University of Greenwich ID Number: 001312184

FPT Student ID Number: GCD220076

Module Code: COMP1810

Module Assessment Title: Data and Web Analytics

Lecturer Name: Trần Trọng Minh

Submission Date: 13/08/2024

Table of Contents

Abstract

This report delves into a comprehensive analysis of the 2008 e-shop clothing datasets, employing various data manipulation and visualization techniques using R. The primary focus is on understanding sales trends, revenue distribution, and product popularity across different dimensions such as location, color, price, and time. The analysis provides insights into consumer behavior, sales performance, and the effectiveness of pricing strategies. Descriptive statistics are also explored to understand salary distributions within a company's dataset, offering a glimpse into the financial structure and compensation trends. The report concludes with a critical evaluation of the findings and their implications for business decision-making.

Keywords: data analysis, visualization, R programming, financial structure

1. Introduction

In the rapidly evolving digital marketplace, understanding consumer behavior and sales trends is crucial for sustaining business growth and competitiveness. This report aims to analyze data from the 2008 e-shop clothing dataset, providing a comprehensive overview of product sales, revenue trends, and consumer preferences. By leveraging R for data analysis and visualization, this study explores key aspects such as the impact of location, color, and price on sales, as well as monthly sales trends across different product categories. Additionally, the report includes a descriptive analysis of salary data to examine the financial landscape within a specific organization. The findings from this analysis are expected to provide valuable insights that can guide strategic decisions in marketing, pricing, and human resource management.

2. TASK A

This section will work primarily with e-shop clothing 2008 and shop clothing infor 2008 files with the aim of analyzing the provided datasets using various statistical, visualization and data manipulation techniques in R. The tasks involve extracting relevant data, performing sales and revenue analysis, conducting descriptive statistics and working with the datasets

```r
{r}
# Load required libraries
library(readxl)
library(dplyr)
library(tidyr)
library(reshape2)
library(ggplot2)

# Import data from the Excel file
data <- read_excel("e-shop clothing 2008.xlsx")

# Extract column names by splitting the first row using semicolons
column_names <- strsplit(colnames(data)[1], ";")[[1]]

# Use semicolons to split the data into separate columns
data <- data %>%
  separate(col = colnames(data)[1],
           into = column_names,
           sep = ";", fill = 'right')

# Remove unwanted columns: "year" and "page 2 (clothing model)"
data <- data %>%
  select(-year, -`page 2 (clothing model)`)

# Rename columns for clarity
data <- data %>%
  rename(click_stream = `order(sells)`,
         session_ID = `session ID`)

# Create a new column 'product_sold' indicating if a product was sold
data <- data %>%
  group_by(session_ID) %>%
  mutate(product_sold = ifelse(click_stream == max(click_stream), 1, 0)) %>%
  ungroup()

# Display the first few rows of the modified dataframe
head(data)
View(data)
```

The code loads necessary libraries, imports data from an Excel file, and processes it by splitting the first column into multiple columns using semicolons. It then removes unwanted columns, renames others for clarity, and adds a new column to indicate if a product was sold based on the highest click stream value within each session. And it displays the first few rows of the modified data and opens it in a viewer for inspection.

As a result, we will have a file with separated columns and format like this

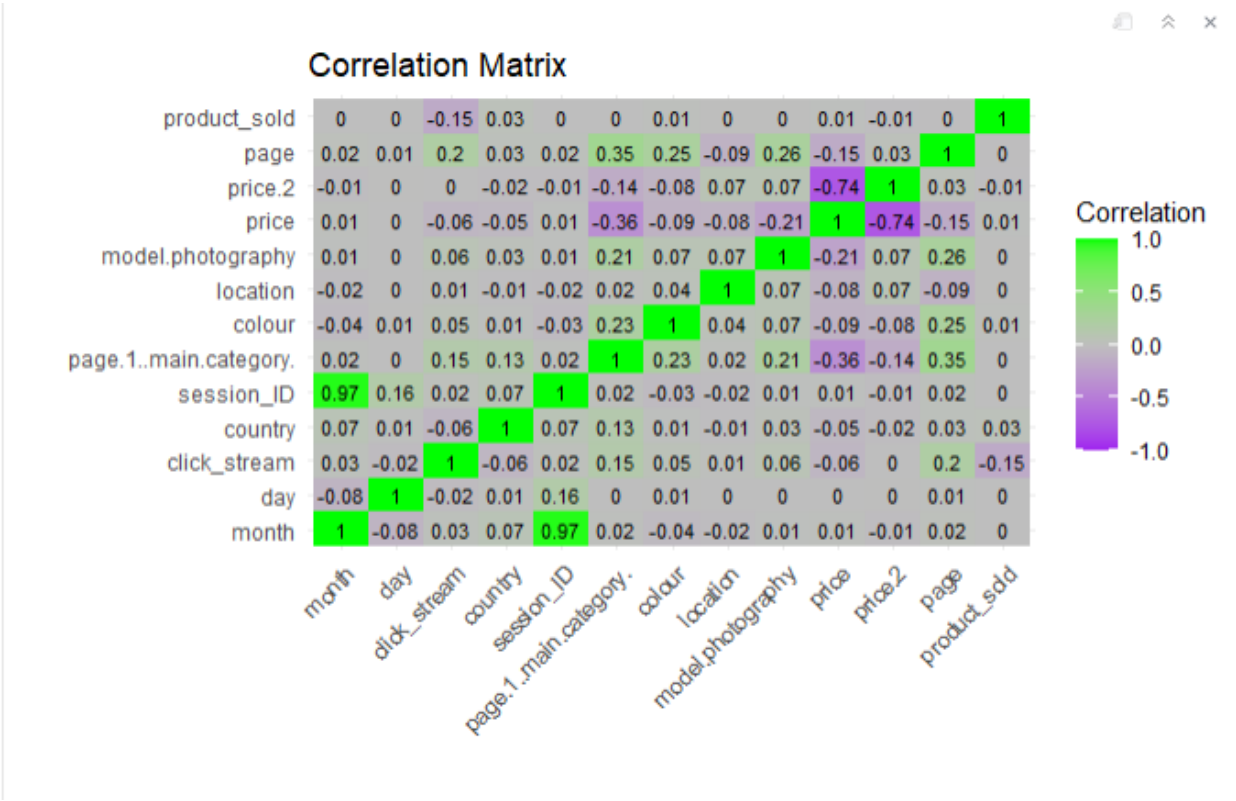
Filter

	month	day	click_stream	country	session_ID	page 1 (main category)	colour	location	model photography	price	price 2	page	product_sold
1	4	1	1	29	1	1	1	5	1	28	2	1	0
2	4	1	2	29	1	1	1	6	1	33	2	1	0
3	4	1	5	29	2	1	3	4	1	38	2	1	0
4	4	1	6	29	2	1	3	4	1	38	2	1	0
5	4	1	1	21	4	1	2	6	1	38	2	2	0
6	4	1	2	21	4	1	2	1	1	62	1	3	0
7	4	1	1	29	7	1	3	4	1	62	1	1	0
8	4	1	1	29	9	1	4	6	1	38	2	1	0
9	4	1	2	29	9	1	8	1	1	28	2	1	0
10	4	1	1	16	10	1	8	1	1	28	2	1	0
11	4	1	1	9	11	1	3	1	1	72	1	1	1
12	4	1	1	38	13	1	8	1	1	28	2	1	0
13	4	1	2	38	13	1	3	1	1	43	2	1	0
14	4	1	3	38	13	1	3	2	1	43	2	1	1
15	4	1	1	9	14	1	4	2	1	43	2	3	1

