Data 622 - Homework #1: Exploratory Data Analysis

Shri Tripathi

2024-10-21

Pre-work

1. Visit the following websites to explore datasets of various sizes (100 to 5 million records):

    - Excel BI Analytics Datasets
    - Kaggle Datasets

2. Select two files to download based on your computer's capabilities (one small and one large).
3. Review the structure and content of both datasets (e.g., size, dependencies, labels).
4. Consider similarities and differences between the datasets.
5. Think about how to analyze and predict outcomes using the datasets.
6. Based on the data, select two suitable machine learning algorithms.

Deliverable

Write an essay explaining your selection of machine learning algorithms, how they relate to the datasets, and your analysis goals.

Exploratory Analysis Using R or Python

1. Perform exploratory analysis on the datasets using R or Python.
2. Submit code, including any errors or warnings encountered during the analysis.
3. Test both datasets and compare the results.

Questions to Address

- Are the columns of your data correlated?
- Are there labels in your data? Did that impact your choice of algorithm?
- What are the pros and cons of each algorithm you selected?
- How does your choice of algorithm relate to the datasets?
- Which result would you trust for a business decision?
- Could the analysis be prone to errors with too much or too little data?
- How does the analysis compare between datasets?

```r
# Upload the libraries needed.
library(rpart.plot)
library(rpart)
library(RColorBrewer)
library(corrplot)
library(GGally)
library(scales)
library(tidyr)
library(tidyverse)
library(dplyr)
library(ggplot2)
library(tabulate)
library(knitr)
library(kableExtra)
library(summarytools)
library(stats)
library(class)
```

```r
# Define URLs for datasets from the external source
small_sales_url <- "https://raw.githubusercontent.com/Shriyanshh/Data-622/refs/heads/main/100%20Sales%20
large_sales_url <- "https://raw.githubusercontent.com/Shriyanshh/Data-622/refs/heads/main/10000%20Sales%

# Read datasets and store them in separate data frames
small_sales_data <- read.csv(small_sales_url)
large_sales_data <- read.csv(large_sales_url)
```

Overview

This analysis focuses on exploring and interpreting sales data from two datasets: one with 100 sales records and another with 10,000 sales records. The objective is to gain insights into the data, understand trends in sales performance across different regions and product types, and identify patterns in financial metrics such as total revenue, total cost, and profit margins. Using various statistical and visual techniques, we aim to assess the correlation between key variables and evaluate the variability in sales volumes, pricing strategies, and overall profitability.

The report will cover both datasets, starting with a detailed exploratory data analysis (EDA) that includes summary statistics, visualizations, and a thorough examination of relationships between variables. Additionally, we will look at potential business implications of these findings, helping to uncover opportunities for improving sales strategies, optimizing costs, and increasing profits.

Finally, the analysis will also provide a comparative view between the small and large datasets, highlighting any notable differences or similarities in the trends observed.

Small Dataset

Content of the Tables:

```r
# The head() function shows the first few rows of the dataframe, providing an initial glimpse of the da
head(small_sales_data)
```

Head of the Dataframe:

```
##                              Region               Country       Item.Type
## 1            Australia and Oceania                Tuvalu       Baby Food
## 2 Central America and the Caribbean               Grenada          Cereal
## 3                            Europe                Russia Office Supplies
## 4               Sub-Saharan Africa Sao Tome and Principe          Fruits
## 5               Sub-Saharan Africa                Rwanda Office Supplies
## 6            Australia and Oceania       Solomon Islands       Baby Food
##   Sales.Channel Order.Priority Order.Date  Order.ID Ship.Date Units.Sold
## 1       Offline              H  5/28/2010 669165933 6/27/2010       9925
## 2        Online              C  8/22/2012 963881480 9/15/2012       2804
## 3       Offline              L   5/2/2014 341417157  5/8/2014       1779
## 4        Online              C  6/20/2014 514321792  7/5/2014       8102
## 5       Offline              L   2/1/2013 115456712  2/6/2013       5062
## 6        Online              C   2/4/2015 547995746 2/21/2015       2974
##   Unit.Price Unit.Cost Total.Revenue Total.Cost Total.Profit
## 1     255.28    159.42    2533654.00 1582243.50    951410.50
## 2     205.70    117.11     576782.80  328376.44    248406.36
## 3     651.21    524.96    1158502.59  933903.84    224598.75
## 4       9.33      6.92      75591.66   56065.84     19525.82
## 5     651.21    524.96    3296425.02 2657347.52    639077.50
## 6     255.28    159.42     759202.72  474115.08    285087.64
```

```
# The summary() function generates key statistics such as mean, median, quartiles, and range, giving an
summary(small_sales_data)
```

Summary Statistic:

```
##     Region             Country           Item.Type         Sales.Channel
##  Length:100         Length:100         Length:100         Length:100
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##
##
##
##  Order.Priority      Order.Date           Order.ID          Ship.Date
##  Length:100         Length:100         Min.   :114606559   Length:100
##  Class :character   Class :character   1st Qu.:338922488   Class :character
##  Mode  :character   Mode  :character   Median :557708561   Mode  :character
##                                        Mean   :555020412
##                                        3rd Qu.:790755081
##                                        Max.   :994022214
##    Units.Sold     Unit.Price       Unit.Cost      Total.Revenue
##  Min.   : 124   Min.   :  9.33   Min.   :  6.92   Min.   :   4870
##  1st Qu.:2836   1st Qu.: 81.73   1st Qu.: 35.84   1st Qu.: 268721
##  Median :5382   Median :179.88   Median :107.28   Median : 752314
##  Mean   :5129   Mean   :276.76   Mean   :191.05   Mean   :1373488
##  3rd Qu.:7369   3rd Qu.:437.20   3rd Qu.:263.33   3rd Qu.:2212045
##  Max.   :9925   Max.   :668.27   Max.   :524.96   Max.   :5997055
##    Total.Cost       Total.Profit
##  Min.   :   3612   Min.   :   1258
##  1st Qu.: 168868   1st Qu.: 121444
```

```
##  Median : 363566   Median : 290768
##  Mean   : 931806   Mean   : 441682
##  3rd Qu.:1613870   3rd Qu.: 635829
##  Max.   :4509794   Max.   :1719922
```

The dataset consists of 100 entries for each variable, which include a mix of data types such as character, integer, and numeric. The descriptive statistics provide valuable insights into the distribution patterns and central tendencies of the numerical variables.

```
# The str() function gives a detailed view of the dataset's structure, showing the data types and forma
str(small_sales_data)
```

Structure of Tables:

```
## 'data.frame':    100 obs. of  14 variables:
##  $ Region        : chr  "Australia and Oceania" "Central America and the Caribbean" "Europe" "Sub-Sa
##  $ Country       : chr  "Tuvalu" "Grenada" "Russia" "Sao Tome and Principe" ...
##  $ Item.Type     : chr  "Baby Food" "Cereal" "Office Supplies" "Fruits" ...
##  $ Sales.Channel : chr  "Offline" "Online" "Offline" "Online" ...
##  $ Order.Priority: chr  "H" "C" "L" "C" ...
##  $ Order.Date    : chr  "5/28/2010" "8/22/2012" "5/2/2014" "6/20/2014" ...
##  $ Order.ID      : int  669165933 963881480 341417157 514321792 115456712 547995746 135425221 871543
##  $ Ship.Date     : chr  "6/27/2010" "9/15/2012" "5/8/2014" "7/5/2014" ...
##  $ Units.Sold    : int  9925 2804 1779 8102 5062 2974 4187 8082 6070 6593 ...
##  $ Unit.Price    : num  255.28 205.7 651.21 9.33 651.21 ...
##  $ Unit.Cost     : num  159.42 117.11 524.96 6.92 524.96 ...
##  $ Total.Revenue : num  2533654 576783 1158503 75592 3296425 ...
##  $ Total.Cost    : num  1582244 328376 933904 56066 2657348 ...
##  $ Total.Profit  : num  951411 248406 224599 19526 639078 ...
```

This dataset comprises 100 records with a total of 14 variables. The variables can be categorized into two types: categorical and numerical. The categorical variables include information such as Region, Country, Item Type, Sales Channel, Order Priority, Order Date, and Ship Date, which help describe the geographic and logistical elements of each transaction.

On the other hand, the numerical variables, which include Order ID, Units Sold, Unit Price, Unit Cost, Total Revenue, Total Cost, and Total Profit, quantify key financial metrics like sales figures, pricing, and associated costs. The diversity in regions, countries, and product categories captured in the dataset provides a solid foundation for exploring trends in sales and financial outcomes.

Dataset Characteristics (Structure, Size, Dependencies, Labels, etc.):

```
# nrow() returns the total number of rows in the dataset.
numbs_of_rows <- nrow(small_sales_data)
print(paste("Total rows in the dataset:", numbs_of_rows))
```

Number of Rows and Columns:

```
## [1] "Total rows in the dataset: 100"
```

```
# ncol() retrieves the total number of columns in the dataset.
numbs_of_columns <- ncol(small_sales_data)
print(paste("Total columns in the dataset:", numbs_of_columns))
```

```
## [1] "Total columns in the dataset: 14"
```

```
# names() lists all the column names (variables) present in the dataset.
names(small_sales_data)
```

```
##  [1] "Region"         "Country"        "Item.Type"      "Sales.Channel"
##  [5] "Order.Priority" "Order.Date"     "Order.ID"       "Ship.Date"
##  [9] "Units.Sold"     "Unit.Price"     "Unit.Cost"      "Total.Revenue"
## [13] "Total.Cost"     "Total.Profit"
```

```
# Calculate the total number of missing values (NA) across the entire dataset.
total_na_count <- sum(is.na(small_sales_data))
print("Total count of missing values in the dataset:")
```

Number of NA's:

```
## [1] "Total count of missing values in the dataset:"
```

```
print(total_na_count)
```

```
## [1] 0
```

```
# Calculate the number of missing values (NA) for each individual column.
na_per_column <- colSums(is.na(small_sales_data))
print("Count of missing values in each column:")
```

```
## [1] "Count of missing values in each column:"
```

```
print(na_per_column)
```

```
##         Region        Country      Item.Type  Sales.Channel Order.Priority
##              0              0              0              0              0
##     Order.Date       Order.ID      Ship.Date     Units.Sold     Unit.Price
##              0              0              0              0              0
##      Unit.Cost  Total.Revenue     Total.Cost   Total.Profit
##              0              0              0              0
```

There are no missing values (NA) present in this dataset

```r
# Remove the "Order.ID" column and keep only numeric variables from the dataset.
numeric_sales_metrics <- small_sales_data[sapply(small_sales_data, is.numeric) & colnames(small_sales_da

# Check for any missing values in the numeric columns.
if (any(is.na(numeric_sales_metrics))) {
  cat("Warning: Missing values found in numeric columns. Handle these before proceeding with calculatio
} else {
  # Compute the standard deviation and variance for all numeric variables.
  standard_deviation <- apply(numeric_sales_metrics, 2, sd)
  variance <- apply(numeric_sales_metrics, 2, var)

  # Create a data frame to display the results for standard deviation and variance.
  result_small_sales_data <- data.frame(Variable = names(numeric_sales_metrics), Standard_Deviation = s

  # Format and print the result as a nicely styled HTML table.
  result_table <- kable(result_small_sales_data, "html") %>%
    kable_styling()

  # Show the table.
  print(result_table)
}
```

Standard Deviation and Variance:

```
## <table class="table" style="margin-left: auto; margin-right: auto;">
##  <thead>
##   <tr>
##    <th style="text-align:left;">   </th>
##    <th style="text-align:left;"> Variable </th>
##    <th style="text-align:right;"> Standard_Deviation </th>
##    <th style="text-align:right;"> Variance </th>
##   </tr>
##  </thead>
## <tbody>
##   <tr>
##    <td style="text-align:left;"> Units.Sold </td>
##    <td style="text-align:left;"> Units.Sold </td>
##    <td style="text-align:right;"> 2794.4846 </td>
##    <td style="text-align:right;"> 7.809144e+06 </td>
##   </tr>
##   <tr>
##    <td style="text-align:left;"> Unit.Price </td>
##    <td style="text-align:left;"> Unit.Price </td>
##    <td style="text-align:right;"> 235.5922 </td>
##    <td style="text-align:right;"> 5.550370e+04 </td>
##   </tr>
##   <tr>
##    <td style="text-align:left;"> Unit.Cost </td>
##    <td style="text-align:left;"> Unit.Cost </td>
##    <td style="text-align:right;"> 188.2082 </td>
##    <td style="text-align:right;"> 3.542232e+04 </td>
##   </tr>
```

```
##   <tr>
##    <td style="text-align:left;"> Total.Revenue </td>
##    <td style="text-align:left;"> Total.Revenue </td>
##    <td style="text-align:right;"> 1460028.7068 </td>
##    <td style="text-align:right;"> 2.131684e+12 </td>
##   </tr>
##   <tr>
##    <td style="text-align:left;"> Total.Cost </td>
##    <td style="text-align:left;"> Total.Cost </td>
##    <td style="text-align:right;"> 1083938.2522 </td>
##    <td style="text-align:right;"> 1.174922e+12 </td>
##   </tr>
##   <tr>
##    <td style="text-align:left;"> Total.Profit </td>
##    <td style="text-align:left;"> Total.Profit </td>
##    <td style="text-align:right;"> 438537.9071 </td>
##    <td style="text-align:right;"> 1.923155e+11 </td>
##   </tr>
## </tbody>
## </table>
```

- **Units Sold**:
  - Exhibits a wide range in the number of items sold.
  - Indicates a diverse sales landscape.
  - Likely influenced by factors such as:
    * Product popularity
    * Market demand
    * Seasonal fluctuations.
- **Unit Price**:
  - Shows stability across different products and time periods.
  - Reflects a consistent pricing strategy.
  - Could indicate a deliberate pricing choice or a potential need for pricing review.
- **Unit Cost**:
  - Displays moderate variability.
  - May be impacted by factors such as:
    * Raw material costs
    * Production efficiency
    * Supplier negotiations.
- **Total Revenue**:
  - Shows significant fluctuations in overall income.
  - Suggests variability in sales performance.
  - Likely affected by external market conditions and internal factors.
- **Total Cost**:
  - Demonstrates notable variation, indicating changes in operational expenses.
  - Understanding these fluctuations is key for cost management and resource allocation.
- **Total Profit**:
  - Reflects changes in net financial outcomes.
  - Further analysis is required to understand what drives these variations.
  - Crucial for devising strategies to improve profitability and business performance.

Relationships between numeric variables using a correlation matrix.

```
# Select numeric columns from the dataset
numeric_data_cols <- sapply(small_sales_data, is.numeric)

# Generate a correlation matrix for the numeric columns
cor_matrix <- cor(small_sales_data[, numeric_data_cols])
cor_matrix
```

See below visualization for Correlation.

```
##                 Order.ID  Units.Sold  Unit.Price   Unit.Cost Total.Revenue
## Order.ID       1.0000000 -0.22290682 -0.19094121 -0.21320058    -0.3146876
## Units.Sold    -0.2229068  1.00000000 -0.07048559 -0.09223245     0.4477845
## Unit.Price    -0.1909412 -0.07048559  1.00000000  0.98726981     0.7523596
## Unit.Cost     -0.2132006 -0.09223245  0.98726981  1.00000000     0.7156226
## Total.Revenue -0.3146876  0.44778449  0.75235960  0.71562263     1.0000000
## Total.Cost    -0.3289444  0.37474592  0.78790543  0.77489522     0.9839277
## Total.Profit  -0.2346378  0.56455046  0.55736525  0.46721391     0.8973269
##               Total.Cost Total.Profit
## Order.ID      -0.3289444   -0.2346378
## Units.Sold     0.3747459    0.5645505
## Unit.Price     0.7879054    0.5573652
## Unit.Cost      0.7748952    0.4672139
## Total.Revenue  0.9839277    0.8973269
## Total.Cost     1.0000000    0.8040911
## Total.Profit   0.8040911    1.0000000
```

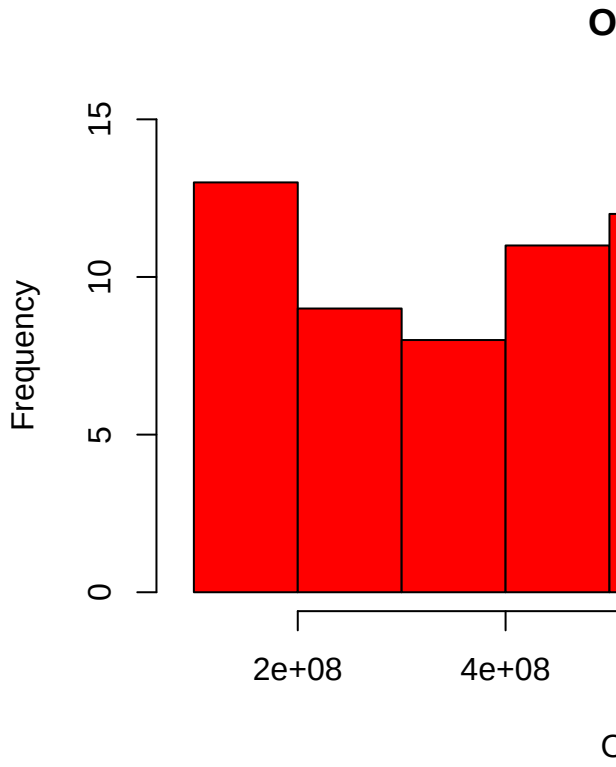
The correlation matrix highlights the relationships between various variable pairs within the dataset. Key findings include:

- **Units Sold** and **Total Profit** show a strong positive correlation of 0.5645, suggesting that higher sales volumes are linked to greater profitability.
- **Unit Price** and **Unit Cost** have a nearly linear relationship, with a strong positive correlation of 0.9873, indicating that as the cost of goods increases, the selling price follows suit.
- **Total Revenue** and **Total Cost** are also highly correlated with a value of 0.9839, signifying that these metrics are closely tied, as expected in most financial data.
- **Order ID** displays a weak negative correlation with most other variables, indicating that it has minimal linear relationships with other features in the dataset.

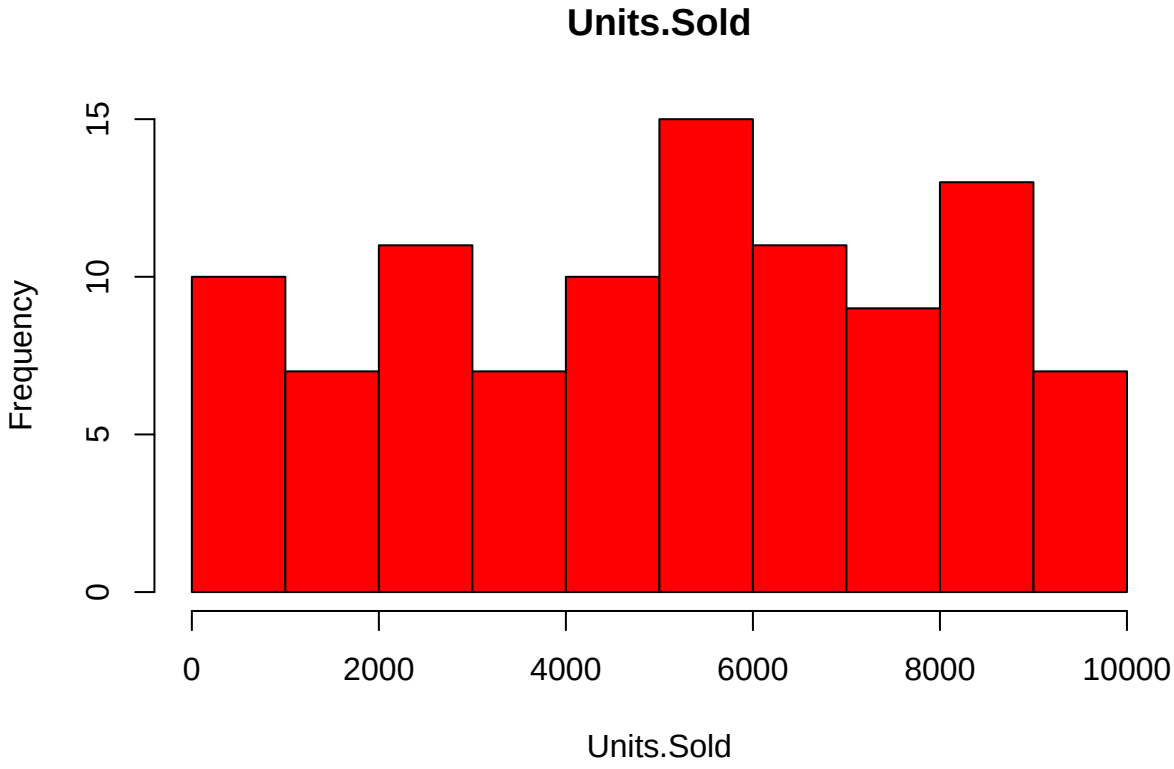
Overall, the correlation matrix provides key insights into how variables interact with each other. Positive correlations suggest that as one variable increases, so does the other, while negative correlations indicate an inverse relationship. The magnitude of these correlations, with values closer to 1 or -1, signifies the strength of the relationship between variables."

Visualization

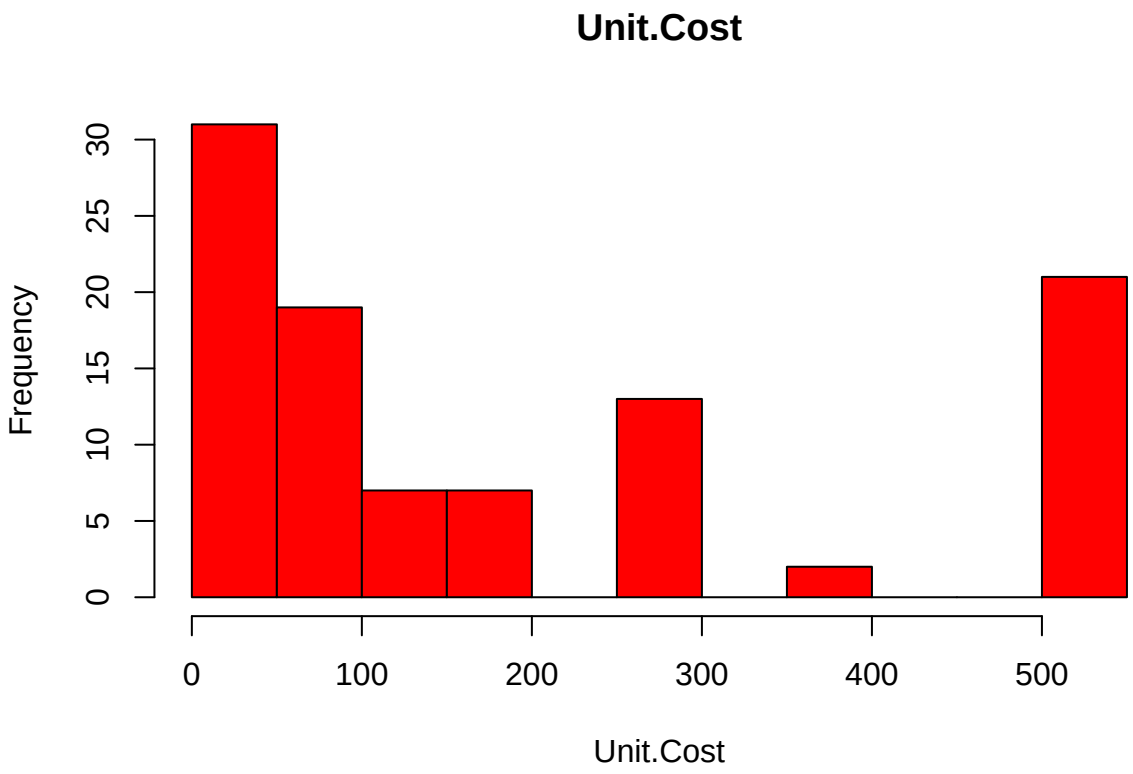
```r
# Identify the numeric columns in the dataset
numeric_data_cols <- sapply(small_sales_data, is.numeric)

# Generate histograms for each numeric column, using red as the color
for (col in names(small_sales_data)[numeric_data_cols]) {
  hist(small_sales_data[[col]], main = col, xlab = col, col = "Red")
}
```

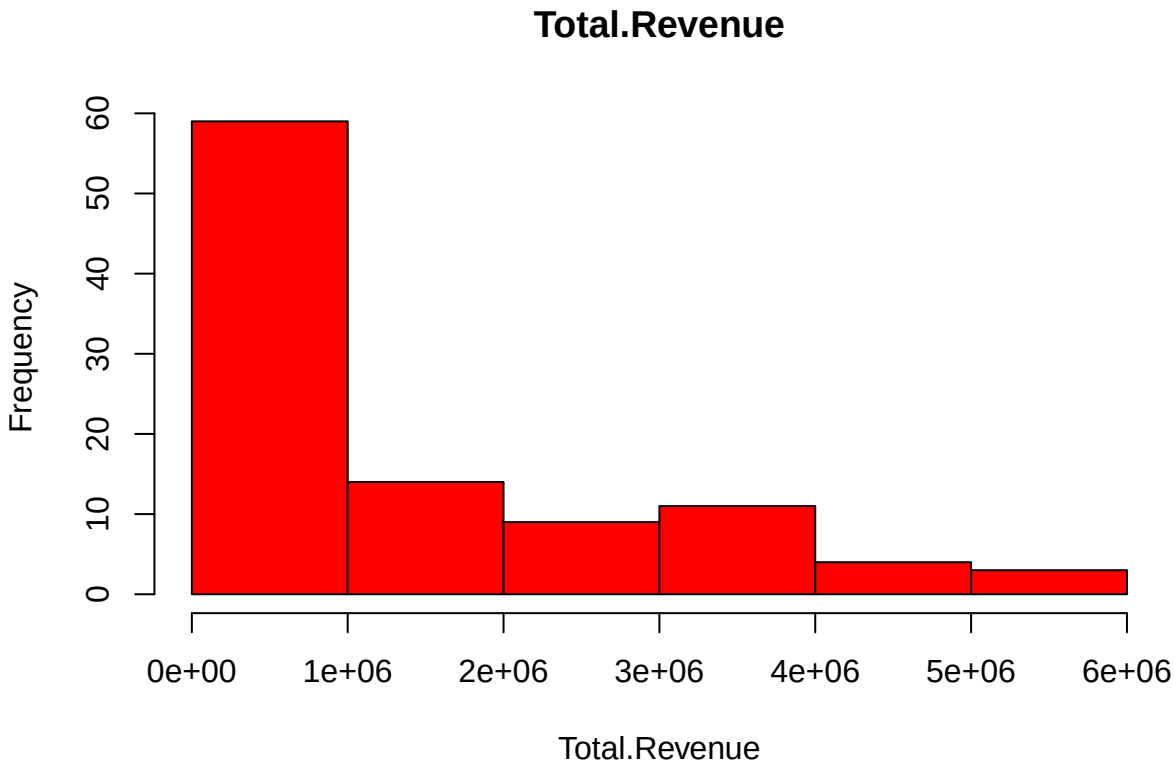


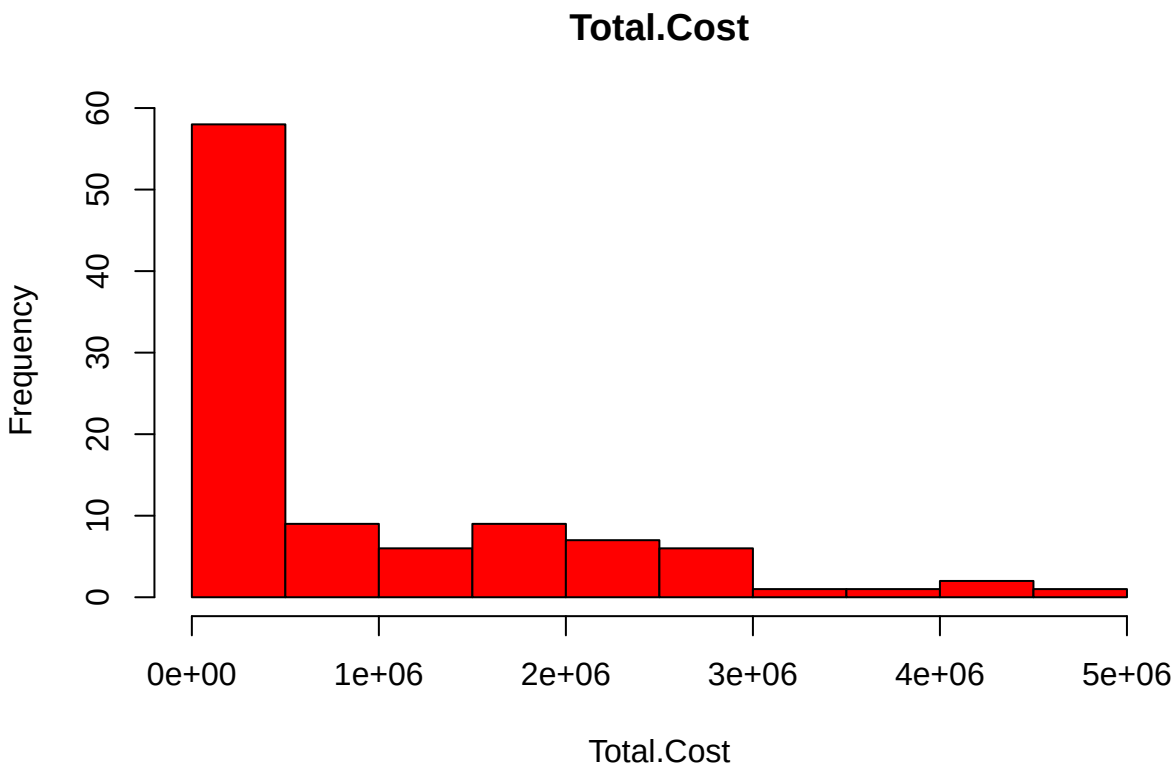
Histograms to visualize the distribution of numeric variables:

