Analyzing sales to comprehend trends in consumer behavior and profitability

By Eunice Lee Wen Jing

Table of Contents

Analyzing sales to comprehend trends in consumer behavior and profitability

Problem Statement

Problem Statement 1

E-commerce companies are facing challenges in managing their inventory, which is a crucial element in determining the business's profitability. The large dataset given includes information from sales platforms that reveal problems like having too much inventory and running out of stock, which are affected by different Maximum Retail Prices (MRP) on sites like Ajio and Amazon. The main task is to synchronise inventory with current demand and sales data, guaranteeing product availability and customer contentment. E-commerce businesses in the digital marketplace require a strong inventory management solution that can combine intricate sales data, such as stock keeping unit (SKU) codes, design numbers, and transaction figures, in order to efficiently allocate inventory, minimize carrying costs, and improve financial health and competitiveness.

Problem Statement 2

Amidst the competitive landscape of e-commerce, e-commerce businesses must work hard to grasp and meet customer preferences successfully. The goal is to examine the e-commerce dataset given in order to understand consumer preferences. The purpose is to discover patterns in consumer buying behaviour by analysing SKU codes, design numbers, and product categories by examining how product features like size, colour, and MRP impact customer preferences on platforms like Ajio, Amazon, and Myntra. The analysis will take into account the attributes like cost per item to grasp purchasing trends. In the end, the objective is to use this data to forecast upcoming consumer actions, and improve profitability within the fiercely competitive e-commerce market.

Problem Statement 3

With the competitiveness in today's e-commerce marketplace, staying competitive is crucial to thrive in the modern marketplace and to drive consistent profitability in sales. This problem statement revolves around encapsulating dynamic sales patterns that often reflect seasonal consumer behaviour by leveraging the sales date provided in the datasets to identify and analyse these seasonal sales trends. Examining the fluctuations in e-commerce sales for a

business across various times of the year and linking these changes to holidays, festivals, and seasonal patterns could assist the company in understanding consumer behavior trends. This enables them to optimize their marketing tactics, coordinate promotional efforts with high-demand periods, and stock up inventory to meet expected needs. These understandings will assist the business to thrive within the competitive market by increasing total profitability and revenue.

Dataset Description

For this report, "Amazon Sale Report.csv", "Sale Report.csv", "May-2022.csv" and "P L March 2021.csv" datasets will be used to gain comprehensive understanding on the profitability and consumer behaviour trends through sales analysis. Through investigating each dataset structure, it is found that the Amazon Sale Report dataset has a dimension of 128975 rows and 24 columns while, Sale Report dataset has a smaller dimension with 9271 rows and 7 columns. Both May-2022 and P L March 2021 datasets have similar dimensions with 1330 rows, but a slight difference in column size with a value of 17 and 18 respectively with near identical information in column name.

Moreover, by exploring the data quality of these datasets, it is revealed that many of them have missing value, inconsistent data formats and inaccurate data. Notably, the Amazon Sale Report and sales report datasets are the only 2 datasets that have missing values present. Missing values in data affect its reliability and usability for analysis, hence data cleaning will be required.

All selected datasets were examined using the **str()** function, which revealed critical insights into the data types and structures of each column. This step was essential to identify the correct handling of date-time data before any plotting or analysis. The **summary()** function was utilized to produce a statistical summary, including minimum, maximum, median, mean, and quartiles for numerical data, as well as frequency counts for categorical data. This summary helped with an initial evaluation of data quality by pointing out missing values and identifying potential outliers or anomalies that may require data cleaning. In addition to that, the **names()** function is used to examine the column names to make it possible to pinpoint which columns are important for additional analysis. The above process guaranteed a comprehensive initial comprehension of the dataset's composition and quality, leading the way for subsequent data cleaning and analysis tasks.

Data Cleaning, Joining and Manipulation Techniques

Data Cleaning

Data cleaning is an essential step in data analysis to guarantee the precision and uniformity of the data prior to extracting any substantial insights from it to conduct analysis. This report focuses on three main data cleaning techniques which are converting selected column variables to appropriate data type, identifying and addressing missing values in the datasets, and detecting and removing duplicate rows from all four datasets.

```
> ################ (1) Convert selected columns to appropriate data type
> # Format required columns from character to numerical formats'
> pl_march_2021$TP.1 <- as.numeric(pl_march_2021$TP.1)
Warning message:
NAs introduced by coercion
> pl_march_2021$TP.2 <- as.numeric(pl_march_2021$TP.2)
Warning message:
NAs introduced by coercion
> may_2022$TP <- as.numeric(may_2022$TP)
Warning message:
NAs introduced by coercion
>
> # compile all MRP columns to call later during reshaping
> mrp_columns <- c("Ajio.MRP", "Amazon.MRP", "Amazon.FBA.MRP", "Flipkart.MRP",
+                  "Limeroad.MRP", "Myntra.MRP", "Paytm.MRP", "Snapdeal.MRP")
>
> # Ensure MRP columns are numeric and gather them into a single column
> pl_march_2021 <- pl_march_2021 %>%
+   mutate(across(all_of(mrp_columns), as.numeric))
Warning message:
There were 8 warnings in `mutate()`.
The first warning was:
i In argument: `across(all_of(mrp_columns), as.numeric)`.
Caused by warning:
! NAs introduced by coercion
i Run dplyr::last_dplyr_warnings() to see the 7 remaining warnings.
> may_2022 <- may_2022 %>%
+   mutate(across(all_of(mrp_columns), as.numeric))
Warning message:
There were 8 warnings in `mutate()`.
The first warning was:
i In argument: `across(all_of(mrp_columns), as.numeric)`.
Caused by warning:
! NAs introduced by coercion
i Run dplyr::last_dplyr_warnings() to see the 7 remaining warnings.
>
> # Convert Date column of character data type to Date data types for amazon_sales
> amazon_sales$Date <- as.Date(amazon_sales$Date, format = "%m-%d-%y")
```

