observation: Shows average revenue contribution per customer by country.

Finding: Enables identifying premium or high-value markets.

Recommendation: Tailor pricing and promotion strategies per market segment.

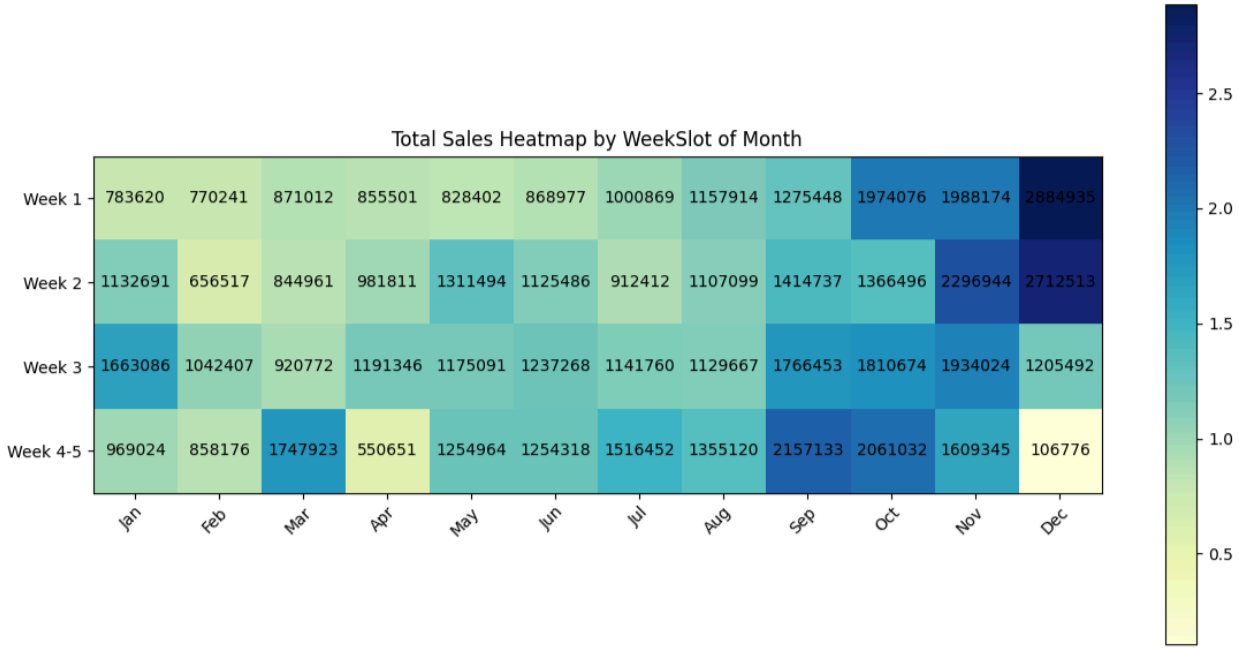
Weekday Sales Analysis

The following graph shows sales on the days of the week



Observation: Identifies which weekdays drive the highest sales.
Finding: Certain days (e.g., weekends) dominate.
Recommendation: Target advertisements and special offers for high-traffic days.

Weekday Slot per Month Heatmap



Observation: Shows sales trend across week numbers and months.
Finding: Specific combinations (e.g., Month X, Week Y) dominate sales.
Recommendation: Allocate staff and marketing resources accordingly.

The heat map shows the sales values in that week of the month. The darker shades represent higher value of sales and lighter shades represent lower value of sales.

Total Sales by WeekSlot

```
#Calculating the total value (Column totals from above)

df["WeekSlot"] = df_positive["Day"].apply(assign_week_slot)

# Group total sales by WeekSlot (all months combined)
total_sales_by_week = df_positive.groupby("WeekSlot")["TotalSales"].sum().reindex(["Week 1", "Week 2", "Week 3", "Week 4-5"])

# Display values
print('Adding up the rows and getting the week total for each week')
print(total_sales_by_week)
```

```
Adding up the rows and getting the week total for each week
WeekSlot
Week 1      15259174.13
Week 2      15863167.52
Week 3      16218044.15
Week 4-5    15440918.74
Name: TotalSales, dtype: float64
```

Observation: Aggregation of sales by weeks within a month.

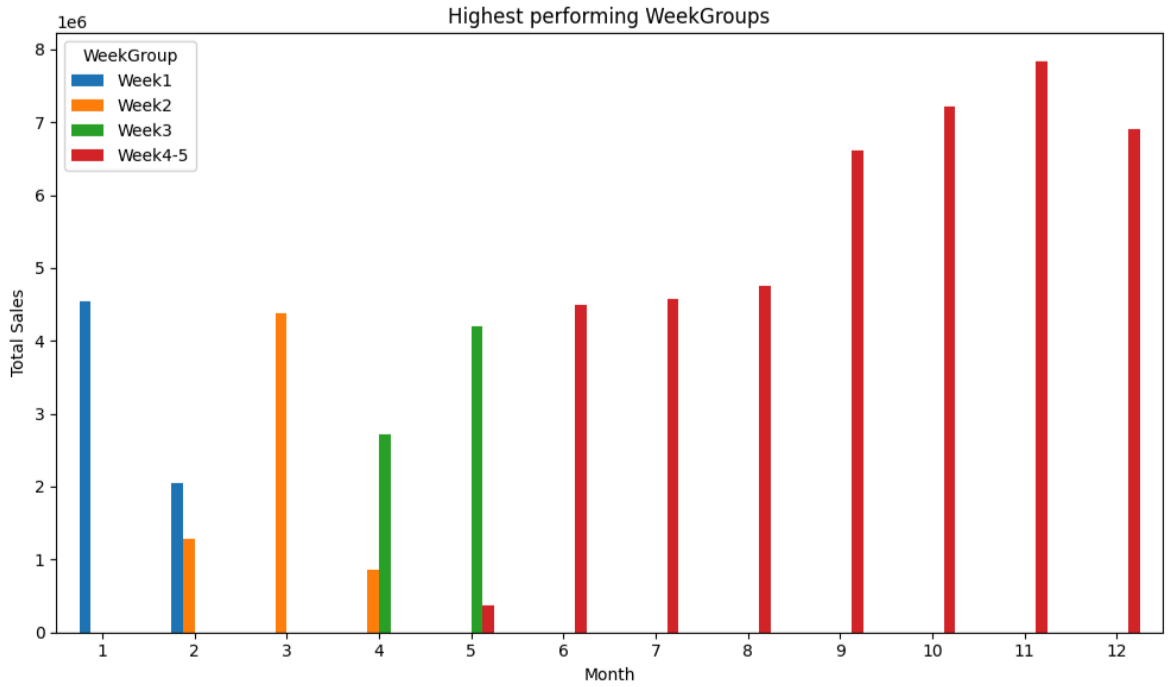
Finding: Strong sales concentration in certain weeks.

Recommendation: Schedule promotions and restocks for high-performing weeks.

Total value of sales based on the weekgroups is almost the same with little variation in values.

Dominant Week Group per Month

	Month	WeekGroup	TotalSales
0	1	Week1	4548423.47
1	2	Week1	2052232.91
2	3	Week2	4384669.82
3	4	Week3	2723808.94
4	5	Week3	4198802.10
5	6	Week4-5	4486050.15
6	7	Week4-5	4571494.88
7	8	Week4-5	4749801.23
8	9	Week4-5	6613772.79
9	10	Week4-5	7212279.85
10	11	Week4-5	7828489.53
11	12	Week4-5	6909717.91

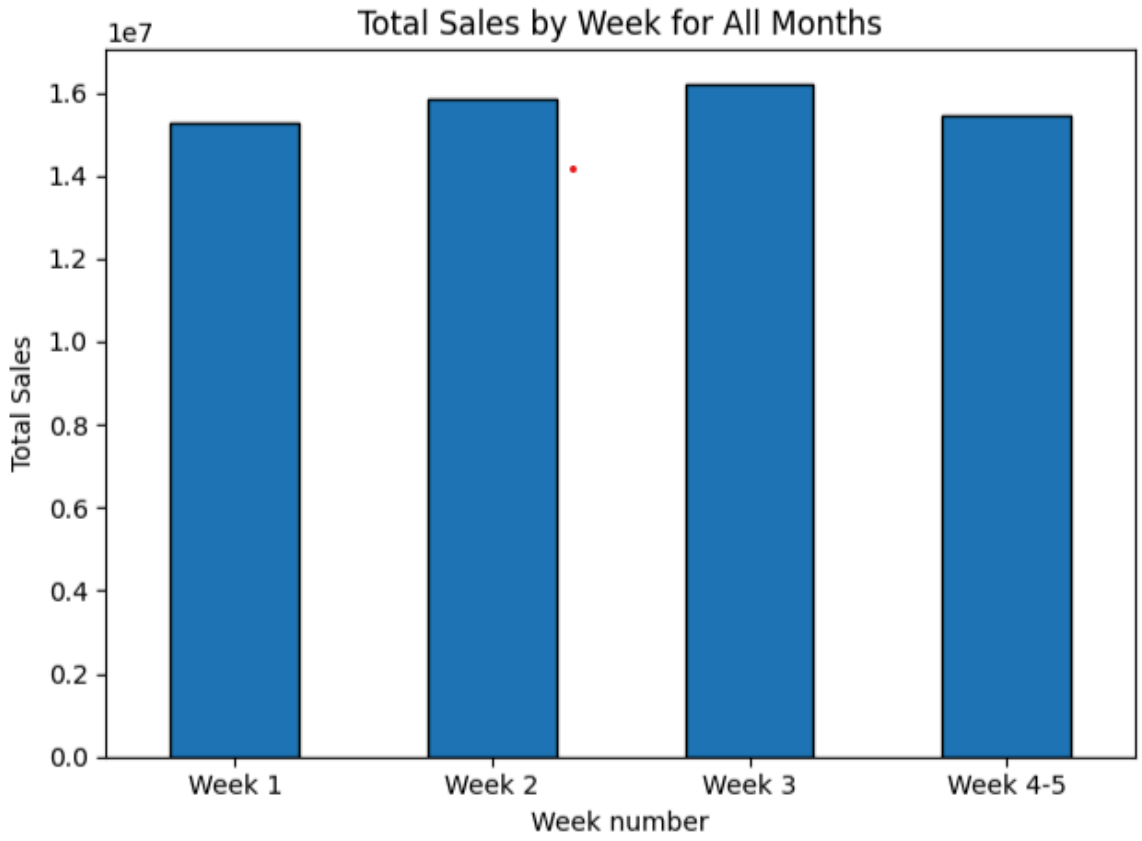


Observation: Identifies which weeks dominate sales for each month.
Finding: Provides predictable patterns for staff and inventory planning.
Recommendation: Build operational strategies aligned with dominant weeks.

It is evident that in the first half of the year, more sales happen in the first weeks of the year. However, as we approach the second half of the year, month ends (last weeks) perform better in sales and dominate it.