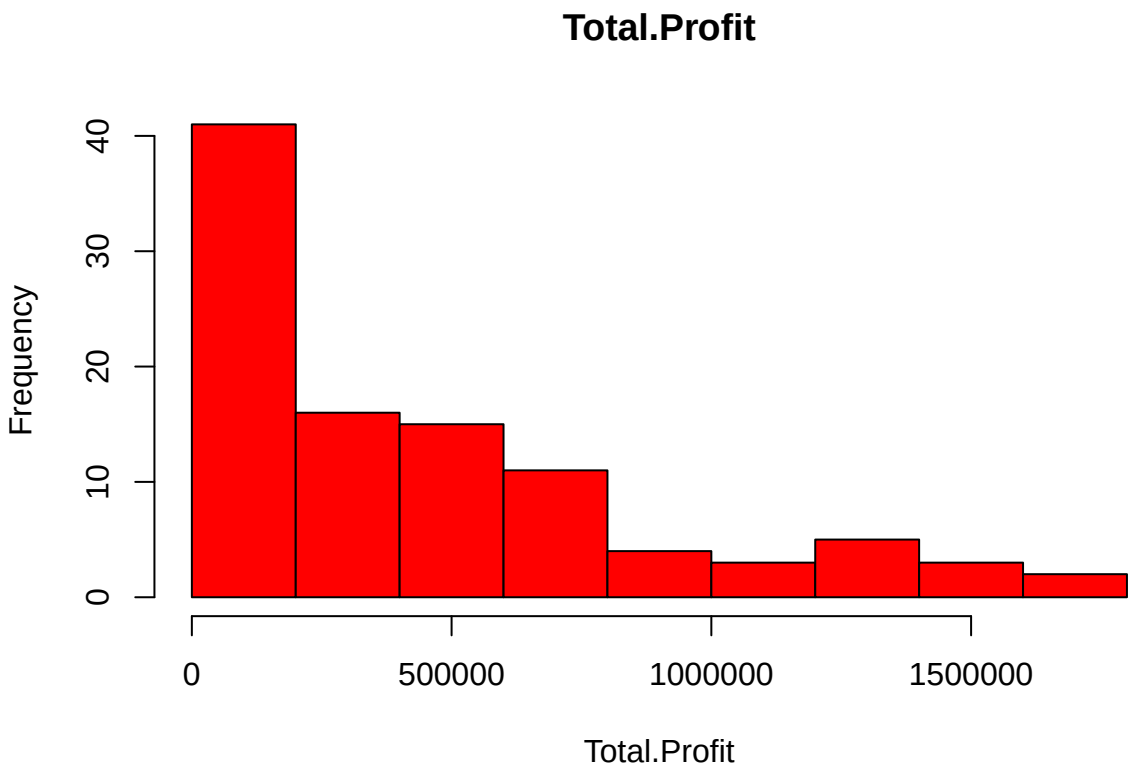



Unit.Price
Frequency
0
5
10
15
20
25
0
100
200
300
400
500
600
700
Unit.Price


Unit.Cost
Frequency
0
5
10
15
20
25
30
0
100
200
300
400
500
Unit.Cost

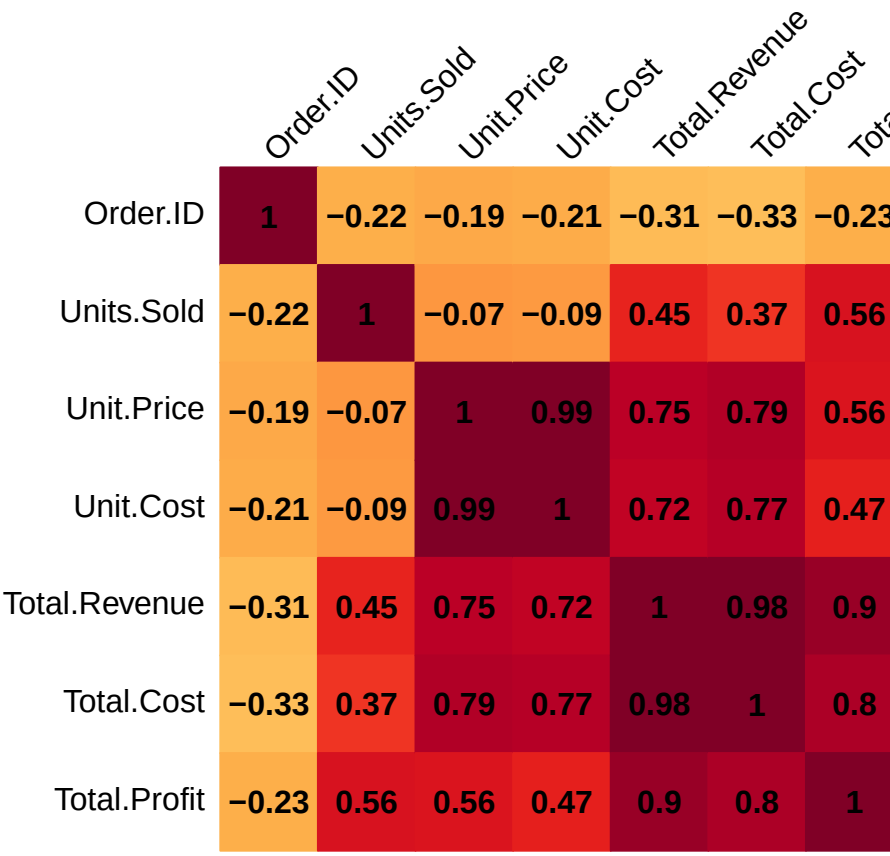

Total.Revenue
Frequency
0
10
20
30
40
50
60
0e+00
1e+06
2e+06
3e+06
4e+06
5e+06
6e+06
Total.Revenue


Total.Cost
Frequency
0
10
20
30
40
50
60
0e+00
1e+06
2e+06
3e+06
4e+06
5e+06
Total.Cost



For both "Order ID" and "Units Sold," the histograms show a uniform distribution. This suggests that values are spread relatively evenly across the range, and there's no clear tendency for values to cluster around specific points. The histogram for "Unit Price" is described as right-skewed. Right-skewed distributions have a tail extending to the right, indicating that there are fewer higher values but with a few extremely high values (outliers) pulling the distribution to the right. The histograms for "Total Revenue," "Total Cost," and "Total Profit" are all described as right-skewed. Similar to the Unit Price, this indicates that these variables have a distribution with a tail extending to the right, implying the presence of relatively fewer higher values.

```r
# Generate a complete correlation plot using a light color scheme
corrplot(cor_matrix, method = "color",
         addCoef.col = "black",     # Display correlation coefficients in black
         tl.col = "black",          # Set the color of variable labels to black
         tl.srt = 45,               # Rotate the labels 45 degrees for readability
         col = colorRampPalette(brewer.pal(9, "YlOrRd"))(100))
```

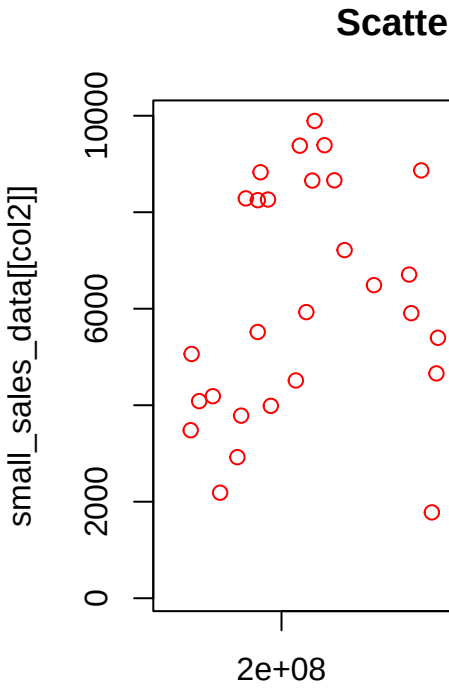


Correlation between variables:

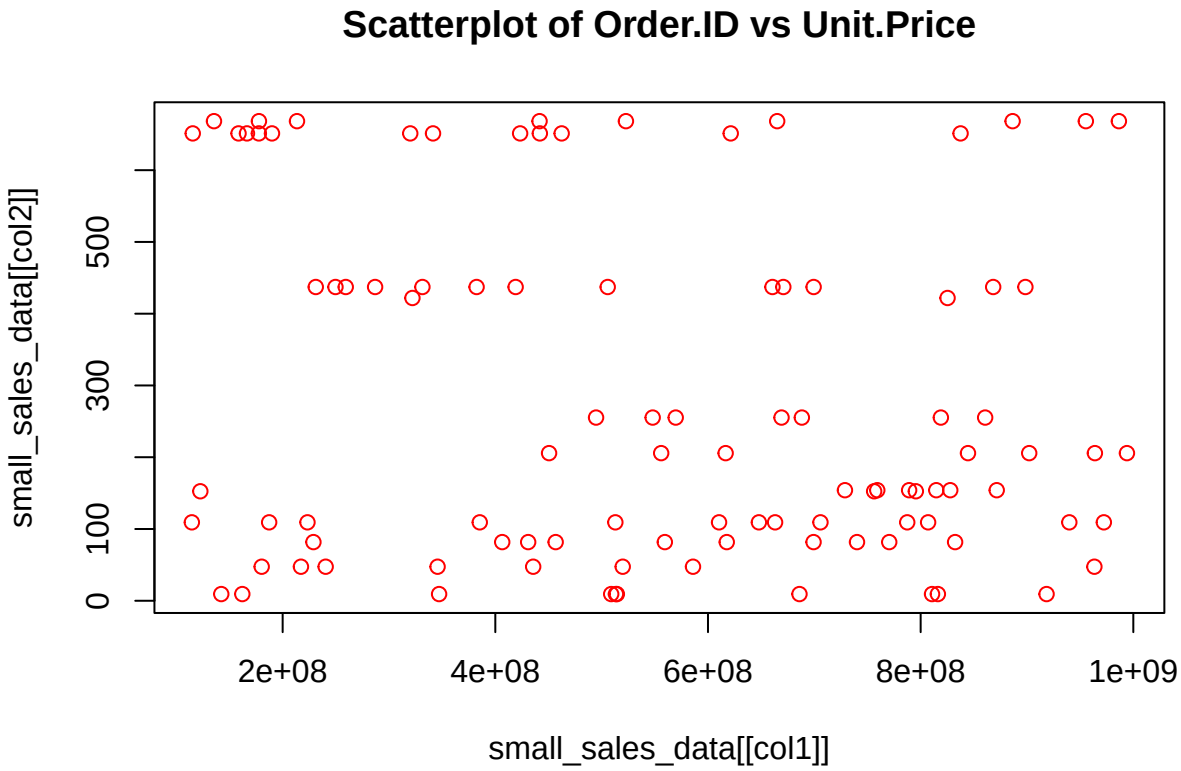
The correlation coefficients in this dataset reveal important relationships between key variables. For instance, the correlation between **Unit Cost** and **Unit Price** is exceptionally strong at 0.99. This suggests that as the cost of producing a unit rises, its selling price increases proportionally. This close relationship highlights a direct link between production costs and pricing strategies. Similarly, the **Total Cost** and **Total Revenue** show a robust positive correlation of 0.98. This indicates that higher production costs are typically associated with higher revenue, emphasizing the interconnected nature of these financial metrics in business operations. Additionally, **Total Profit** and **Total Revenue** have a strong positive correlation of 0.90, though slightly less pronounced than the others. This relationship suggests that as profits rise, revenue tends to increase as well, highlighting a meaningful connection between profitability and overall sales performance.

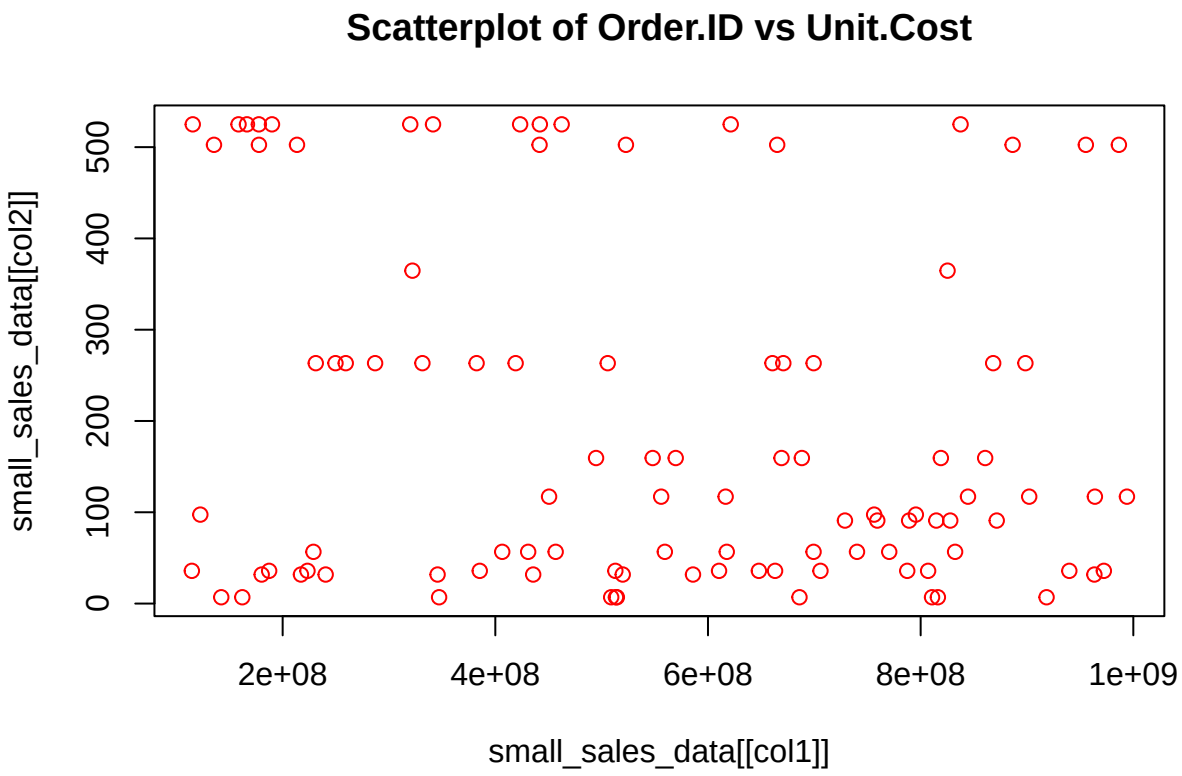
```r
# Assuming your dataset is 'small_sales_data'
# Select numeric columns from the dataset
numeric_data_cols <- sapply(small_sales_data, is.numeric)

# Generate scatterplots for each pair of numeric columns
for (col1 in names(small_sales_data)[numeric_data_cols]) {
  for (col2 in names(small_sales_data)[numeric_data_cols]) {
    if (col1 != col2) {
      # Create scatterplot for each unique pair of numeric columns
      plot(small_sales_data[[col1]], small_sales_data[[col2]],
           main = paste("Scatterplot of", col1, "vs", col2),
           col = "Red")
    }
```

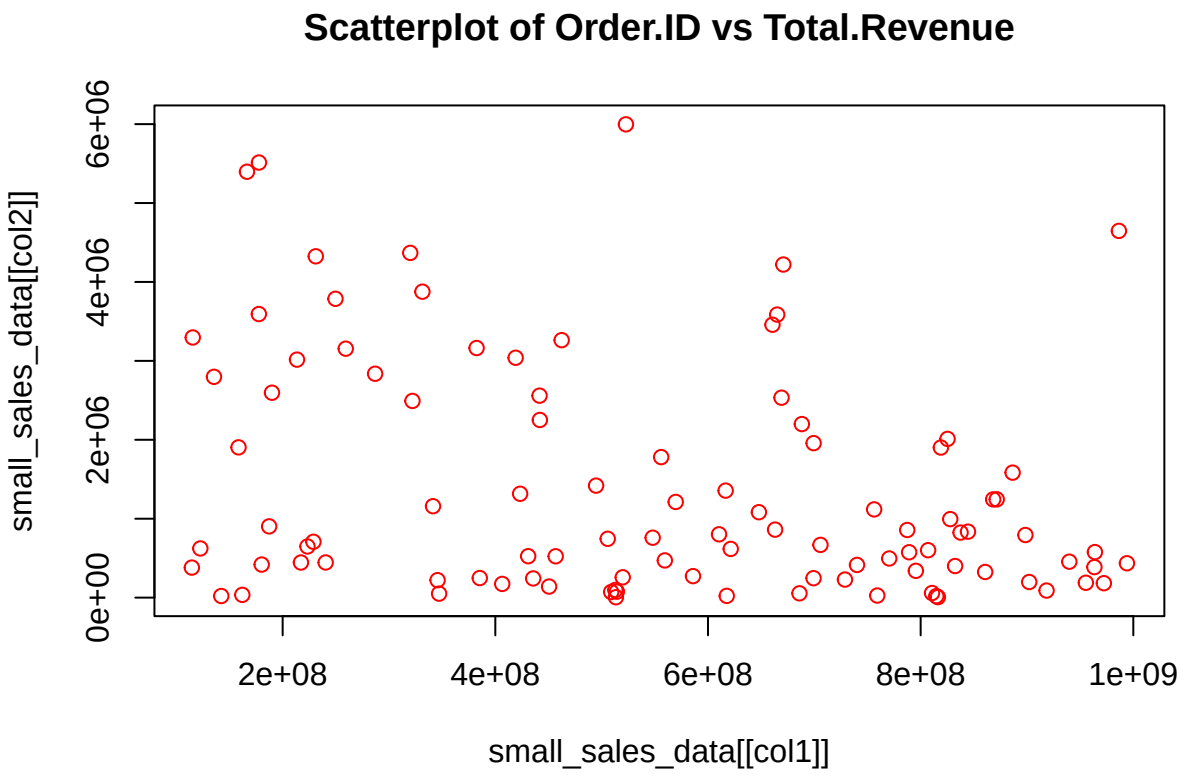
```
  }
}
```

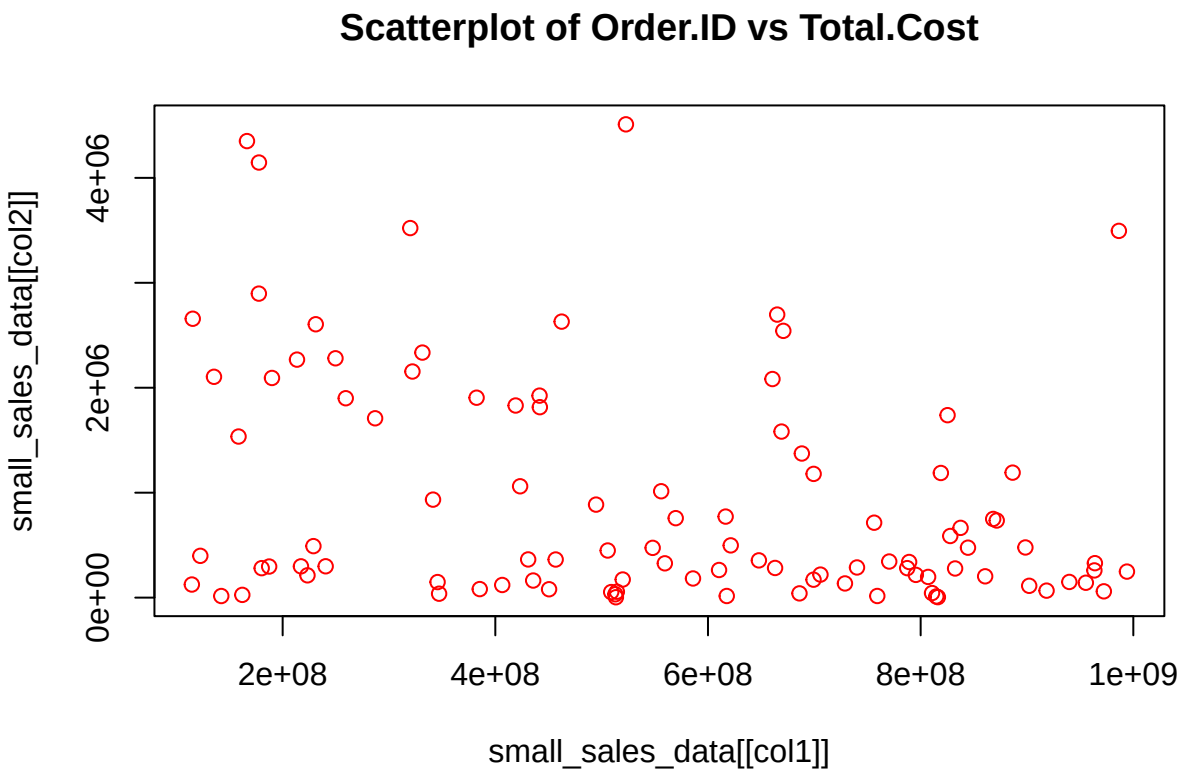


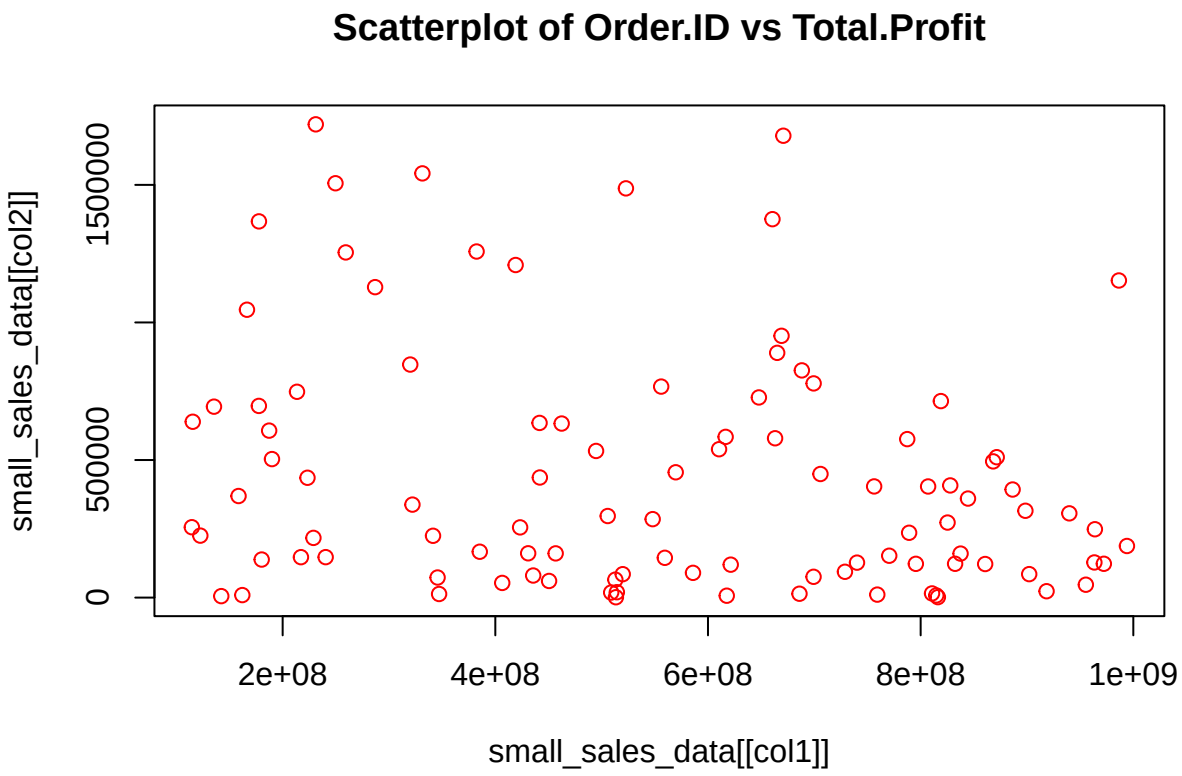
Individual Scatter Plots to explore relationships for each pair of variables:

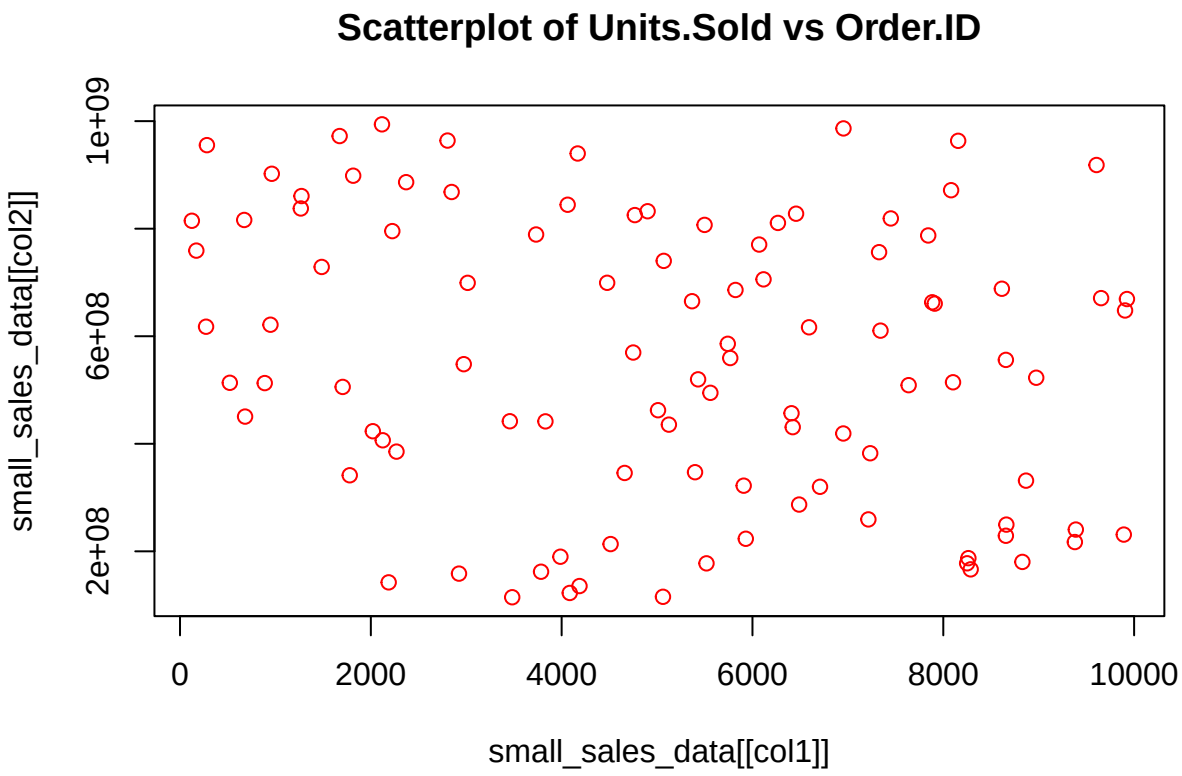


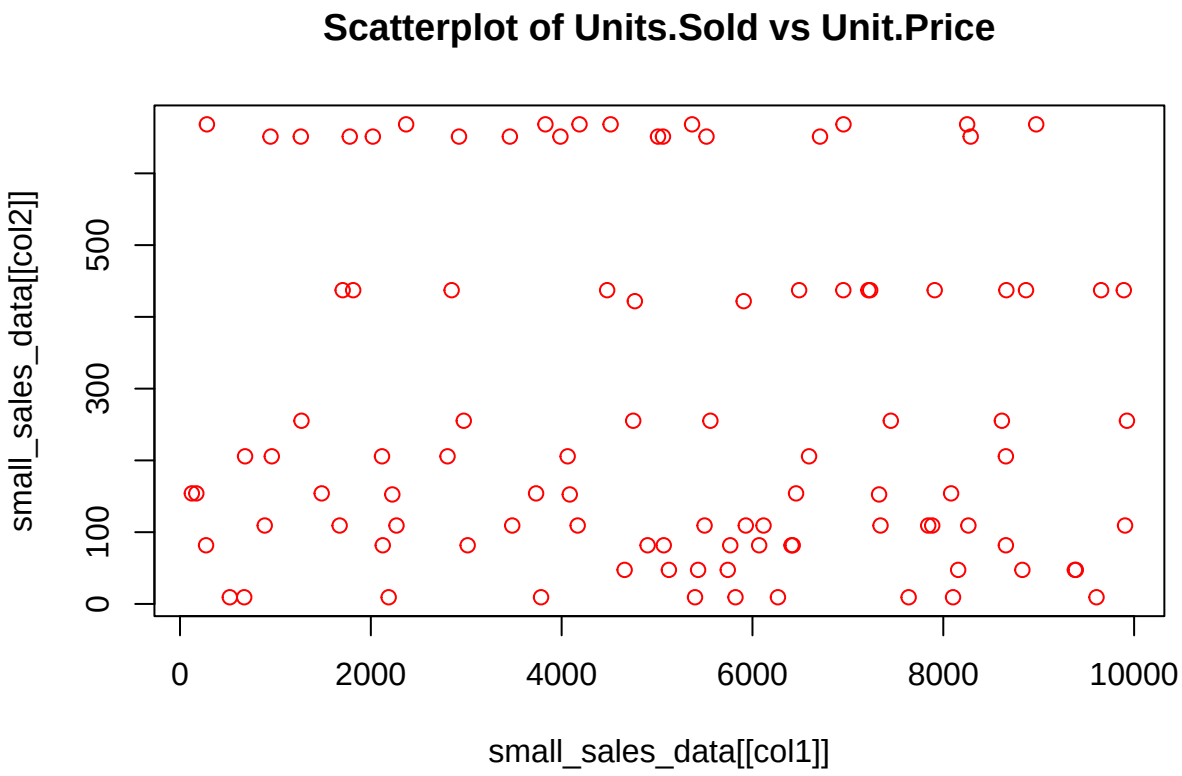

Scatterplot of Order.ID vs Unit.Cost
small_sales_data[[col2]]
0
100
200
300
400
500
2e+08
4e+08
6e+08
8e+08
1e+09
small_sales_data[[col1]]