Code Fragment 1: Convert selected columns to appropriate data type

```
> #################(2) Check for missing values for each datasets
> # Check for amazon_sales
> colSums(is.na(amazon_sales))
            index          Order.ID              Date            Status        Fulfilment      Sales.Channel ship.service.level
                0                 0                 0                 0                 0                 0                 0
            Style               SKU          Category              Size              ASIN     Courier.Status               Qty
                0                 0                 0                 0                 0                 0                 0
         currency            Amount         ship.city        ship.state ship.postal.code      ship.country     promotion.ids
                0              7795                 0                 0                33                 0                 0
              B2B      fulfilled.by       Unnamed..22
                0                 0                 0
>
> # Check for sale_report
> colSums(is.na(sale_report))
    index   SKU.Code Design.No.      Stock   Category       Size      Color
        0          0          0         36          0          0          0
>
> # Check for pl_march_2021
> colSums(is.na(pl_march_2021))
        index           Sku      Style.Id       Catalog      Category        Weight           TP.1           TP.2       MRP.Old  Final.MRP.Old
            0             0             0             0             0             0             6             6             0             0
     Ajio.MRP    Amazon.MRP Amazon.FBA.MRP   Flipkart.MRP   Limeroad.MRP     Myntra.MRP      Paytm.MRP  Snapdeal.MRP
            0             0             0             0             0             0             0             0
>
> # Check for may_2022
> colSums(is.na(may_2022))
        index           Sku      Style.Id       Catalog      Category        Weight            TP       MRP.Old  Final.MRP.Old      Ajio.MRP
            0             0             0             0             0             0             6             0             0             0
   Amazon.MRP Amazon.FBA.MRP   Flipkart.MRP   Limeroad.MRP     Myntra.MRP      Paytm.MRP  Snapdeal.MRP
            0             0             0             0             0             0             0
>
> #### Remove rows with missing values for each datasets
> amazon_sales_cleaned <- na.omit(amazon_sales)
> sale_report_cleaned <- na.omit(sale_report)
> pl_march_2021_cleaned <- na.omit(pl_march_2021)
> may_2022_cleaned <- na.omit(may_2022)
```

Code Fragment 2: Obtain missing value counts in all the variables from each dataset

Converting chosen columns to the correct data types before performing any visualizations and analysation data is an important data cleaning step. Recognizing dates as datetime and numbers as numerical or integer values ensure precise calculations and comparisons. Furthermore, uniform data types in all columns facilitate accurate comparisons and summarizations. This is crucial because it will impact the outcomes when combining data from various sources or when conducting group-by actions in subsequent data manipulation stages. In addition to that, visualizations frequently need data to be in a specific format, such as the time series graph that necessitates date-time data types in order to accurately plot the x-axis over time.

The action of checking for missing values in each selected dataset is done by using the **is.na()** function. This action reveals that all the datasets contain missing values. Hence, the **na.omit()** function is used to remove rows with missing values from all the datasets. Omitting missing values helps to maintain dataset integrity as missing values can introduce bias, reduce the statistical power, and ultimately lead to invalid analysis or conclusions. By removing these incomplete records, it can be ensured that the datasets used in the following conclusions and analysis are based on data that is as accurate and reliable as possible.

```
> ##################(3) Check for duplicates for each datasets
> # Count the number of duplicate rows
> # sale_report_cleaned
> if (sum(duplicated(sale_report_cleaned)) == 0) {
+   paste("There are no duplicate values in sale_report_cleaned")
+ } else {
+   paste("Number of duplicate values in sale_report_cleaned:", sum(duplicated(sale_report_cleaned)))
+ }
[1] "There are no duplicate values in sale_report_cleaned"
> # amazon_sales_cleaned
> if (sum(duplicated(amazon_sales_cleaned)) == 0) {
+   paste("There are no duplicate values in amazon_sales_cleaned")
+ } else {
+   paste("Number of duplicate values in amazon_sales_cleaned:", sum(duplicated(amazon_sales_cleaned)))
+ }
[1] "There are no duplicate values in amazon_sales_cleaned"
> # pl_march_2021_cleaned
> if (sum(duplicated(pl_march_2021_cleaned)) == 0) {
+   paste("There are no duplicate values in pl_march_2021_cleaned")
+ } else {
+   paste("Number of duplicate values in pl_march_2021_cleaned:", sum(duplicated(pl_march_2021_cleaned)))
+ }
[1] "There are no duplicate values in pl_march_2021_cleaned"
> # may_2022_cleaned
> if (sum(duplicated(may_2022_cleaned)) == 0) {
+   paste("There are no duplicate values in may_2022_cleaned")
+ } else {
+   paste("Number of duplicate values in may_2022_cleaned:", sum(duplicated(may_2022_cleaned)))
+ }
[1] "There are no duplicate values in may_2022_cleaned"
```

Code Fragment 3: Results of duplicates for each dataset

The third data cleaning method used is to check and remove duplicate rows from the 4 datasets. This is done by using the **duplicated()** function which flags repeated entries. If there are indeed duplicated rows, one of the rows of the dataset will be removed directly from the original NA-free dataset. Duplicate entries can severely distort the analysis, leading to unreliable results. **If else()** function is also used to identify and report duplicate rows. The if statement evaluates whether this count is zero, indicating no duplicates. If true, it prints a message stating that there are no duplicate values in that dataset. If false, the else block executes, printing the number of duplicate values found. This approach is repeated for each dataset to ensure a thorough check for duplicates across all datasets. However, upon inspection, it was found that the datasets were free of any duplicates, negating the need for further row deletions. Hence, by removing duplicate rows is an essential data cleaning step to avoid skewed analysis results.