Total Weekly Sales Bar Graph

This graph shows the total sale in each week



Top 20 and Bottom 20 Products in Sales

```
for idx, row in bottom_20_named.iterrows():
    print(f"ProductNo: {idx} |  Name: {row['ProductName']} | Total Sales: {row['TotalSales']:.2
```

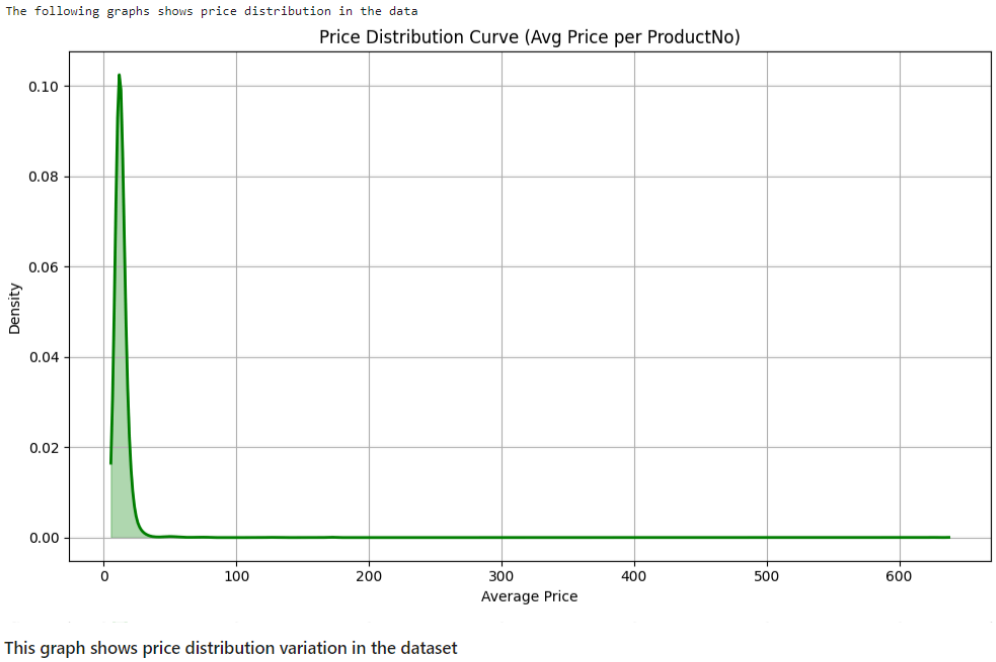
```
Top 20 Products by Total Sales:
ProductNo: 23843 |  Name: Paper Craft Little Birdie | Total Sales: 1002718.10
ProductNo: 23166 |  Name: Medium Ceramic Top Storage Jar | Total Sales: 881990.18
ProductNo: 22197 |  Name: Popcorn Holder | Total Sales: 587222.66
ProductNo: 84077 |  Name: World War 2 Gliders Asstd Designs | Total Sales: 568722.59
ProductNo: 85123A |  Name: Cream Hanging Heart T-Light Holder | Total Sales: 484354.72
ProductNo: 84879 |  Name: Assorted Colour Bird Ornament | Total Sales: 420132.72
ProductNo: 21212 |  Name: Pack Of 72 Retrospot Cake Cases | Total Sales: 391241.08
ProductNo: 23084 |  Name: Rabbit Night Light | Total Sales: 328529.51
ProductNo: 22423 |  Name: Regency Cakestand 3 Tier | Total Sales: 306900.94
ProductNo: 85099B |  Name: Jumbo Bag Red Retrospot | Total Sales: 296584.47
ProductNo: 22492 |  Name: Mini Paint Set Vintage | Total Sales: 287001.66
ProductNo: 47566 |  Name: Party Bunting | Total Sales: 286412.55
ProductNo: 22178 |  Name: Victorian Glass Hanging T-Light | Total Sales: 273977.99
ProductNo: 22616 |  Name: Pack Of 12 London Tissues | Total Sales: 270858.04
ProductNo: 21977 |  Name: Pack Of 60 Pink Paisley Cake Cases | Total Sales: 264266.48
ProductNo: 15036 |  Name: Assorted Colours Silk Fan | Total Sales: 261513.88
ProductNo: 21915 |  Name: Red Harmonica In Box | Total Sales: 245657.57
ProductNo: 17003 |  Name: Brocade Ring Purse | Total Sales: 239414.20
ProductNo: 22086 |  Name: Paper Chain Kit 50'S Christmas | Total Sales: 234089.64
ProductNo: 84946 |  Name: Antique Silver T-Light Glass | Total Sales: 216431.62

  Bottom 20 Products by Total Sales:
ProductNo: 23609 |  Name: Set 10 Cards Snowy Robin 17099 | Total Sales: 6.19
ProductNo: 84550 |  Name: Crochet Lilac/Red Bear Keyring | Total Sales: 6.19
ProductNo: 84227 |  Name: Hen House W Chick In Nest | Total Sales: 10.68
ProductNo: 90084 |  Name: Pink Crystal Guitar Phone Charm | Total Sales: 11.12
ProductNo: 35597B |  Name: Blackchristmas Tree 30cm | Total Sales: 11.53
ProductNo: 84743C |  Name: Orange Felt Vase + Flowers | Total Sales: 11.53
ProductNo: 84977 |  Name: Wire Flower T-Light Holder | Total Sales: 11.53
ProductNo: 35597A |  Name: Dusty Pink Christmas Tree 30cm | Total Sales: 11.53
ProductNo: 21009 |  Name: Etched Glass Star Tree Decoration | Total Sales: 11.53
ProductNo: 84569C |  Name: Pack 4 Flower/Butterfly Patches | Total Sales: 11.53
ProductNo: 37461 |  Name: Funky Monkey Mug | Total Sales: 11.53
ProductNo: 23664 |  Name: Flower Shop Design Mug | Total Sales: 11.94
ProductNo: 90092 |  Name: Blue Crystal Boot Phone Charm | Total Sales: 11.95
ProductNo: 21491 |  Name: Set Of Three Vintage Gift Wraps | Total Sales: 12.25
ProductNo: 22146 |  Name: Easter Craft Ivy Wreath With Chick | Total Sales: 12.25
ProductNo: 22323 |  Name: Pink Polkadot Kids Bag | Total Sales: 12.25
ProductNo: 21414 |  Name: Scallop Shell Soap Dish | Total Sales: 12.40
ProductNo: 85170A |  Name: Set/6 Ivory Bird T-Light Candles | Total Sales: 12.40
ProductNo: 47579 |  Name: Tea Time Breakfast Basket | Total Sales: 12.40
ProductNo: 20860 |  Name: Gold Cosmetics Bag With Butterfly | Total Sales: 12.40
```

World Map Plot of Sale Value

Observation: Displays global sales distribution.
Finding: Highlights strong and underperforming regions.
Recommendation: Use for international expansion planning.

Plotting Price Distribution for Products

Observation: Shows price range and concentration across products.
Finding: Helps identify pricing patterns and potential for premium pricing.
Recommendation: Adjust pricing strategy for competitive advantage.



Customers with Most Frequent Transactions

```
Top 20 Customers with highest number of transactions and their Purchase Value:

CustomerNo: 17841 | Transactions: 7803.0 | Total Purchase Value: 250424.13
CustomerNo: 14911 | Transactions: 5794.0 | Total Purchase Value: 872608.57
CustomerNo: 14096 | Transactions: 5093.0 | Total Purchase Value: 177814.26
CustomerNo: 12748 | Transactions: 4456.0 | Total Purchase Value: 249639.54
CustomerNo: 14606 | Transactions: 2751.0 | Total Purchase Value: 69497.40
CustomerNo: 15311 | Transactions: 2451.0 | Total Purchase Value: 385548.04
CustomerNo: 14646 | Transactions: 2066.0 | Total Purchase Value: 2108959.95
CustomerNo: 13089 | Transactions: 1884.0 | Total Purchase Value: 349979.58
CustomerNo: 13263 | Transactions: 1661.0 | Total Purchase Value: 51951.15
CustomerNo: 14298 | Transactions: 1638.0 | Total Purchase Value: 633533.19
CustomerNo: 17434 | Transactions: 1481.0 | Total Purchase Value: 64531.55
CustomerNo: 15039 | Transactions: 1479.0 | Total Purchase Value: 99459.24
CustomerNo: 14156 | Transactions: 1399.0 | Total Purchase Value: 683106.77
CustomerNo: 16729 | Transactions: 1378.0 | Total Purchase Value: 77194.24
CustomerNo: 16549 | Transactions: 1369.0 | Total Purchase Value: 39429.91
CustomerNo: 18118 | Transactions: 1262.0 | Total Purchase Value: 32801.82
CustomerNo: 16592 | Transactions: 1226.0 | Total Purchase Value: 68414.34
CustomerNo: 14159 | Transactions: 1179.0 | Total Purchase Value: 46502.18
CustomerNo: 15005 | Transactions: 1152.0 | Total Purchase Value: 45334.06
CustomerNo: 14796 | Transactions: 1148.0 | Total Purchase Value: 66186.89
```

Observation: Lists the top buyers by transaction count.

Finding: Identifies high-frequency buyers critical for loyalty.

Recommendation: Build loyalty and retention strategies for these customers.

Customers That Gave the Most Value

```
Top 20 Customers by Purchase Value:

CustomerNo: 14646 | Total Purchase Value: 2112282.03 | Transactions: 2064.0
CustomerNo: 16446 | Total Purchase Value: 1002741.57 | Transactions: 3.0
CustomerNo: 14911 | Total Purchase Value: 914204.19 | Transactions: 5574.0
CustomerNo: 12415 | Total Purchase Value: 900545.54 | Transactions: 715.0
CustomerNo: 18102 | Total Purchase Value: 897137.36 | Transactions: 431.0
CustomerNo: 17450 | Total Purchase Value: 891069.53 | Transactions: 336.0
CustomerNo: 12346 | Total Purchase Value: 840113.80 | Transactions: 1.0
CustomerNo: 14156 | Total Purchase Value: 694202.51 | Transactions: 1380.0
CustomerNo: 13694 | Total Purchase Value: 646116.78 | Transactions: 790.0
CustomerNo: 17511 | Total Purchase Value: 639006.19 | Transactions: 963.0
CustomerNo: 14298 | Total Purchase Value: 635526.37 | Transactions: 1635.0
CustomerNo: 16684 | Total Purchase Value: 528587.34 | Transactions: 275.0
CustomerNo: 16029 | Total Purchase Value: 434930.14 | Transactions: 240.0
CustomerNo: 15311 | Total Purchase Value: 391423.54 | Transactions: 2339.0
CustomerNo: 16422 | Total Purchase Value: 381043.45 | Transactions: 369.0
CustomerNo: 17404 | Total Purchase Value: 365681.20 | Transactions: 195.0
CustomerNo: 13089 | Total Purchase Value: 354356.08 | Transactions: 1845.0
CustomerNo: 16333 | Total Purchase Value: 349966.24 | Transactions: 45.0
CustomerNo: 17949 | Total Purchase Value: 348200.27 | Transactions: 69.0
CustomerNo: 15061 | Total Purchase Value: 333977.12 | Transactions: 403.0
```

Observation: Shows top revenue-generating customers.

Finding: Identifies highest value customers.

Recommendation: Personalize offerings and incentives for these customers.

Customers with Both High Frequency and High Value

```
Common (No of Transactions and high value) Customers in Top 20 no of transcations and Top 20 Value Buyers:

CustomerNo: 14911 | Transactions: 5574.0 | Total Purchase Value: 914204.19
CustomerNo: 15311 | Transactions: 2339.0 | Total Purchase Value: 391423.54
CustomerNo: 14646 | Transactions: 2064.0 | Total Purchase Value: 2112282.03
CustomerNo: 13089 | Transactions: 1845.0 | Total Purchase Value: 354356.08
CustomerNo: 14298 | Transactions: 1635.0 | Total Purchase Value: 635526.37
CustomerNo: 14156 | Transactions: 1380.0 | Total Purchase Value: 694202.51
```

These are the highest value customers and company should focus on maintaing relationship with these customers.