After that, we will visualize the data to understand the correlations between them and the relationships between different fields.

```r
{r}
# Convert all columns to numeric
data_numeric <- as.data.frame(lapply(data, as.integer))

# Calculate the correlation matrix
correlation_matrix <- cor(data_numeric)

# Convert the correlation matrix to long format
cor_long_format <- reshape2::melt(correlation_matrix)

# Plot the correlation matrix using ggplot2 with a different color scheme
ggplot(cor_long_format, aes(x = Var1, y = Var2, fill = value)) +
  geom_tile() +
  scale_fill_gradient2(low = "purple", mid = "gray", high = "green",
                       midpoint = 0, limits = c(-1, 1),
                       guide = guide_colorbar(title = "Correlation")) +
  labs(title = "Correlation Matrix", x = "", y = "") +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1)) +
  geom_text(aes(label = round(value, 2)), color = "black", size = 3)
```

The code converts all columns in the dataframe to numeric values, calculates the correlation matrix to show the relationships between variables, and then reshapes the matrix into a long format for plotting. Using ggplot2, it creates a heatmap of the correlation matrix, where each tile represents the correlation between variables, colored from purple (low correlation) to green (high correlation). The plot includes a color bar for correlation values, a title, rotated x-axis labels for readability, and text labels showing the exact correlation values.



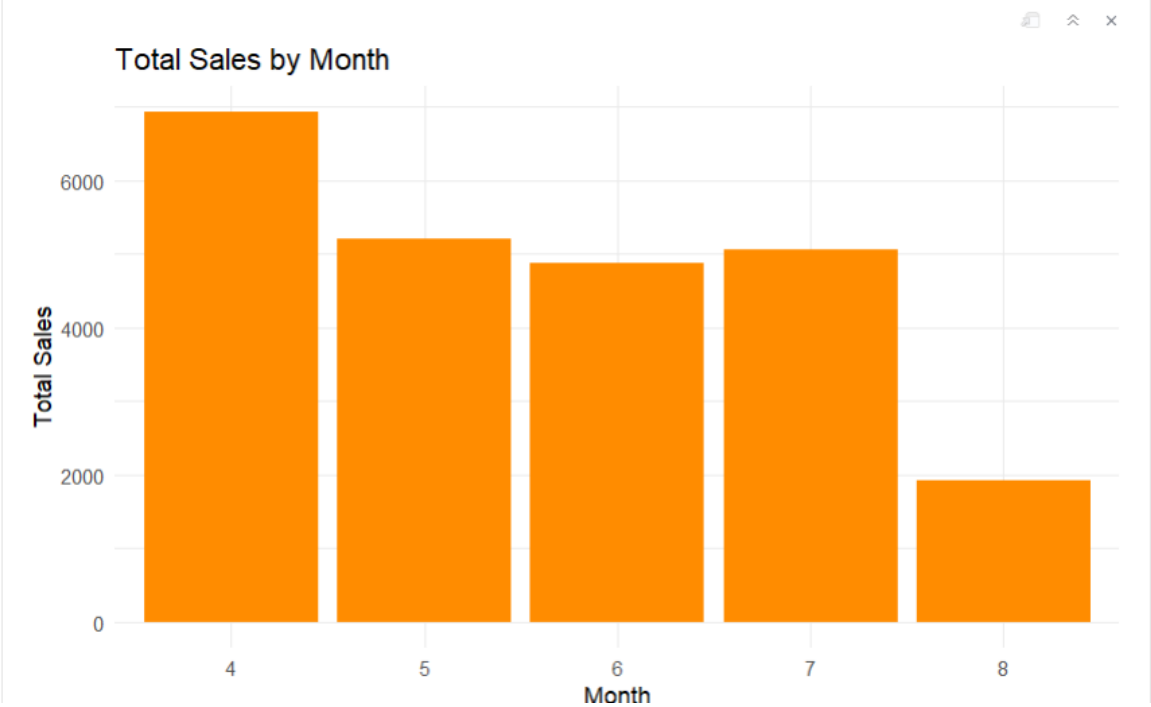
The correlation matrix provides insights into the relationships between different variables:

1. **Colour, Month, Day, Location**: These variables exhibit very weak or no correlation with other variables, suggesting that they may be relatively independent or not significantly related to numerical values such as sales, clicks, or prices.
2. **Strong Negative Correlation between Price and Price 2 (-0.74)**: This strong negative correlation indicates that as the primary price (price) increases, the secondary price (price 2) tends to decrease. This could imply a discounting strategy where one price reflects a promotional or reduced rate.
3. **Moderate Negative Correlation between Page 1 (Main Category) and Price (-0.36)**: This moderate negative correlation suggests that certain product categories are associated with lower prices. This may reflect different pricing strategies across various product lines, where specific categories may be priced lower, possibly to appeal to a broader customer base or to meet competitive pricing in the market.

3. TASK B

The modified code creates a bar chart to visualize total sales by month. It first summarizes the sales data by grouping it by month and calculating the total sales for each month. The ggplot function is then used to generate the bar chart, with geom_bar(stat = "identity") creating bars where the height directly represents the total sales values. The bars are colored dark orange (fill = "darkorange"), and the chart is labeled with appropriate titles for clarity. The theme_minimal() function is applied to give the chart a clean, simple look.

```
# Summarize product_sold by month
monthly_sales_summary <- data %>%
  group_by(month) %>%
  summarize(total_sales = sum(product_sold), .groups = 'drop')

# Bar chart of sales by month
ggplot(monthly_sales_summary, aes(x = month, y = total_sales)) +
  geom_bar(stat = "identity", fill = "darkorange") +
  labs(title = "Total Sales by Month", x = "Month", y = "Total Sales") +
  theme_minimal()
```