Data Manipulation and Data Joining

In relation to problem statement 1, histogram and scatter plot will be used to gain insights on the relationship between sales volume and inventory level by utilizing the cleaned Amazon Sale Report, Sale Report, May-2022 and P L March 2021 datasets.

```
> #------------------------------ Histogram -------------------------------------#
> ##----DATA MANIPULATION-----##
> # Histogram 1: Inventory Distribution of MRPs Across Different Platforms throughout May 2022 and March 2021
> # Remove the 'TP.1' and 'TP.2' columns as its not needed
> pl_march_2021_cleaned <- pl_march_2021_cleaned[, !(names(pl_march_2021_cleaned) %in% c("TP.1", "TP.2"))]
>
> # Remove the 'TP' column as its not needed
> may_2022_cleaned <- may_2022_cleaned[, !(names(may_2022_cleaned) %in% c("TP"))]
>
> # Perform full outer join
> full_out_dataset <- full_join(pl_march_2021_cleaned, may_2022_cleaned)
Joining with `by = join_by(index, Sku, Style.Id, Catalog, Category, Weight, MRP.Old, Final.MRP.Old, Ajio.MRP, Amazon.MRP,
Amazon.FBA.MRP, Flipkart.MRP, Limeroad.MRP, Myntra.MRP, Paytm.MRP,
Snapdeal.MRP)`
>
> # Inspect dataset after data joining
> names(full_out_dataset)
 [1] "index"          "Sku"            "Style.Id"       "Catalog"        "Category"       "Weight"         "MRP.Old"
"Final.MRP.Old"  "Ajio.MRP"       "Amazon.MRP"     "Amazon.FBA.MRP"
[12] "Flipkart.MRP"   "Limeroad.MRP"   "Myntra.MRP"     "Paytm.MRP"      "Snapdeal.MRP"
>
> # compile all MRP columns to call later during reshaping
> mrp_columns <- c("Ajio.MRP", "Amazon.MRP", "Amazon.FBA.MRP", "Flipkart.MRP",
+                  "Limeroad.MRP", "Myntra.MRP", "Paytm.MRP", "Snapdeal.MRP")
>
> # Reshaping the MRP columns into a long format
> histogram_dataset <- full_out_dataset %>%
+   pivot_longer(                       # Transform data from wide format to long format
+     cols = all_of(mrp_columns),       # Select all MRP columns to pivot
+     names_to = "Platform_MRP",        # New column for the MRP platform names
+     values_to = "MRP_Value"           # New column for the MRP values
+   )         # Filter to select rows where the MRP_Value is not missing values
>
> # Check outcome
> head(histogram_dataset)
# A tibble: 6 x 10
  index Sku          Style.Id   Catalog Category Weight MRP.Old Final.MRP.Old Platform_MRP   MRP_Value
  <int> <chr>        <chr>      <chr>   <chr>    <chr>  <chr>   <chr>         <chr>          <dbl>
1     0 Os206_3141_S Os206_3141 Moments Kurta    0.3    2178    2295          Ajio.MRP        2295
2     0 Os206_3141_S Os206_3141 Moments Kurta    0.3    2178    2295          Amazon.MRP      2295
3     0 Os206_3141_S Os206_3141 Moments Kurta    0.3    2178    2295          Amazon.FBA.MRP  2295
4     0 Os206_3141_S Os206_3141 Moments Kurta    0.3    2178    2295          Flipkart.MRP    2295
5     0 Os206_3141_S Os206_3141 Moments Kurta    0.3    2178    2295          Limeroad.MRP    2295
6     0 Os206_3141_S Os206_3141 Moments Kurta    0.3    2178    2295          Myntra.MRP      2295
```

Code Fragment 4: Data Manipulation for Histogram

Before manipulating May-2022 and P L March 2021 datasets for the histogram, P L March 2021 dataset, TP.1 and TP.2 columns and the TP column from the May-2022 dataset were removed as it is not needed. Following that, the cleaned datasets are used to perform full outer join, which is further manipulated by reshaping the MRP related columns into long format to prepare the plotting of histogram on the inventory distribution of MRPs across different platforms.

```
> #------------------------------ Scatter Plot -------------------------------------#
> # Scatter Plot 1: Relationship Between Sales Volume and Inventory Levels
> ##----DATA MANIPULATION-----##
> # Summarize the sales data to get total sales quantity per SKU
> total_sales_per_sku <- amazon_sales_cleaned %>%
+   group_by(SKU) %>%
+   summarise(Total_Sales_Qty = sum(Qty, na.rm = TRUE))
>
> # Left Join the sales data with the inventory data
> sales_inventory_data <- total_sales_per_sku %>%
+   left_join(sale_report_cleaned, by = c("SKU" = "SKU.Code"))
>
> # Check Missing value
> colSums(is.na(sales_inventory_data))
            SKU Total_Sales_Qty           index       Design.No.           Stock        Category            Size
              0               0             571             571             571             571             571
          Color
            571
>
> # Remove Rows with Missing values
> sales_inventory_data = na.omit(sales_inventory_data)
```

Code Fragment 5: Data Manipulation for Scatter Plot

While manipulating Amazon Sale Report and Sale Report dataset for scatter plot, we first summarise the Amazon total sales quantity by SKU and perform left join on the inventory data that Sales Report dataset holds. NAs were found to be present after the left join so it is also removed.

Continuing on regarding problem statement 2, barchart and pie chart will be used to analyse customer preference in terms of their buying behaviour using the cleaned Amazon Sale Report and Sale Report datasets.

```
> #------------------------------ Bar Chart -------------------------------------#
> ##### Bar Chart 1: Category Distribution
> ##----DATA MANIPULATION-----##
> # Get number of counts for product categories
> category_counts <- sale_report_cleaned %>%
+   count(Category) %>%
+   arrange(desc(n))
>
> # Rename the column name n to Counts
> names(category_counts)[names(category_counts) == "n"] <- "Counts"
>
> # Delete row category with empty string in category
> # Delete row for duplicate category Kurta Set
> category_counts = category_counts %>%
+   filter(Category != "" & Category != "KURTA SET")
```