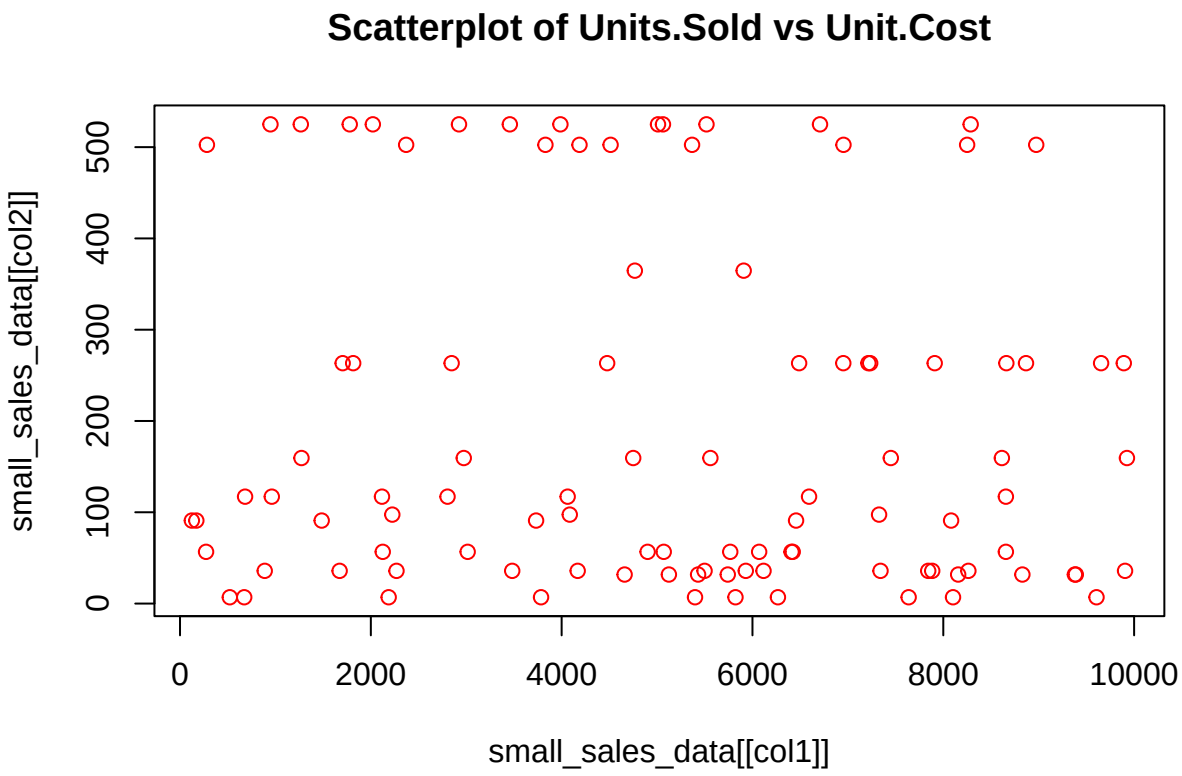

Scatterplot of Order.ID vs Total.Revenue
small_sales_data[[col2]]
0e+00
2e+06
4e+06
6e+06
2e+08
4e+08
6e+08
8e+08
1e+09
small_sales_data[[col1]]

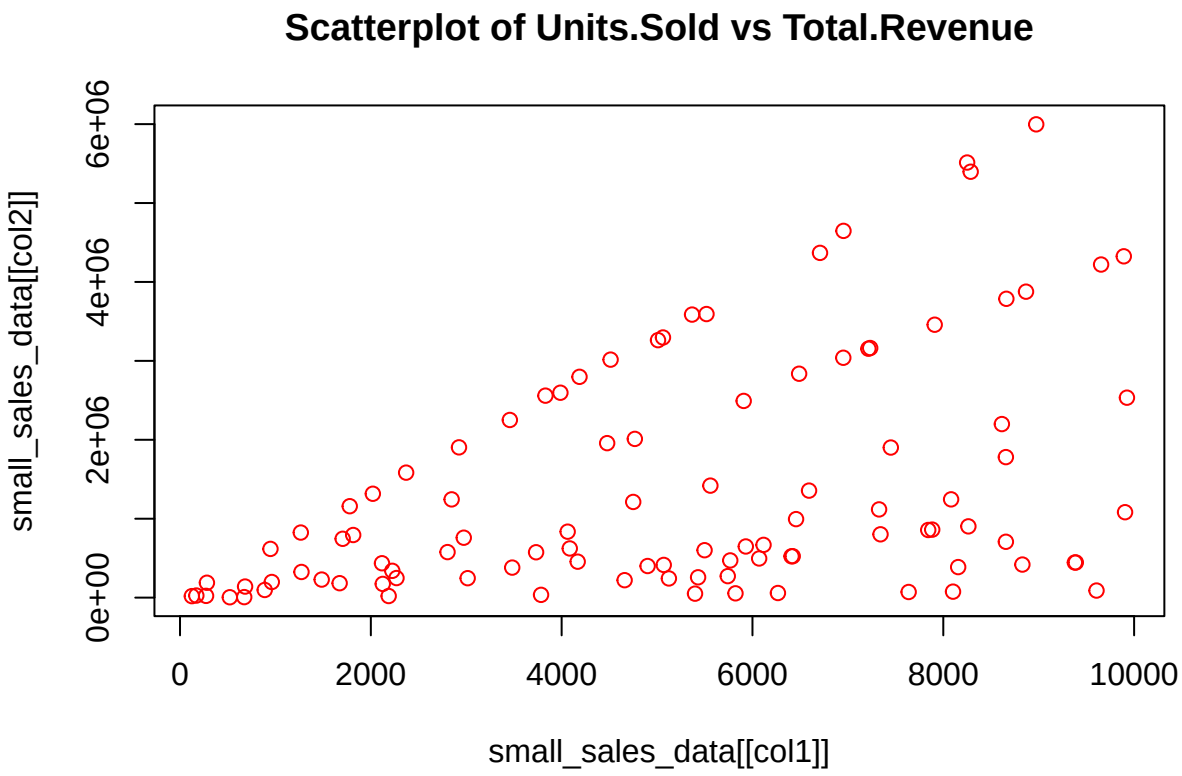

Scatterplot of Order.ID vs Total.Cost
small_sales_data[[col2]]
0e+00
2e+06
4e+06
2e+08
4e+08
6e+08
8e+08
1e+09
small_sales_data[[col1]]

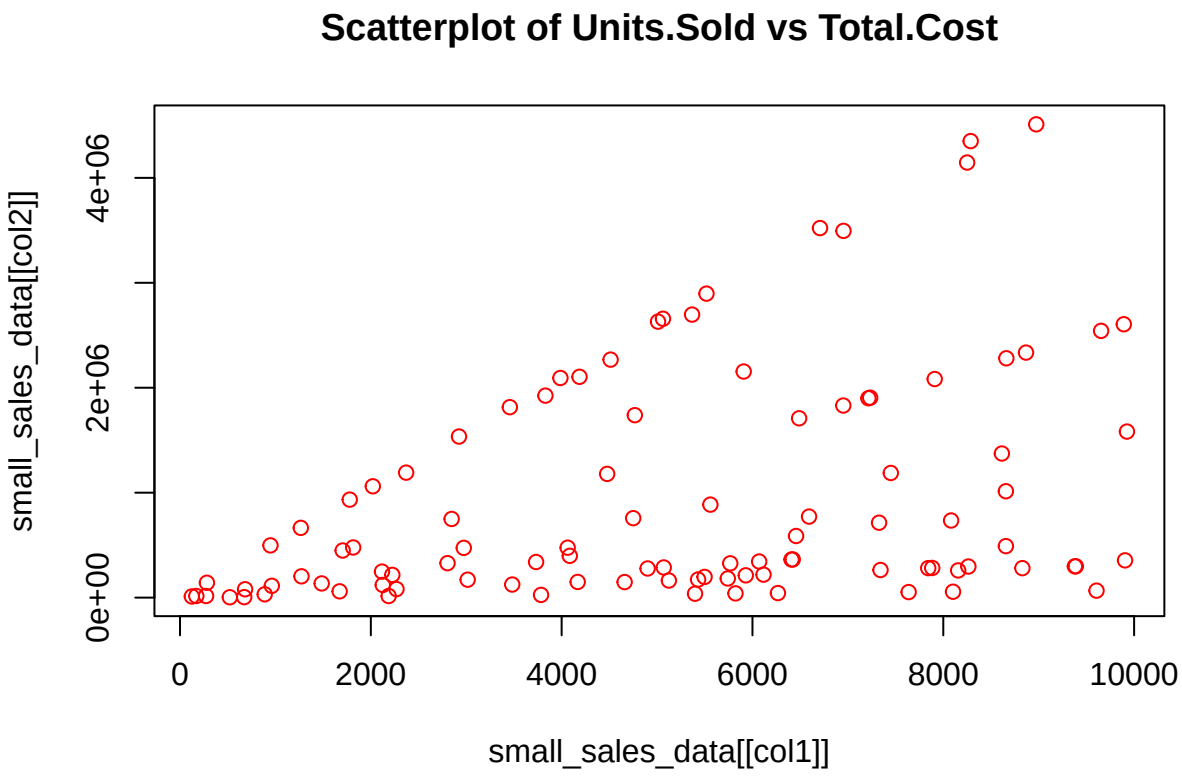

Scatterplot of Order.ID vs Total.Profit
small_sales_data[[col2]]
0
500000
1500000
2e+08
4e+08
6e+08
8e+08
1e+09
small_sales_data[[col1]]

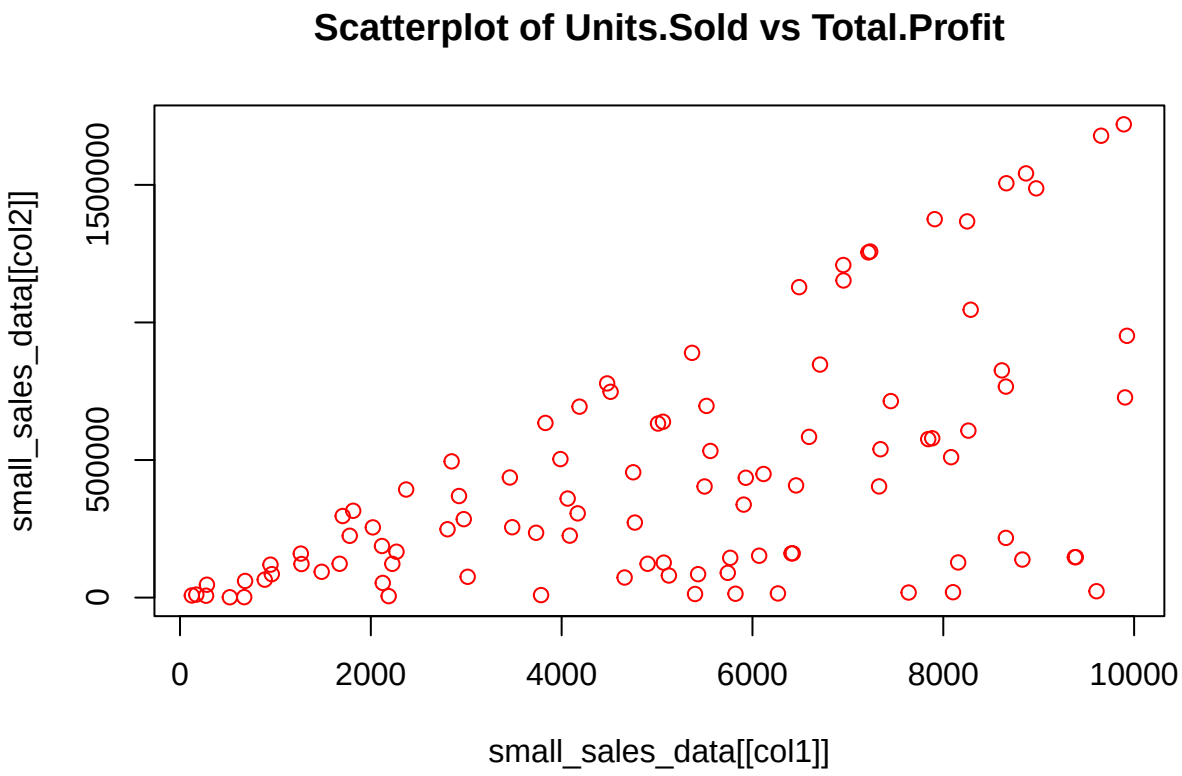

Scatterplot of Units.Sold vs Order.ID
small_sales_data[[col2]]
1e+09
6e+08
2e+08
0
2000
4000
6000
8000
10000
small_sales_data[[col1]]

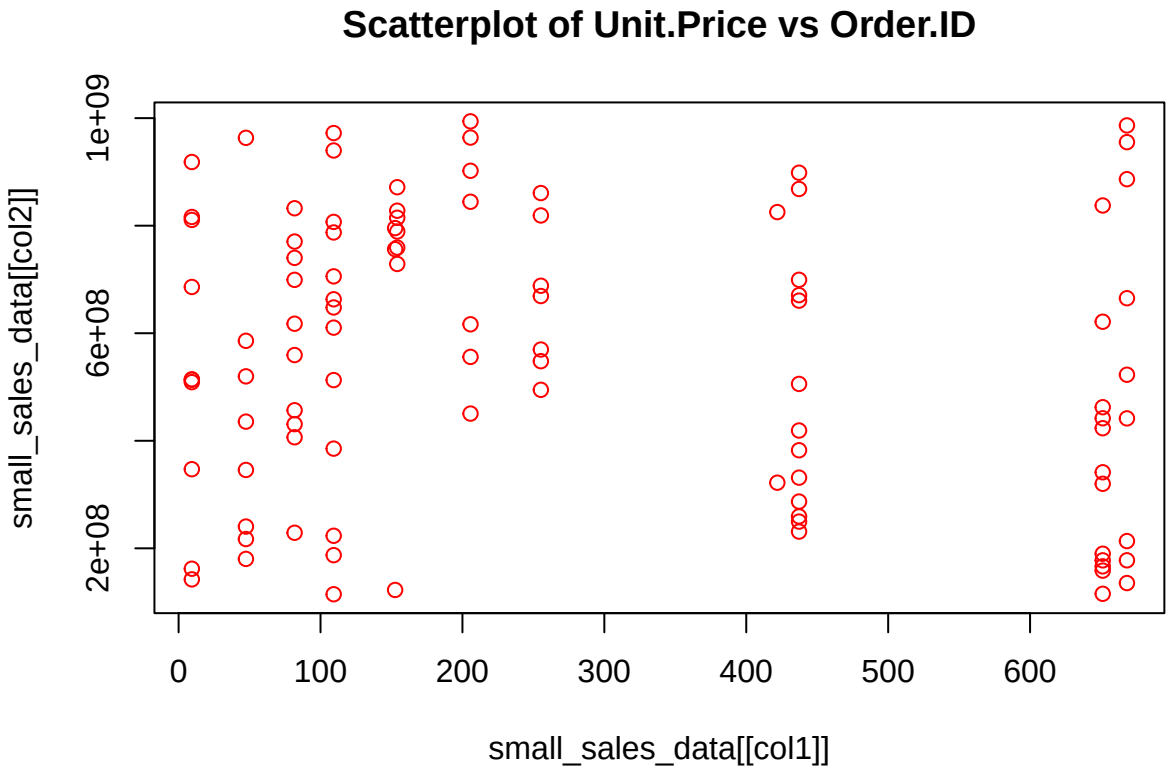

Scatterplot of Units.Sold vs Unit.Price
small_sales_data[[col2]]
0
100
300
500
0
2000
4000
6000
8000
10000
small_sales_data[[col1]]

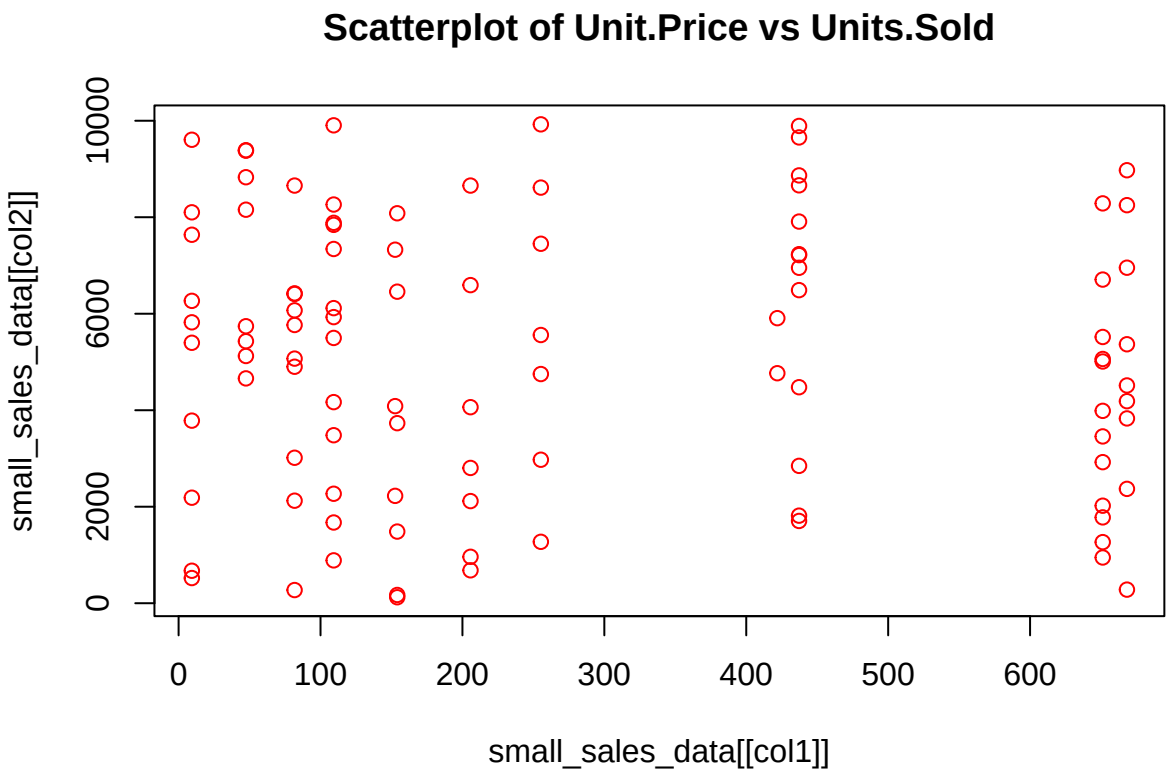

Scatterplot of Units.Sold vs Unit.Cost
small_sales_data[[col2]]
0
100
200
300
400
500
0
2000
4000
6000
8000
10000
small_sales_data[[col1]]

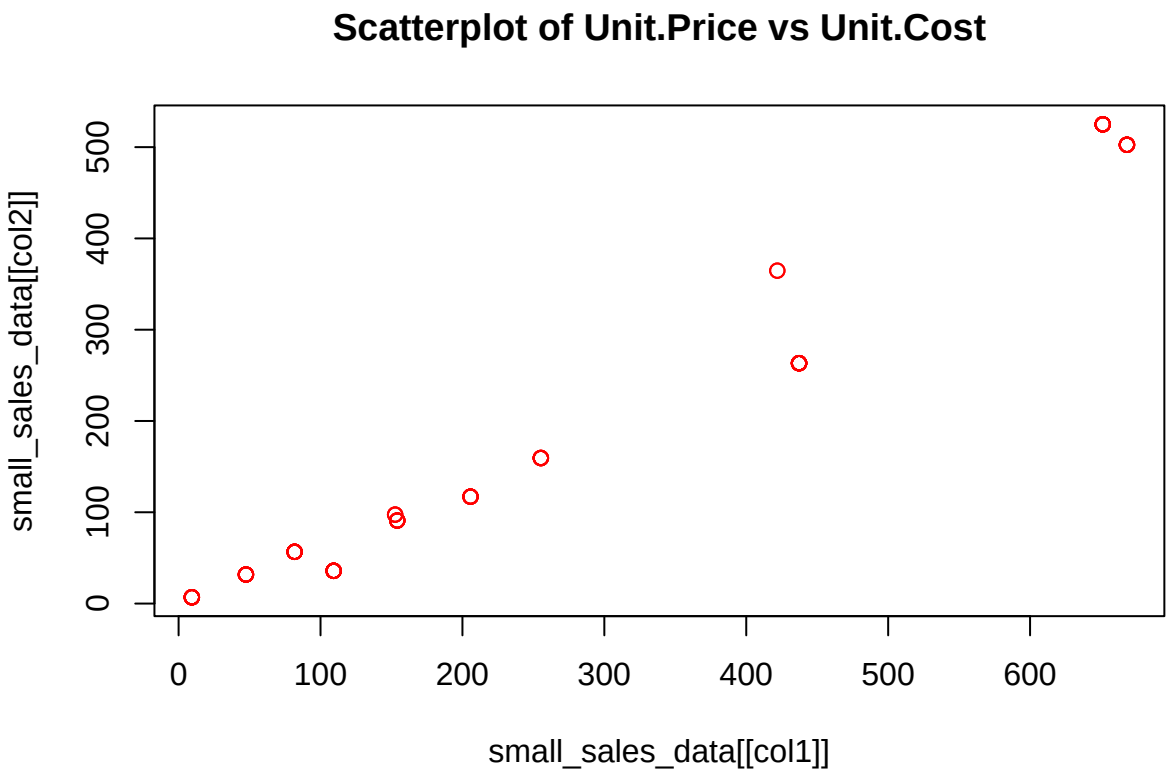

Scatterplot of Units.Sold vs Total.Revenue
small_sales_data[[col2]]
0e+00
2e+06
4e+06
6e+06
0
2000
4000
6000
8000
10000
small_sales_data[[col1]]

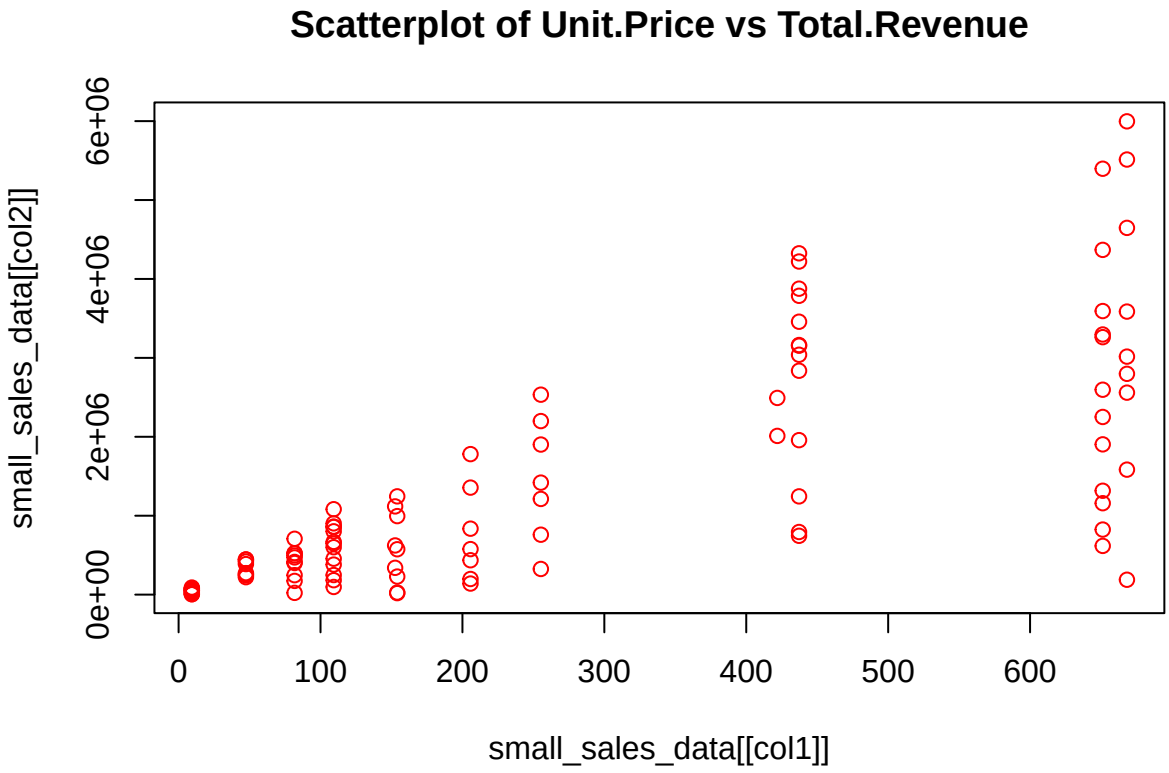

Scatterplot of Units.Sold vs Total.Cost
small_sales_data[[col2]]
0e+00
2e+06
4e+06
0
2000
4000
6000
8000
10000
small_sales_data[[col1]]