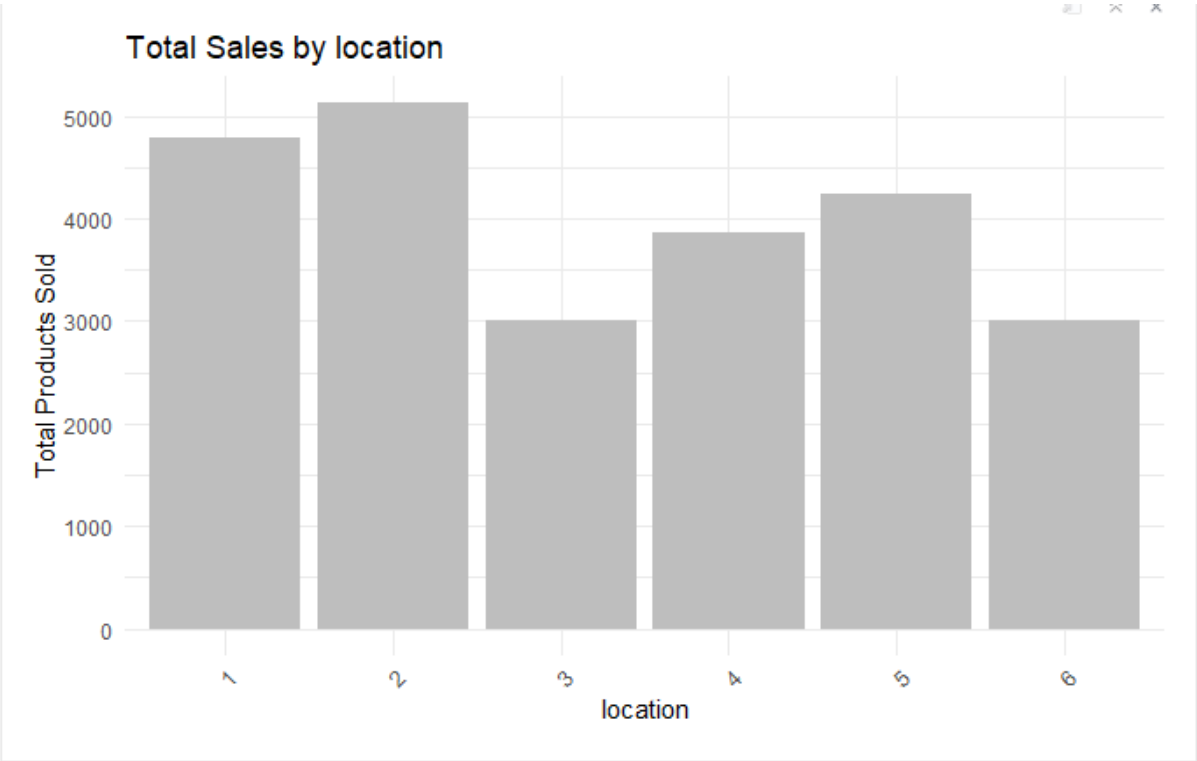


In the table of Total Sales by month, April is the month with the highest sales, which could be due to increased customer demand during the summer season, with a total of 7,000 products sold. People have more time to relax and shop. From May to July, sales remain stable, fluctuating around 5,000. August has the lowest sales, with a total of nearly 2,000 products sold. This may be because, at the end of the summer break, customers are preparing for a busy work season, so their shopping time is limited, leading to a significant decrease in sales.

Next comes the total sale by location section, here is the code to compare revenue based on each different location

```
# Summarize total products sold by location
sales_by_location <- df %>%
  group_by(location) %>%
  summarize(total_sold = sum(product_sold, na.rm = TRUE), .groups = 'drop')

# Plot total sales by location
ggplot(sales_by_location, aes(x = factor(location), y = total_sold)) +
  geom_bar(stat = "identity", fill = "gray") +
  labs(title = "Total Sales by location", x = "location", y = "Total Products Sold"
) +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```

And when visualized, we will get a bar chart graph showing total sales based on local

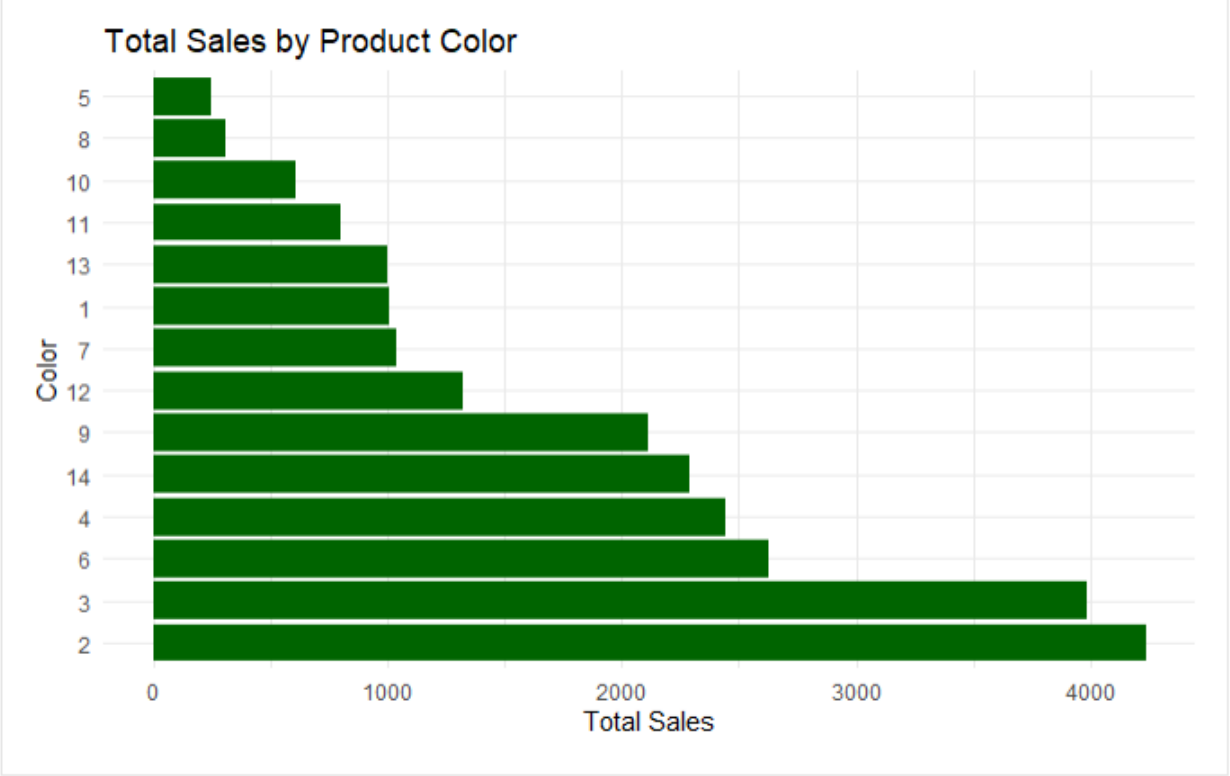


Based on the graph, we can see that the left area of the graph has a higher total number of products than the center and right areas. Location 2 has the highest total product sales with more than 5000 products sold, followed by location 1 which is about 500 products less than location 2 and the total products sold in location 1 is about 4800 to 4900. The total number of products sold from the center area 3 increases gradually to Location 5. In which Location 3 is 3000 lowest equal to location 6, location 4 is about 3900 products and location 5 is 4200 to 4300 products.

Below is a chart showing the number of products based on color:

```r
print(colnames(df))

# Verify the first few rows to confirm 'product_sold' exists and contains data
head(df)

# Summarize total sales by product color
color_sales_summary <- df %>%
  group_by(colour) %>%
  summarize(total_sales = sum(product_sold, na.rm = TRUE), .groups = 'drop')

# Create a bar plot to visualize total sales by product color
ggplot(color_sales_summary, aes(x = reorder(colour, -total_sales), y = total_sales
)) +
  geom_bar(stat = "identity", fill = "darkgreen") +
  labs(title = "Total Sales by Product Color", x = "Color", y = "Total Sales") +
  theme_minimal() +
  coord_flip()
```

Here is the chart:



Based on the chart, we see that the number of products is shown to increase gradually with color 2 being the most sold color and color 5 being the least sold and the difference is very large with color 2 being around 4500 and color 5 being 200-300 products. With the product having color 3 having a sales volume of around 4000, we see that customers really like color 3 and 2 and the product having color 5 8 is underrated with a total sales volume of less than 1000.

Here is the monthly product sales by category chart:

```r
{r}
# Summarize total products sold by category and month
category_sales_summary <- df %>%
  group_by(month, page.1..main.category.) %>%
  summarize(total_sold = sum(product_sold, na.rm = TRUE), .groups = 'drop')

# Plot sales by category and month
ggplot(category_sales_summary, aes(x = month, y = total_sold, color = factor(page.1..main.category.))) +
  geom_line(size = 1) +   # Draws lines connecting the points
  geom_point(size = 3) +  # Adds points to the plot
  labs(title = "Monthly Product Sales by Category", x = "Month", y = "Total Products Sold") +
  scale_color_brewer(palette = "Set1", name = "Product Category") +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```