Code Fragment 6: Data Manipulation for Bar Chart 1

For bar charts in using the Sale Report dataset, the dataset is manipulated by aggregating the count number for each product category using **count()** function and having another separate manipulation for another bar chart to get the colour preference by size. For the count number for each product category result, the dataset was further cleaned by removing the empty string category and KURTA SET category duplicate that was found.

```
> ##### Bar Chart 2: Color Preference by Size
> ##----DATA MANIPULATION-----##
> # Check out the color column values for data inconsistency and inaccurate data by
> # using unique function on sales report color column
> unique(sale_report_cleaned$Color)
 [1] "Red"            "Orange"         "Maroon"         "Purple"         "Yellow"         "Green"
 [7] "Pink"           "Beige"          "Navy Blue"      "Black"          "White"          "Brown"
[13] "Gold"           "Chiku"          "Blue"           "Multicolor"     "Peach"          "Grey"
[19] "Olive"          "Dark Green"     "Turquoise Blue" "Mustard"        "Teal"           "Khaki"
[25] "Olive Green"    "TEAL BLUE "     "Cream"          "OFF WHITE"      "Light Green"    "Light Pink"
[31] "Lemon Yellow"   "Sea Green"      "Turquoise Green" "LEMON "        "LEMON"          "Sky Blue"
[37] "LIME GREEN"     ""               "Light Blue"     "Dark Blue"      "Indigo"         "Rust"
[43] "BURGUNDY"       "Wine"           "Light Brown"    "Mauve"          "MINT GREEN"     "CORAL ORANGE"
[49] "CORAL PINK"     "Turquoise"      "AQUA GREEN"     "LIGHT YELLOW"   "Magenta"        "Powder Blue"
[55] "CORAL "         "TEAL GREEN "    "Taupe"          "Charcoal"       "Teal Green"     "NAVY"
[61] "MINT"           "NO REFERENCE"   "CORAL"
>
> #! GOAL: Find if there is any trends in color preferences for different sizes
> # Convert all strings in the 'Color' column to uppercase for data format consistency
> sale_report_cleaned$Color <- toupper(sale_report_cleaned$Color)
>
> # Filter out unrelated and nonspecific color values such as empty string, "NO REFERENCE"  and "MULTICOLOR".
> sale_report_cleaned <- sale_report_cleaned %>% filter(Color != "" & Color != "NO REFERENCE" & Color != "MULTICOLOR")
>
> # Grouping specific colors under a general color category to get overall main colour genre
> sale_report_cleaned <- sale_report_cleaned %>%
+   mutate(Color_Group = case_when(
+     Color %in% c("RED", "MAROON", "RUST", "WINE", "BURGUNDY") ~ "RED",
+     Color %in% c("PINK", "PEACH", "LIGHT PINK", "CORAL PINK", "CORAL ", "CORAL") ~ "PINK",
+     Color %in% c("ORANGE", "BROWN", "KHAKI", "TAUPE", "CORAL ORANGE", "LIGHT BROWN") ~ "ORANGE",
+     Color %in% c("YELLOW", "BEIGE", "GOLD", "CHIKU", "MUSTARD", "CREAM", "LEMON YELLOW", "LEMON ", "LEMON", "LIGHT YELLOW") ~ "YE
LLOW",
+     Color %in% c("GREEN", "OLIVE", "DARK GREEN", "TEAL", "OLIVE GREEN", "LIGHT GREEN", "SEA GREEN", "TEAL GREEN", "LIME GREEN",
"AQUA GREEN", "MINT", "TEAL GREEN ", "MINT GREEN") ~ "GREEN",
+     Color %in% c("BLUE", "NAVY BLUE", "TURQUOISE BLUE", "TEAL BLUE ", "SKY BLUE", "LIGHT BLUE", "DARK BLUE", "POWDER BLUE", "NAV
Y", "TURQUOISE GREEN", "TURQUOISE") ~ "BLUE",
+     Color %in% c("PURPLE", "TEAL GREEN","INDIGO", "MAUVE", "MAGENTA") ~ "PURPLE",
+     Color %in% c("BLACK", "WHITE", "GREY", "OFF WHITE", "CHARCOAL") ~ "GREYSCALE",
+     TRUE ~ as.character(Color) # Keeps the original color if it doesn't match the above
+   ))
> View(sale_report_cleaned)
```

Code Fragment 7: Data Manipulation for Bar Chart 2

Before retrieving the result on colour preference by size, color column character values were converted to uppercase characters for data format consistency as few color character values were found to be either in uppercase, sentence case or initial case format, where you capitalize the first letter of every word. Moreover, filtering was performed to remove unrelated and nonspecific color values such as empty string, "NO REFERENCE" and "MULTICOLOR". Lastly, we group similar color values under its respective general color category to get an overall main colour genre that customers preferred.

```
> #------------------------------ Pie Chart -------------------------------------#
> ##### Pie Chart 1: Orders Fulfillment Type Distribution
> ##----DATA MANIPULATION-----##
> fulfilment_distribution <- table(amazon_sales_cleaned$Fulfilment)
> fulfilment_distribution

  Amazon Merchant
   83621    37528
```

Code Fragment 8: Data Manipulation for Pie Chart

Moving onto manipulating the Amazon Sale Report dataset for the pie chart to get order fulfillment type distribution, we count the order fulfillment by Merchant or Amazon by using **table()** function on the dataset Fulfilment column.

Lastly in relation to problem statement 3, time series plot and line graph will be used to examine seasonal sales trends using the cleaned Amazon Sale Report dataset.

```
> #----------------------------- Time Series -----------------------------------#
> ##----DATA MANIPULATION-----##
> # Select required column variables that is Order.ID and Date
> amazon_sales_ts <- amazon_sales_cleaned %>%
+   select(Order.ID, Date)
>
> # Summarize the number of sales by date
> amazon_sales_summary <- amazon_sales_ts %>%
+   group_by(Date) %>%
+   summarise(Sales_Count = n())
> amazon_sales_summary
# A tibble: 91 × 2
   Date       Sales_Count
   <date>           <int>
 1 2022-03-31         162
 2 2022-04-01        1362
 3 2022-04-02        1456
 4 2022-04-03        1592
 5 2022-04-04        1374
 6 2022-04-05        1517
 7 2022-04-06        1474
 8 2022-04-07        1434
 9 2022-04-08        1582
10 2022-04-09        1528
# i 81 more rows
# i Use `print(n = ...)` to see more rows
```

Code Fragment 9: Data Manipulation for Time Series Plot

For the time series plot, the dataset is manipulated by selecting the required column variables, which are Order.ID and Date. With that, the number of sales by Date column variable was summarised by using **group_by()** and **summarise()** functions.

```
> #----------------------------- Line Graph -----------------------------------#
> ##----DATA MANIPULATION-----##
> # Line Graph 1: Quantity Sold by Category Over Time
> # Summarize the sales by category and date
> sales_by_category <- amazon_sales %>%
+   group_by(Date, Category) %>%
+   summarise(Total_Qty = sum(Qty, na.rm = TRUE))
`summarise()` has grouped output by 'Date'. You can override using the `.groups` argument.
>
> # Convert to all uppercase so it shows nicely in the legend as originally it is a combination of uppercase and lowercase charac
ters
> sales_by_category$Category <- toupper(sales_by_category$Category)
> sales_by_category
# A tibble: 703 × 3
# Groups:   Date [91]
   Date       Category      Total_Qty
   <date>     <chr>             <int>
 1 2022-03-31 BLOUSE                1
 2 2022-03-31 ETHNIC DRESS          1
 3 2022-03-31 SET                  68
 4 2022-03-31 TOP                   9
 5 2022-03-31 WESTERN DRESS         6
 6 2022-03-31 KURTA                71
 7 2022-04-01 BLOUSE               14
 8 2022-04-01 BOTTOM                2
 9 2022-04-01 ETHNIC DRESS         10
10 2022-04-01 SET                 557
# i 693 more rows
# i Use `print(n = ...)` to see more rows
```

Code Fragment 10: Data Manipulation for Line Graph

While for the line graph, the number of sales by Date and Category column variables were summarised by using **group_by()** and **summarise()** functions. With that, the sales by date and category result is further manipulated by converting Category column values to uppercase for nicer legend label display.