Observation: Segments customers excelling in both metrics.

Finding: Identifies best customers overall.

Recommendation: Target these as premium brand advocates.

Bottom 20 Customers by Purchase Value

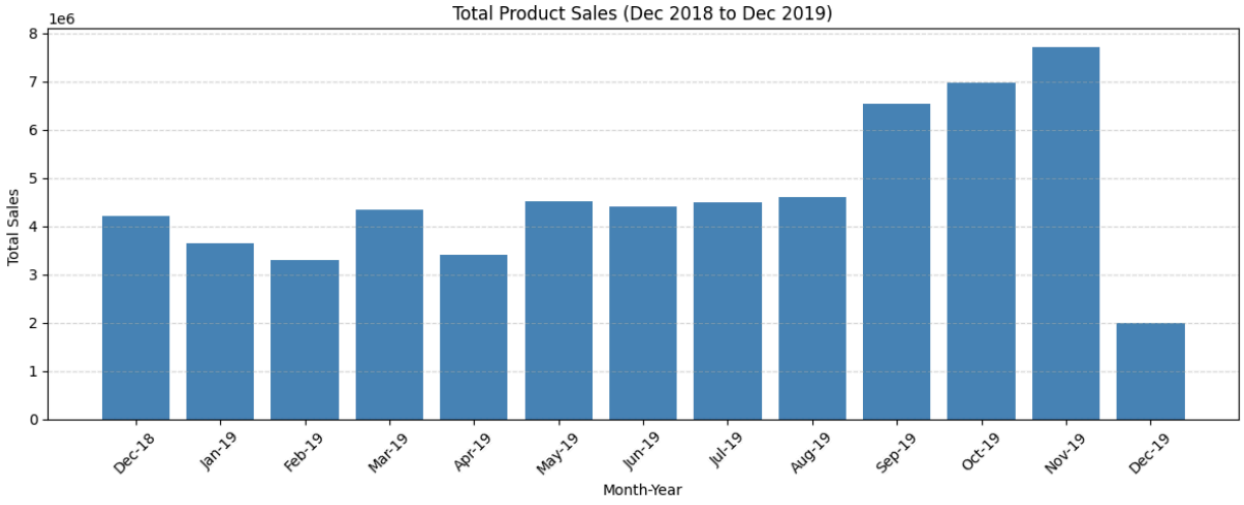
```
Bottom 20 Customers by Purchase Value:

CustomerNo: 16937 | Total Purchase Value: 5.97 | Transactions: 1.0
CustomerNo: 13775 | Total Purchase Value: 11.53 | Transactions: 1.0
CustomerNo: 12810 | Total Purchase Value: 11.98 | Transactions: 1.0
CustomerNo: 14435 | Total Purchase Value: 12.38 | Transactions: 2.0
CustomerNo: 12309 | Total Purchase Value: 12.86 | Transactions: 1.0
CustomerNo: 16724 | Total Purchase Value: 13.27 | Transactions: 1.0
CustomerNo: 12081 | Total Purchase Value: 13.27 | Transactions: 1.0
CustomerNo: 14025 | Total Purchase Value: 14.09 | Transactions: 1.0
CustomerNo: 17683 | Total Purchase Value: 14.48 | Transactions: 2.0
CustomerNo: 17699 | Total Purchase Value: 14.48 | Transactions: 2.0
CustomerNo: 17246 | Total Purchase Value: 14.50 | Transactions: 1.0
CustomerNo: 16476 | Total Purchase Value: 14.61 | Transactions: 1.0
CustomerNo: 17401 | Total Purchase Value: 14.61 | Transactions: 1.0
CustomerNo: 13773 | Total Purchase Value: 15.02 | Transactions: 1.0
CustomerNo: 15085 | Total Purchase Value: 15.84 | Transactions: 1.0
CustomerNo: 17116 | Total Purchase Value: 18.06 | Transactions: 1.0
CustomerNo: 16703 | Total Purchase Value: 18.40 | Transactions: 1.0
CustomerNo: 15818 | Total Purchase Value: 19.42 | Transactions: 1.0
CustomerNo: 14922 | Total Purchase Value: 21.17 | Transactions: 1.0
CustomerNo: 13674 | Total Purchase Value: 22.24 | Transactions: 1.0
```

Efforts should be made to analyze the purchasing behavior of these customers

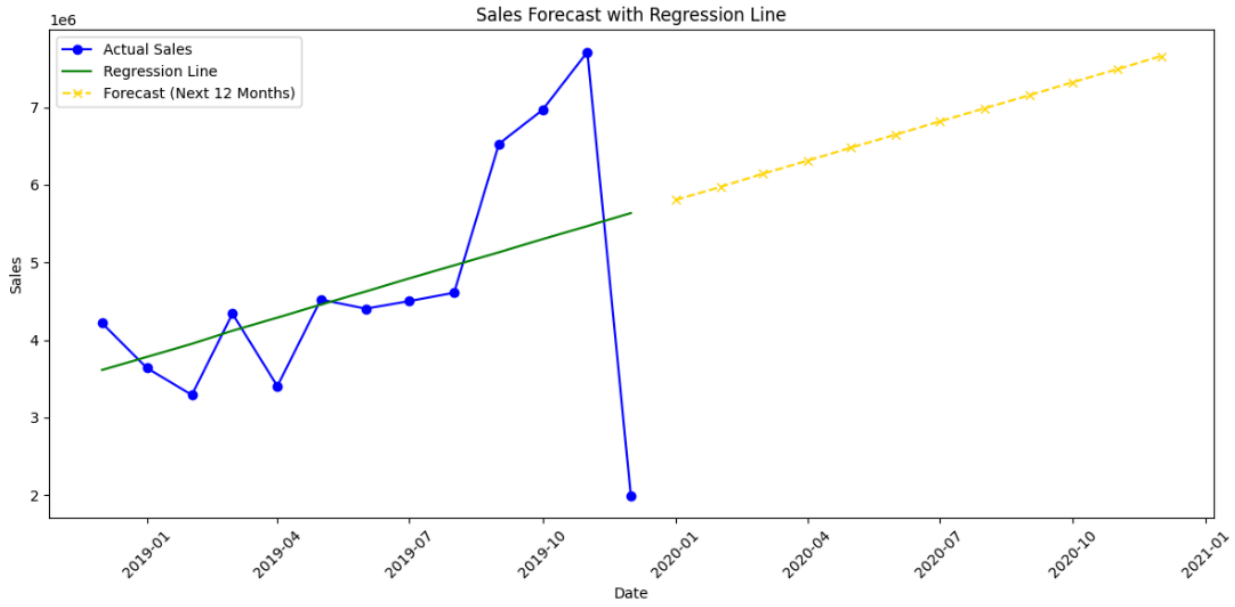
Observation: Lists customers with the lowest contribution.
Finding: Shows low-engagement segment.
Recommendation: Consider re-engagement campaigns or focus elsewhere.

Total Product Sales (December 2018–December 2019)

Observation: Aggregation across 13 months.The data for Dec 2019 consists of 9 days (1-9 Dec 2019).
Finding: Provides annual trend of sales per product.
Recommendation: Use for long-term forecasting and seasonal planning.



Sales Forecast with Linear Regression



We can see that the trend of sales is linear with positive growth

Observation: Model predicts future sales trend.
Finding: Enables forecasting revenue and making data-driven decisions.
Recommendation: Incorporate into sales planning and target setting.

Cancellation Analysis

```
Total Value of Cancellations: -2646715.27
Number of Cancelled Transactions: 8494
Cancellation percentage is  4.4 %
```

Recommendations:

Since cancellation percentage is very high, at 4.4%, efforts should be made to reduce this value.

Observation: Evaluates total cancelled sales across the dataset.
Finding: Identifies patterns and hotspots for cancellations.
Recommendation: Address root causes and review return policies.

Cancellations by Country

```
cancel_by_country = cancelled_orders.groupby('Country')['TotalSales'].sum().sort_values()
print(cancel_by_country.head(20))  # Most negative = highest cancellations
```

```
Country
United Kingdom   -2491406.76
EIRE               -52765.88
Germany            -21277.87
USA                -15418.27
Spain              -15273.47
France             -13758.64
Japan               -9861.97
Australia           -6851.56
Sweden              -5837.28
Netherlands         -3742.20
Switzerland         -3545.64
Italy               -1139.02
Norway              -1067.73
Czech Republic       -917.64
Portugal             -840.95
Belgium              -784.90
Denmark              -644.87
Austria              -598.96
Finland              -374.29
Malta                -208.12
Name: TotalSales, dtype: float64
```

Observation: Shows cancellation trends across countries.

Finding: Identifies markets with higher cancellation rates.

Recommendation: Tailor policies and review fulfillment quality.

Top 20 Cancelled Products Analysis

```
#Group by both ProductName and ProductNo to include Product ID
#and get Total Cancellations and Average Price
summary = top_20_df.groupby(['ProductName', 'ProductNo']).agg({
    'TotalSales': 'sum',
    'Price': 'mean'
}).sort_values(by='TotalSales')

# Display the result
summary = summary.rename(columns={'TotalSales': 'CancelledValue', 'Price': 'AvgPrice'})
print(summary)
```

ProductName	ProductNo	CancelledValue	AvgPrice
Medium Ceramic Top Storage Jar	23166	-843277.96	11.446000
Paper Craft Little Birdie	23843	-501359.05	6.190000
Rotating Silver Angels T-Light Hldr	84347	-96310.82	12.000000
Fairy Cake Flannel Assorted Colour	21108	-38921.40	9.833333
Cream Hanging Heart T-Light Holder	85123A	-33190.99	12.796190
Gin And Tonic Diet Metal Sign	21175	-24684.61	12.588571
Feltcraft Doll Molly	22273	-18424.21	11.911667
Tea Time Party Bunting	47566B	-18376.16	13.952857
Regency Cakestand 3 Tier	22423	-18148.61	21.787833
Herb Marker Basil	22920	-16508.19	10.865000
Coloured Glass Star T-Light Holder	71477	-15694.95	11.861613
Pink Blue Felt Craft Trinket Box	20971	-14984.89	11.482500
Pantry Chopping Board	23113	-14613.37	13.853333
Paper Pocket Traveling Fan	15034	-14304.54	10.355000
World War 2 Gliders Asstd Designs	84077	-12564.96	10.496000
Home Sweet Home Mug	21877	-11949.07	11.546000
Doormat Fairy Cake	48185	-11578.12	18.093333
Feltcraft Butterfly Hearts	22147	-10119.58	11.713750
Place Setting White Heart	22151	-9453.36	10.656000
Revolver Wooden Ruler	22992	-8487.94	12.161429

Observation: Lists most cancelled products.

Finding: Products with higher return or cancellation rates.

Recommendation: Investigate quality issues.