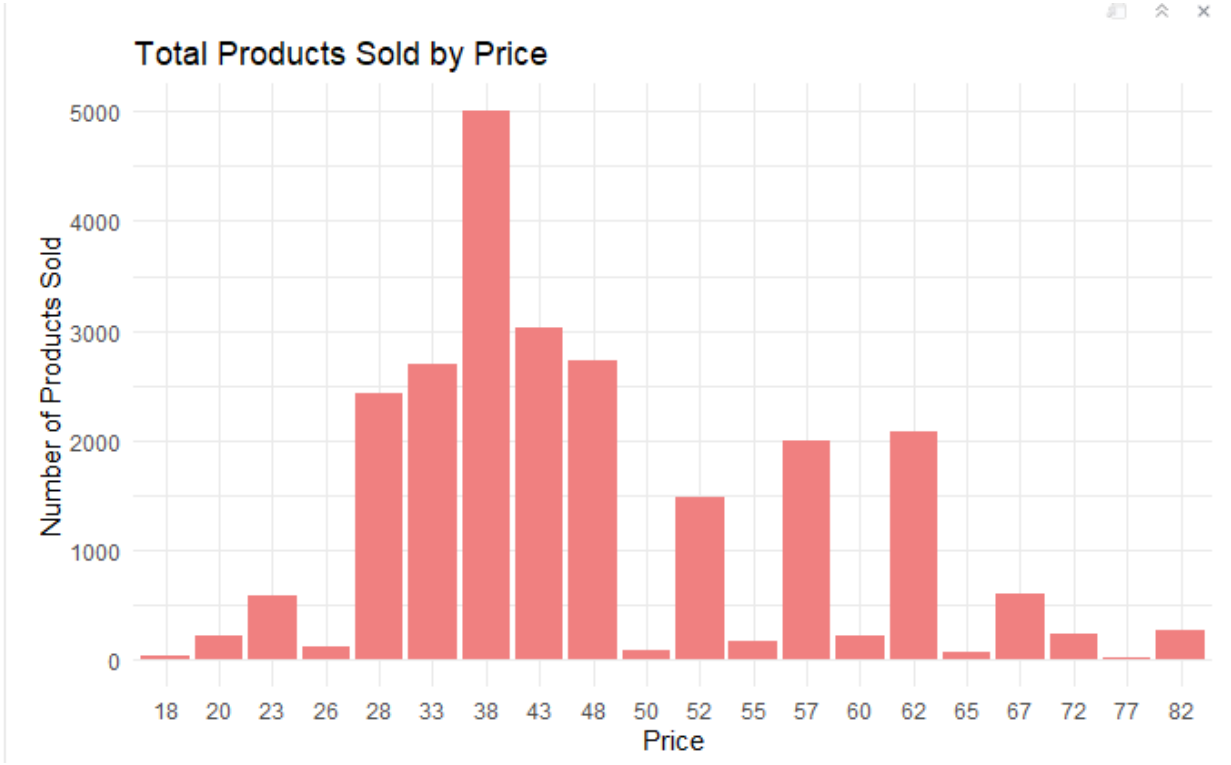
The code summarizes total products sold by category and month, then creates a line plot to visualize these sales trends. It groups the data by month and product category, calculates the total products sold, and plots this information with lines connecting the points and different colors for each category. The plot includes a title, labeled axes, a color legend for product categories, and rotated x-axis labels for better readability.

Based on the graph, we observe that product sales are declining over time. Product Category 1 initially met user demand well, with sales nearing 2000 units, but then experienced a sharp decline. Product Category 2, while not decreasing as quickly as Product 1, showed a steady

decline from April to July, with a significant drop in August. Products 3 and 4, despite being popular in April, also suffered a significant drop in sales.

Next is the chart showing sales by price:

```{r}
# Summarize product sales by price
price_sales_summary <- df %>%
  group_by(price) %>%
  summarize(total_sold = sum(product_sold, na.rm = TRUE), .groups = 'drop')

# Bar plot for product sales by price with updated colors
ggplot(price_sales_summary, aes(x = price, y = total_sold)) +
  geom_bar(stat = "identity", fill = "lightcoral") +
  labs(title = "Total Products Sold by Price", x = "Price", y = "Number of Products Sold") +
  theme_minimal()
```



The chart indicates that most users are willing to spend between 28 and 48, with total product sales rising from 2500 to a peak of 5000 at a price of 38, and then gradually decreasing to around 2800 at a price of 48. Additionally, some higher-priced products, such as those priced at 57 and 62, still attract significant purchases, with a total of 2000 units sold. This suggests that while the ideal price range for products is between 28 and 48, indicating a strong value perception, higher-priced items can also be popular if they offer perceived value or benefits.

Discussion:

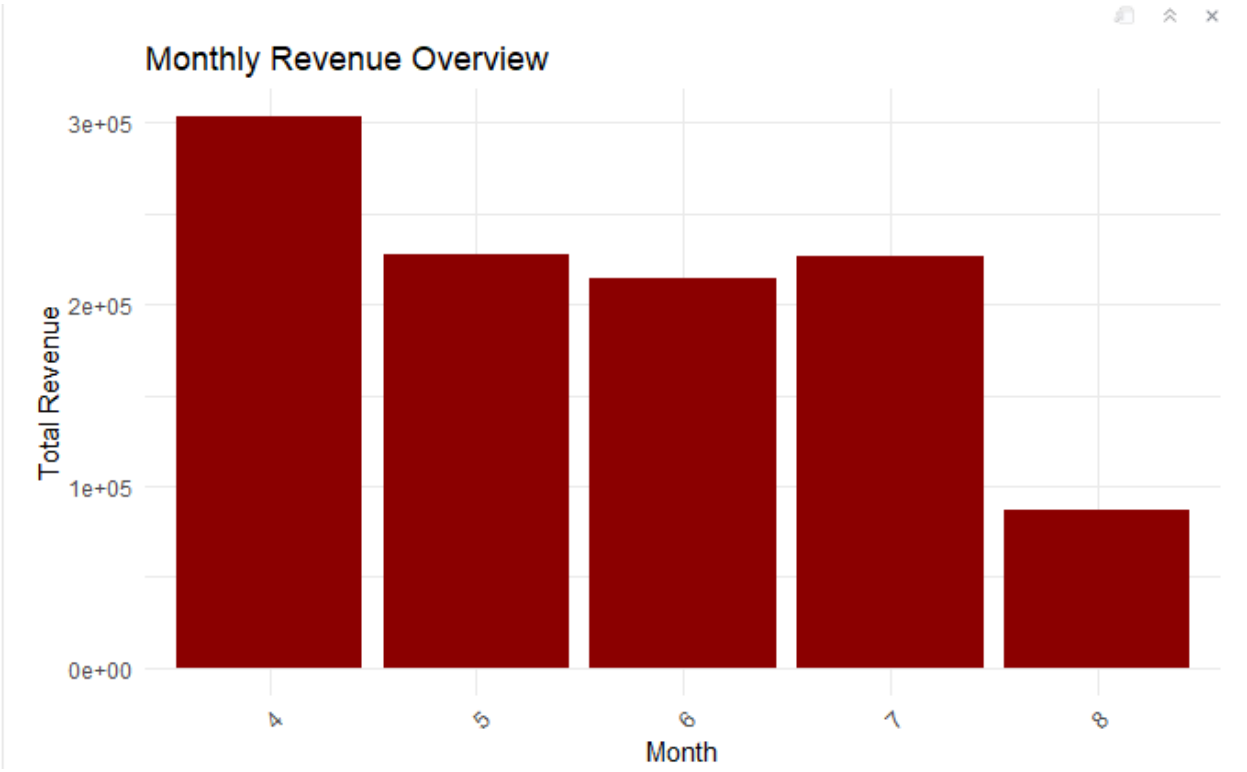
The total sales graph reflects various key insights from the data tables. Location 2 stands out with the highest sales, exceeding 5,000 products, significantly impacting the total sales graph with a prominent peak. Location 1 follows closely, contributing substantially with around 4,800 to 4,900 products sold. Sales from Locations 3 to 5 show a gradual increase, with Location 5 achieving the highest in this range, while Location 6's lower sales dampen the overall performance. Color preferences also play a significant role: Color 2 leads with around 4,500 products sold, while Color 5 lags behind with only 200-300 products. This disparity in color sales affects the total sales graph, with peaks driven by Color 2 and dips corresponding to less popular colors. Additionally, sales trends by product category show an initial high for Category 1, which then sharply declines, and a steady decrease for Category 2. Categories 3 and 4 also experience significant drops in sales. These declining trends are visible as downward slopes on the graph. Lastly, the chart highlights a peak in sales at a price of 38, with a total of 5,000 units sold, while higher prices show decreased sales. Despite the overall decline at higher price points, the graph reflects continued significant sales for premium-priced products, indicating varied consumer behavior across price ranges.