Visualisations

Histogram

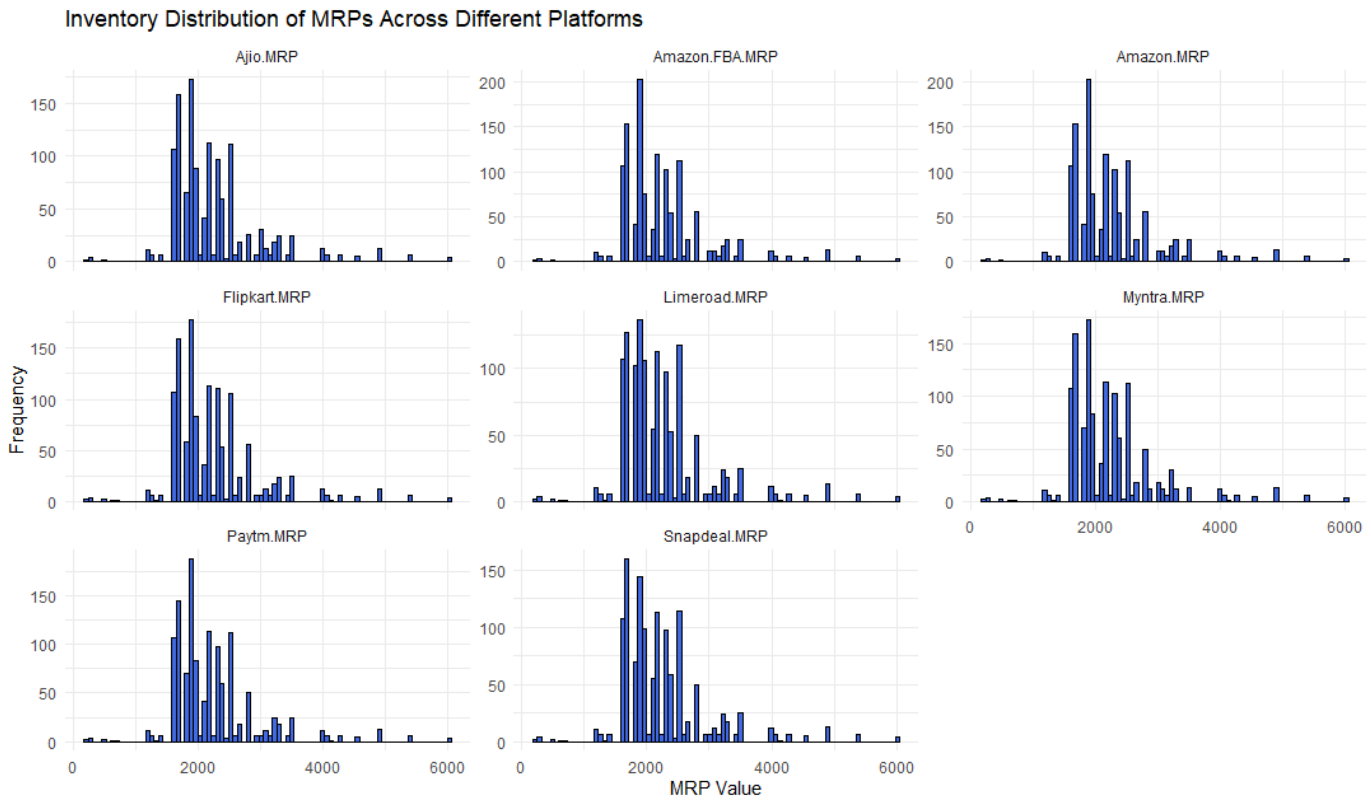

Figure 1. Inventory Distribution of MRPs Across Different Platforms (Histogram)

The histogram above illustrates the distribution of Maximum Retail Prices (MRPs) across eight different e-commerce platforms: Ajio, Amazon, Amazon FBA, Flipkart, Limeroad, Myntra, Paytm, and Snapdeal. This histogram addresses problem statement 1 on inventory management issues.

Each subplot shows the frequency of products available at various MRP values on each platform. For instance, both Ajio and Flipkart exhibit a positive correlation within the MRP range of 1000 to 2000. As the MRP value goes up in this range, the frequency of products also goes up. Still, a clear trend can be observed towards higher MRPs reaching up to 4000, showing a negative correlation. To put it differently, when MRPs go over 2000, the number of products available goes down. This implies that these platforms focus on products priced between 1000-2000 MRP, but also have more expensive items available.

Amazon, including both Amazon.FBA and Amazon.MRP, also shows a distribution following a comparable trend to Ajio and Flipkart, with a significant number of products priced between 1000 and 2000 MRP. This distribution exhibits a positive skew towards higher MRPs within this range but maintains a broader spread, suggesting a more varied price range in its inventory. Limeroad and Snapdeal present similar distributions with peaks in lower MRP ranges which indicate a negative correlation, where higher MRPs correspond to lower product frequencies. Hence, it is concluded that both Limeroad and Snapdeal seem to focus on more affordable price points, catering to budget-conscious consumers. Myntra and Paytm emphasize their range of products primarily priced between 1000 and 3000 MRPs. Myntra shows a relatively equal distribution of products in different price ranges, indicating a fair and well-rounded strategy. On the flip side, Paytm demonstrates a positive relationship in the beginning until about 2000, then shifts to a noticeable downward trend. This shows an emphasis on products with moderate prices.

By analyzing these distributions, companies can identify which price points have the highest inventory levels and which are underrepresented. This helps in synchronizing inventory with current demand and sales data, ensuring that popular price ranges are adequately stocked while avoiding excess inventory in less demanded price ranges. If a platform shows a high frequency of products in the 1000-2000 MRP range, it indicates a significant inventory focus there, suggesting a need for careful stock monitoring and possibly more aggressive sales strategies for higher-priced items. Conversely, price points with low frequency might indicate potential gaps in the product range that need addressing to meet customer demand. Effective inventory management based on these insights can help minimize carrying costs and avoid stockouts, thus ensuring better product availability and customer satisfaction.