Low value product sales (< £50)

```
# Group by product and sum total sales
product_sales = df_positive.groupby(['ProductNo', 'ProductName'])['TotalSales'].sum().reset_index()

# Filter products whose total sales < 50
low_sales_products = product_sales[product_sales['TotalSales'] < 50]

# Display the result
print(low_sales_products[['ProductNo', 'ProductName', 'TotalSales']])
```

```
     ProductNo                         ProductName  TotalSales
105     17191A   Rose Flower Candle+Incense 16x16cm       42.66
139      20678       Large Black Diamante Hairslide       47.76
155      20703              Blue Padded Soft Mobile       14.61
241      20860    Gold Cosmetics Bag With Butterfly       12.40
242      20861          Gold Cosmetic Bag Pink Star       24.80
...        ...                                 ...         ...
3654    90178B    Purple Chunky Glass+Bead Necklace       22.50
3656    90179B   Purple Fine Bead Necklace W Tassel       28.39
3662    90181C     Black Glass/Shell/Pearl Necklace       19.63
3678    90187A    Blue Drop Earrings W Bead Cluster       13.65
3681    90189A  Silver 2 Strand Necklace-Leaf Charm       35.88

[146 rows x 3 columns]
```

Observation: Identifies low sales products.

Finding: 146 products identified with sales less than £50.

Recommendation: It is recommended to remove these products from the inventory.

Cancellation Analysis for Products Priced > £100

```
ts_100=df_positive[df_positive['Price']>100]['TotalSales'].sum()
co=cancelled_orders[cancelled_orders['Price']>100]['TotalSales'].sum()
print('Total value of orders  for product price>£100 is', round((ts_100), 2))
print('Total value of cancelled orders for product price>£100 is', round((-co), 2))
print('Precentage of cancelled orders(Price >£100)  ', round((-co*100/ts_100),2))
```

```
Total value of orders  for product price>£100 is 67626.68
Total value of cancelled orders for product price>£100 is 10555.46
Precentage of cancelled orders(Price >£100)   15.61
```

Recommendation:

Percentage of cancelled orders in high value items is very high (Almost 16%, as compared to average 4.4%).

The company should take steps to either remove these items from inventory or take these a feedback from customers based on their preferences.

Observation: Identifies patterns for higher-priced products.

Finding: Premium products have notable cancellation patterns.

Recommendation: Strengthen quality controls and customer service for premium products.

Exploratory Data Analysis - Part 2

Contributor: **Aishwarya Gulhane**

Cohort Analysis

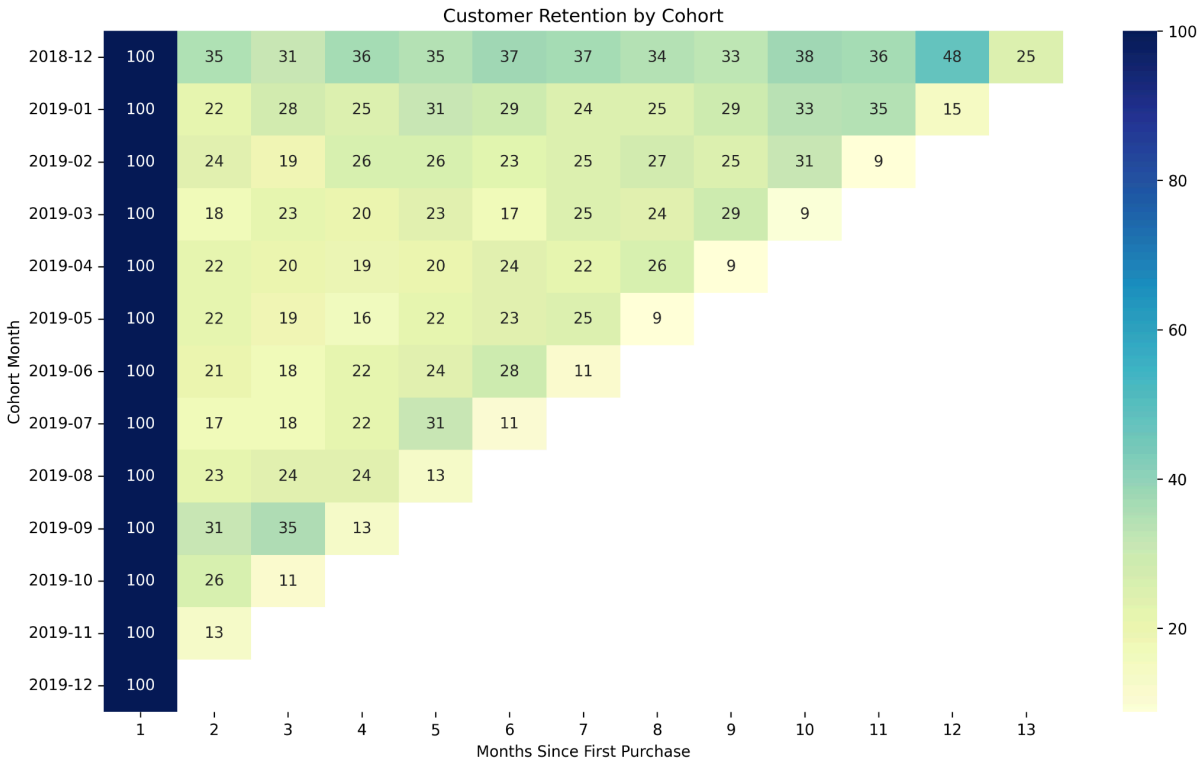
Cohort analysis is a technique that groups data into specific time-based segments (or "cohorts") to understand how behavior or performance evolves over time.

```
]: # Ensure the Date column is in datetime format
sales_data['Date'] = pd.to_datetime(sales_data['Date'], errors='coerce')

# Extract cohort information
sales_data['OrderMonth'] = sales_data['Date'].dt.to_period('M')
# Grouping customers based on the month they joined
sales_data['CohortMonth'] = sales_data.groupby('CustomerNo')['OrderMonth'].transform('min')

# Getting the number of months between OrderMonth and CohortMonth
sales_data['CohortIndex'] = (sales_data['OrderMonth'].dt.year - sales_data['CohortMonth'].dt.year) * 12 + (sales_data['OrderMonth'].dt

# Count unique customers per cohort per CohortIndex
cohort_data = sales_data.groupby(['CohortMonth', 'CohortIndex'])['CustomerNo'].nunique().reset_index()

# Pivot to create cohort retention table
cohort_pivot = cohort_data.pivot_table(index='CohortMonth', columns='CohortIndex', values='CustomerNo')

# Divide each row by its first column (month 1) to get percentages
cohort_percent = cohort_pivot.divide(cohort_pivot.iloc[:, 0], axis=0) * 100

# Plot retention heatmap
plt.figure(figsize=(14, 8))
sns.heatmap(cohort_percent, annot=True, fmt='.0f', cmap='YlGnBu')
plt.title('Customer Retention by Cohort')
plt.ylabel('Cohort Month')
plt.xlabel('Months Since First Purchase')
plt.yticks(rotation=0)
plt.show()

print("""
The heatmap above visualizes customer retention over time, where:
- Rows represent each cohort month (the first month the customers purchased).
- Columns represent the number of months since the cohort's first purchase.
- Cell values show the percentage of unique customers retained in each subsequent month.
""")
```

Observations:

- **Strongest Retention Cohort:**
  - December 2018 has the strongest retention: 35% stayed in Month 2, 36% in Month 4, and 48% came back even in Month 12.
  - Suggests successful acquisition or better onboarding strategies in this period.
- **Decline Over Time:**
  - Later cohorts (from mid to late 2019) show a steep drop in retention:E.g., June 2019 cohort drops from 100% to 18% in Month 3, and 11% by Month 7.
  - December 2019: Only 13% retention by Month 2.

- **Spike:**
    - September 2019 cohort shows: 31% in Month 2 → 35% in Month 3 → drop to 13% in Month 4.
    - Could indicate a successful short-term campaign or promo that didn't sustain.



NOTE:

Incomplete Data: Recent cohorts (Oct–Dec 2019) have fewer months tracked, so it's too early to evaluate long-term retention.

Recommendations

- **Repeating What Worked in Dec 2018-** Understand what was done differently in December 2018 — like marketing, product offers, pricing, or how new customers were welcomed. Try to apply those winning strategies again.
- **Stop Drop-Off in Months 2–4** Most customers stop coming back after the first month. Focus on Months 2 to 4 to bring them back — maybe through emails, reminders, offers, or better support.
- **Try Loyalty Programs or Personalized Messages** Many customers from mid-2019 left quickly. To keep them longer, try loyalty rewards or more personal outreach (like offers based on what they bought earlier).

- **Check the Spike in Sep 2019** The September 2019 group suddenly became active in Month 3. Find out what caused it — was there a promo or event? If yes, try that idea again with other groups.

Model Selection

Contributor: **Aishwarya Gulhane**

ARIMA Model Evaluation

Weekly Forecasting

- **Best Model:** ARIMA (p=2, d=1, q=2)
- **Error Metric:** ~25% (MAPE), indicating a satisfactory fit for sales forecasting.
- **Observations:** Weekly forecasting captures longer trend dynamics effectively, making it well-suited for sales prediction where variations occur gradually.

Daily Forecasting

- No combination of (p, d, q) achieved **SMAPE < 30%**.
- **Reason:** Daily sales data is highly granular and noisy, making it challenging for ARIMA to model effectively.
- **Implication:** Daily forecasting may not be ideal for this dataset due to the high level of variability and noise.

Why ARIMA?

- ARIMA is a **classical time series forecasting technique** designed for sequential data.
- *In this case, sales data is ordered by date, making ARIMA an appropriate choice.*
- The model captures:
    - **Autoregression (AR):** Leverages past sales values for prediction.
    - **Integration (I):** Removes trend and stabilizes the series.
    - **Moving Averages (MA):** Smoothens random fluctuations and noise.
- These characteristics make ARIMA especially well-suited for **weekly sales forecasting**, where trend and seasonality dominate, yielding more reliable results.

Summary of Results

Granularity	Model (Best)	Error (SMAPE)	Outcome
Weekly	ARIMA(2,1,2)	~25%	Good fit for forecasting
Daily	All Trials	>30%	Not suitable due to high noise and low predictive reliability

Model Analysis

Contributor: **Aishwarya Gulhane**

Weekly forecasting with ARIMA provides a robust approach for sales trend prediction, achieving an error rate (~25%) acceptable for practical use.

```
]: p_values = range(0, 3)  # Try 0,1,2
   d_values = range(0, 3)  # Try 0,1,2
   q_values = range(0, 3)  # Try 0,1,2
   results = []
   forecast_steps = len(test)

   # Grid search over p, d, qdaily_sales
   for p, d, q in itertools.product(p_values, d_values, q_values):
       try:
           model = ARIMA(train, order=(p, d, q))
           model_fit = model.fit()
           forecast = model_fit.forecast(steps=forecast_steps)
           actual_values = test
           predicted_values = forecast
           mae = np.mean(np.abs(actual_values - predicted_values))
           mape = np.mean(np.abs(actual_values - predicted_values) / np.abs(actual_values) * 100)
           smape = 100 * np.mean(2 * np.abs(actual_values - predicted_values) /(np.abs(actual_values) + np.abs(predicted_values)))
           results.append({'p': p, 'd': d, 'q': q, 'AIC': model_fit.aic, 'MAE': mae,'MAPE':mape, 'SMAPE': smape, 'Forecast': predicted_va
           #print(results)
       except:
           continue
```