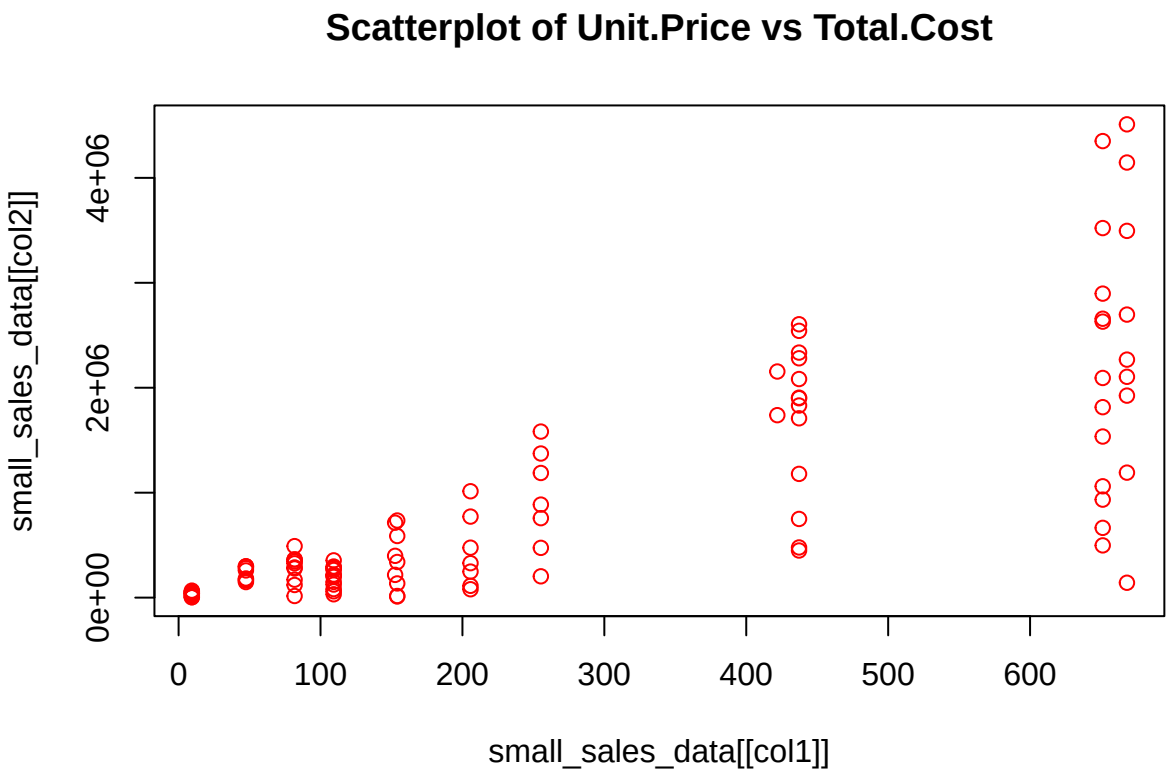

Scatterplot of Units.Sold vs Total.Profit
small_sales_data[[col2]]
0
500000
1500000
0
2000
4000
6000
8000
10000
small_sales_data[[col1]]

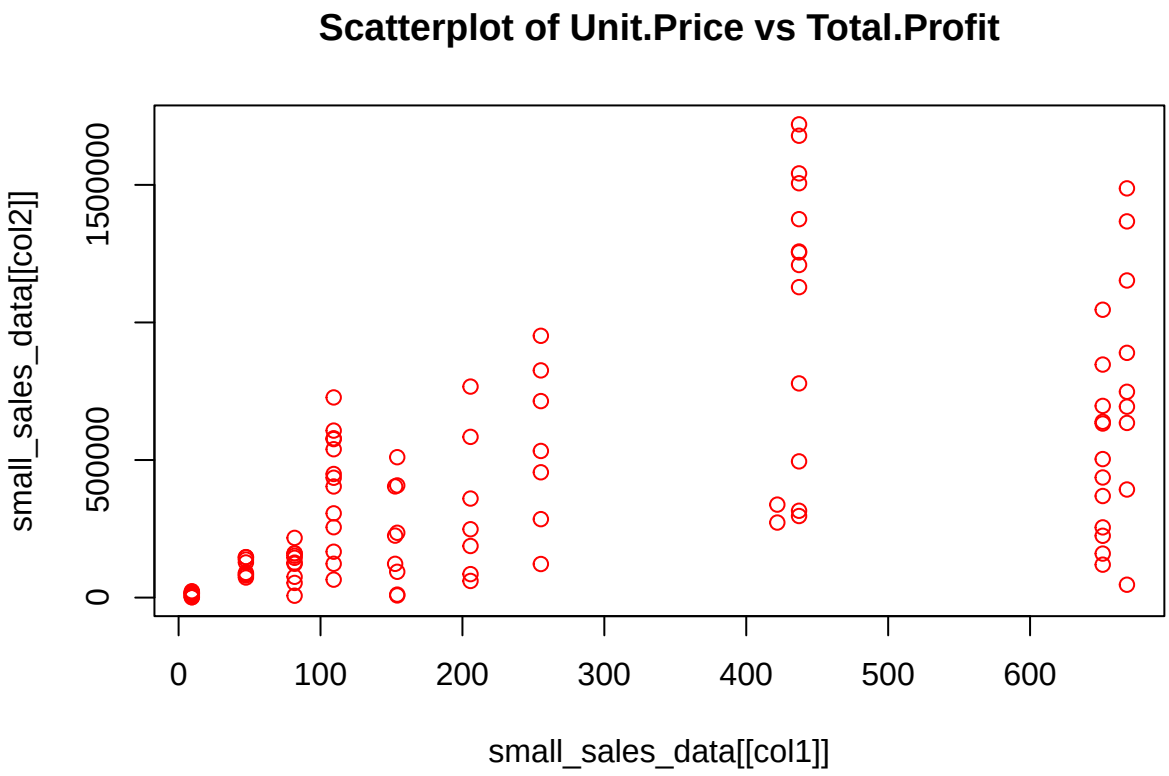

Scatterplot of Unit.Price vs Order.ID
small_sales_data[[col2]]
1e+09
6e+08
2e+08
0
100
200
300
400
500
600
small_sales_data[[col1]]

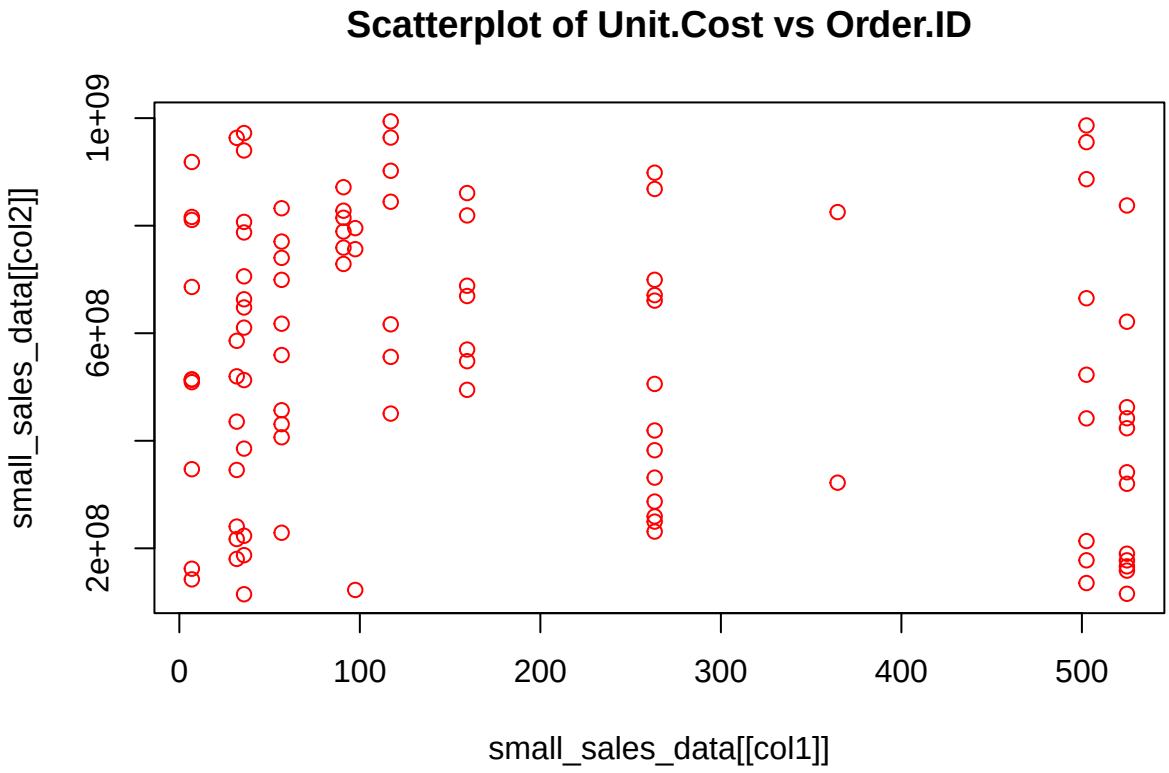

Scatterplot of Unit.Price vs Units.Sold
small_sales_data[[col2]]
0
2000
6000
10000
0
100
200
300
400
500
600
small_sales_data[[col1]]

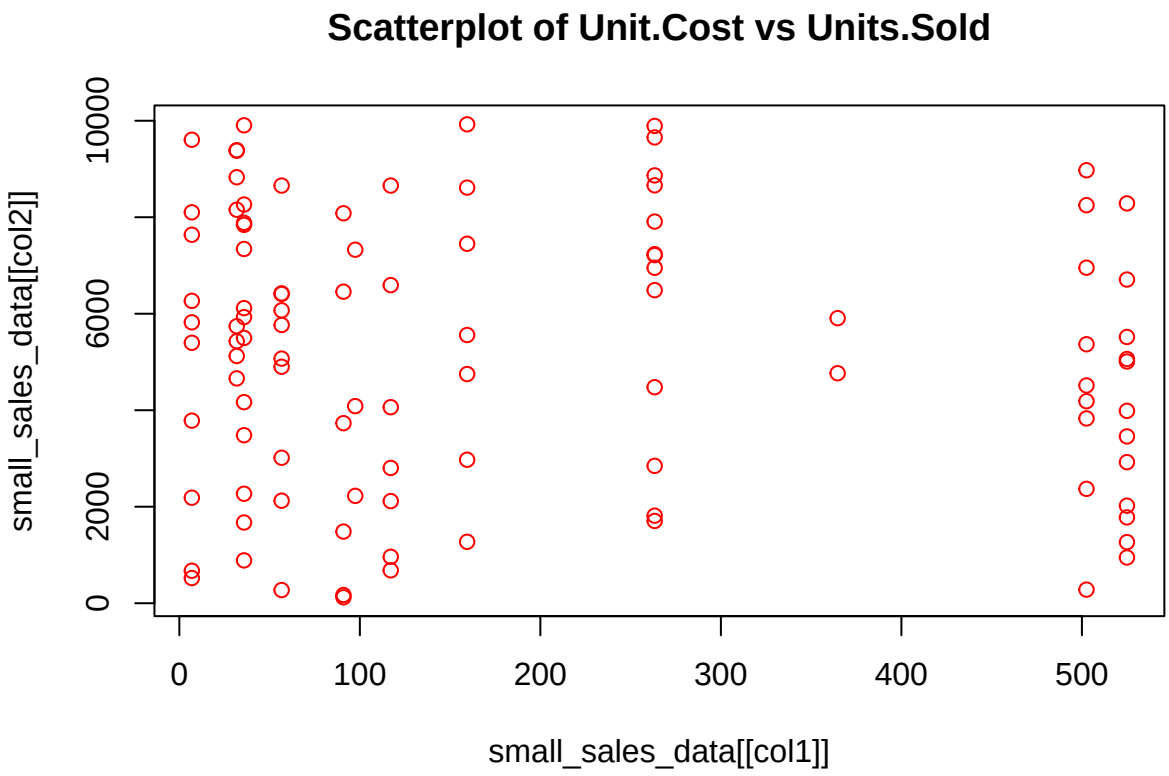

Scatterplot of Unit.Price vs Unit.Cost
small_sales_data[[col2]]
0
100
200
300
400
500
small_sales_data[[col1]]
0
100
200
300
400
500
600

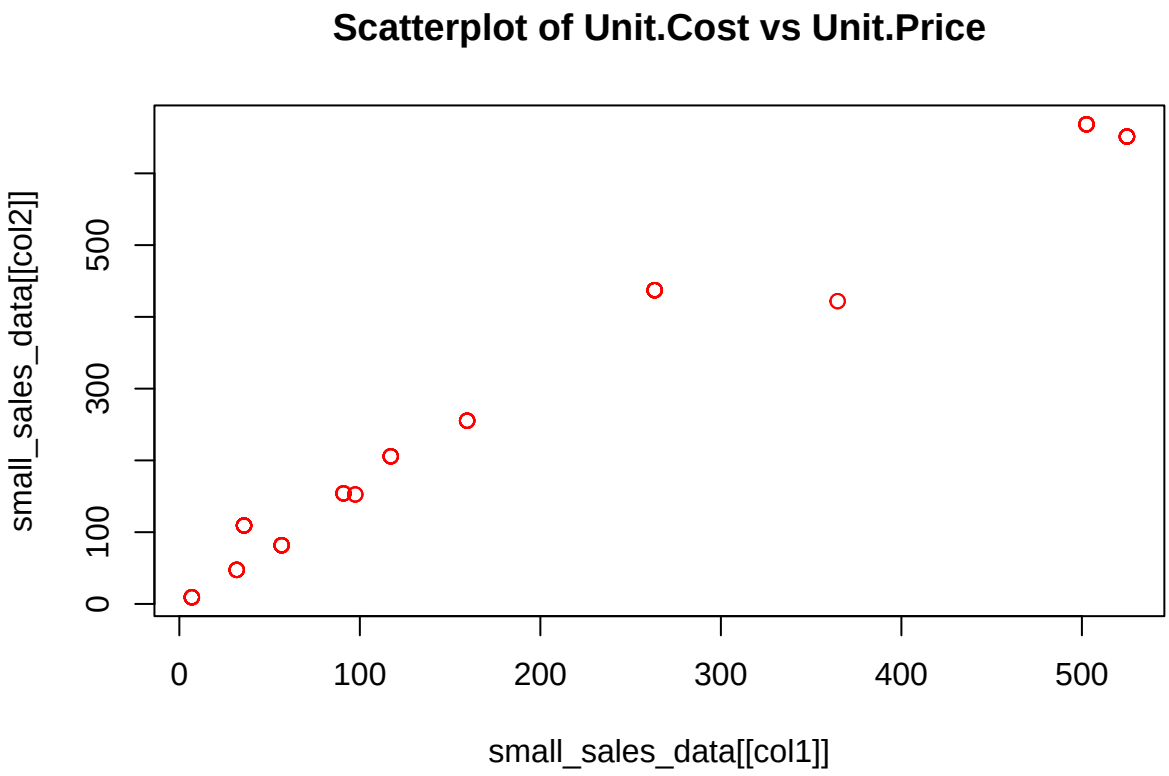

Scatterplot of Unit.Price vs Total.Revenue
small_sales_data[[col2]]
0e+00
2e+06
4e+06
6e+06
0
100
200
300
400
500
600
small_sales_data[[col1]]

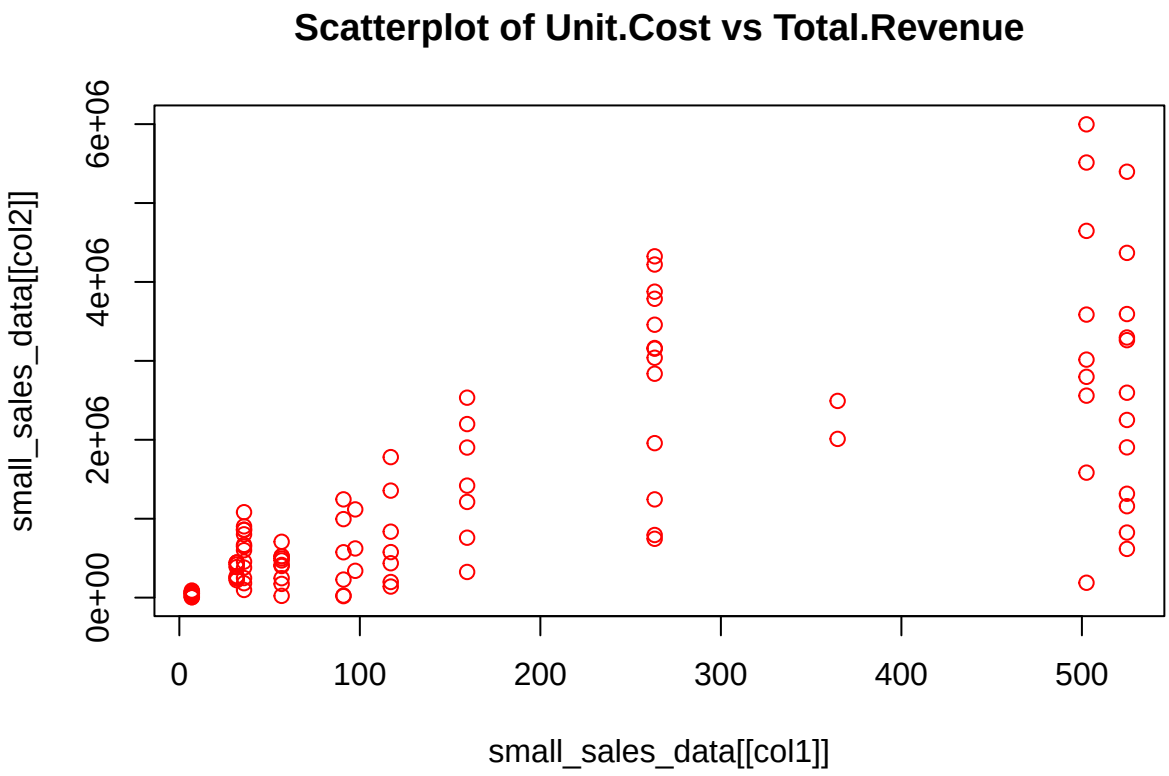

Scatterplot of Unit.Price vs Total.Cost
small_sales_data[[col2]]
0e+00
2e+06
4e+06
0
100
200
300
400
500
600
small_sales_data[[col1]]

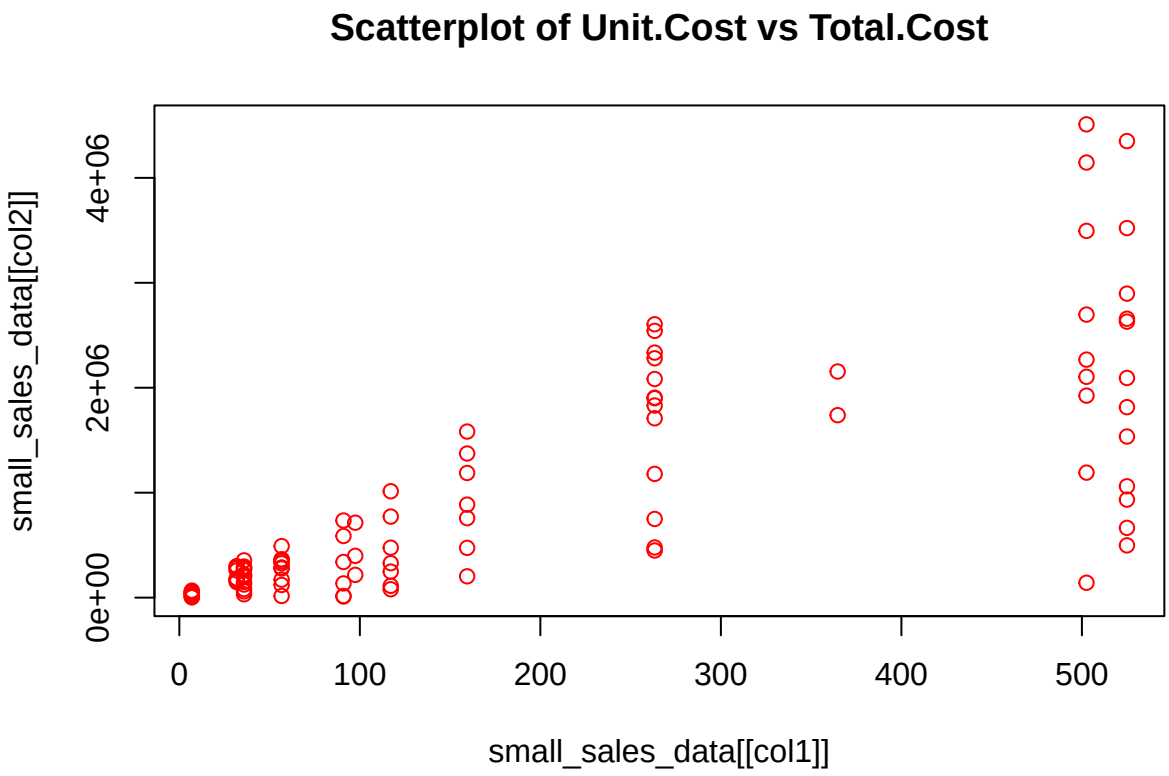

Scatterplot of Unit.Price vs Total.Profit
small_sales_data[[col2]]
0
500000
1500000
0
100
200
300
400
500
600
small_sales_data[[col1]]

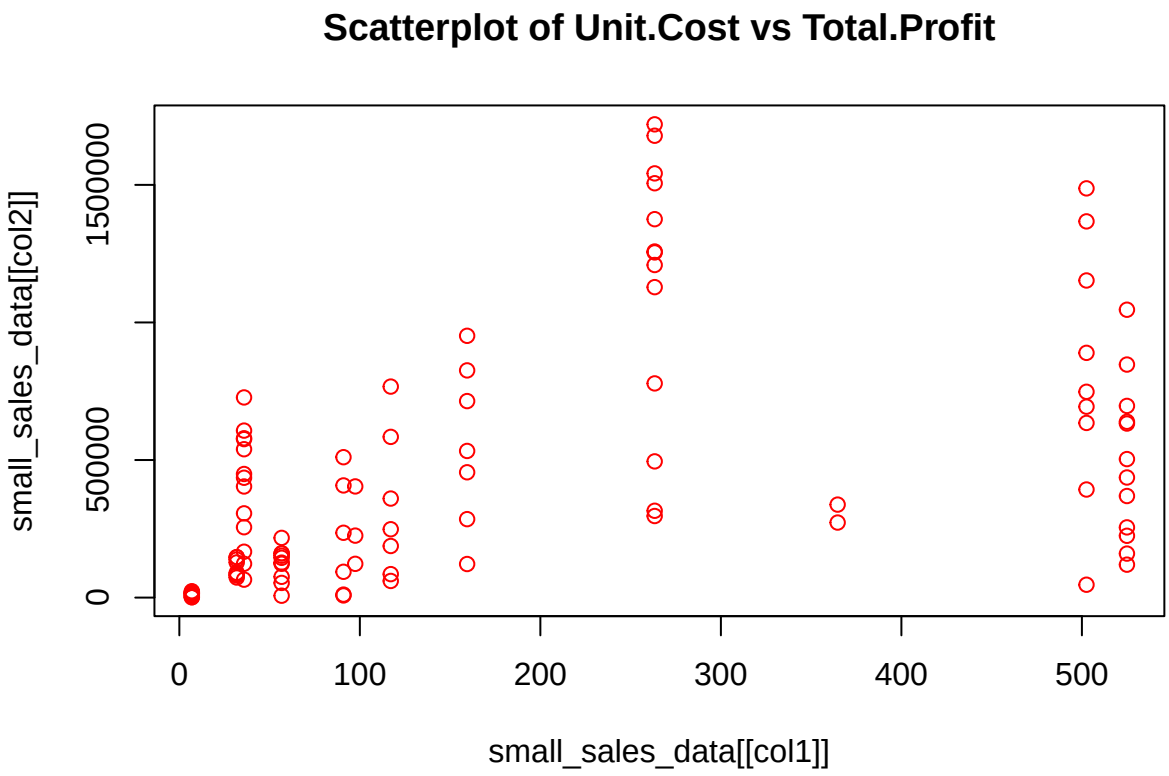

Scatterplot of Unit.Cost vs Order.ID
small_sales_data[[col2]]
1e+09
6e+08
2e+08
0
100
200
300
400
500
small_sales_data[[col1]]

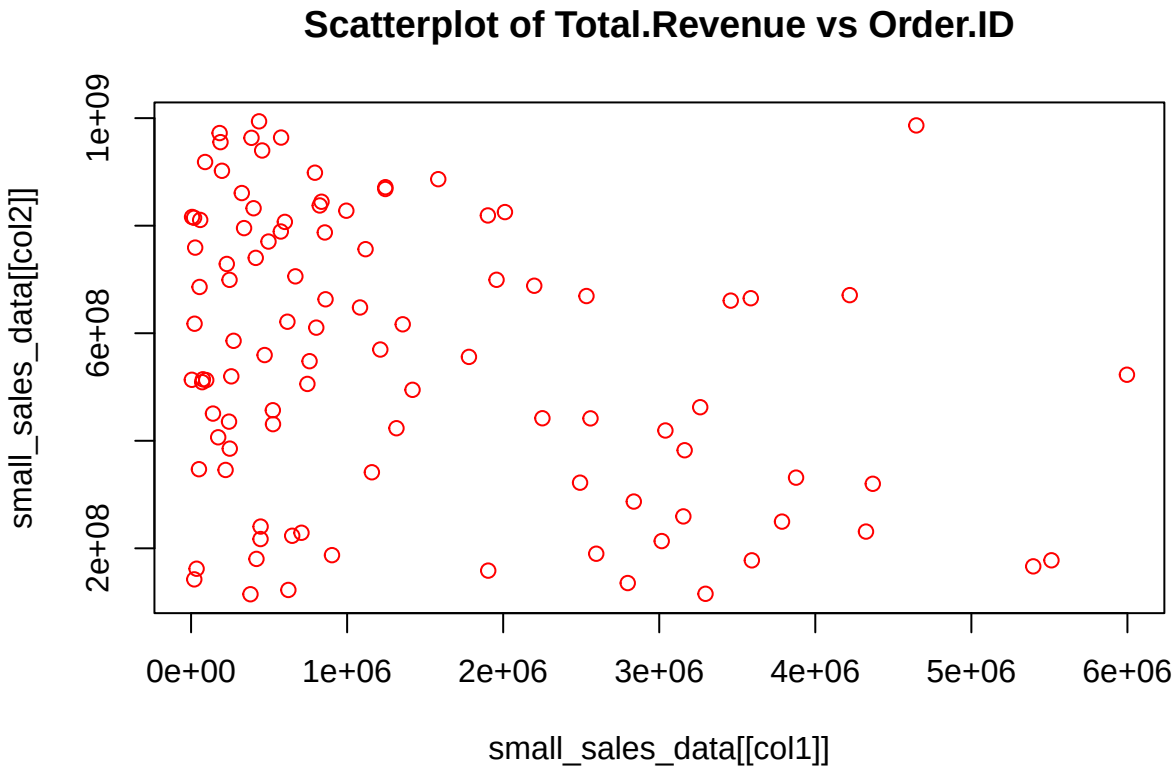

Scatterplot of Unit.Cost vs Units.Sold
small_sales_data[[col2]]
0
2000
6000
10000
0
100
200
300
400
500
small_sales_data[[col1]]

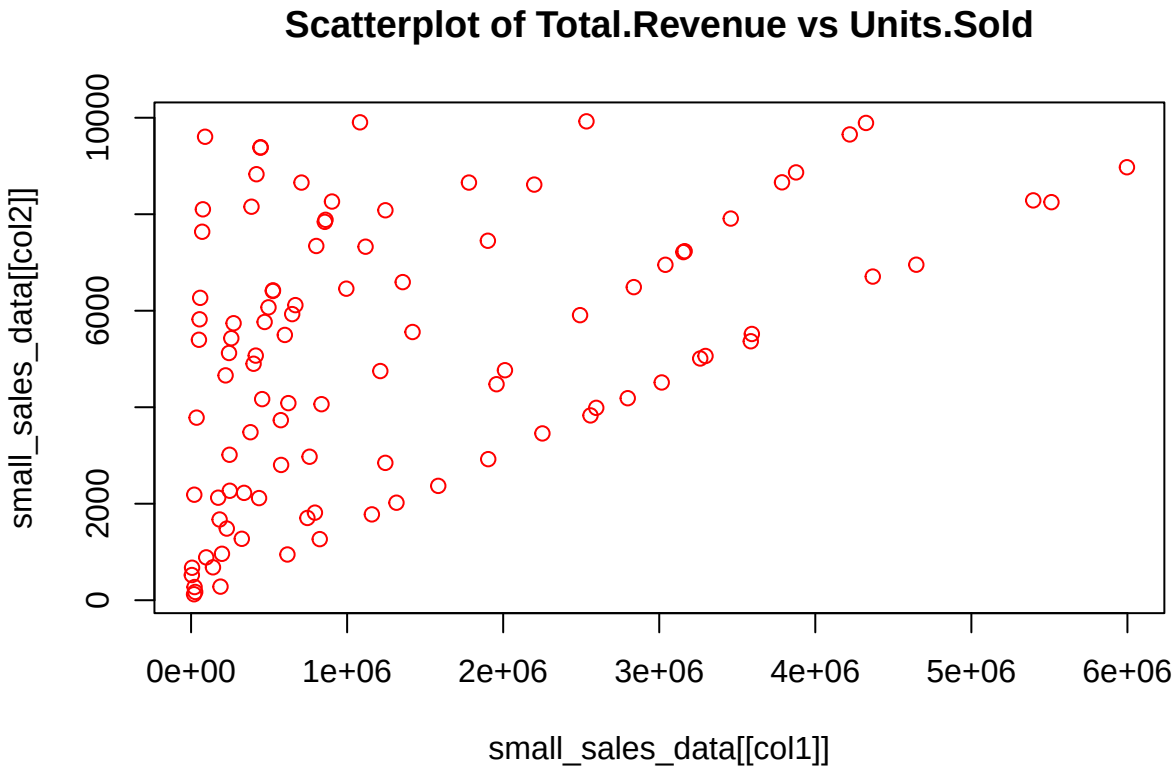

Scatterplot of Unit.Cost vs Unit.Price
small_sales_data[[col2]]
0
100
300
500
0
100
200
300
400
500
small_sales_data[[col1]]