Scatter Plot

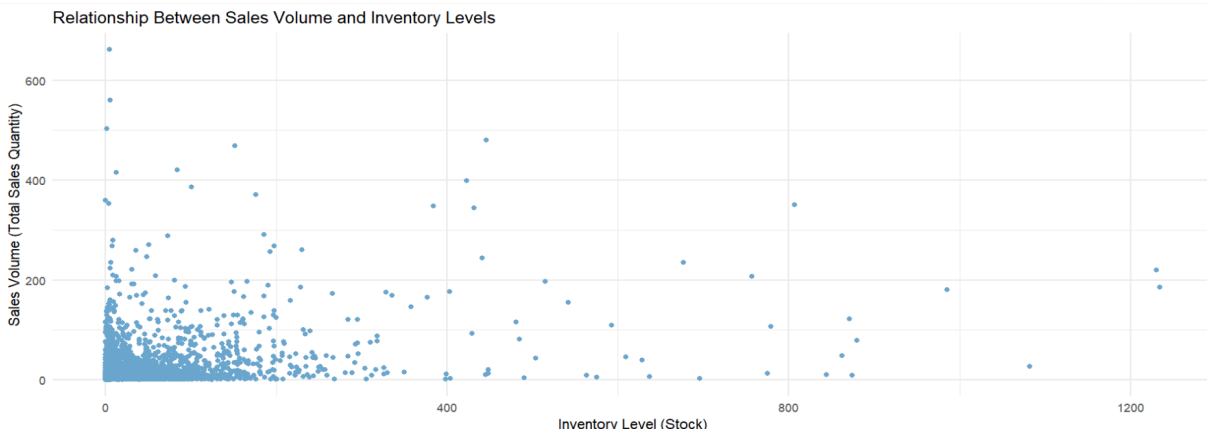

Figure 2. Relationship Between Sales Volume and Inventory Levels (Scatter Plot)

The scatter plot in Figure 2 displays the relationship between sales volume and inventory level. This scatter plot addresses problem statement 1 on inventory management issues. The inventory level is shown on the x-axis of the graph, while the y-axis represents the total sales quantity known as sales volume. The scatter plot is utilized to tackle problem statement 1 regarding the difficulties of e-commerce business effectively managing inventory and how it affects profitability.

It is observed that the majority data points cluster at the lower end of the inventory level spectrum, specifically between 0 and 400 units. Correspondingly, the sales volumes in this cluster are also lower, generally ranging from 0 to 200 units sold. This indicates that lower inventory levels are typically associated with lower sales volumes. On the other hand, higher inventory levels tend to correlate with higher sales volumes. However, this trend is not absolute, as the scatter plot shows variability. Additionally, we notice there are a few outliers where sales volume is high, (the data points exceed 400 units quantity sold)despite relatively low inventory levels (0 to 200 units). This shows us that some products may achieve high sales volumes even with limited stock.

In summary, the scatter plot does not exhibit a strong linear relationship between inventory levels and sales volumes. Instead, it shows that sales volumes can vary greatly at different inventory levels, indicating the complexity of managing inventory and sales in e-commerce.

Bar Chart

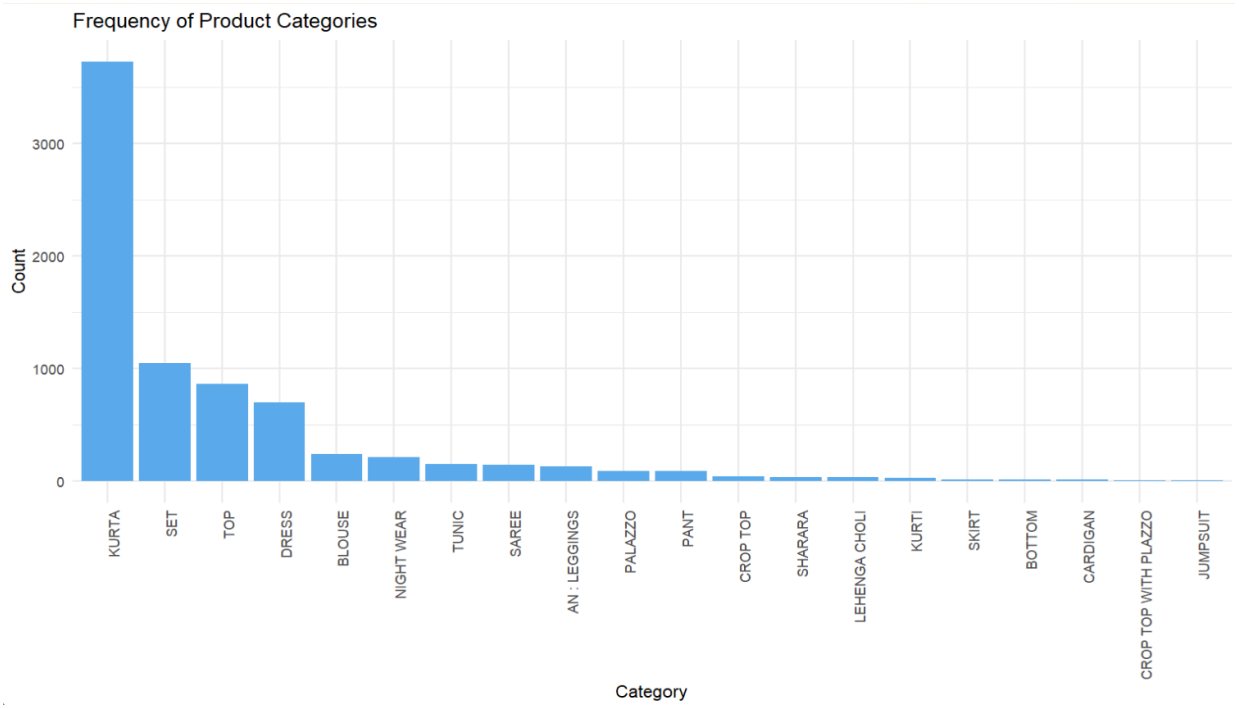

Figure 3. Frequency of Product Categories (Bar Chart)

The bar chart above displays the frequency of different product categories in the sales report, which addresses problem statement 2 understanding customer preferences.

KURTA has the highest frequency with approximately 3500 units. This shows that KURTA is a popular category and has significant stock levels among businesses, making it the dominant product. This suggests that understanding preferences within the KURTA clothing line could be crucial for the business to improve their profitability. Hence, businesses could consider exploring why KURTA is so popular between the consumers and consider tailoring its offerings accordingly to attract more consumers. Categories such as SET, TOP and DRESS have moderate frequency ranging from the 1000 to 2000 units, showing that these categories are also important in the sales data but not as important as the KURTA category. Additionally, we know that categories like BLOUSE, NIGHT WEAR, TUNIC, SAREE, LEGGINGS have low frequencies, each ranging between 200 and 500 units. The remaining categories, such as PALAZZO, PANT, CROP TOP, LEHENGA CHOLI, and KURTI have very low frequencies, below 200 units. There are also categories with almost zero counts such as CROP TOP WITH PALAZZO and JUMPSUIT, representing their minimal presence in sales data.