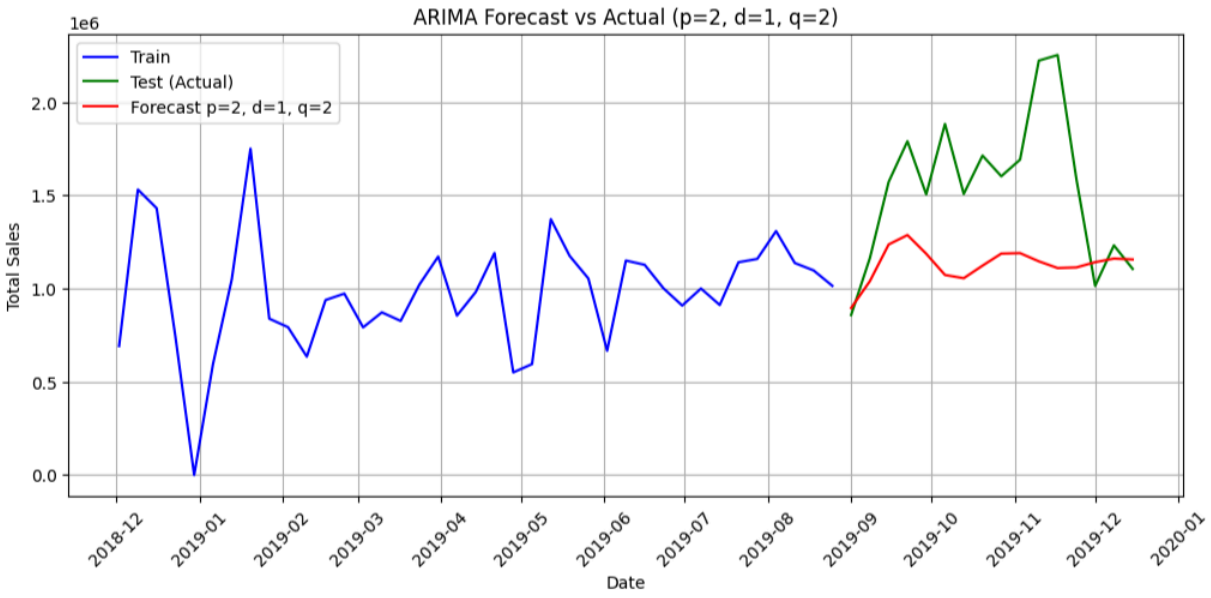
```
# Printing and plot results only if MAPE <= 30 [good fit]
for res in results:
    if res['MAPE'] <= 30:
        p, d, q = res['p'], res['d'], res['q']
        prediction = res['Forecast']
        comparison = pd.DataFrame({
            'Actual': test,
            'Predicted': prediction,
            'Difference': test - prediction
        })

        print("----------------------------")
        print(f"Model: p={p}, d={d}, q={q} | AIC={res['AIC']:.2f} ,MAE={res['MAE']:.2f},MAPE ={res['MAPE']:.2f}")
        print("Top 5 Actual vs Predicted:")
        print(comparison.head())

        # Plot
        plt.figure(figsize=(12, 5))
        plt.plot(train_w, label='Train', color='blue')
        plt.plot(test_w, label='Test (Actual)', color='green')
        plt.plot(prediction, label=f'Forecast p={p}, d={d}, q={q}', color='red')
        plt.title(f'ARIMA Forecast vs Actual (p={p}, d={d}, q={q})')
        plt.xlabel("Date")
        plt.ylabel("Total Sales")
        plt.legend()
        plt.grid(True)

        ax = plt.gca()
        ax.xaxis.set_major_locator(mdates.MonthLocator(interval=1))
        ax.xaxis.set_major_formatter(mdates.DateFormatter("%Y-%m"))
        for label in ax.get_xticklabels():
            label.set_rotation(45)
        plt.show()
```

```
print("----------------------------")
best_model = min(results, key=lambda x: x['MAPE'])
print("Best Model Parameters: (p,d,q)", best_model['p'], best_model['d'], best_model['q'])
print("AIC:", best_model['AIC'])
# AIC (Akaike Information Criterion):A measure of model quality – balances fit vs complexity - Lower is better.
## Shows how well model fits th#### The best ARIMA model (2,1,2) captures your weekly sales trend well.
#### An error of roughly 25% suggests the model is doing a good job (within good range), especially given the variability that often occurs in sales data. e data relative to the
print("MAE:", best_model['MAE'])
# MAE (Mean Absolute Error):
## The average absolute difference between actual and predicted values.Lower MAE = closer prediction.
print("MAPE:", best_model['MAPE'])
# MAPE (Mean Absolute Percentage Error):
## Percentage error between actual and predicted. MAPE < 30  is Good
print("----------------------------")
## A low MAPE (and low AIC) means the model captures the data well and predicts reliably.
```

Result:

```
----------------------------
Best Model Parameters: (p,d,q) 2 1 2
AIC: 1056.9960096460984
MAE: 439540.422175562
MAPE: 25.03344512843016
----------------------------
```

```
----------------------------
Model: p=2, d=1, q=2 | AIC=1057.00 ,MAE=439540.42,MAPE =25.03
Top 5 Actual vs Predicted:
                Actual     Predicted     Difference
2019-09-01   858515.00  8.963537e+05  -37838.697680
2019-09-08  1164535.99  1.040567e+06  123969.311642
2019-09-15  1572890.84  1.237107e+06  335783.563235
2019-09-22  1790987.50  1.287607e+06  503380.984672
2019-09-29  1505963.44  1.187349e+06  318614.677185
```



The best ARIMA model (2,1,2) captures your weekly sales trend well.

An error of roughly 25% suggests the model is doing a good job (within good range), especially given the variability that often occurs in sales data.

Conclusion

This project provided an end-to-end analysis of a UK-based e-commerce dataset, uncovering key insights across sales performance, customer behavior, product dynamics, and operational inefficiencies like cancellations. Using a combination of exploratory data analysis, cohort modeling, RFM segmentation, and predictive forecasting, the team extracted actionable intelligence from over 500,000 transactions spanning December 2018 to December 2019.

The analysis highlighted high-performing countries and products, identified sales patterns across time periods (e.g., strong weekday/week group effects), and exposed gaps such as high cancellation rates for premium items and low-value product inventory. Furthermore, cohort retention analysis emphasized the importance of sustained customer engagement beyond the first month. Predictive modeling using ARIMA, enabled forecasting future sales trends with reasonable accuracy. Together, these findings support strategic decisions around inventory, marketing, pricing, and retention, empowering the business to move toward data-driven growth and operational efficiency.

Analyzed e-commerce sales data (1 year) to understand trends and customer behavior.

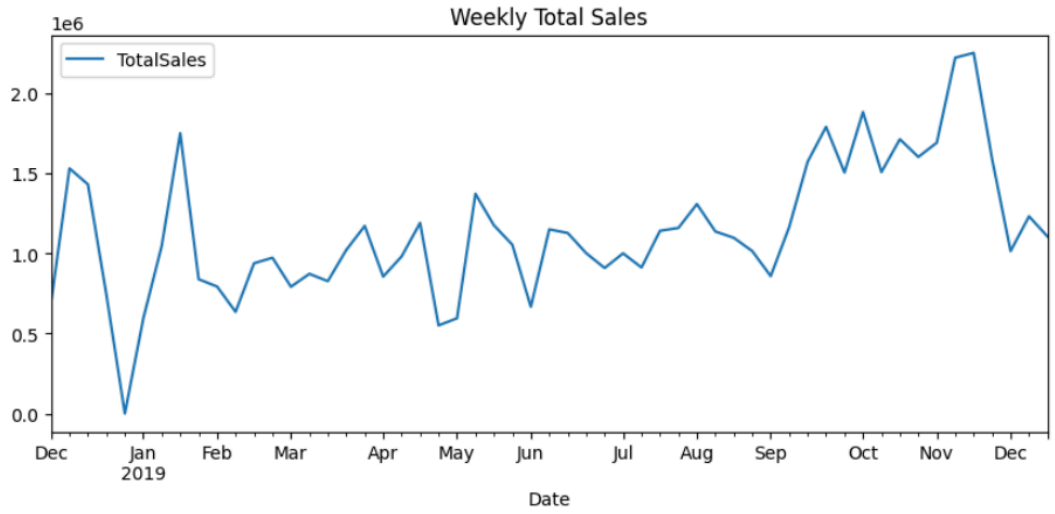
- Created a robust end-to-end workflow:

Data Cleaning & Preparation ➔ EDA ➔ Modeling (ARIMA) ➔ Insights & Recommendations

- Key Findings
  – Quantity is the primary driver of Total Sales.
  – Weekly sales patterns captured well by ARIMA (25% error rate).
- Limitations:
  – Limited data (1 year) impacts long-term forecasting precision.
  – Daily forecasts using ARIMA prone to noise and higher error.
- Future Scope:
  – Incorporate multi-year data for improved trend and seasonality detection.
  – Refine forecasting and feature engineering for greater accuracy.

Appendix

WEEKLY FORECAST : ARIMA




```
output saved at: /home/Isha/USD/usd_project_aai_500/AAI-500-IN1_PROJECT/visuals/modeling_charts/Weekly_total_sales.png
------------------------------
ADF Statistic: -2.6516
p-value: 0.0828
```

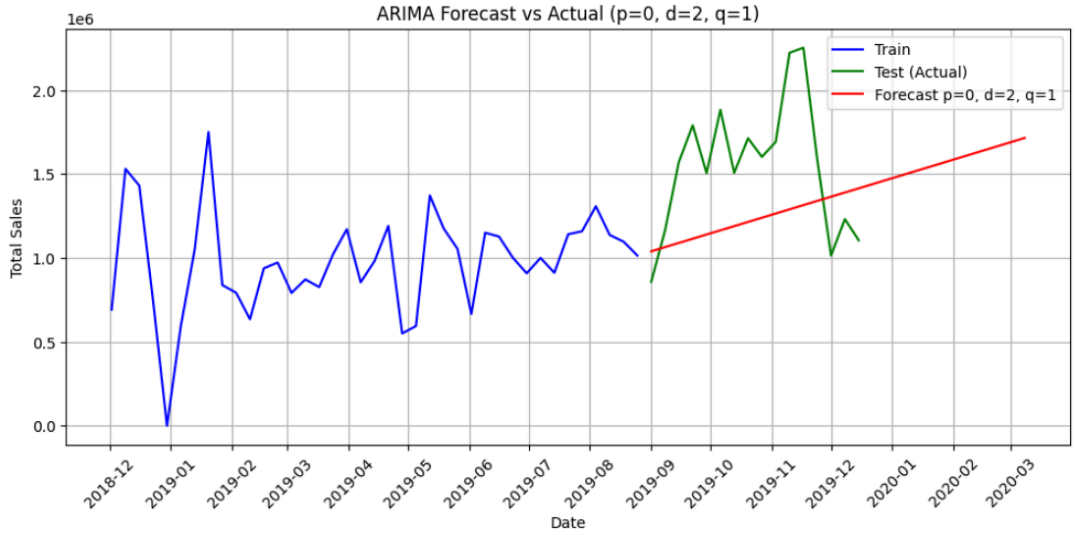
```
: Date
2018-12-02     692086.45
2018-12-09    1531721.91
2018-12-16    1431665.15
2018-12-23     742174.88
2018-12-30          0.00
Freq: W-SUN, Name: TotalSales, dtype: float64
```

Split Test data and Train data

```
Date range: 2018-12-02 00:00:00 -> 2019-12-15 00:00:00
------------------------------
Train range: 2018-12-02 00:00:00 -> 2019-08-25 00:00:00 (39 points)
Test range : 2019-09-01 00:00:00 -> 2019-12-15 00:00:00 (16 points)
```