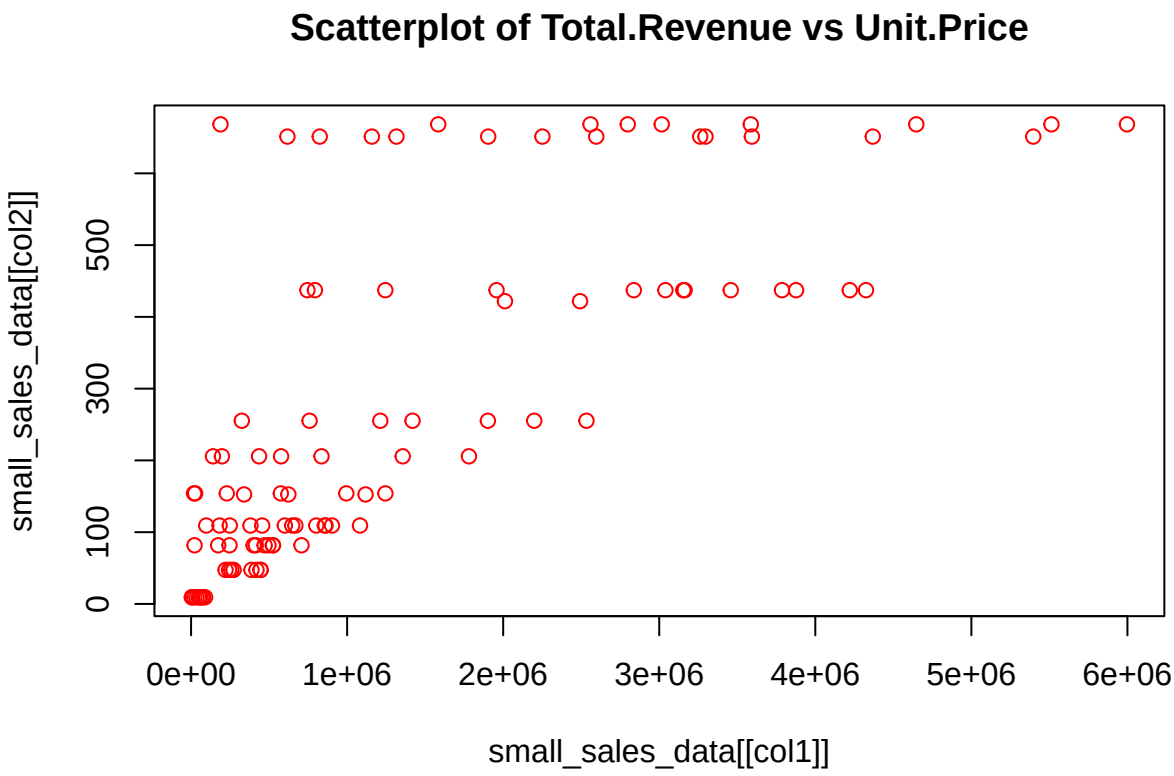

Scatterplot of Unit.Cost vs Total.Revenue
small_sales_data[[col2]]
0e+00
2e+06
4e+06
6e+06
0
100
200
300
400
500
small_sales_data[[col1]]

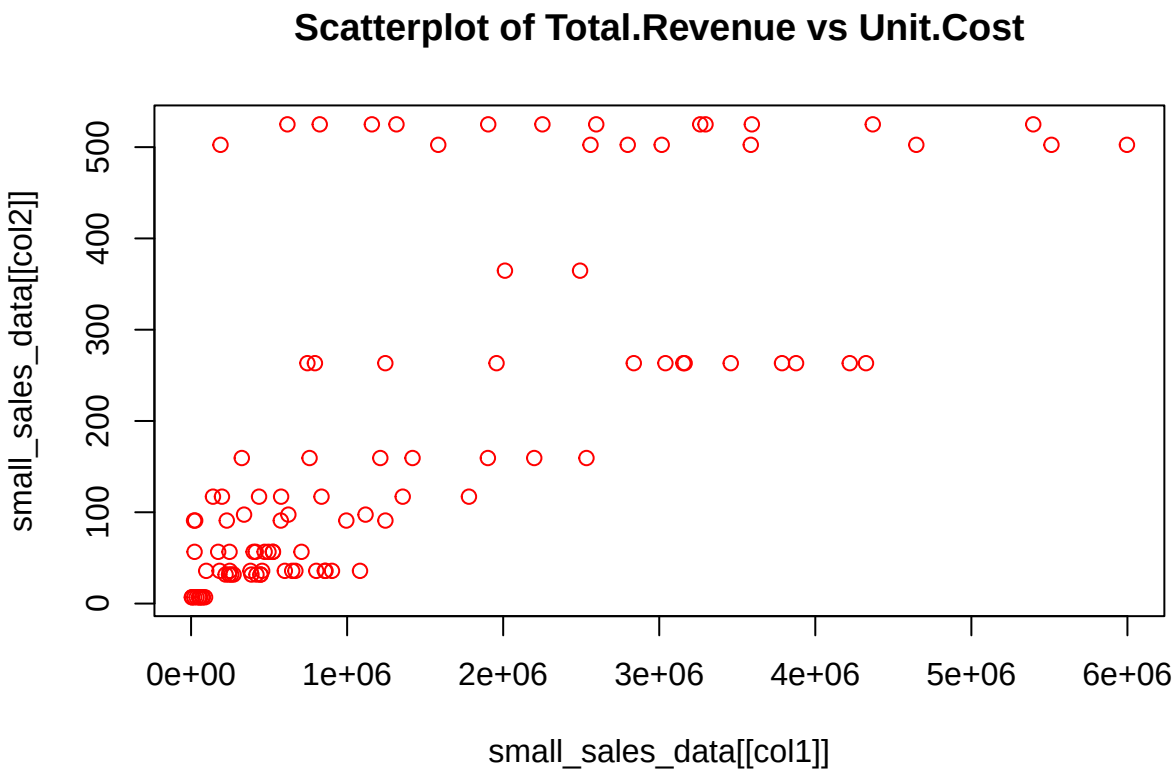

Scatterplot of Unit.Cost vs Total.Cost
small_sales_data[[col2]]
0e+00
2e+06
4e+06
0
100
200
300
400
500
small_sales_data[[col1]]

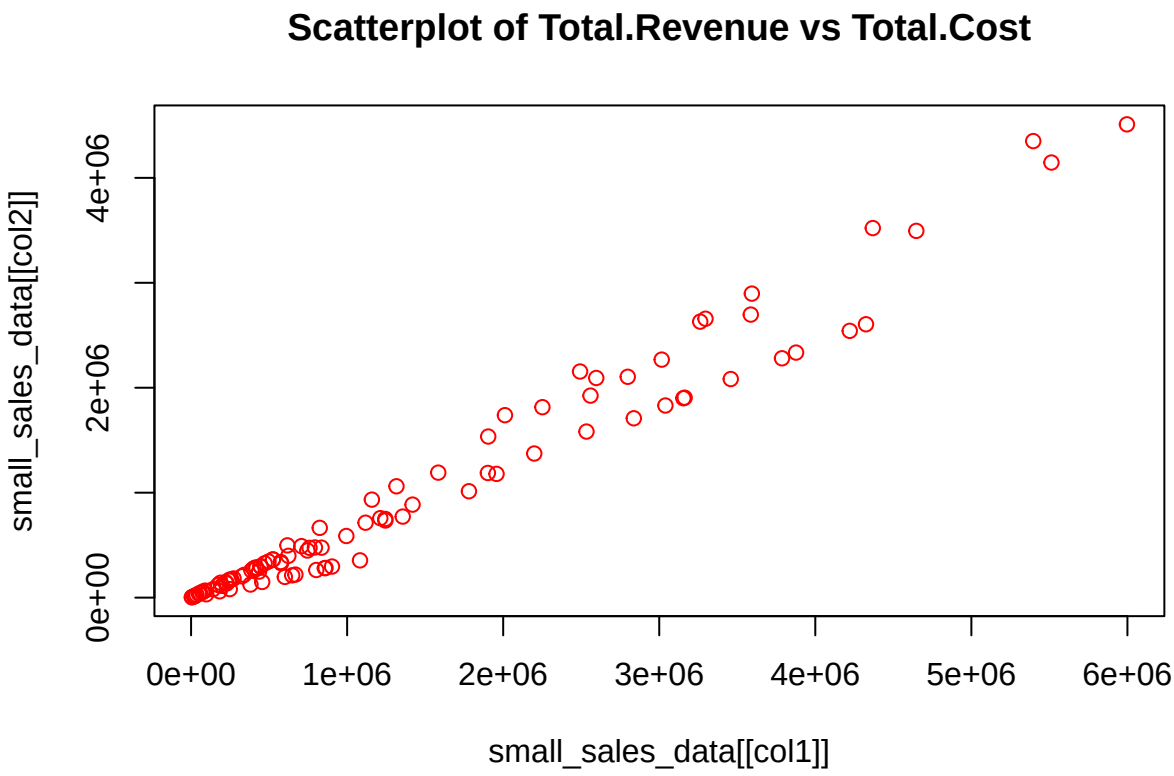

Scatterplot of Unit.Cost vs Total.Profit
small_sales_data[[col2]]
0
500000
1500000
0
100
200
300
400
500
small_sales_data[[col1]]

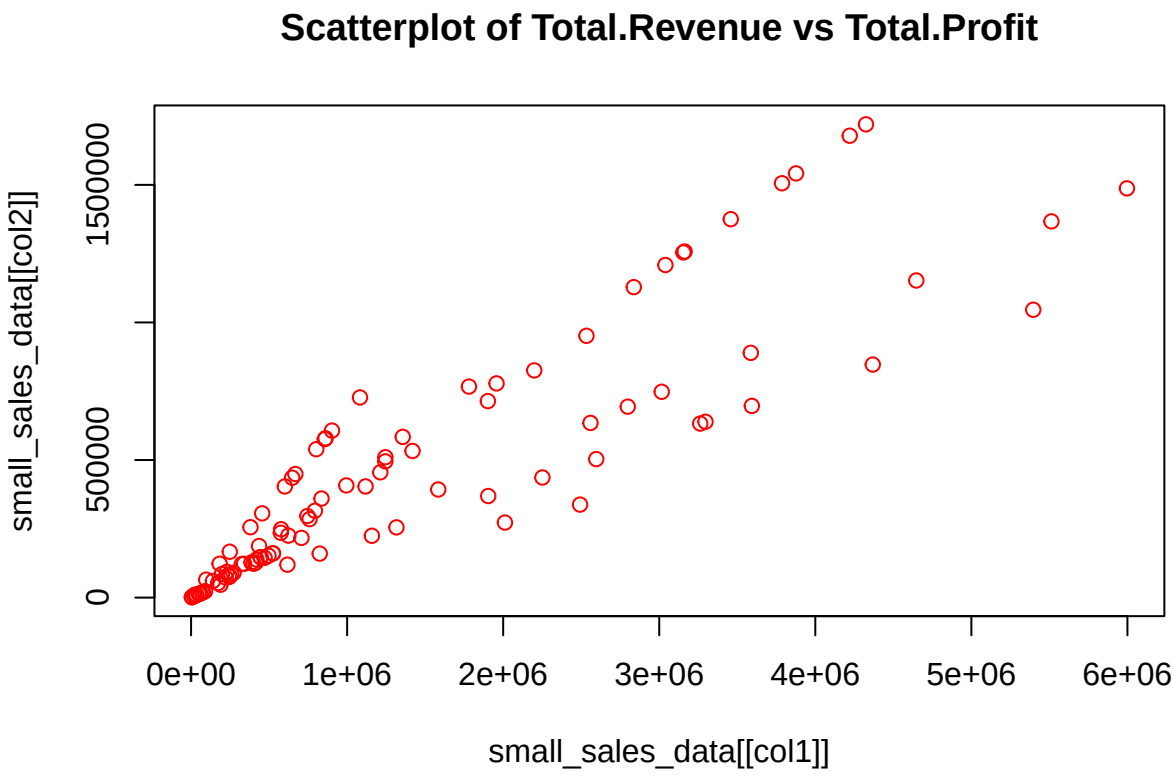

Scatterplot of Total.Revenue vs Order.ID
small_sales_data[[col2]]
1e+09
6e+08
2e+08
0e+00
1e+06
2e+06
3e+06
4e+06
5e+06
6e+06
small_sales_data[[col1]]

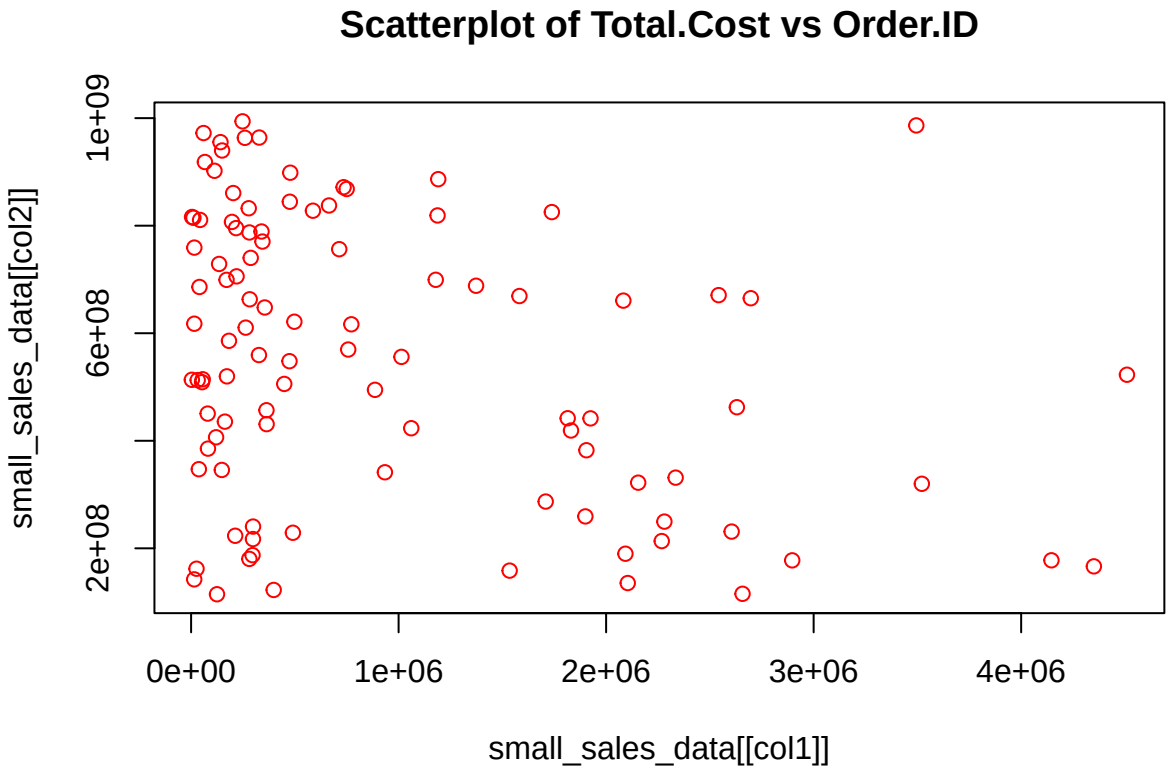

Scatterplot of Total.Revenue vs Units.Sold
small_sales_data[[col2]]
0
2000
6000
10000
0e+00
1e+06
2e+06
3e+06
4e+06
5e+06
6e+06
small_sales_data[[col1]]

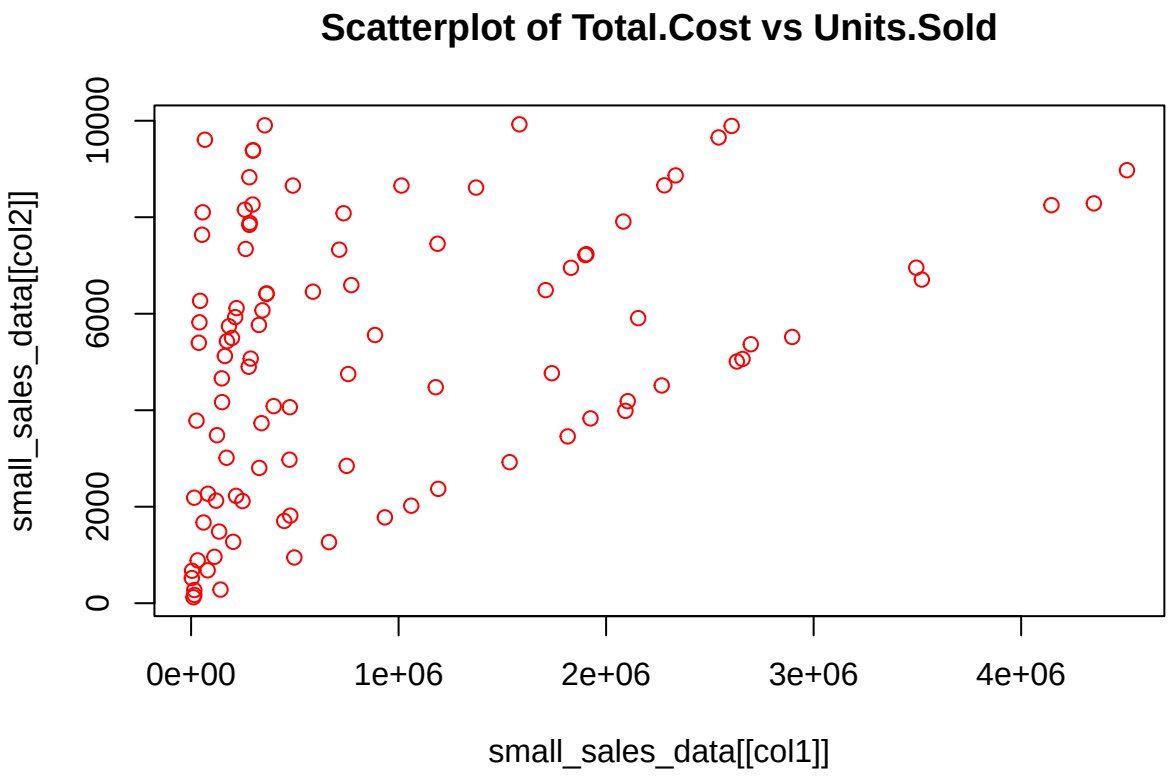

Scatterplot of Total.Revenue vs Unit.Price
small_sales_data[[col2]]
0
100
300
500
0e+00
1e+06
2e+06
3e+06
4e+06
5e+06
6e+06
small_sales_data[[col1]]

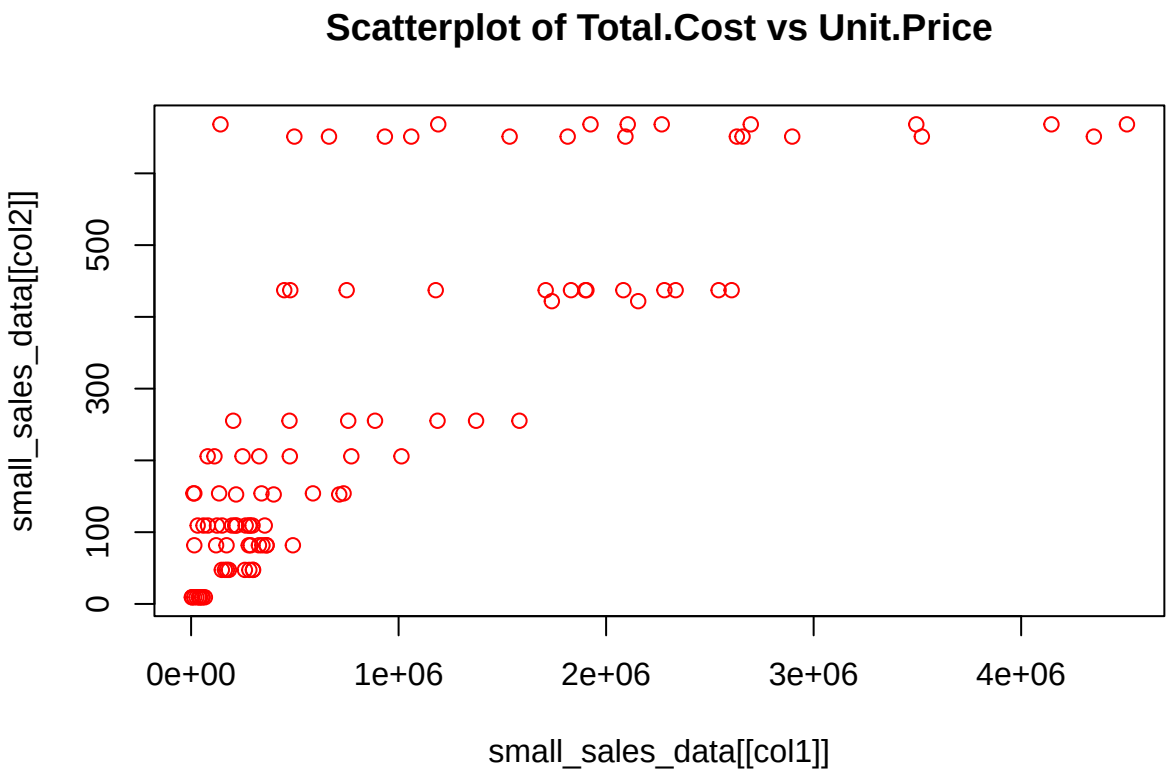

Scatterplot of Total.Revenue vs Unit.Cost
small_sales_data[[col2]]
0
100
200
300
400
500
0e+00
1e+06
2e+06
3e+06
4e+06
5e+06
6e+06
small_sales_data[[col1]]

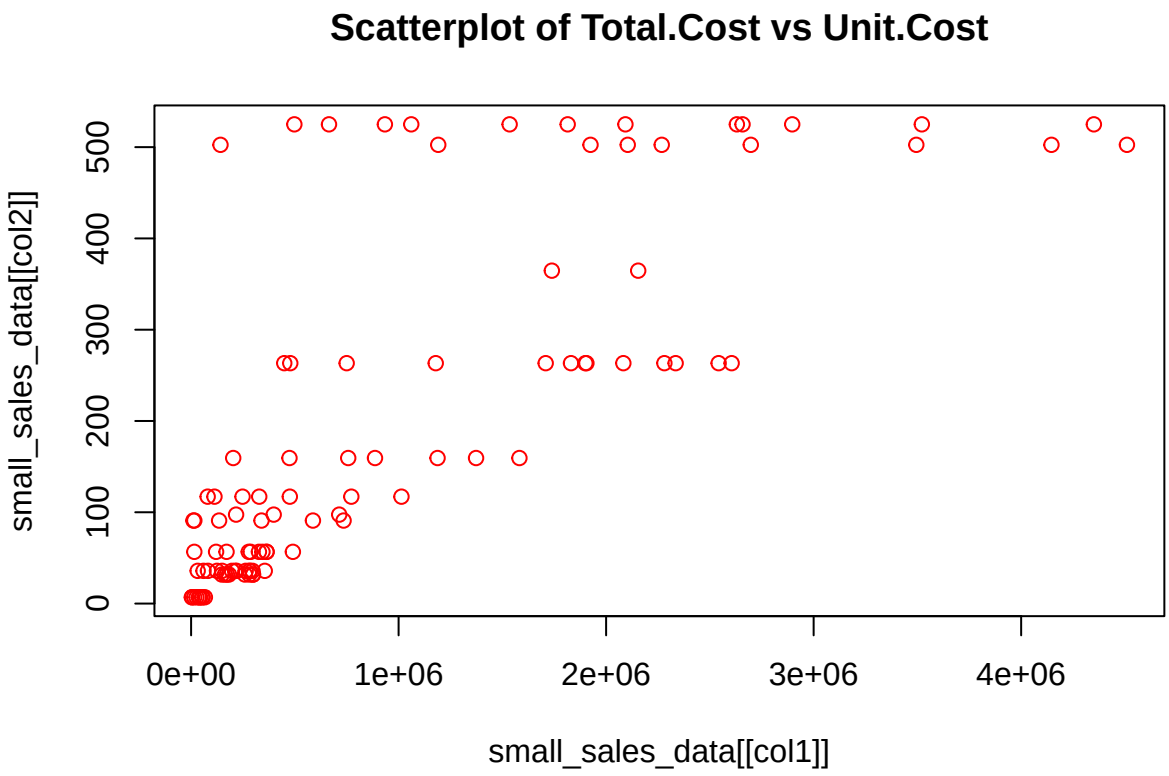

Scatterplot of Total.Revenue vs Total.Cost
small_sales_data[[col2]]
0e+00
2e+06
4e+06
0e+00
1e+06
2e+06
3e+06
4e+06
5e+06
6e+06
small_sales_data[[col1]]

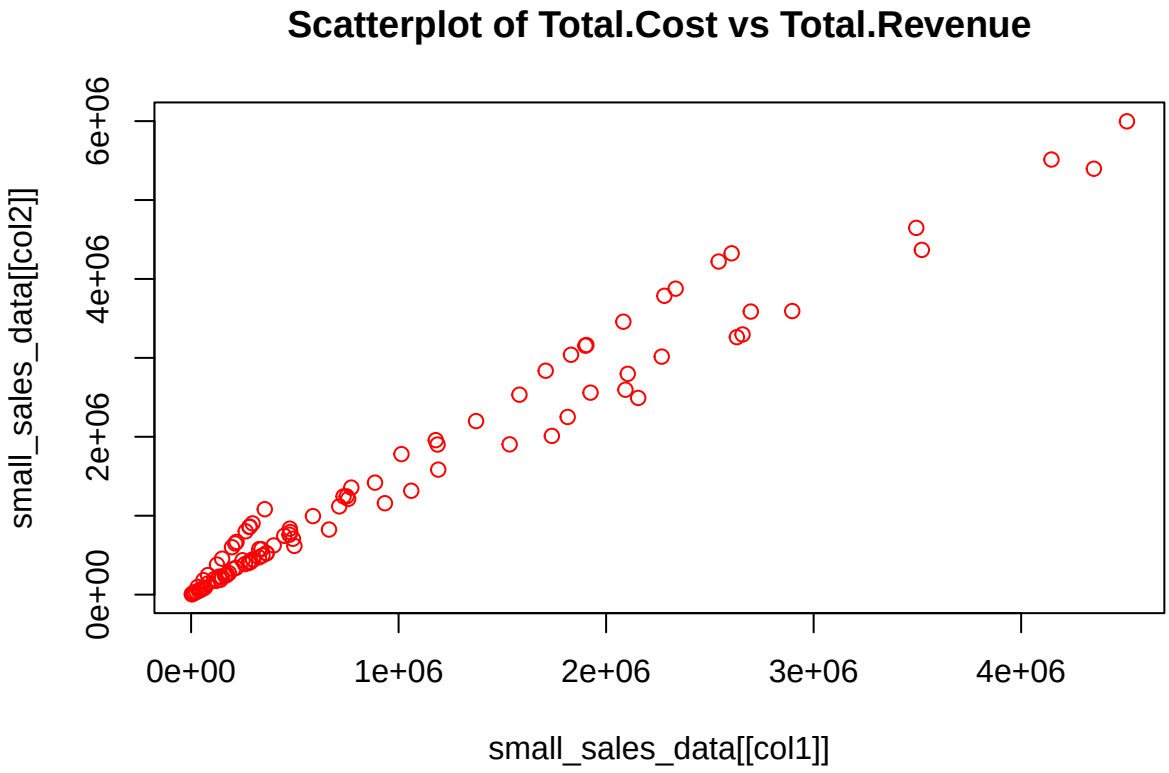

Scatterplot of Total.Revenue vs Total.Profit
small_sales_data[[col2]]
0
500000
1500000
0e+00
1e+06
2e+06
3e+06
4e+06
5e+06
6e+06
small_sales_data[[col1]]