4. TASK C

```r
# Ensure columns are numeric
df <- df %>%
  mutate(
    price = as.numeric(price),
    product_sold = as.numeric(product_sold)
  ) %>%
  mutate(Revenue = price * product_sold)
```

```r
# Calculate total revenue by month
monthly_revenue_summary <- df %>%
  group_by(month) %>%
  summarize(total_revenue = sum(Revenue, na.rm = TRUE), .groups = 'drop')

# Plot the total revenue by month using a bar chart
ggplot(monthly_revenue_summary, aes(x = month, y = total_revenue)) +
  geom_bar(stat = "identity", fill = "darkred") +
  labs(title = "Monthly Revenue Overview", x = "Month", y = "Total Revenue") +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```

The code first ensures that the price and product_sold columns in the dataframe are numeric and then calculates the Revenue for each record by multiplying these columns. It then summarizes the total revenue for each month by grouping the data by month and summing up the Revenue. Finally, it creates a bar chart of the total monthly revenue using ggplot2, with bars filled in dark red, a title, labeled axes, and rotated x-axis labels for better readability.



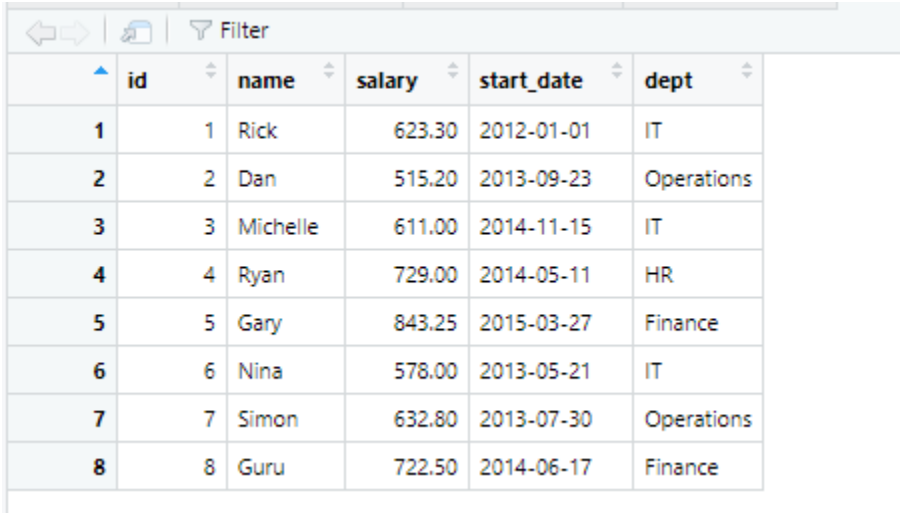
Below is the revenue chart from April to August for the store. Revenue in April is the highest of the five months, while August has the lowest. Revenue in May and July are nearly identical, and June's revenue is lower than both May and July. The high revenue in April may be due to new products attracting a wide range of customers, whereas the lower revenue in subsequent months could be because older products lost their appeal, and new products did not meet the needs of most customers.

5. TASK D

```{r}
# Read data from an Excel file into a data frame
dframe <- read_excel('input.xlsx')

# Separate the first column into six new columns based on whitespace
dframe <- dframe %>%
  separate(col = colnames(dframe)[1], into = c("id", "id2", "name", "salary",
"start_date", "dept"), sep = "\\s+", convert = TRUE)

# Remove the 'id2' column from the data frame
dframe <- dframe %>%
  select(-id2)

dframe$salary <- as.numeric(dframe$salary)

# Display the transformed data frame
View(dframe)
```

The code reads data from an Excel file into a dataframe named dframe, then splits the first column into six new columns (id, id2, name, salary, start_date, dept) using whitespace as the separator. It removes the unnecessary id2 column and converts the salary column to numeric type. Finally, it displays the transformed dataframe in a viewer, though the correct name for the dataframe in the View function should be dframe.

And this is a result after transformed data:



Filter

	id	name	salary	start_date	dept
1	1	Rick	623.30	2012-01-01	IT
2	2	Dan	515.20	2013-09-23	Operations
3	3	Michelle	611.00	2014-11-15	IT
4	4	Ryan	729.00	2014-05-11	HR
5	5	Gary	843.25	2015-03-27	Finance
6	6	Nina	578.00	2013-05-21	IT
7	7	Simon	632.80	2013-07-30	Operations
8	8	Guru	722.50	2014-06-17	Finance

Below is the code to find the mean, mode, standard deviation and variance of salary column:

```r
{r}
# Calculate Mean Salary
mean_salary <- mean(dframe$salary, na.rm = TRUE)

# Calculate Median Salary
median_salary <- median(dframe$salary, na.rm = TRUE)

# Calculate Mode Salary
# Note: Mode is not directly available in base R; we need to define a custom
function
mode_salary <- as.numeric(names(sort(table(dframe$salary), decreasing = TRUE)[1]))