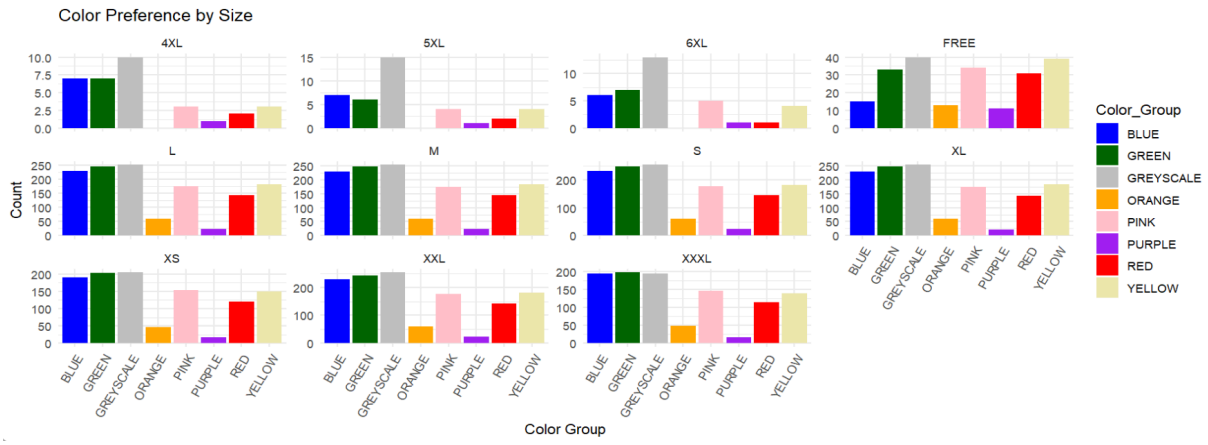

Figure 4. Color Preference by Size (Bar Chart)

The above bar chart illustrates color preferences by size in sales data, which also tackles problem statement 2 which assists e-commerce businesses by providing insights to comprehend customer preferences.

For larger sizes, specifically 4XL, 5XL, and 6XL, the highest frequency is observed for the Greyscale colour group, followed closely by Blue and Green, which have similar frequencies. Bright colours such as Pink, Purple, Red, and Yellow show lower preferences, with Orange colour shirts having a frequency of zero for these sizes.

The color preference patterns are similar for other sizes, including XXXL, XXL, XL, L, M, S, XS, and Free size. Darker shades like Blue, Green, and Greyscale are the top preferences across

these sizes. Pink, Red, and Yellow have moderate frequencies, while Orange and Purple are the least preferred.

Overall, Greyscale clothes are observed to be the most preferred across all sizes indicating a strong consumer preference for neutral colors. Greyscale hues have great versatility. They can be effortlessly combined with other items in a closet so this might be one of the reasons why grayscale colors are more preferred by the consumers. Other than that, grayscale colors are commonly linked with a sense of professionalism. A lot of workplaces have dress codes that either suggest or mandate employees to wear neutral colors. Hence, it can be inferred that dark shades are more popular than bright shades for both everyday and professional wear. This information can be useful for inventory management and marketing strategies, enabling businesses to align their stock and promotional efforts with customer preferences based on size and colour.

Pie Chart

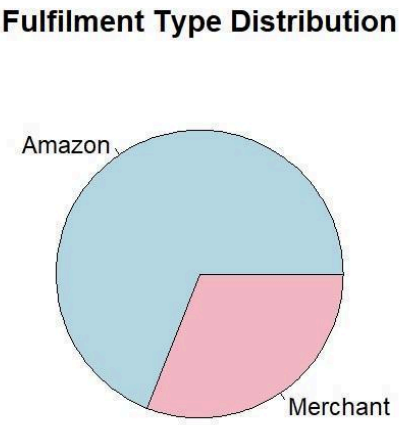

Figure 5. Fulfilment Type Distribution (Pie Chart)

The pie chart titled "Fulfilment Type Distribution" in Figure 5 shows the breakdown of order fulfilment methods for sales made through the Amazon platform, which addresses the problem statement 2: understand customer preferences. It distinguishes between two types of fulfillment: Amazon Fulfilment and Merchant Fulfilment. The larger segment, colored in blue, represents Amazon-fulfilled orders, indicating that a significant majority of products are handled through Amazon's fulfillment services. The smaller segment, colored in pink, represents merchant-fulfilled orders, which account for a lesser portion of the total orders. This distribution suggests a preference for Amazon's fulfillment services, possibly due to benefits such as faster shipping times, reliability, and enhanced customer service. On the other hand, merchant fulfillment, this segment indicates that merchants prefer to manage their fulfillment

independently, which could be due to specific product requirements, cost considerations, or other business strategies. This distribution provides insights into seller preferences and the logistics strategies employed in e-commerce, highlighting the competitive advantage of utilizing Amazon's fulfilment services.

Time Series Plot

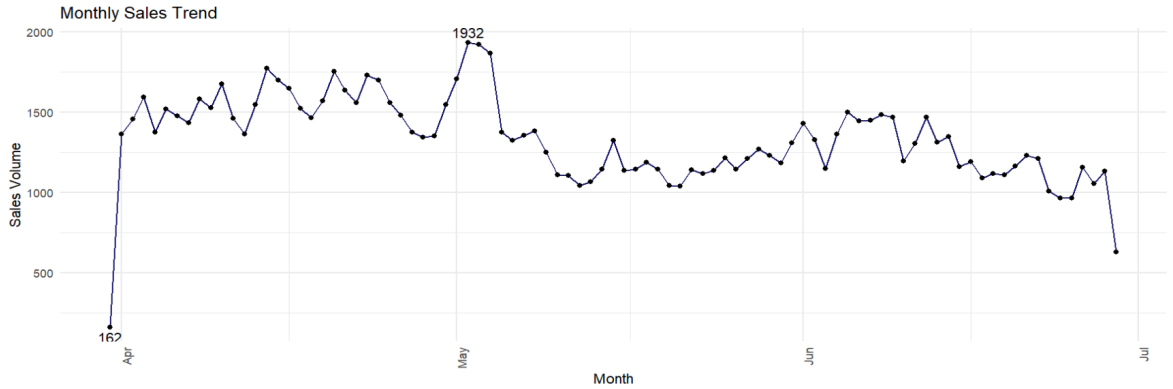

Figure 6. Monthly Sales Trend (Time Series Plot)

The time series plot visualises the monthly sales trend of Amazon sales volume from April till July. This visualisation is created using amazon sales highlights significant patterns in sales activity over this period. This analysis is used to help solve problem statement 3, which revolves around encapsulating dynamic sales patterns that often reflect seasonal consumer behaviour by leveraging the sales data provided in the datasets to identify and analyse these seasonal sales trends.