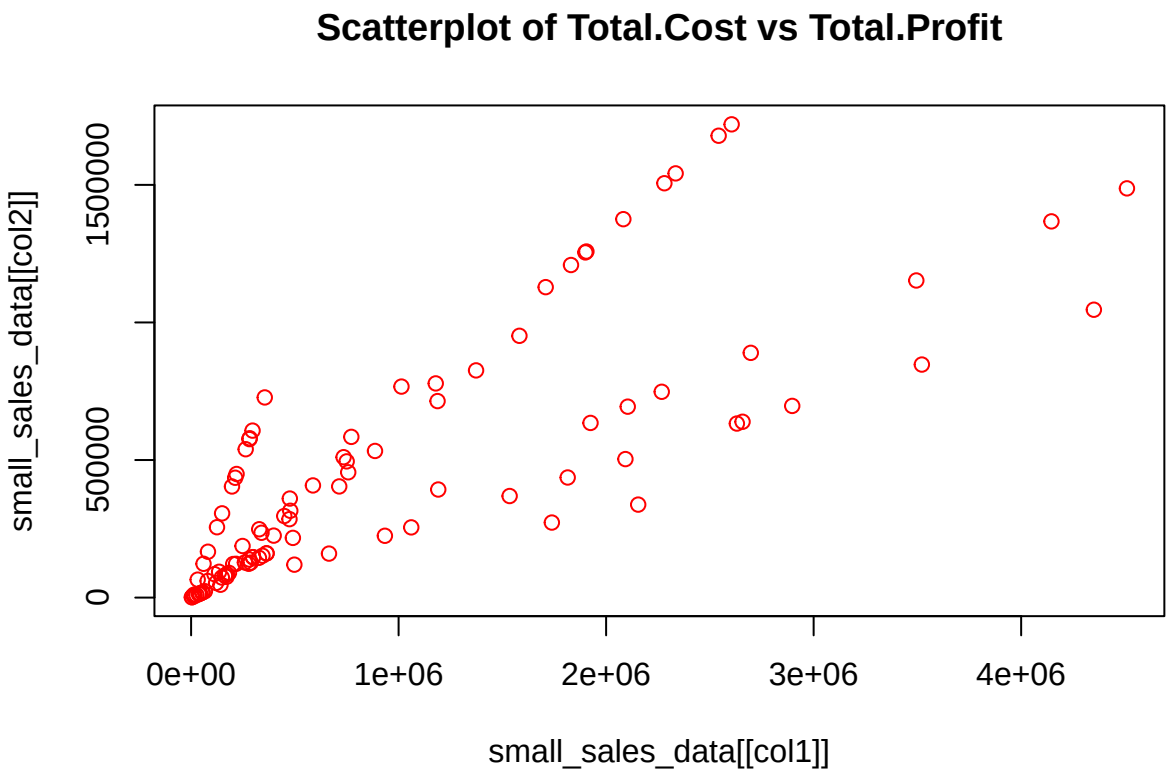

Scatterplot of Total.Cost vs Order.ID
small_sales_data[[col2]]
1e+09
6e+08
2e+08
0e+00
1e+06
2e+06
3e+06
4e+06
small_sales_data[[col1]]

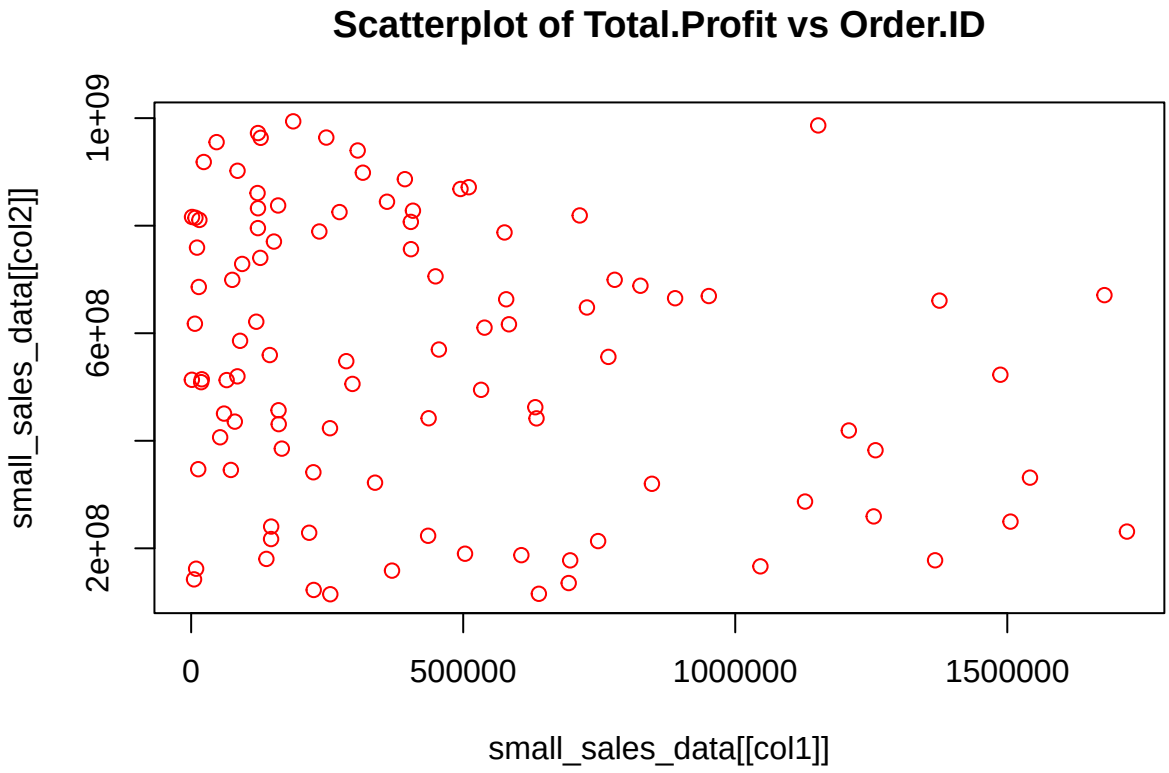

Scatterplot of Total.Cost vs Units.Sold
small_sales_data[[col2]]
0
2000
6000
10000
0e+00
1e+06
2e+06
3e+06
4e+06
small_sales_data[[col1]]

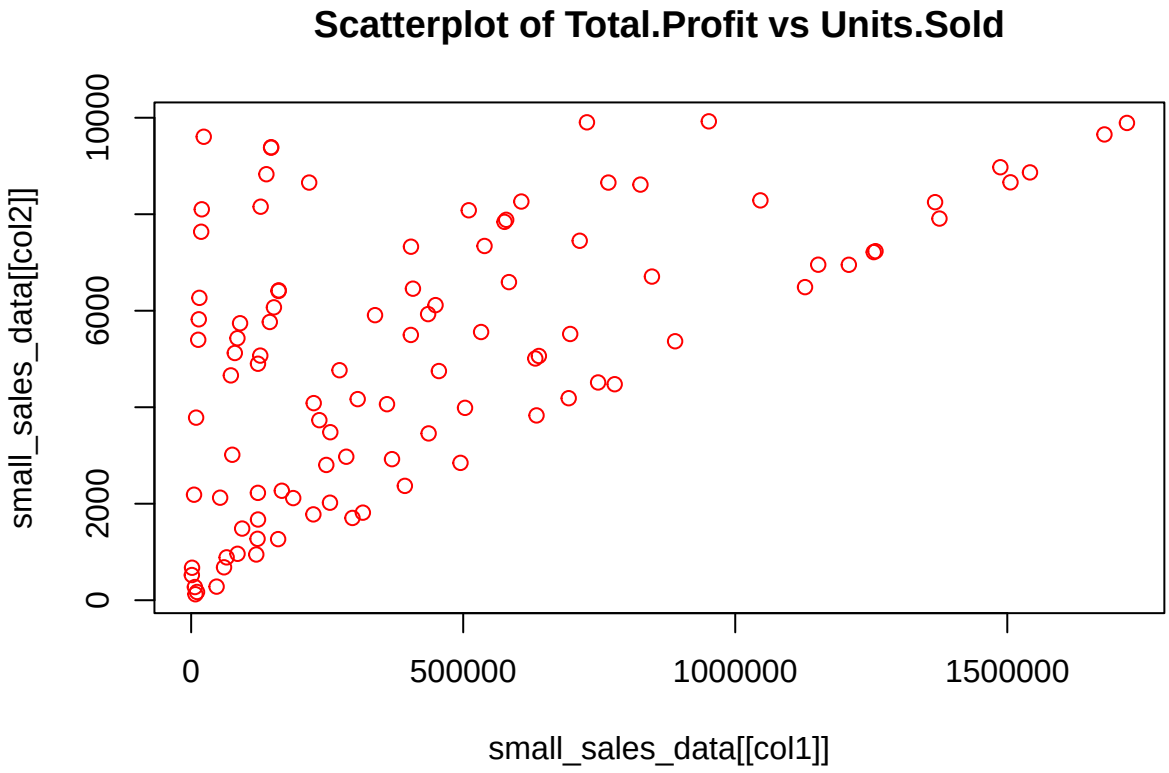

Scatterplot of Total.Cost vs Unit.Price
small_sales_data[[col2]]
0
100
300
500
0e+00
1e+06
2e+06
3e+06
4e+06
small_sales_data[[col1]]

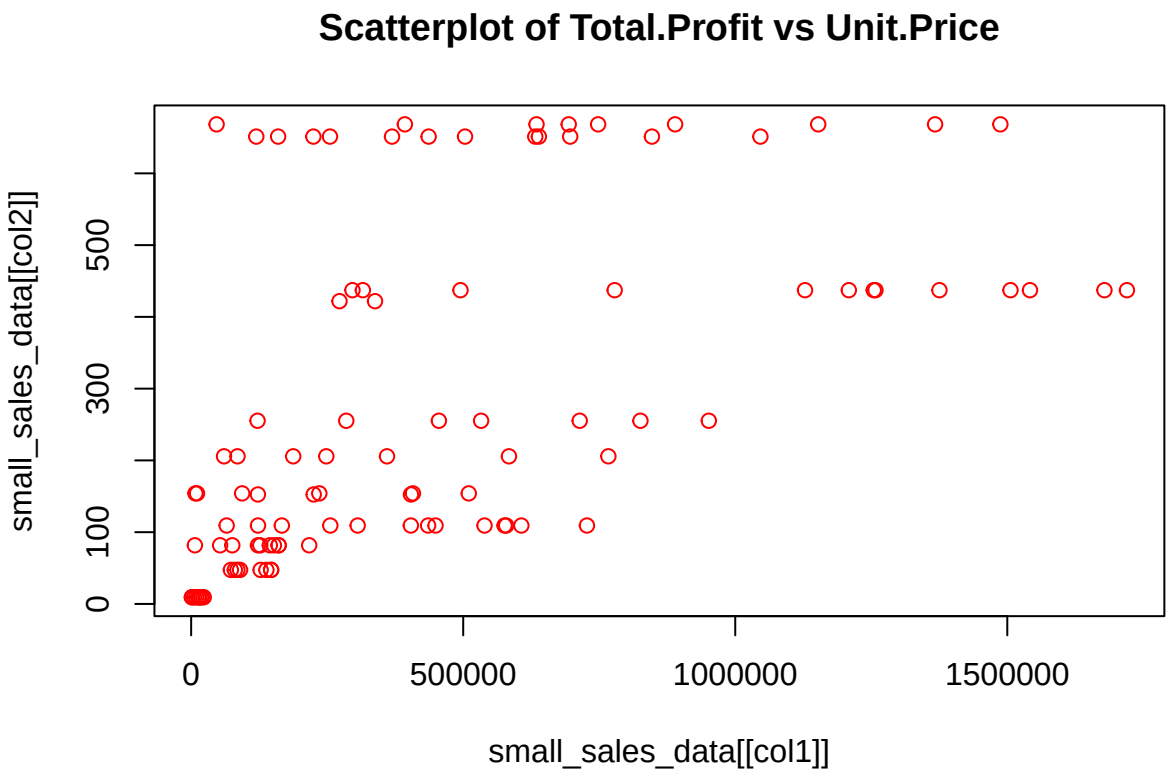

Scatterplot of Total.Cost vs Unit.Cost
small_sales_data[[col2]]
0
100
200
300
400
500
0e+00
1e+06
2e+06
3e+06
4e+06
small_sales_data[[col1]]

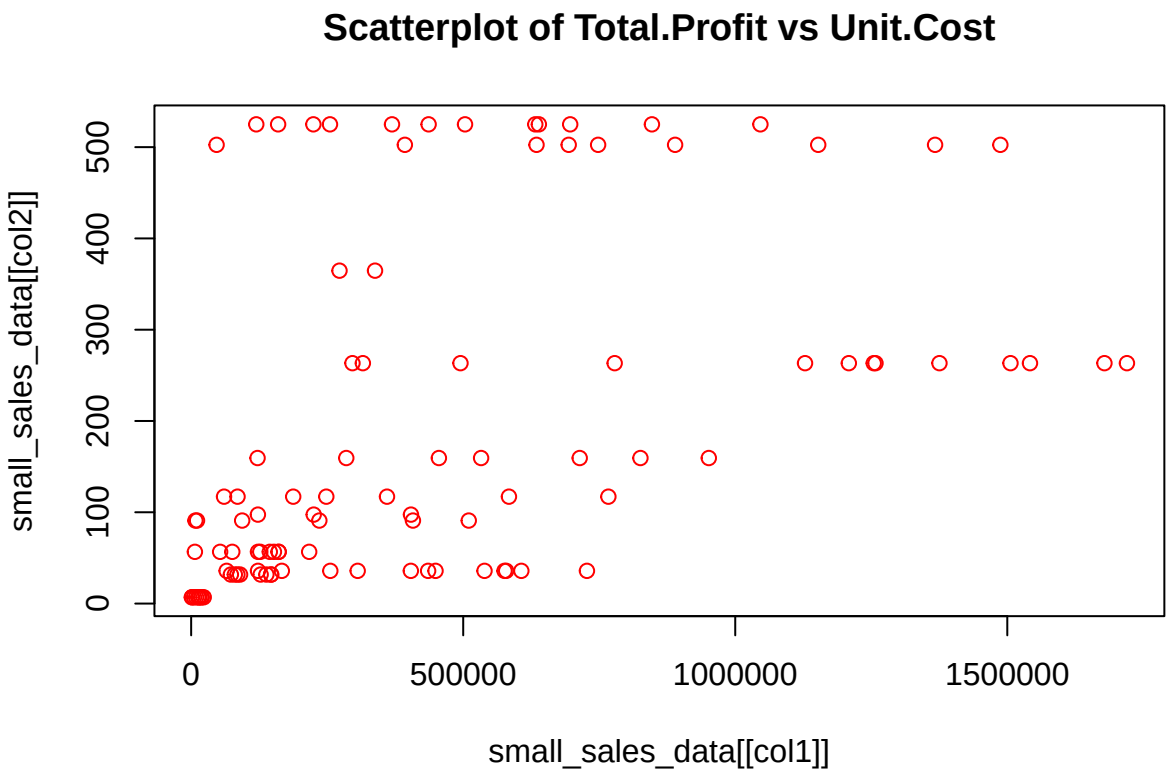

Scatterplot of Total.Cost vs Total.Revenue
small_sales_data[[col2]]
0e+00
2e+06
4e+06
6e+06
0e+00
1e+06
2e+06
3e+06
4e+06
small_sales_data[[col1]]

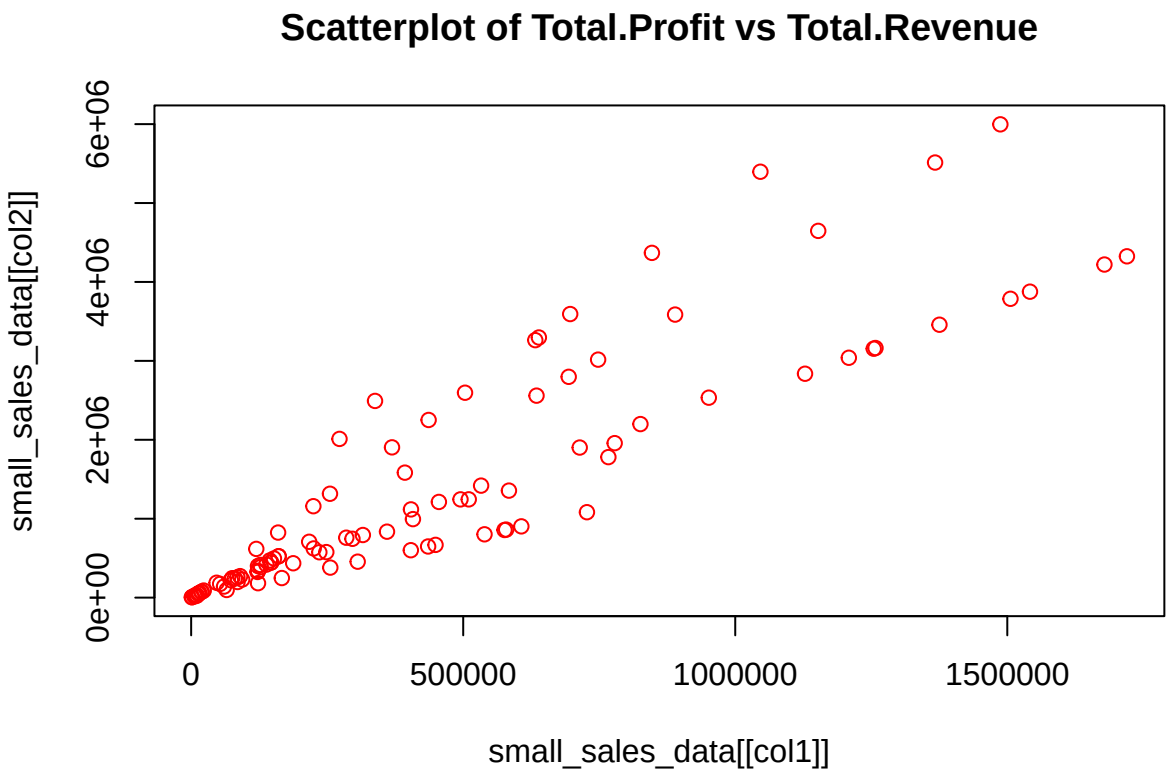

Scatterplot of Total.Cost vs Total.Profit
small_sales_data[[col2]]
0
500000
1500000
0e+00
1e+06
2e+06
3e+06
4e+06
small_sales_data[[col1]]

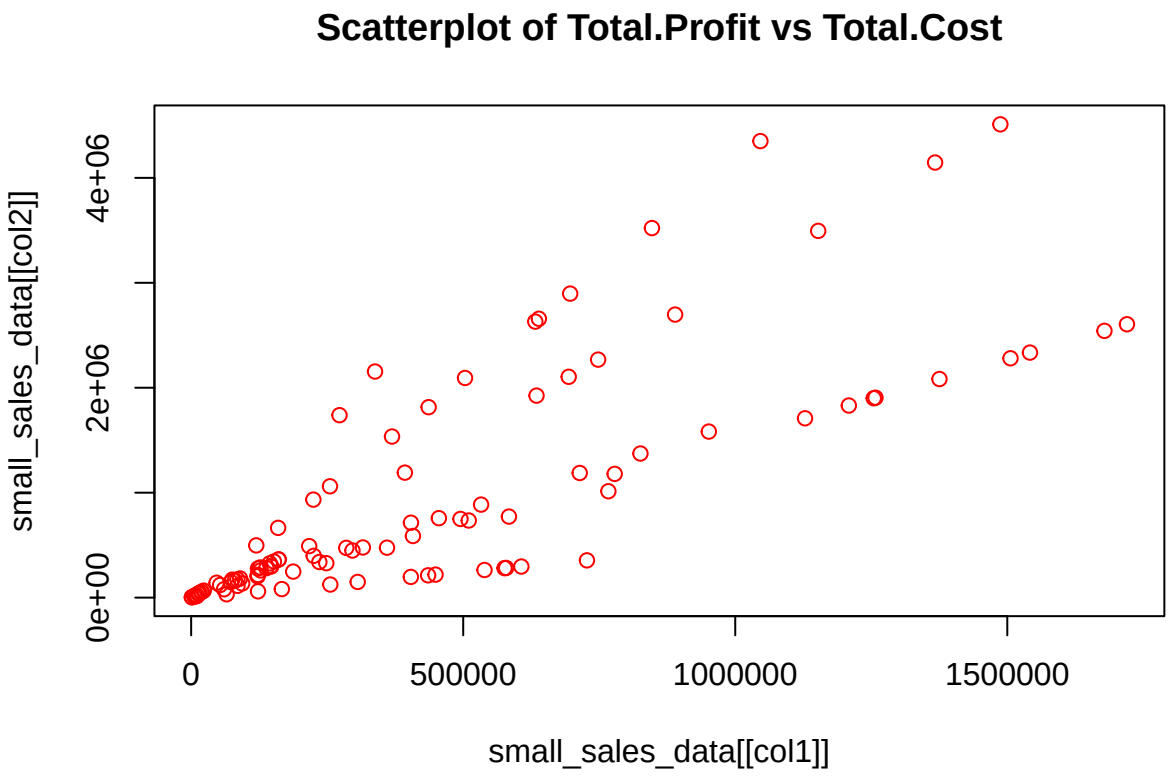

Scatterplot of Total.Profit vs Order.ID
small_sales_data[[col2]]
1e+09
6e+08
2e+08
0
500000
1000000
1500000
small_sales_data[[col1]]


Scatterplot of Total.Profit vs Units.Sold
small_sales_data[[col2]]
0
2000
6000
10000
0
500000
1000000
1500000
small_sales_data[[col1]]


Scatterplot of Total.Profit vs Unit.Price
small_sales_data[[col2]]
0
100
300
500
0
500000
1000000
1500000
small_sales_data[[col1]]


Scatterplot of Total.Profit vs Unit.Cost
small_sales_data[[col2]]
0
100
200
300
400
500
0
500000
1000000
1500000
small_sales_data[[col1]]


Scatterplot of Total.Profit vs Total.Revenue
small_sales_data[[col2]]
0e+00
2e+06
4e+06
6e+06
0
500000
1000000
1500000
small_sales_data[[col1]]

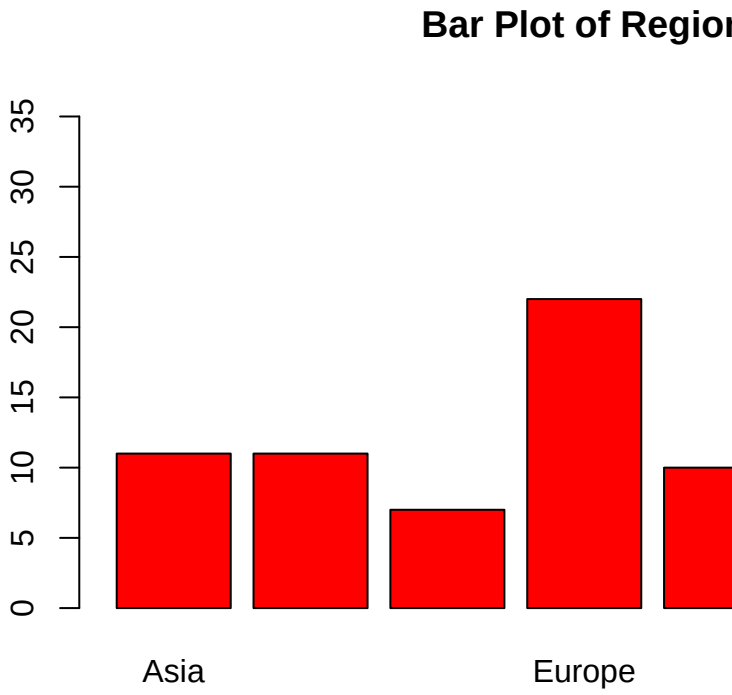


A scatter plot with dispersed points often indicates a weak or nonexistent linear relationship between the variables, suggesting their independence. The random distribution of points implies no clear or predictable connection between the variables. This randomness can occur in both numeric and categorical data, and in statistical terms, such a pattern may indicate homoscedasticity, where variability is consistent across different levels. Although the lack of a discernible pattern suggests no correlation, additional analysis may be required to fully understand the data dynamics and potential hidden relationships. When a scatter plot shows clusters of points at the top and bottom with a straight line running through the middle, it hints at a possible linear relationship. This pattern points toward a positive correlation, meaning that as one variable increases, the other tends to increase as well. However, to confirm the strength and nature of this relationship, further analysis like calculating the correlation coefficient is necessary. Scatter plots that exhibit outliers concentrated at one extreme, with most points clustered at the other, suggest the presence of influential data points or non-linear relationships. These outliers can skew the analysis, potentially distorting the results of correlation or regression studies. It's important to examine these outliers carefully, as they may represent anomalies or unique cases in the data. Further investigations, such as residual analysis or non-linear modeling, may be needed to accurately capture the underlying trends.

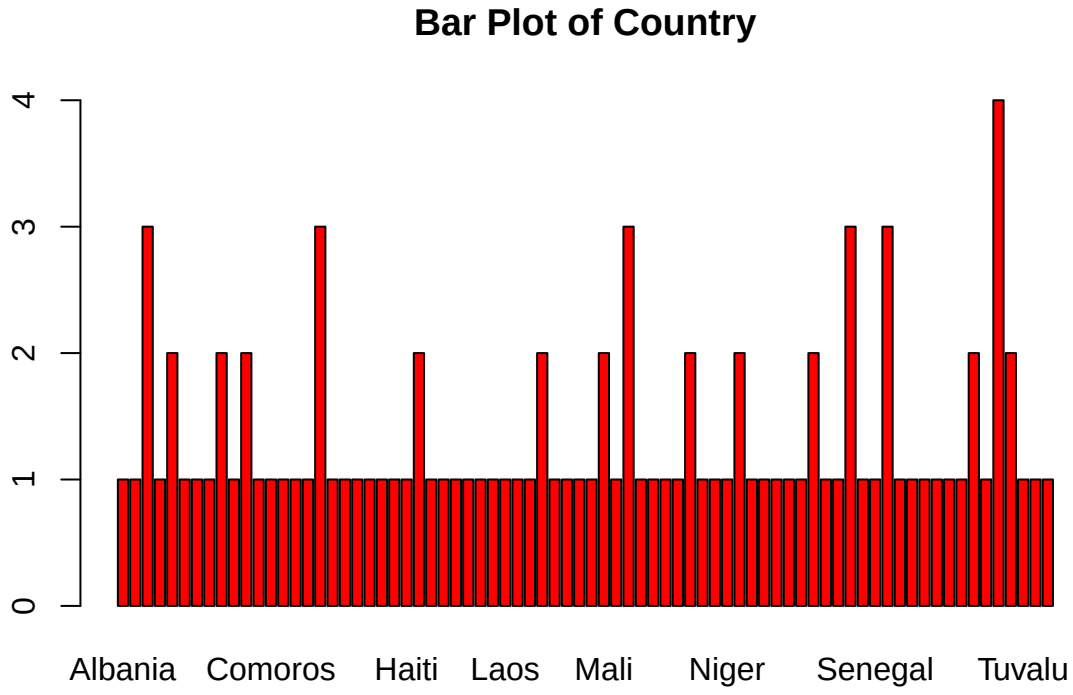
A diagonal line from the bottom-left to the top-right of a scatter plot signifies a positive linear relationship, where an increase in one variable corresponds with an increase in the other. The steeper the line, the stronger the correlation. To better quantify this relationship, calculating correlation coefficients or performing regression analysis would provide deeper insights into the strength and nature of the association. A straight vertical line in a scatter plot, extending from the bottom to the top, suggests a perfect positive correlation. This indicates that as one variable increases, the other increases proportionally, leading to a correlation coefficient of +1. However, this type of perfect correlation is rare in real-world datasets, as factors like measurement error and other variables usually introduce deviations. Lastly, a horizontal alignment of points in a scatter plot implies a perfect negative linear relationship, where as one variable increases, the other decreases at a consistent rate. This results in a correlation coefficient of -1. Similar to positive correlations, real-world data often exhibit variations due to external influences or measurement inconsistencies, making

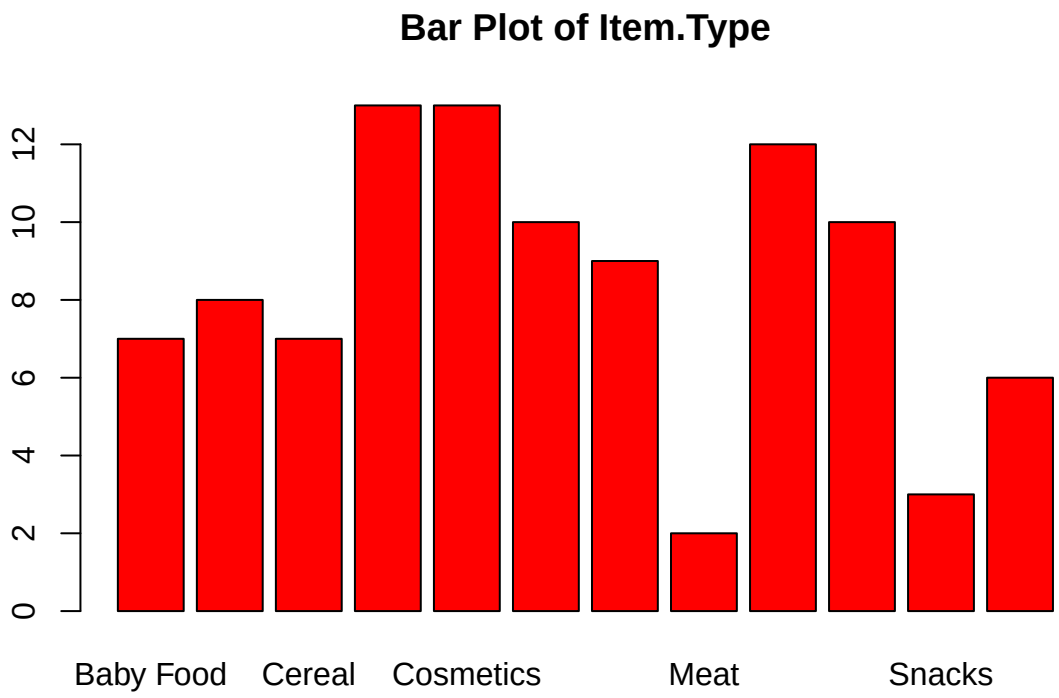
such idealized scenarios uncommon. This version provides the same information in a fresh format with slight elaboration and additional context for clarity.