# Calculate Standard Deviation of Salary
sd_salary <- sd(dframe$salary, na.rm = TRUE)

# Calculate Variance of Salary
var_salary <- var(dframe$salary, na.rm = TRUE)

# Print the results
cat("Mean Salary:", mean_salary, "\n")
cat("Median Salary:", median_salary, "\n")
cat("Mode Salary:", mode_salary, "\n")
cat("Standard Deviation of Salary:", sd_salary, "\n")
cat("Variance of Salary:", var_salary, "\n")
```

The salary data for the company yields the following statistics:

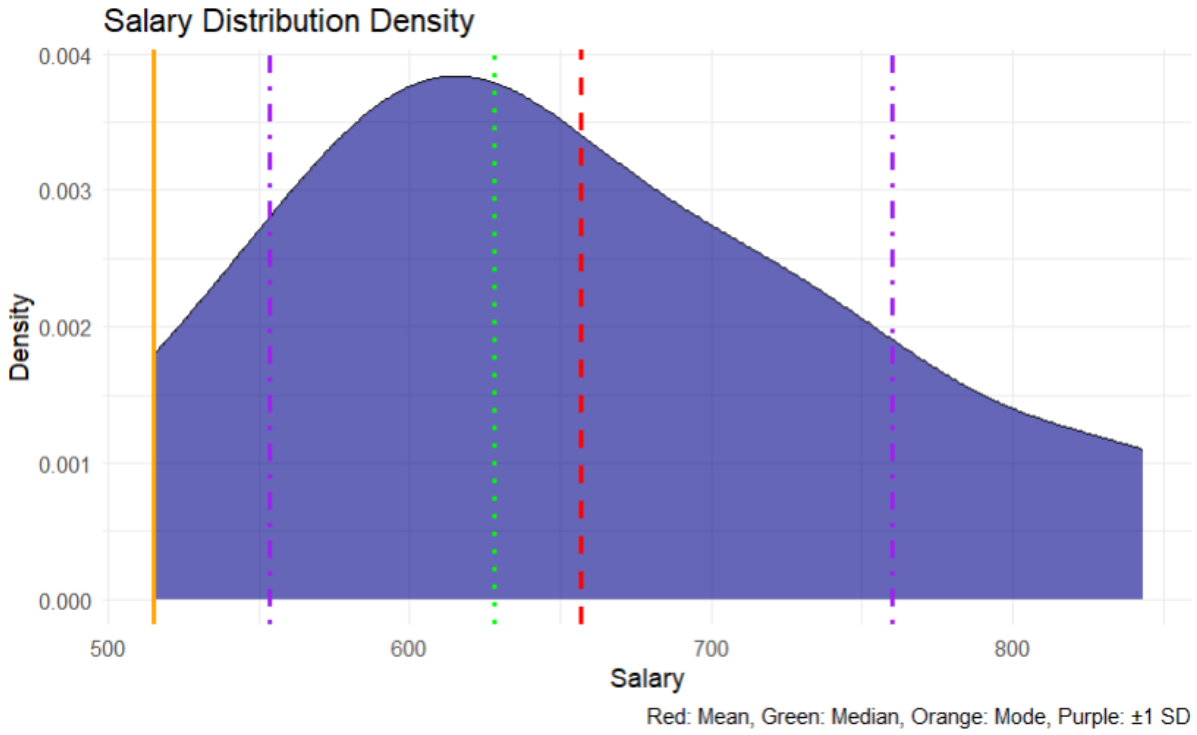
- **Mean Salary:** 656.88
- **Median Salary:** 628.05
- **Mode Salary:** 515.20
- **Standard Deviation of Salary:** 103.06
- **Variance of Salary:** 10621.25

These metrics provide a comprehensive view of the salary distribution within the company, indicating the average, central tendency, and dispersion of salaries.

To gain deeper insights into salary issues, we can examine the salary density plot:

```r
{r}
# Calculate Mean Salary
mean_salary <- mean(dframe$salary, na.rm = TRUE)

# Calculate Median Salary
median_salary <- median(dframe$salary, na.rm = TRUE)

# Calculate Mode Salary
mode_salary <- as.numeric(names(sort(table(dframe$salary), decreasing = TRUE)[1]))

# Calculate Standard Deviation of Salary
sd_salary <- sd(dframe$salary, na.rm = TRUE)

# Calculate Variance of Salary (already calculated using sd_salary)
var_salary <- var(dframe$salary, na.rm = TRUE)

# Plot the density of salaries with mean, median, mode, and standard deviation
lines
ggplot(dframe, aes(x = salary)) +
  geom_density(fill = "darkblue", alpha = 0.6) +
  geom_vline(aes(xintercept = mean_salary), color = "red", linetype = "dashed",
size = 1, show.legend = TRUE) +
  geom_vline(aes(xintercept = median_salary), color = "green", linetype = "dotted",
size = 1, show.legend = TRUE) +
  geom_vline(aes(xintercept = mode_salary), color = "orange", linetype = "solid",
size = 1, show.legend = TRUE) +
  geom_vline(aes(xintercept = mean_salary + sd_salary), color = "purple", linetype
= "dotdash", size = 1, show.legend = TRUE) +
  geom_vline(aes(xintercept = mean_salary - sd_salary), color = "purple", linetype
= "dotdash", size = 1, show.legend = TRUE) +
  labs(title = "Salary Distribution Density", x = "Salary", y = "Density",
       caption = "Red: Mean, Green: Median, Orange: Mode, Purple: ±1 SD") +
  theme_minimal()
```

This code calculates key statistical measures for salary data—mean, median, mode, and standard deviation—and visualizes them on a density plot. The density plot shows the distribution of salaries, with vertical lines added to represent each calculated statistic. A red dashed line marks the mean salary, a green dotted line shows the median, an orange solid line indicates the mode, and purple dot-dashed lines highlight one standard deviation above and below the mean. This visualization allows for easy comparison of these central tendency measures and the overall salary distribution.



 Salary Distribution: The curve represents the density of salary values, showing how common different salary ranges are. The highest peak of the curve indicates the salary range where most employees are concentrated.

 Mean Salary (Red dashed line): The mean salary is marked by the red dashed line. It falls towards the higher end of the distribution, indicating that the average salary is somewhat higher than the most common salaries (the mode).

 Median Salary (Green dotted line): The median salary, represented by the green dotted line, lies slightly to the left of the mean. This suggests that the distribution is right-skewed, meaning that there are more lower salary values, but a few high salaries are pulling the mean upward.

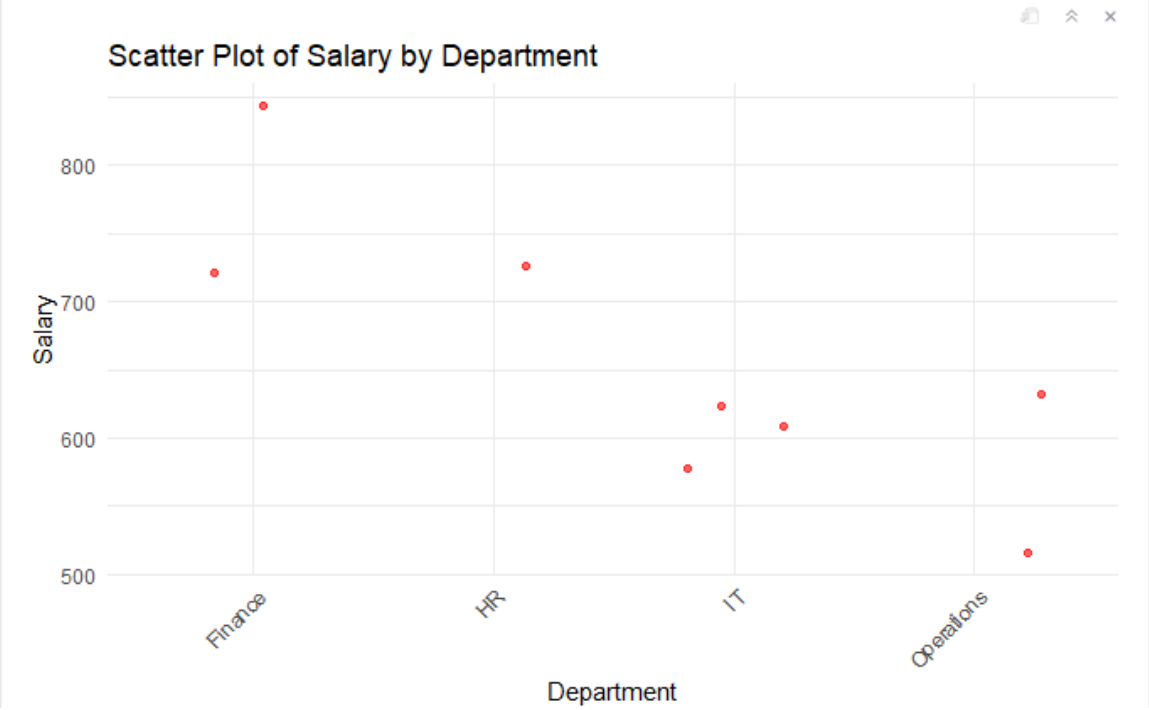
 Mode Salary (Orange solid line): The mode is represented by the orange solid line, located at the peak of the distribution curve. This is the most frequently occurring salary value, and it is lower than both the median and the mean, reinforcing the skewness observed.

 Standard Deviation (Purple dot-dashed lines): The purple dot-dashed lines mark one standard deviation above and below the mean. These lines enclose the central portion of the data, showing where the bulk of salary values lie. The placement of these lines indicates the range within which most employees' salaries fall.

The right-skewed distribution indicates that while most salaries are clustered around a certain value, there are some higher salaries that pull the average up. The fact that the mode is lower than the median and mean suggests that more employees earn less than the average, with fewer employees earning significantly more, contributing to the skew. Additionally, the spread of the standard deviation shows that although most employees' salaries are close to the average, there is noticeable variability in pay within the company.

To explore salaries across different departments in the company, we can visualize this using a scatter plot. The scatter plot will show how salaries are distributed among various departments, helping to identify patterns or discrepancies:

```r
{r}
# Convert 'dept' to a factor to ensure it's treated as a categorical variable
dframe$dept <- as.factor(dframe$dept)

# Create a scatter plot showing the relationship between department and salary
ggplot(dframe, aes(x = dept, y = salary)) +
  geom_point(color = "red", alpha = 0.6, position = position_jitter(width = 0.3)) +
 # Scatter plot with jitter to avoid overlapping points
  labs(title = "Scatter Plot of Salary by Department",
       x = "Department",
       y = "Salary") +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))  # Rotate x-axis labels
for better readability
```



The code converts the dept column to a factor to ensure it's treated as a categorical variable and then creates a scatter plot to visualize the relationship between department and salary. Points are plotted in red with some horizontal jitter to prevent overlap, and the plot includes a title and axis labels. The minimal theme is used for a clean look, and x-axis labels are rotated 45 degrees for improved readability.

The graph highlights salary variations across departments. It shows that the Finance department generally offers higher salaries compared to others. In contrast, the IT department exhibits a broader salary range, with some employees earning significantly less and others earning comparable salaries to those in Finance. The Operations department has a more limited salary range, with most salaries clustered around the mid-level.

6. Task E

6.1 Task Ei

```{r}
# Load the starwars dataset
star_wars_df <- starwars

# Replace "none" with NA
star_wars_df[star_wars_df == "none"] <- NA

# View the cleaned dataframe
View(star_wars_df)
```

The code replaces "none" values in the starwars dataset with NA to handle missing data and then displays the cleaned dataframe.

And it will make the file look like this

	name	height	mass	hair_color	skin_color	eye_color	birth_year	sex	gender	homeworld	species	films	vehicles	starsh
1	Luke Skywalker	172	77.0	blond	fair	blue	19.0	male	masculine	Tatooine	Human	c("A New Hope", "The Empire Strikes Back", "Return [...]	c("Snowspeeder", "Imperial Speeder Bike")	c("X-
2	C-3PO	167	75.0	NA	gold	yellow	112.0	NA	masculine	Tatooine	Droid	c("A New Hope", "The Empire Strikes Back", "Return [...]	character(0)	chara
3	R2-D2	96	32.0	NA	white, blue	red	33.0	NA	masculine	Naboo	Droid	c("A New Hope", "The Empire Strikes Back", "Return [...]	character(0)	chara
4	Darth Vader	202	136.0	NA	white	yellow	41.9	male	masculine	Tatooine	Human	c("A New Hope", "The Empire Strikes Back", "Return [...]	character(0)	TIE A
5	Leia Organa	150	49.0	brown	light	brown	19.0	female	feminine	Alderaan	Human	c("A New Hope", "The Empire Strikes Back", "Return [...]	Imperial Speeder Bike	chara
6	Owen Lars	178	120.0	brown, grey	light	blue	52.0	male	masculine	Tatooine	Human	c("A New Hope", "Attack of the Clones", "Revenge o [...]	character(0)	chara
7	Beru Whitesun Lars	165	75.0	brown	light	blue	47.0	female	feminine	Tatooine	Human	c("A New Hope", "Attack of the Clones", "Revenge o [...]	character(0)	chara
8	R5-D4	97	32.0	NA	white, red	red	NA	NA	masculine	Tatooine	Droid	A New Hope	character(0)	chara
9	Biggs Darklighter	183	84.0	black	light	brown	24.0	male	masculine	Tatooine	Human	A New Hope	character(0)	X-wir
10	Obi-Wan Kenobi	182	77.0	auburn, white	fair	blue-gray	57.0	male	masculine	Stewjon	Human	c("A New Hope", "The Empire Strikes Back", "Return [...]	Tribubble bongo	c("Je
11	Anakin Skywalker	188	84.0	blond	fair	blue	41.9	male	masculine	Tatooine	Human	c("The Phantom Menace", "Attack of the Clones", "R [...]	c("Zephyr-G swoop bike", "XJ-6 airspeeder")	c("Na
12	Wilhuff Tarkin	180	NA	auburn, grey	fair	blue	64.0	male	masculine	Eriadu	Human	c("A New Hope", "Revenge of the Sith")	character(0)	chara
13	Chewbacca	228	112.0	brown	unknown	blue	200.0	male	masculine	Kashyyyk	Wookiee	c("A New Hope", "The Empire Strikes Back", "Return [...]	AT-ST	c("Mi

The code filters the star_wars_df dataframe to include only those rows where eye_color is not "black" and height is greater than 150. This results in a subset of the data that meets both criteria.

```{r}
# Filter by eyes color and height
star_wars_df %>%
  filter(eye_color != "black",
         height >150)
```

And this is the result of that code

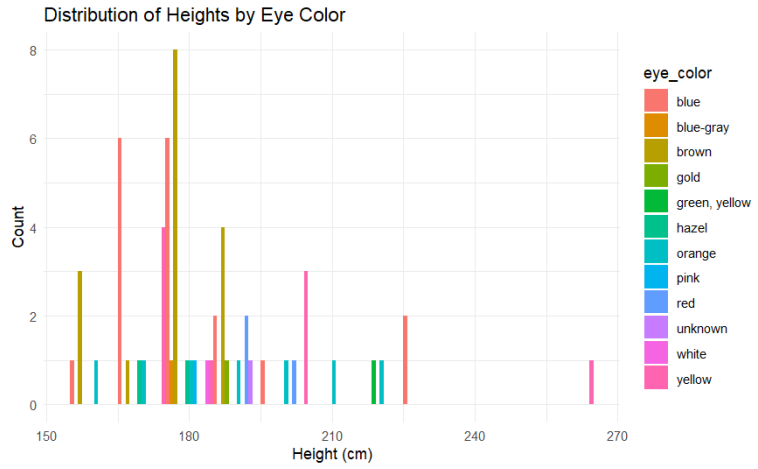
A tibble: 61 × 14

name <chr>	height <int>	mass <dbl>	hair_color <chr>
Luke Skywalker	172	77.0	blond
C-3PO	167	75.0	*NA*
Darth Vader	202	136.0	*NA*
Owen Lars	178	120.0	brown, grey
Beru Whitesun Lars	165	75.0	brown
Biggs Darklighter	183	84.0	black
Obi-Wan Kenobi	182	77.0	auburn, white
Anakin Skywalker	188	84.0	blond
Wilhuff Tarkin	180	*NA*	auburn, grey
Chewbacca	228	112.0	brown

1-10 of 61 rows | 1-4 of 14 columns Previous 1 2 3 4 5 6 7 Next

And here is the visualization code to create the graph after filtering
The code uses the ggplot2 and dplyr libraries to filter and visualize data from the star_wars_df dataframe. It first filters the data to exclude rows where eye_color is "black" and height is 150 cm or less. Then, it creates a histogram of the heights, grouping and filling the bars by eye_color. The histogram bins are set to 10 cm in width and are positioned side by side for each eye color. The plot is titled "Distribution of Heights by Eye Color," with axes labeled "Height (cm)" and "Count," and uses a minimal theme for a clean appearance. Note that the comment refers to creating a scatter plot, but the code actually generates a histogram.