The sales volume shows a sharp increase at the beginning of April, starting from a low point of 162 transactions at the end of March. This dramatic rise indicates the beginning of a strong sales period, likely driven by new promotions or the launch of new products.

We can observe from the graph above that the maximum sales fell in early May with the highest sales counts of 1932 transactions. The reason may be because Mid-year Sales and Promotions as businesses often run significant sales events in May, including mid-year sales and Mother's Day promotions. The month of May also has seasonal change from spring to summer, and so businesses tend to clear out spring inventory, often leading to discounts and thus leading to increased consumer purchasing power. Moreover, consumers also tend to buy new apparel in preparation for the summer season, driving higher sales volumes.

After the peak in early May, there is a visible decline in sales volume, followed by a relatively stable but gradually decreasing trend through the end of June. Despite occasional spikes, the overall trend is downward, indicating the end of major promotional events and a return to regular sales activity.

The monthly sales trend from April to July showcases the dynamic nature of retail sales on Amazon. Promotional events and seasonal changes significantly influence consumer behaviour, resulting in marked peaks and troughs in sales volume.

Line Graph

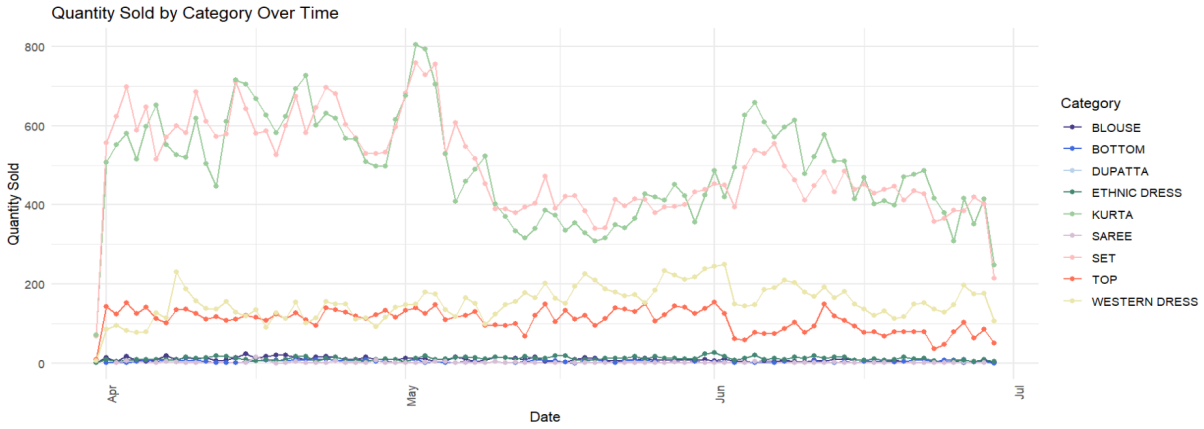

Figure 7. Quantity Sold by Category Over Time (Line Graph)

The line graph in Figure 7 illustrates the quantity of various product categories sold over time from the end of March to July. Every clothes' category is represented by a different coloured line. The y-axis represents the quantity sold, ranging from 0 to 800, and the x-axis represents the time period. This analysis also addresses problem statement 3, which focuses on identifying and analysing seasonal sales trends.

From the graph above, we observed that Set Clothes (pink line) and Kurta (light green line) have the highest quantities sold, frequently reaching peaks around 600-800 units. This suggests that Set clothes and Kurta are popular choices for daily occasions. Kurta is an Indian traditional garment. Additionally, Top (orange line) and Western Dress (yellow line) show moderate sales, ranging between 100 and 300 units. Western Dress reaches a peak between the end of May and early June, likely due to specific events or seasonal trends during that period. Blouse, Bottom, Dupatta, Ethnic Dress, and Saree have relatively lower sales, mostly staying below 100 units. Despite their lower demand, these categories maintain a stable and consistent sales trend throughout the period.

In the time period of April to early May, it can be seen that the highest quantity of clothes sold across most categories, indicating a peak season for clothing purchases. This could be due to seasonal sales during that period that drive higher demand for new apparel. However, starting from June, there is a downward trend in sales across all categories. This decline might suggest a post-peak season slowdown in consumer purchasing behaviour.

Conclusion

Examining Amazon's monthly sales figures and trends in the clothing category provides important insights for improving business strategies. Peak times, such as early April to early May, due to promotions and seasonal events, highlight the significance of aligning marketing strategies to boost sales during these busy periods. It is important to keep a good stock of popular items to meet consumer demand, while less popular items can have a more conservative inventory management.

In addition to that, the analysis on MRP distributions on different e-commerce platforms emphasizes the strategic inventory control that matches the demand based on price points. E-commerce platforms that have a large number of products priced between 1000-2000 MRP require careful inventory management and flexible pricing tactics to increase sales and avoid excess inventory.

Other than that, the analysis on the distribution of fulfillment types, color preferences based on size, and frequencies of product categories gives important insights on consumer preferences and purchasing habits in the competitive e-commerce industry. Customers' preference for Amazon Fulfillment shows that they prioritize quick delivery and dependable service. The analysis of color preferences shows a consistent shift towards darker hues in all sizes, with fewer people preferring brighter colors. This understanding assists in adjusting inventory control and marketing tactics to match consumer color choices, guaranteeing that in-demand colors are adequately supplied and advertised. The analysis also suggests that these popular categories like Kurta, Set, and, Top should be given preference in inventory and marketing strategies. Increasing the inventory of popular items, sticking to favorite color schemes, and making use of fast fulfillment services can greatly enhance customer happiness and boost revenue.

In conclusion, the analysis provides e-commerce companies with the information to make decisions that would adjust to consumer trends, and succeed in a competitive market by better meeting customer preferences.