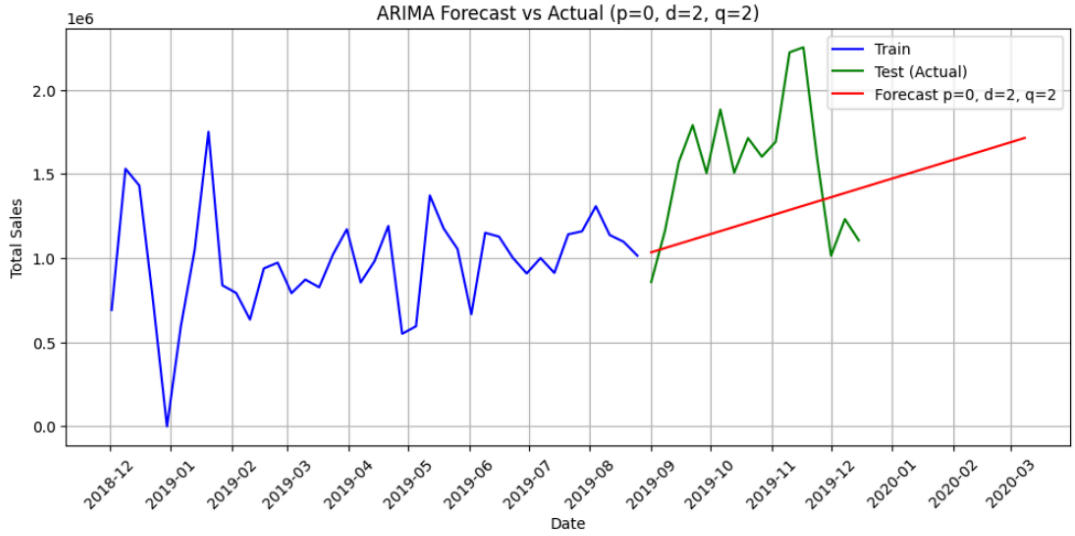
Iterating over different p, d, q values to fit ARIMA

```
Model: p=0, d=2, q=1 | AIC=1064.62 , MAE=441761.34, MAPE=27.07
Top 5 Actual vs Predicted:
                Actual     Predicted     Difference
2019-09-01   858515.00  1.040193e+06 -181678.176931
2019-09-08  1164535.99  1.065216e+06   99319.496138
2019-09-15  1572890.84  1.090240e+06  482651.029207
2019-09-22  1790987.50  1.115263e+06  675724.372276
2019-09-29  1505963.44  1.140286e+06  365676.995345
```



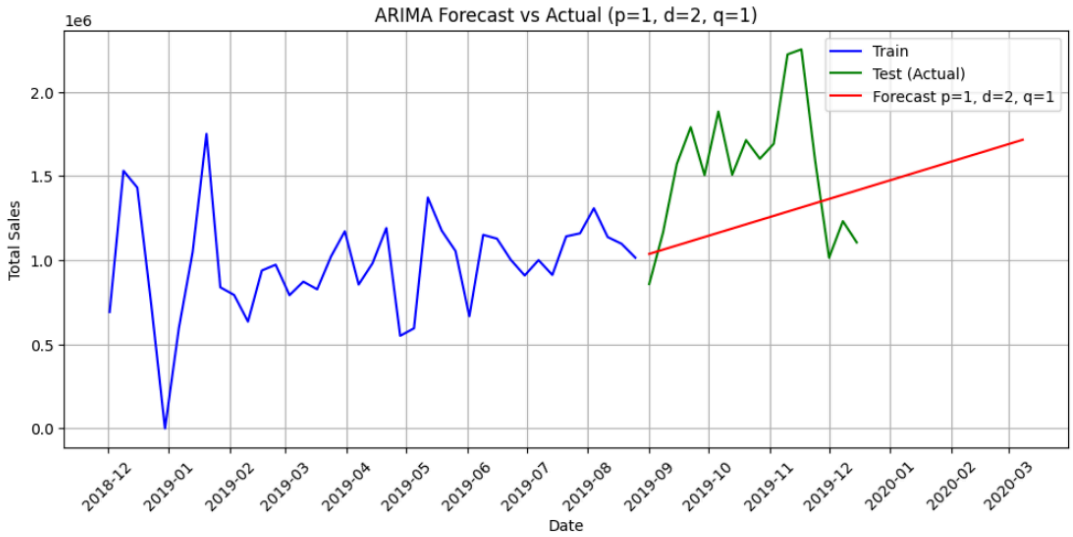

```
 Output saved at: /home/Isha/USD/usd_project_aai_500/AAI-500-IN1_PROJECT/visuals/modeling_charts/ARIMA_weekly_total_sales_prediction_p=0_d=2_q=
1.png
```

```
Model: p=0, d=2, q=2 | AIC=1066.48 , MAE=444000.98, MAPE=27.18
Top 5 Actual vs Predicted:
                Actual     Predicted     Difference
2019-09-01   858515.00  1.035033e+06 -176517.986470
2019-09-08  1164535.99  1.060208e+06  104328.369510
2019-09-15  1572890.84  1.085382e+06  487508.585489
2019-09-22  1790987.50  1.110557e+06  680430.611468
2019-09-29  1505963.44  1.135732e+06  370231.917447
```

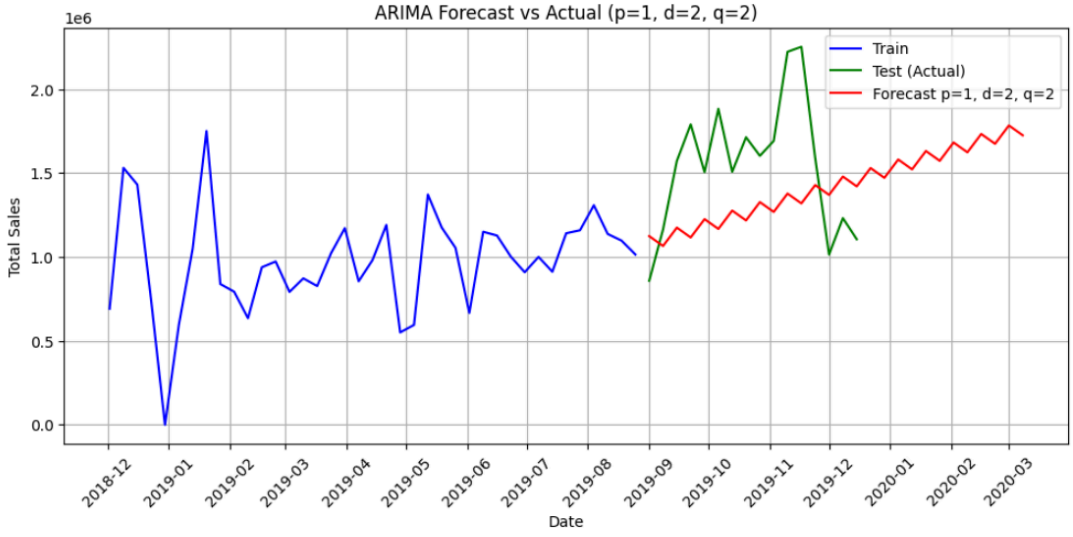



```
 Output saved at: /home/Isha/USD/usd_project_aai_500/AAI-500-IN1_PROJECT/visuals/modeling_charts/ARIMA_weekly_total_sales_prediction_p=0_d=2_q=
2.png
```

```
Model: p=1, d=2, q=1 | AIC=1066.54 , MAE=442821.70, MAPE=27.12
Top 5 Actual vs Predicted:
                Actual     Predicted     Difference
2019-09-01   858515.00  1.037666e+06 -179150.972032
2019-09-08  1164535.99  1.062732e+06  101804.266416
2019-09-15  1572890.84  1.087860e+06  485030.597422
2019-09-22  1790987.50  1.112990e+06  677997.205234
2019-09-29  1505963.44  1.138120e+06  367843.055596
```

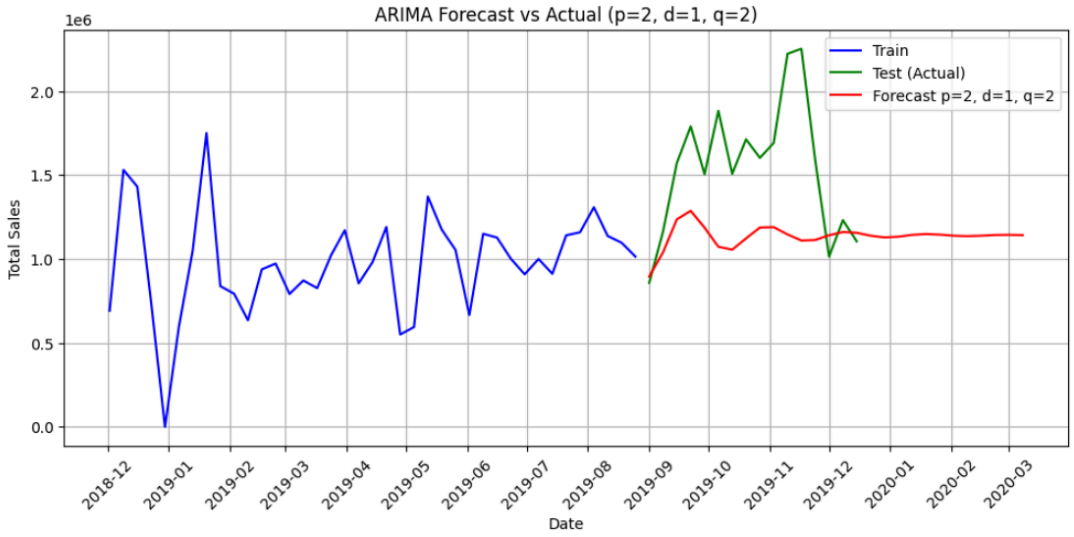



```
 Output saved at: /home/Isha/USD/usd_project_aai_500/AAI-500-IN1_PROJECT/visuals/modeling_charts/ARIMA_weekly_total_sales_prediction_p=1_d=2_q=
1.png

---------------------------
Model: p=1, d=2, q=2 | AIC=1073.48 , MAE=419781.56, MAPE=26.16
Top 5 Actual vs Predicted:
                Actual     Predicted     Difference
2019-09-01   858515.00  1.124560e+06 -266044.554227
2019-09-08  1164535.99  1.065929e+06   98606.978556
2019-09-15  1572890.84  1.175316e+06  397574.943726
2019-09-22  1790987.50  1.116688e+06  674299.337159
2019-09-29  1505963.44  1.226072e+06  279891.201632
```

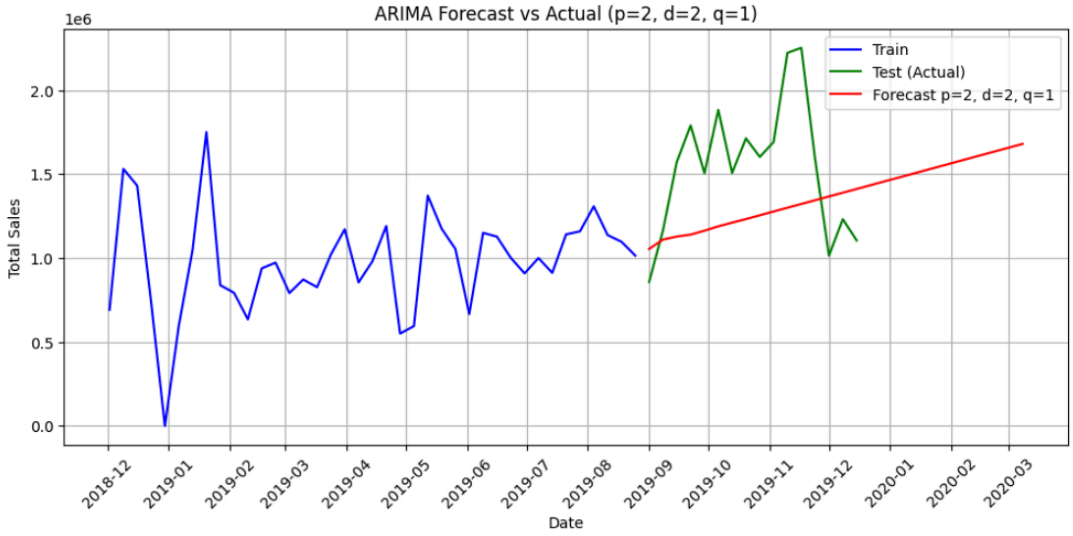



```
 Output saved at: /home/Isha/USD/usd_project_aai_500/AAI-500-IN1_PROJECT/visuals/modeling_charts/ARIMA_weekly_total_sales_prediction_p=1_d=2_q=
2.png
```