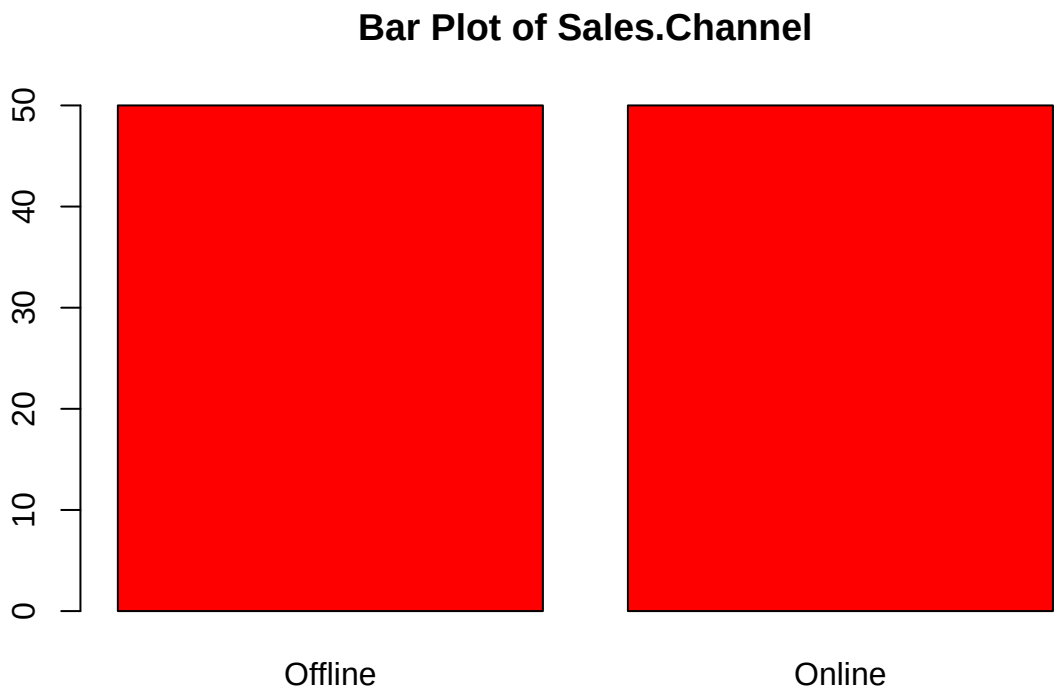
```
# Loop through each column in the dataset
for (column_name in names(small_sales_data)) {

  # Check if the column is not numeric
  if (!is.numeric(small_sales_data[[column_name]])) {

    # Create a bar plot for non-numeric (categorical) columns with red color
    barplot(table(small_sales_data[[column_name]]),
            main = paste("Bar Plot of", column_name),   # Set title for each bar plot
            col = "red")
  }
}
```

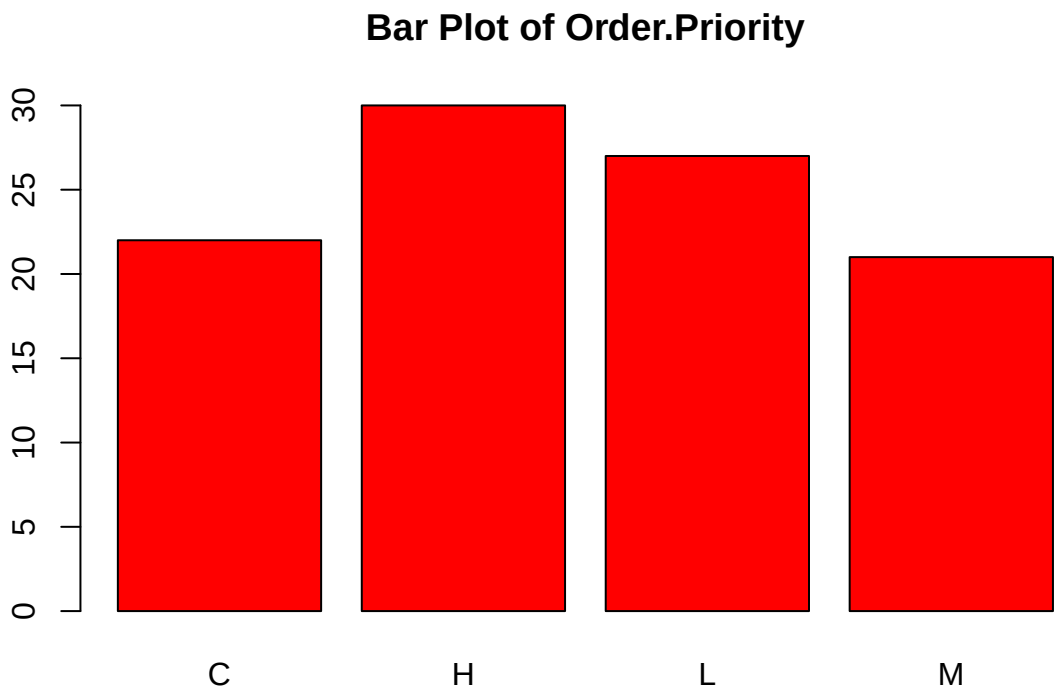


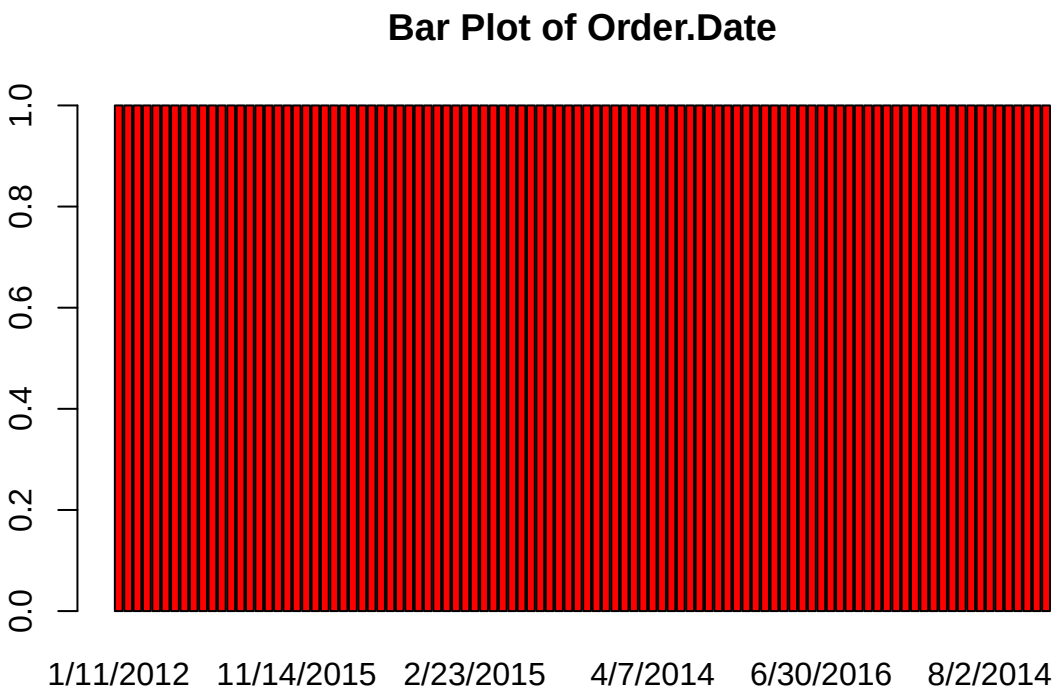
Frequency for each variables for each columns:

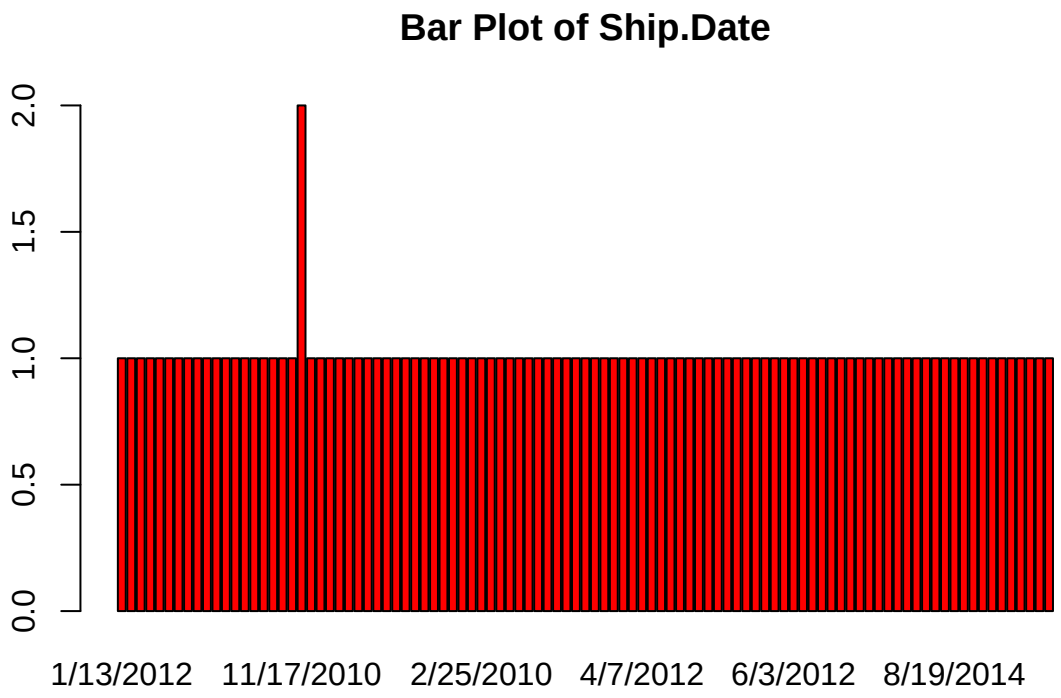



Bar Plot of Item.Type
0
2
4
6
8
10
12
Baby Food
Cereal
Cosmetics
Meat
Snacks


Bar Plot of Sales.Channel
0
10
20
30
40
50
Offline
Online

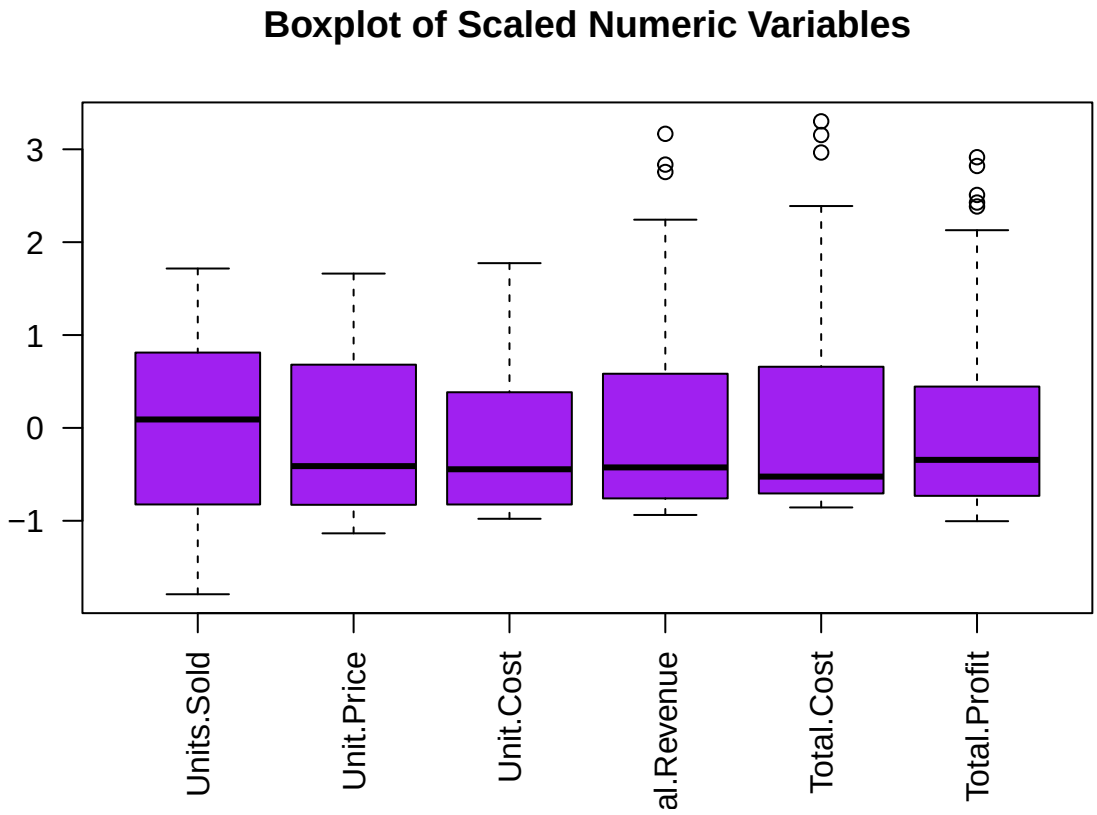

Bar Plot of Order.Priority
0
5
10
15
20
25
30
C
H
L
M


Bar Plot of Order.Date
0.0
0.2
0.4
0.6
0.8
1.0
1/11/2012
11/14/2015
2/23/2015
4/7/2014
6/30/2016
8/2/2014



Boxplots for all numeric columns: Based on the dataset summary, the numeric variables (Units Sold, Unit Price, Unit Cost, Total Revenue, Total Cost, Total Profit) exhibit a wide range of values, each with different scales. For example, the Units Sold variable spans from a minimum of 124 to a maximum of 9,925, while Total Revenue ranges from 4,870 to 5,997,055. Due to these varying scales, when all variables are plotted together in a boxplot, certain variables may dominate the plot, making it difficult to observe the finer details of others. To address this issue and improve the clarity of the visualization, normalizing or scaling the variables would bring them onto a similar scale, allowing for a more balanced and insightful comparison.

```r
# Extract relevant numeric columns from the dataset for scaling
scaled_df <- small_sales_data[, c("Units.Sold", "Unit.Price", "Unit.Cost", "Total.Revenue", "Total.Cost

# Apply scaling to standardize the numeric variables
scaled_df <- scale(scaled_df)

# Create a boxplot for the scaled numeric variables with a purple color scheme
boxplot(scaled_df, col = "purple",
        main = "Boxplot of Scaled Numeric Variables",
        las = 2)  # Adjust the axis labels to display vertically for improved readability
```



The boxplot analysis indicates a right-skewed distribution, with a concentration of data points toward higher values. This suggests that most observations are clustered at the upper end of the range. The presence of outliers, which are values significantly deviating from the overall trend, adds complexity to the dataset and suggests the existence of exceptional cases or anomalies that warrant further investigation. These outliers may reflect rare events or extreme variations in the data that could have a substantial impact on the analysis.

Large Dataset:

Content of the Tables:

```
# The head() function shows the initial rows of the dataframe, providing a snapshot of the data for a q
head(large_sales_data)
```

Head of the Dataframe:

```
##                           Region                         Country       Item.Type
## 1           Sub-Saharan Africa                            Chad Office Supplies
## 2                         Europe                          Latvia       Beverages
## 3 Middle East and North Africa                        Pakistan      Vegetables
## 4           Sub-Saharan Africa Democratic Republic of the Congo       Household
## 5                         Europe                  Czech Republic       Beverages
## 6           Sub-Saharan Africa                    South Africa       Beverages
##   Sales.Channel Order.Priority Order.Date  Order.ID Ship.Date Units.Sold
```

```
## 1        Online              L  1/27/2011 292494523 2/12/2011       4484
## 2        Online              C 12/28/2015 361825549 1/23/2016       1075
## 3       Offline              C  1/13/2011 141515767  2/1/2011       6515
## 4        Online              C  9/11/2012 500364005 10/6/2012       7683
## 5        Online              C 10/27/2015 127481591 12/5/2015       3491
## 6       Offline              H  7/10/2012 482292354 8/21/2012       9880
##   Unit.Price Unit.Cost Total.Revenue Total.Cost Total.Profit
## 1     651.21    524.96    2920025.64 2353920.64    566105.00
## 2      47.45     31.79      51008.75   34174.25     16834.50
## 3     154.06     90.93    1003700.90  592408.95    411291.95
## 4     668.27    502.54    5134318.41 3861014.82   1273303.59
## 5      47.45     31.79     165647.95  110978.89     54669.06
## 6      47.45     31.79     468806.00  314085.20    154720.80
```

```r
# The summary() function provides key statistics such as mean, median, quartiles, minimum, and maximum
summary(large_sales_data)
```

Summary Statistic:

```
##     Region             Country           Item.Type         Sales.Channel
##  Length:10000       Length:10000       Length:10000       Length:10000
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##
##
##
##  Order.Priority      Order.Date           Order.ID          Ship.Date
##  Length:10000       Length:10000       Min.   :100089156   Length:10000
##  Class :character   Class :character   1st Qu.:321806669   Class :character
##  Mode  :character   Mode  :character   Median :548566305   Mode  :character
##                                        Mean   :549871874
##                                        3rd Qu.:775998104
##                                        Max.   :999934232
##    Units.Sold      Unit.Price       Unit.Cost      Total.Revenue
##  Min.   :    2   Min.   :  9.33   Min.   :  6.92   Min.   :    168
##  1st Qu.: 2531   1st Qu.:109.28   1st Qu.: 56.67   1st Qu.: 288551
##  Median : 4962   Median :205.70   Median :117.11   Median : 800051
##  Mean   : 5003   Mean   :268.14   Mean   :188.81   Mean   :1333355
##  3rd Qu.: 7472   3rd Qu.:437.20   3rd Qu.:364.69   3rd Qu.:1819143
##  Max.   :10000   Max.   :668.27   Max.   :524.96   Max.   :6680027
##    Total.Cost       Total.Profit
##  Min.   :    125   Min.   :     43.4
##  1st Qu.: 164786   1st Qu.:  98329.1
##  Median : 481606   Median : 289099.0
##  Mean   : 938266   Mean   : 395089.3
##  3rd Qu.:1183822   3rd Qu.: 566422.7
##  Max.   :5241726   Max.   :1738178.4
```

The dataset consists of 10,000 entries for each variable, with a mix of data types including character, integer, and numeric. The descriptive statistics provide valuable insights into the distribution and central tendencies of the numeric variables, helping to summarize their overall patterns and variability.

```r
# The str() function gives a summary of the dataset's structure, detailing the data type and format of
str(large_sales_data)
```

Structure of Tables:

```
## 'data.frame':    10000 obs. of  14 variables:
##  $ Region        : chr  "Sub-Saharan Africa" "Europe" "Middle East and North Africa" "Sub-Saharan Af
##  $ Country       : chr  "Chad" "Latvia" "Pakistan" "Democratic Republic of the Congo" ...
##  $ Item.Type     : chr  "Office Supplies" "Beverages" "Vegetables" "Household" ...
##  $ Sales.Channel : chr  "Online" "Online" "Offline" "Online" ...
##  $ Order.Priority: chr  "L" "C" "C" "C" ...
##  $ Order.Date    : chr  "1/27/2011" "12/28/2015" "1/13/2011" "9/11/2012" ...
##  $ Order.ID      : int  292494523 361825549 141515767 500364005 127481591 482292354 844532620 564251
##  $ Ship.Date     : chr  "2/12/2011" "1/23/2016" "2/1/2011" "10/6/2012" ...
##  $ Units.Sold    : int  4484 1075 6515 7683 3491 9880 4825 3330 2431 6197 ...
##  $ Unit.Price    : num  651.2 47.5 154.1 668.3 47.5 ...
##  $ Unit.Cost     : num  525 31.8 90.9 502.5 31.8 ...
##  $ Total.Revenue : num  2920026 51009 1003701 5134318 165648 ...
##  $ Total.Cost    : num  2353921 34174 592409 3861015 110979 ...
##  $ Total.Profit  : num  566105 16835 411292 1273304 54669 ...
```

The dataset contains 10,000 observations across 14 variables, including categorical features like Region, Country, Item Type, Sales Channel, and Order Priority, and numerical features such as Units Sold, Unit Price, Total Revenue, and Total Profit. The categorical variables provide insights into the geographic and logistical details of each transaction, while the numerical variables capture the financial aspects. With a wide range of regions, countries, and products, the dataset offers a solid foundation for analyzing sales trends and financial performance.

Dataset Characteristics (Structure, Size, Dependencies, Labels, etc.):

```r
# Get the total count of rows in the dataset
numbs_of_rows <- nrow(large_sales_data)
print(paste("Number of rows:", numbs_of_rows))
```

Number of Rows and Columns:

```
## [1] "Number of rows: 10000"
```

```r
# Get the total count of columns in the dataset
numbs_of_columns <- ncol(large_sales_data)
print(paste("Number of columns:", numbs_of_columns))
```

```
## [1] "Number of columns: 14"
```

```r
# Display the names of all columns in the dataset
names(large_sales_data)
```

```
##  [1] "Region"         "Country"        "Item.Type"      "Sales.Channel"
##  [5] "Order.Priority" "Order.Date"     "Order.ID"       "Ship.Date"
##  [9] "Units.Sold"     "Unit.Price"     "Unit.Cost"      "Total.Revenue"
## [13] "Total.Cost"     "Total.Profit"
```

```r
# Calculate the total number of missing values (NA) in the entire dataset
total_na_count <- sum(is.na(large_sales_data))
print("Total missing values in the dataset:")
```

Number of NA's:

```
## [1] "Total missing values in the dataset:"
```

```r
print(total_na_count)
```

```
## [1] 0
```

```r
# Calculate the number of missing values (NA) for each individual column
na_per_column <- colSums(is.na(large_sales_data))
print("Missing values per column:")
```

```
## [1] "Missing values per column:"
```

```r
print(na_per_column)
```

```
##         Region        Country      Item.Type  Sales.Channel Order.Priority
##              0              0              0              0              0
##     Order.Date       Order.ID      Ship.Date     Units.Sold     Unit.Price
##              0              0              0              0              0
##      Unit.Cost  Total.Revenue     Total.Cost   Total.Profit
##              0              0              0              0
```

There are no missing values (NA) in this dataset.

```r
# Remove the "Order.ID" column and keep only the numeric columns from the dataset
numeric_large_sales_data <- large_sales_data[sapply(large_sales_data, is.numeric) & colnames(large_sales

# Check if any missing values exist in the numeric columns
if (any(is.na(numeric_large_sales_data))) {
  # Display a warning message if missing values are found
  cat("Warning: Missing values detected in numeric columns. Handle them before proceeding with calculati
} else {
  # Calculate the standard deviation for each numeric variable
  standard_deviation <- apply(numeric_large_sales_data, 2, sd)
```

```
  # Calculate the variance for each numeric variable
  variance <- apply(numeric_large_sales_data, 2, var)

  # Combine the calculated standard deviation and variance into a data frame
  result_large_sales_data <- data.frame(Variable = names(numeric_large_sales_data),
                                        Standard_Deviation = standard_deviation,
                                        Variance = variance)

  # Format the results as an HTML table for better presentation
  result_table <- kable(result_large_sales_data, "html") %>%
    kable_styling()

  # Display the formatted table
  print(result_table)
}
```

Standard Deviation and Variance:

```
## <table class="table" style="margin-left: auto; margin-right: auto;">
##  <thead>
##   <tr>
##    <th style="text-align:left;">   </th>
##    <th style="text-align:left;"> Variable </th>
##    <th style="text-align:right;"> Standard_Deviation </th>
##    <th style="text-align:right;"> Variance </th>
##   </tr>
##  </thead>
## <tbody>
##   <tr>
##    <td style="text-align:left;"> Units.Sold </td>
##    <td style="text-align:left;"> Units.Sold </td>
##    <td style="text-align:right;"> 2873.2465 </td>
##    <td style="text-align:right;"> 8.255545e+06 </td>
##   </tr>
##   <tr>
##    <td style="text-align:left;"> Unit.Price </td>
##    <td style="text-align:left;"> Unit.Price </td>
##    <td style="text-align:right;"> 217.9441 </td>
##    <td style="text-align:right;"> 4.749963e+04 </td>
##   </tr>
##   <tr>
##    <td style="text-align:left;"> Unit.Cost </td>
##    <td style="text-align:left;"> Unit.Cost </td>
##    <td style="text-align:right;"> 176.4459 </td>
##    <td style="text-align:right;"> 3.113316e+04 </td>
##   </tr>
##   <tr>
##    <td style="text-align:left;"> Total.Revenue </td>
##    <td style="text-align:left;"> Total.Revenue </td>
##    <td style="text-align:right;"> 1465026.1739 </td>
##    <td style="text-align:right;"> 2.146302e+12 </td>
##   </tr>
##   <tr>
##    <td style="text-align:left;"> Total.Cost </td>
```

```
##    <td style="text-align:left;"> Total.Cost </td>
##    <td style="text-align:right;"> 1145914.0694 </td>
##    <td style="text-align:right;"> 1.313119e+12 </td>
##   </tr>
##   <tr>
##    <td style="text-align:left;"> Total.Profit </td>
##    <td style="text-align:left;"> Total.Profit </td>
##    <td style="text-align:right;"> 377554.9607 </td>
##    <td style="text-align:right;"> 1.425477e+11 </td>
##   </tr>
## </tbody>
## </table>
```

The dataset provides valuable insights into several key variables, each showing unique patterns. For example, 'Units Sold' exhibits considerable variation, with a standard deviation of 2,873.25 and a variance of 8.26 million, indicating a broad range of sales volumes. In contrast, 'Unit Price' shows lower variability, with a standard deviation of 217.94 and a variance of 47,499.63, suggesting a more consistent pricing approach.

'Unit Cost' displays moderate fluctuation, with a standard deviation of 176.45 and a variance of 31,133.32, hinting at factors influencing production costs. Financial metrics such as 'Total Revenue' and 'Total Cost' also show substantial variation, with standard deviations of 1,465,026.17 and 1,145,914.07, respectively, reflecting the complexity of managing revenue and expenses at scale.

Finally, 'Total Profit' has notable variability, with a standard deviation of 377,554.96 and a variance of 142,547.7, pointing to potential opportunities for further investigation to enhance profitability. This overview highlights the dataset's numeric features, setting the stage for deeper analysis and informed strategic decisions.