```r
{r}
# Filter the data
filtered_data <- star_wars_df %>%
  filter(eye_color != "black", height > 150)

# Create the scatter plot
ggplot(filtered_data, aes(x = height, fill = eye_color)) +
  geom_histogram(binwidth = 10, position = "dodge") +
  labs(title = "Distribution of Heights by Eye Color",
       x = "Height (cm)",
       y = "Count") +
  theme_minimal()
```



Looking at the chart, we see that brown eye color is the most common eye color with a count of 8, followed by blue and yellow with an equal count of 6.

6.2 Task Eii

```r
{r}
# Calculate BMI and round to 2 decimal places
star_wars_df <- star_wars_df %>%
  mutate(bmi = mass / (height / 100)^2) %>%
  mutate(bmi = round(bmi, 2))

# View the updated dataframe
View(star_wars_df)
```

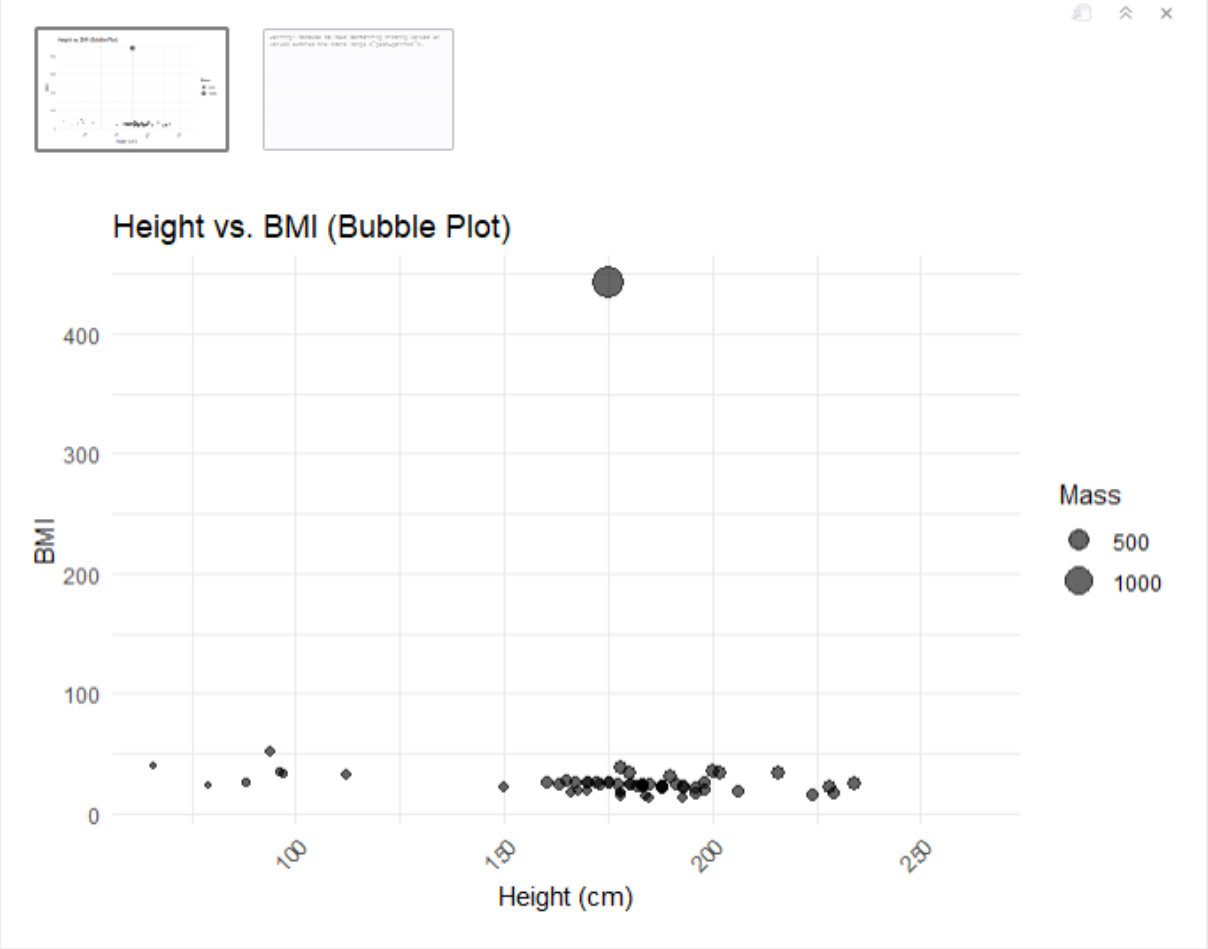
Here is how to calculate BMI and round to 2 decimal places

And after calculating BMI, the data table will have a bmi column like this

vehicles	starships	bmi
c("Snowspeeder", "Imperial Speeder Bike")	c("X-wing", "Imperial shuttle")	26.03
character(0)	character(0)	26.89
character(0)	character(0)	34.72
character(0)	TIE Advanced x1	33.33
Imperial Speeder Bike	character(0)	21.78
character(0)	character(0)	37.87
character(0)	character(0)	27.55
character(0)	character(0)	34.01
character(0)	X-wing	25.08
Tribubble bongo	c("Jedi starfighter", "Trade Federation cruiser", […]	23.25

Here is plot a graph of height against BMI, and provide an explanation for your graph

```r
{r}
ggplot(star_wars_df, aes(x = height, y = bmi, size = mass)) +
  geom_point(alpha = 0.6) +
  labs(title = "Height vs. BMI (Bubble Plot)",
       x = "Height (cm)",
       y = "BMI",
       size = "Mass") +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```



The bubble plot visualizes the relationship between height and BMI for characters in the star_wars_df dataframe, with bubble size representing mass. The X-axis shows height in centimeters, and the Y-axis displays BMI. Larger bubbles indicate higher mass, while smaller bubbles denote lower mass, with semi-transparent blue bubbles allowing overlap visibility. The plot reveals how BMI varies with height, potentially indicating trends where taller characters may have higher or lower BMI. Large bubbles at various heights and BMIs help illustrate how mass influences BMI, while smaller bubbles suggest a diverse mass range among characters. The plot also highlights clusters of characters with similar height, BMI, and mass, and outliers with atypical mass values, offering insights into unique or exceptional cases.

7. Conclusion

The analysis of the 2008 e-shop clothing datasets reveals significant patterns in consumer preferences and sales performance across various dimensions. The study highlights the importance of location, color, and pricing in driving sales, with certain locations and product attributes consistently outperforming others. Monthly sales trends indicate a decline in product demand over time, underscoring the need for continuous innovation and adaptation in product offerings. The examination of salary data further provides a detailed understanding of the company's compensation structure, suggesting areas for potential improvement in salary distribution. Overall, the insights gained from this report can inform more effective business strategies, ultimately enhancing customer satisfaction and profitability.