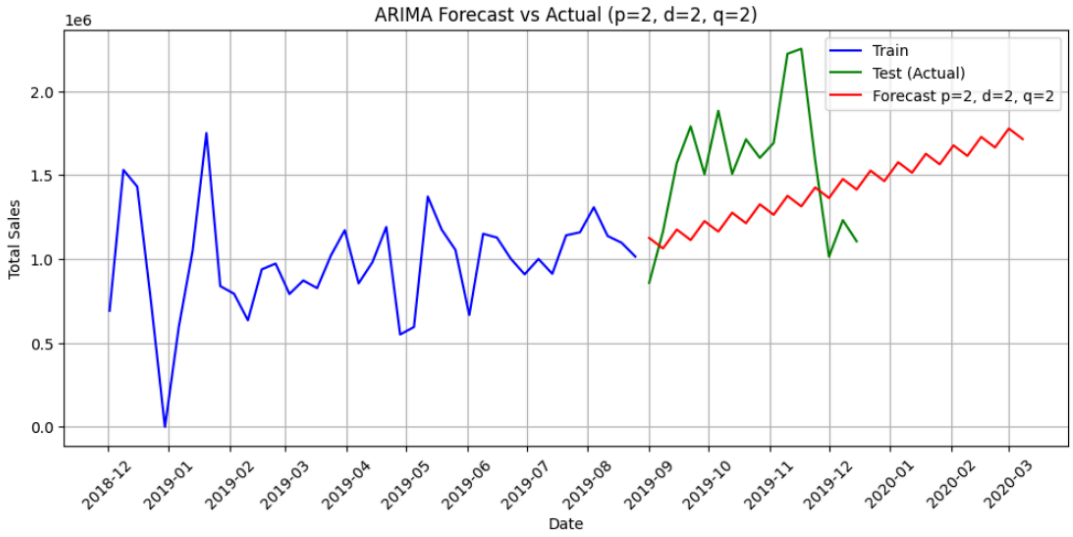
```
----------------------------
Model: p=2, d=1, q=2 | AIC=1057.00 , MAE=439540.42, MAPE=25.03
Top 5 Actual vs Predicted:
                Actual     Predicted     Difference
2019-09-01   858515.00  8.963537e+05  -37838.697680
2019-09-08  1164535.99  1.040567e+06  123969.311642
2019-09-15  1572890.84  1.237107e+06  335783.563235
2019-09-22  1790987.50  1.287607e+06  503380.984672
2019-09-29  1505963.44  1.187349e+06  318614.677185
```



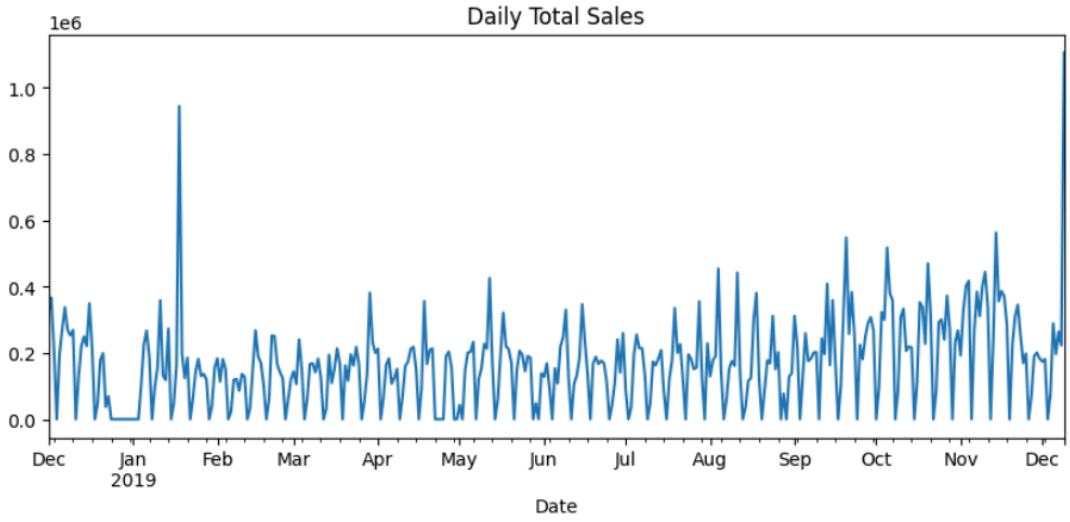

```
Model: p=2, d=2, q=1 | AIC=1064.63 , MAE=427225.89, MAPE=26.18
Top 5 Actual vs Predicted:
                Actual     Predicted     Difference
2019-09-01   858515.00  1.054776e+06 -196261.064143
2019-09-08  1164535.99  1.110864e+06   53671.560801
2019-09-15  1572890.84  1.128576e+06  444315.153830
2019-09-22  1790987.50  1.140163e+06  650824.105125
2019-09-29  1505963.44  1.163770e+06  342192.973221
```



```
--------------------------
Model: p=2, d=2, q=2 | AIC=1074.94 , MAE=420639.06, MAPE=26.19
Top 5 Actual vs Predicted:
                Actual     Predicted     Difference
2019-09-01   858515.00  1.126198e+06 -267682.889218
2019-09-08  1164535.99  1.063490e+06  101046.271320
2019-09-15  1572890.84  1.176194e+06  396696.354696
2019-09-22  1790987.50  1.113627e+06  677360.511502
2019-09-29  1505963.44  1.226343e+06  279619.944112
```



DAILY FORECAST : ARIMA

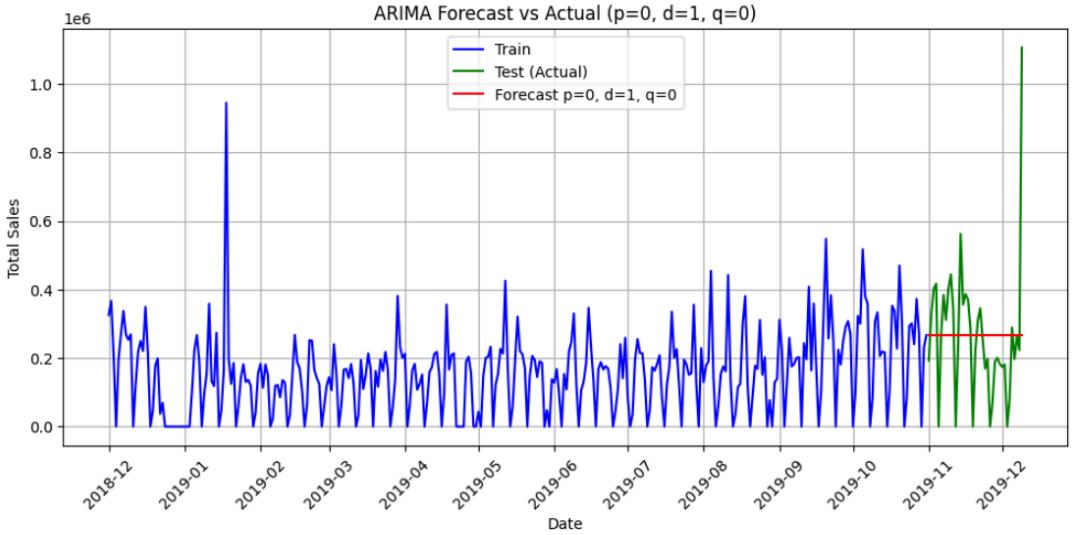



```
output saved at: /home/Isha/USD/usd_project_aai_500/AAI-500-IN1_PROJECT/visuals/modeling_charts/daily_total_sales.png
```

Split Test data and Train data

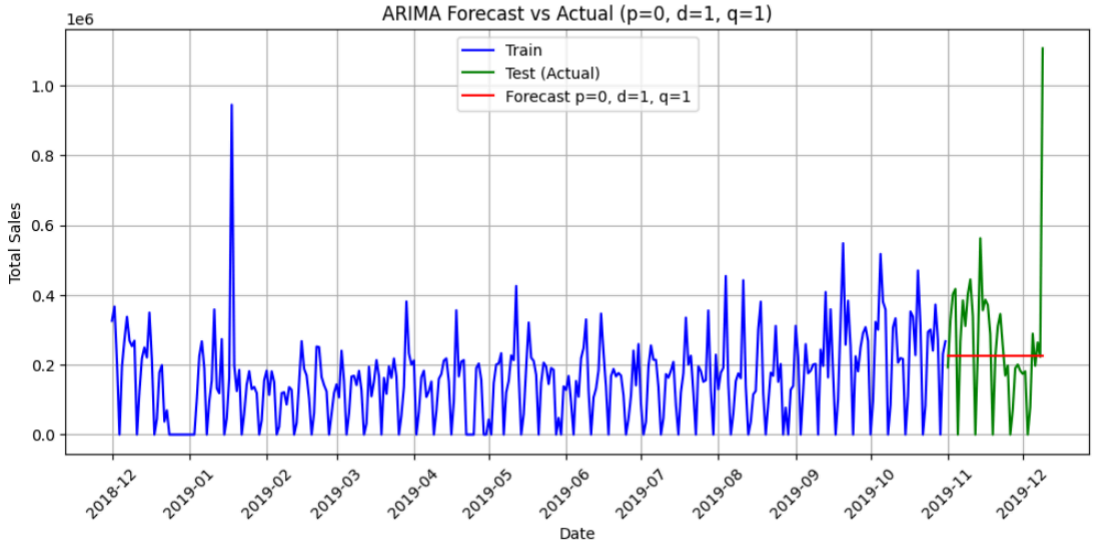
```
------------------------------
ADF Statistic: -2.6516
p-value: 0.0828
------------------------------
Train range: 2018-12-01 00:00:00 -> 2019-10-31 00:00:00 (335 points)
Test range : 2019-11-01 00:00:00 -> 2019-12-09 00:00:00 (39 points)
```

```
Model: p=0, d=1, q=0 | MAE=130052.18, SMAPE=56.74%
Top 5 Actual vs Predicted:
               Actual  Predicted  Difference
2019-11-01  192621.80  267148.72   -74526.92
2019-11-02  327067.21  267148.72    59918.49
2019-11-03  401757.07  267148.72   134608.35
2019-11-04  416964.74  267148.72   149816.02
2019-11-05       0.00  267148.72  -267148.72
```



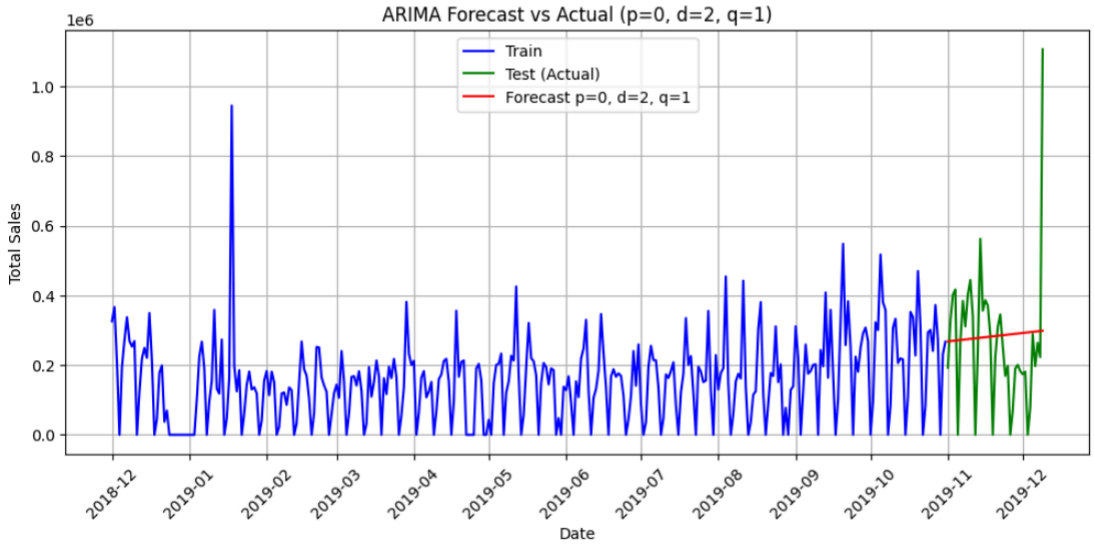

```
Output saved at: /home/Isha/USD/usd_project_aai_500/AAI-500-IN1_PROJECT/visuals/modeling_charts/ARIMA_daily_total_sales_prediction_p=0_d=1_q=0.
png
```

```
Model: p=0, d=1, q=1 | MAE=130909.43, SMAPE=59.15%
Top 5 Actual vs Predicted:
               Actual      Predicted     Difference
2019-11-01  192621.80  225433.891705  -32812.091705
2019-11-02  327067.21  225433.891705  101633.318295
2019-11-03  401757.07  225433.891705  176323.178295
2019-11-04  416964.74  225433.891705  191530.848295
2019-11-05       0.00  225433.891705 -225433.891705
```



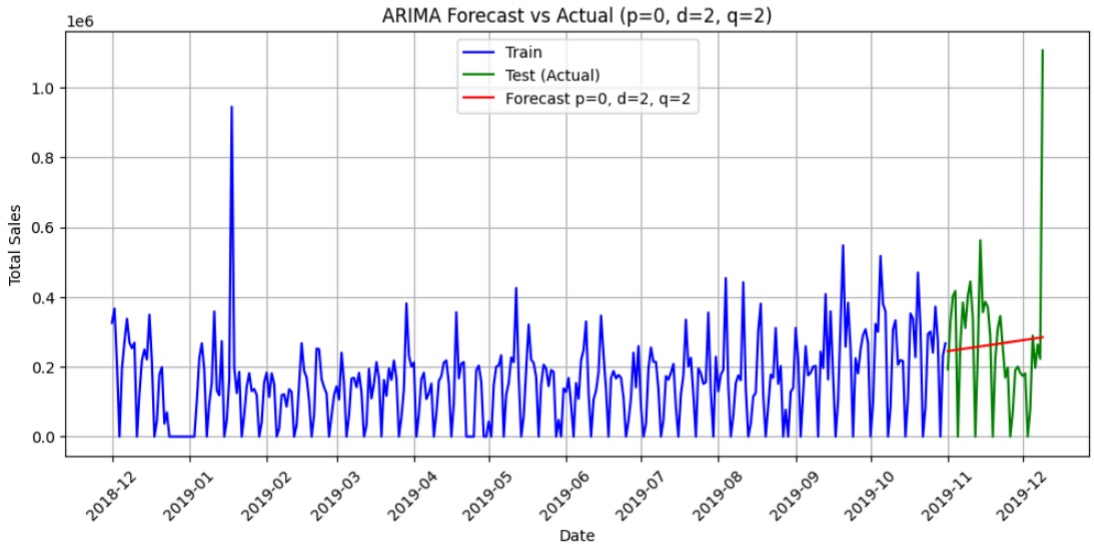

```
Output saved at: /home/Isha/USD/usd_project_aai_500/AAI-500-IN1_PROJECT/visuals/modeling_charts/ARIMA_daily_total_sales_prediction_p=0_d=1_q=1
png
```

```
Model: p=0, d=2, q=1 | MAE=136114.06, SMAPE=58.10%
Top 5 Actual vs Predicted:
               Actual      Predicted     Difference
2019-11-01  192621.80  267952.303168  -75330.503168
2019-11-02  327067.21  268755.886336   58311.323664
2019-11-03  401757.07  269559.469504  132197.600496
2019-11-04  416964.74  270363.052672  146601.687328
2019-11-05       0.00  271166.635840 -271166.635840
```



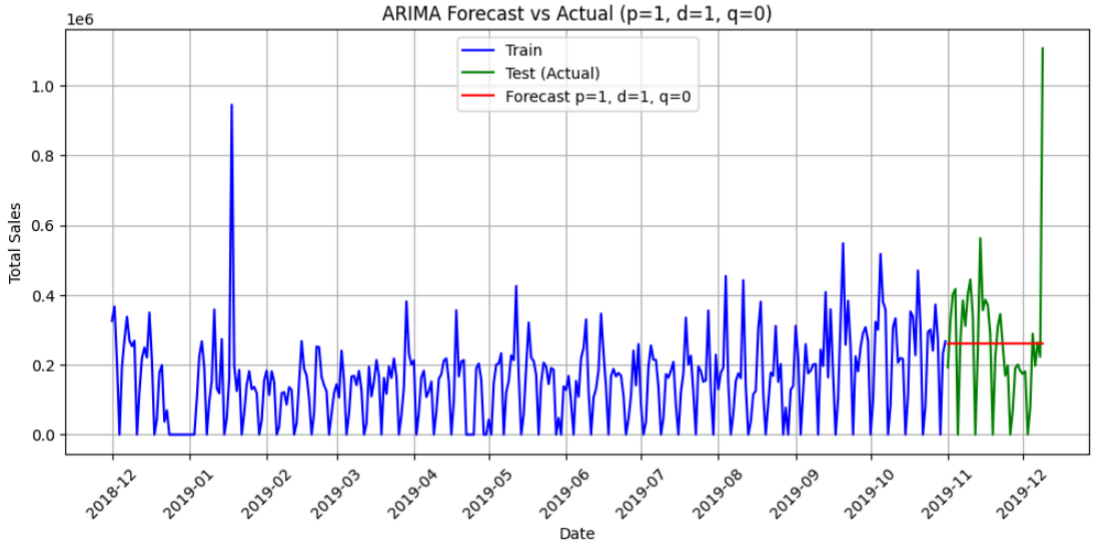

```
 Output saved at: /home/Isha/USD/usd_project_aai_500/AAI-500-IN1_PROJECT/visuals/modeling_charts/ARIMA_daily_total_sales_prediction_p=0_d=2_q=1.
png
```

```
---------------------------
Model: p=0, d=2, q=2 | MAE=135285.23, SMAPE=58.83%
Top 5 Actual vs Predicted:
               Actual      Predicted     Difference
2019-11-01  192621.80  244815.118251  -52193.318251
2019-11-02  327067.21  245857.396447   81209.813553
2019-11-03  401757.07  246899.674642  154857.395358
2019-11-04  416964.74  247941.952838  169022.787162
2019-11-05       0.00  248984.231034 -248984.231034
```



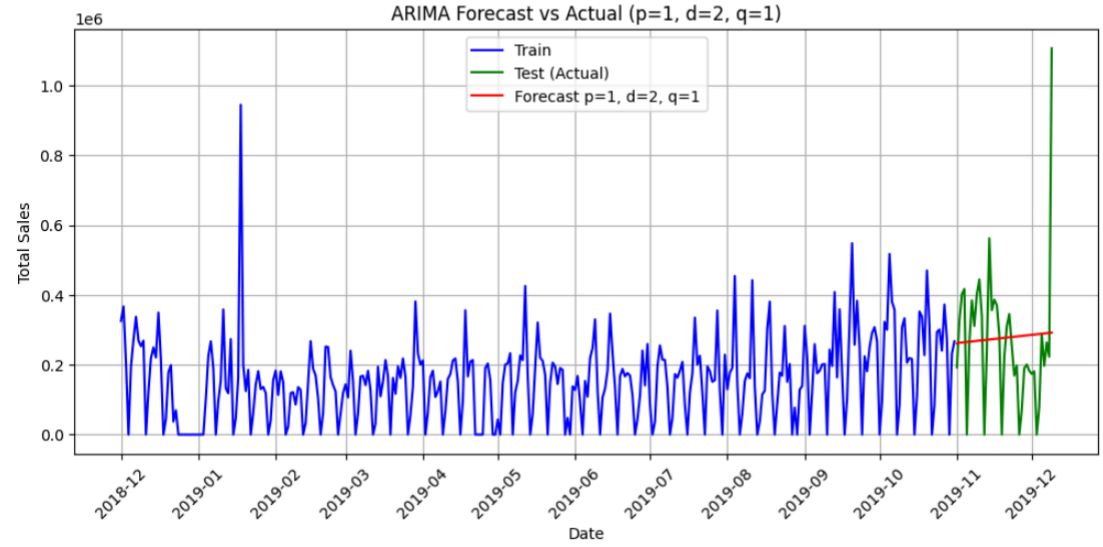

```
 Output saved at: /home/Isha/USD/usd_project_aai_500/AAI-500-IN1_PROJECT/visuals/modeling_charts/ARIMA_daily_total_sales_prediction_p=0_d=2_q=2.
```

```
Model: p=1, d=1, q=0 | MAE=129607.30, SMAPE=56.88%
Top 5 Actual vs Predicted:
               Actual      Predicted     Difference
2019-11-01  192621.80  259642.991590  -67021.191590
2019-11-02  327067.21  261223.348398   65843.861602
2019-11-03  401757.07  260890.598862  140866.471138
2019-11-04  416964.74  260960.660416  156004.079584
2019-11-05       0.00  260945.908716 -260945.908716
```




```
Output saved at: /home/Isha/USD/usd_project_aai_500/AAI-500-IN1_PROJECT/visuals/modeling_charts/ARIMA_daily_total_sales_prediction_p=1_d=1_q=0.
png
```

```
Model: p=1, d=2, q=1 | MAE=134837.90, SMAPE=57.99%
Top 5 Actual vs Predicted:
               Actual      Predicted     Difference
2019-11-01  192621.80  260822.487335  -68200.687335
2019-11-02  327067.21  263054.631672   64012.578328
2019-11-03  401757.07  263541.741607  138215.328393
2019-11-04  416964.74  264384.660250  152580.079750
2019-11-05       0.00  265155.030273 -265155.030273
```




```
Output saved at: /home/Isha/USD/usd_project_aai_500/AAI-500-IN1_PROJECT/visuals/modeling_charts/ARIMA_daily_total_sales_prediction_p=1_d=2_q=1
png
```