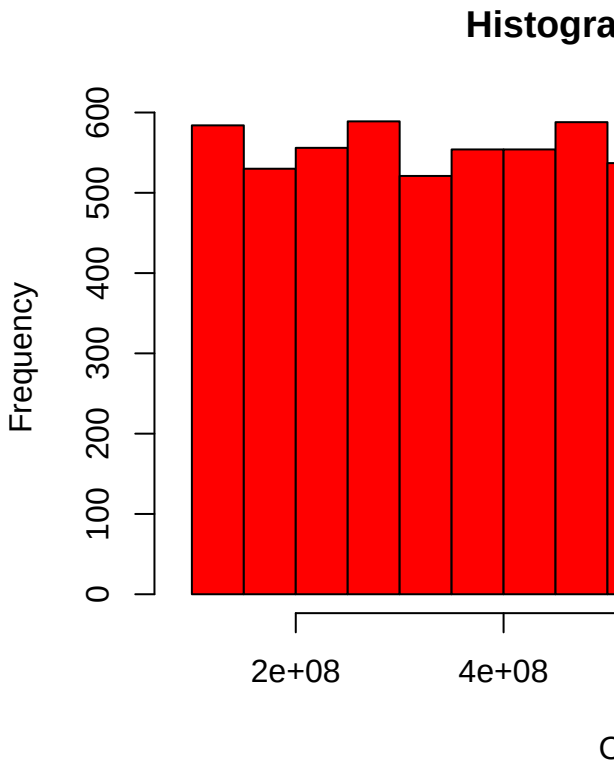
```r
# Calculate the correlation matrix for the numeric columns in the dataset
cor_matrix <- cor(large_sales_data[, numeric_data_cols])

# Display the resulting correlation matrix
cor_matrix
```

Relationships between numeric variables using a correlation matrix.

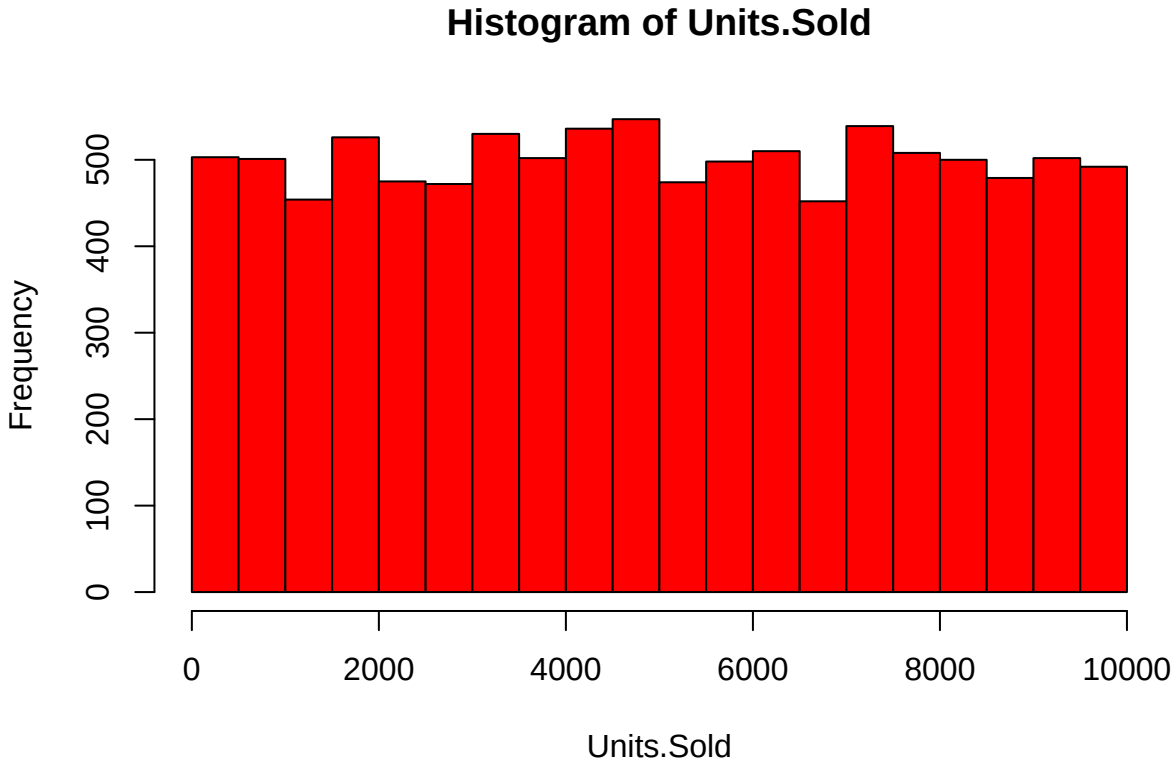
```
##                  Order.ID  Units.Sold  Unit.Price   Unit.Cost Total.Revenue
## Order.ID       1.000000000 -0.01973240  0.01217491  0.01044871  -0.009288123
## Units.Sold    -0.019732401  1.00000000 -0.01297840 -0.01244102   0.518615102
## Unit.Price     0.012174909 -0.01297840  1.00000000  0.98632386   0.733225977
## Unit.Cost      0.010448711 -0.01244102  0.98632386  1.00000000   0.723267379
## Total.Revenue -0.009288123  0.51861510  0.73322598  0.72326738   1.000000000
## Total.Cost    -0.009530296  0.46617752  0.74924307  0.75983568   0.987873673
## Total.Profit  -0.007115369  0.59749001  0.57111439  0.50032251   0.882011543
##                Total.Cost Total.Profit
## Order.ID      -0.009530296 -0.007115369
## Units.Sold     0.466177522  0.597490009
## Unit.Price     0.749243073  0.571114386
## Unit.Cost      0.759835678  0.500322512
## Total.Revenue  0.987873673  0.882011543
## Total.Cost     1.000000000  0.798153247
## Total.Profit   0.798153247  1.000000000
```

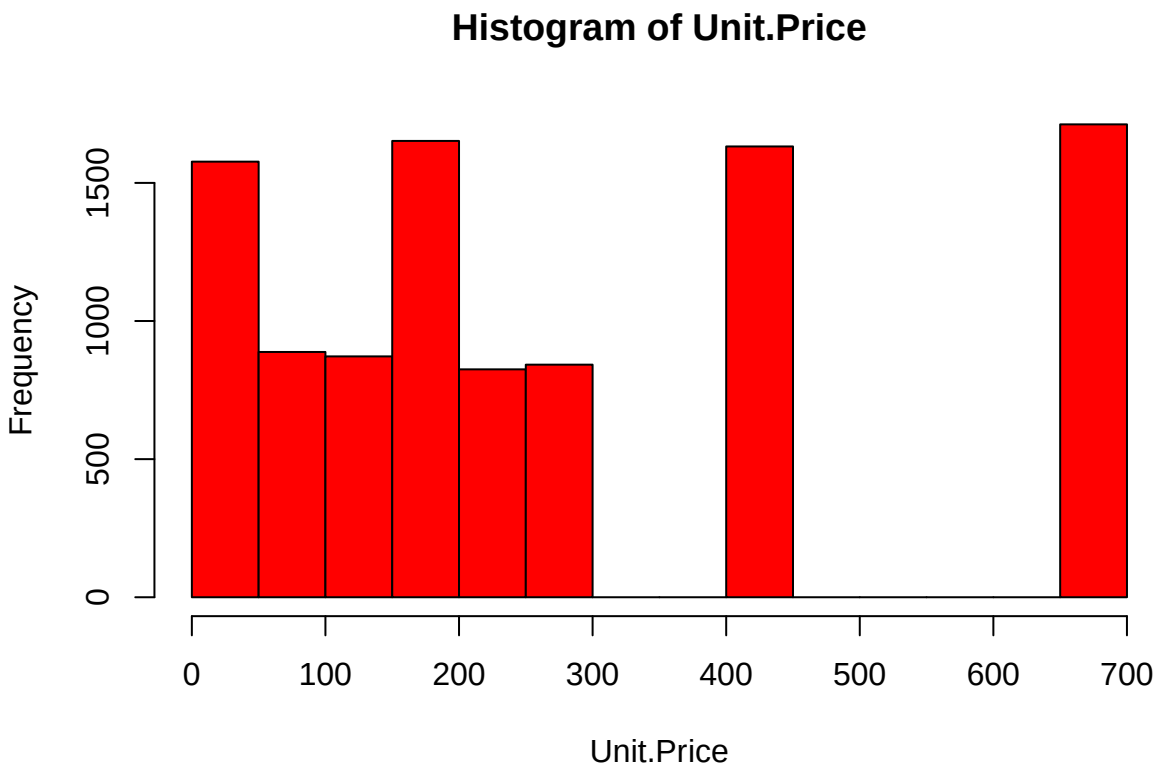
Visualization

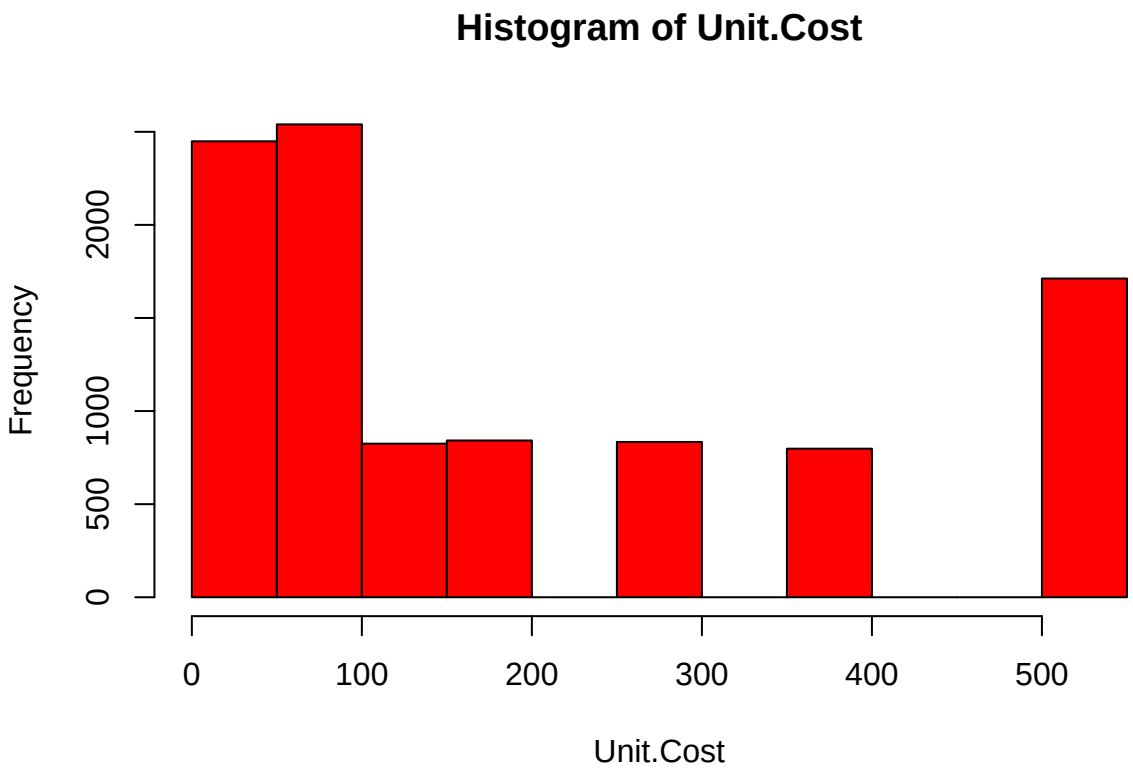
```
# Identify numeric columns in the dataset
numeric_data_cols <- sapply(large_sales_data, is.numeric)

# Generate histograms for each numeric column, using a purple color scheme
for (col in names(large_sales_data)[numeric_data_cols]) {
  hist(large_sales_data[[col]],
       main = paste("Histogram of", col),  # Set the title for each histogram
       xlab = col,                         # Label the x-axis with the column name
       col = "red")                     # Set the color of the bars to red
}
```

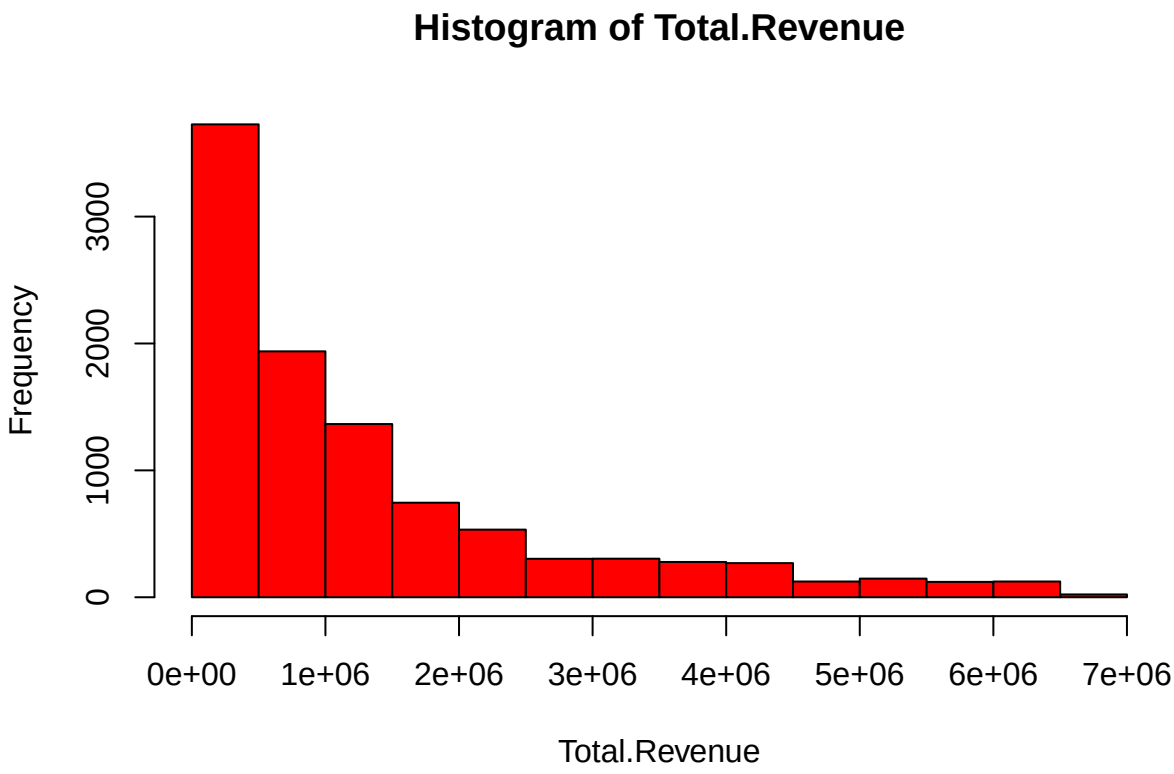


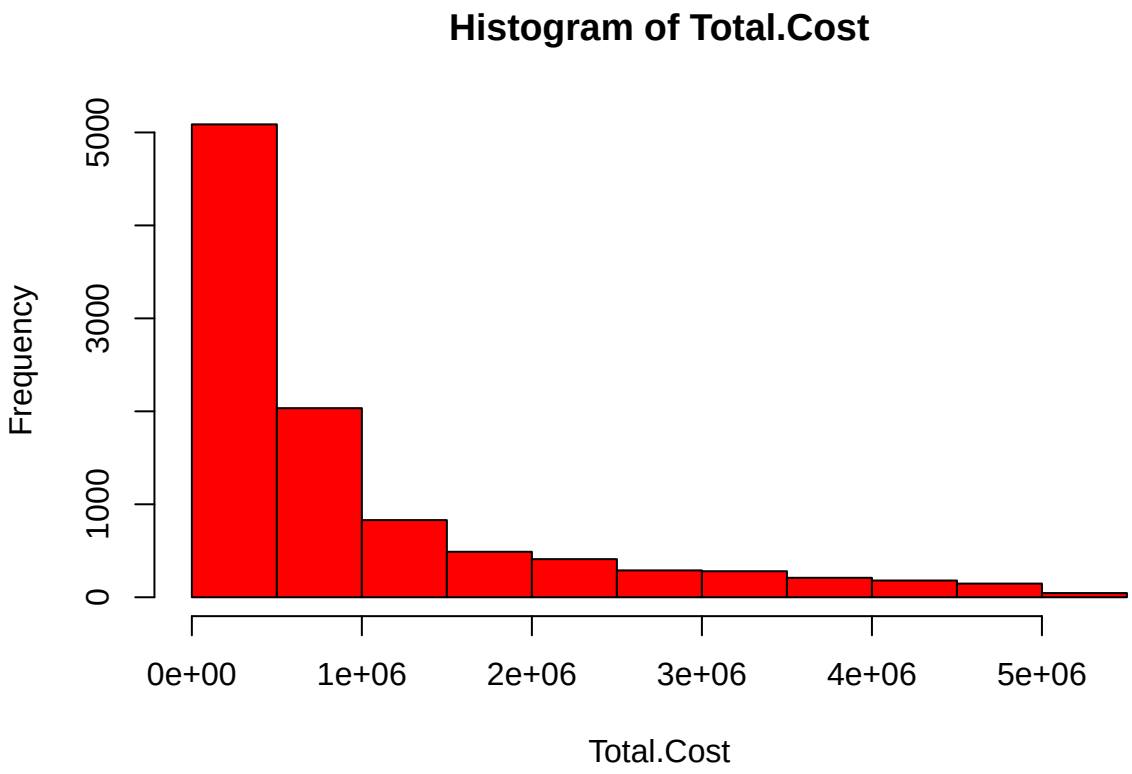
Histograms to visualize the distribution of numeric variables:

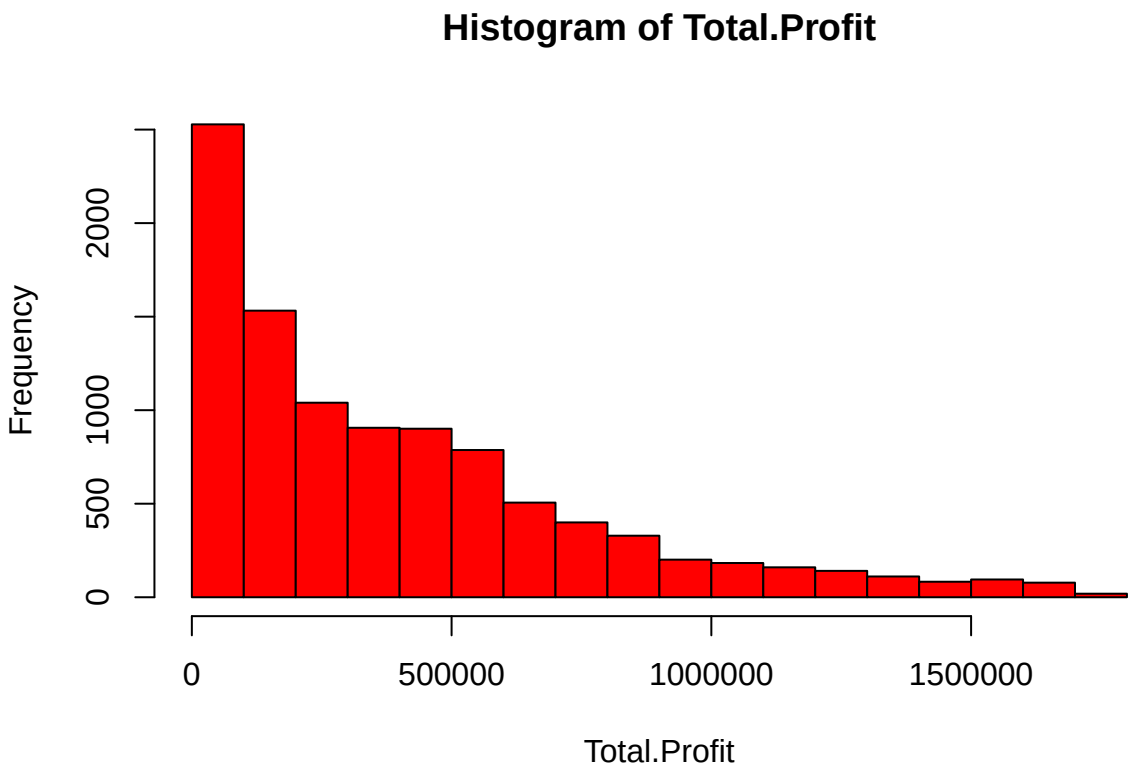



Histogram of Unit.Price
Frequency
0
500
1000
1500
0
100
200
300
400
500
600
700
Unit.Price


Histogram of Unit.Cost
Frequency
0
500
1000
2000
0
100
200
300
400
500
Unit.Cost

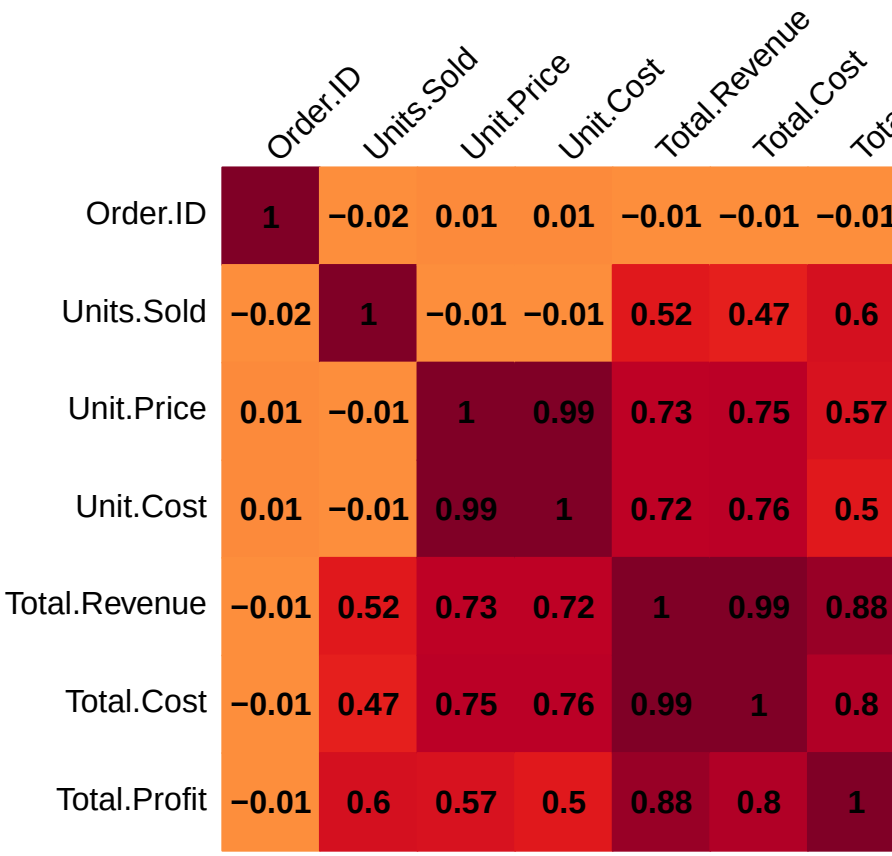

Histogram of Total.Revenue
Frequency
0
1000
2000
3000
0e+00
1e+06
2e+06
3e+06
4e+06
5e+06
6e+06
7e+06
Total.Revenue


Histogram of Total.Cost
Frequency
0
1000
3000
5000
0e+00
1e+06
2e+06
3e+06
4e+06
5e+06
Total.Cost



The resulting histograms for the numeric variables show a similar distribution pattern as observed in the smaller dataset.

```
# Generate a correlation plot using a light color scheme
corrplot(cor_matrix, method = "color",
         addCoef.col = "black",     # Display correlation coefficients in black
         tl.col = "black",          # Set the color of the variable labels to black
         tl.srt = 45,               # Rotate the labels 45 degrees for better readability
         col = colorRampPalette(brewer.pal(9, "YlOrRd"))(100))  # Use a yellow-red color palette
```



Correlation between variables:

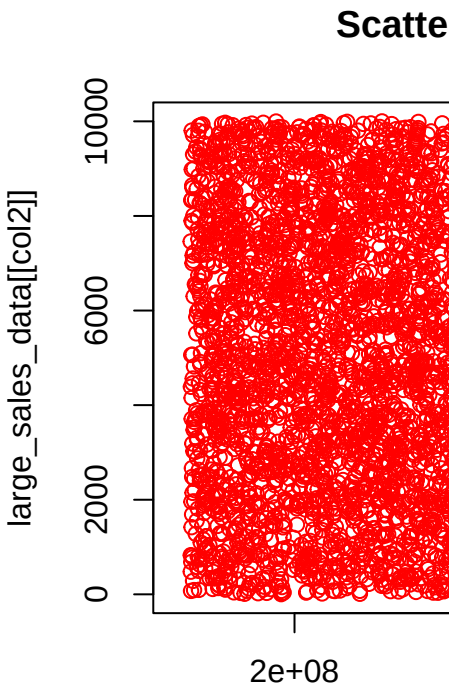
Similar to the small dataset, the correlation coefficients in this larger dataset reveal important insights into the relationships between key variables. For instance, a correlation coefficient of 0.99 between **Unit Cost** and **Unit Price** suggests a very strong positive correlation, indicating that as production costs increase, there is a proportional rise in the selling price. This close relationship highlights a direct connection between production expenses and pricing strategies.

Additionally, the correlation coefficient between **Total Cost** and **Total Revenue** is 0.99, slightly higher than in the small dataset, further demonstrating a robust positive correlation. This indicates that an increase in production costs tends to be accompanied by a proportional increase in revenue, underscoring the close link between costs and revenue generation.

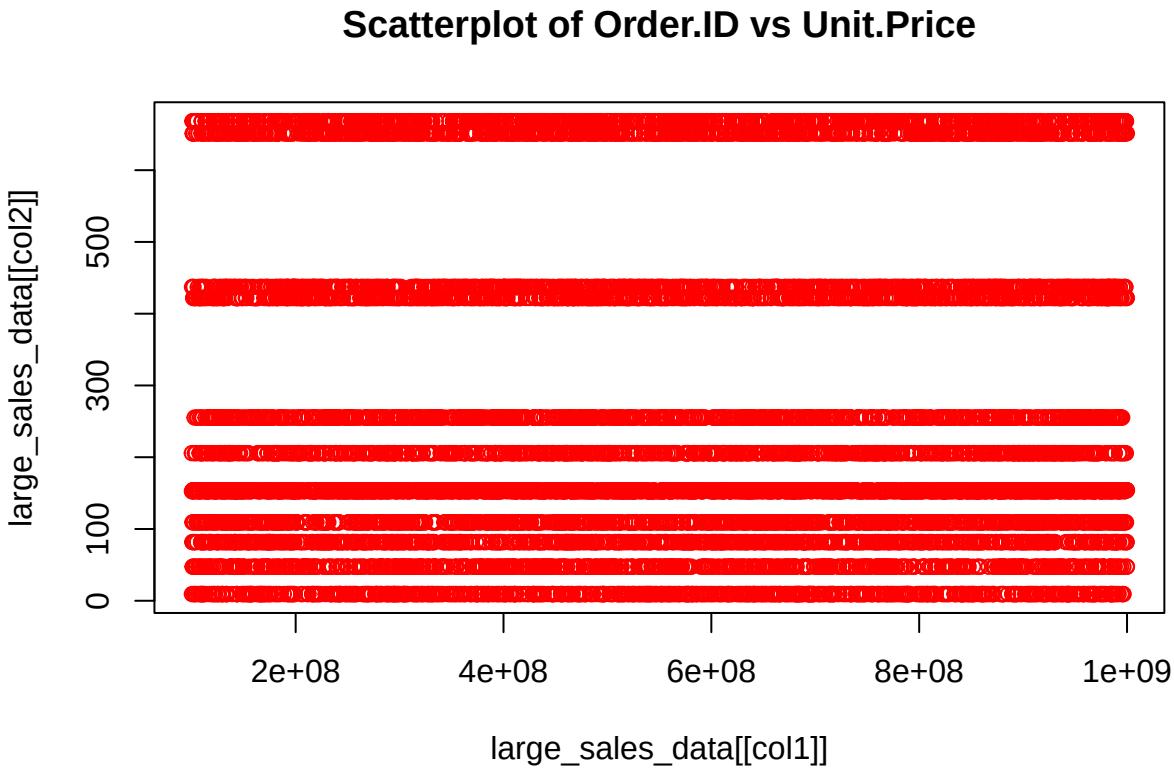
The correlation between **Total Profit** and **Total Revenue** is slightly lower at 0.88, compared to the small dataset, though still a strong positive correlation. This suggests that as profits rise, revenue generally increases as well, although the relationship is not as intense as in other examples. Nevertheless, this correlation still highlights a meaningful link between profitability and overall revenue performance.

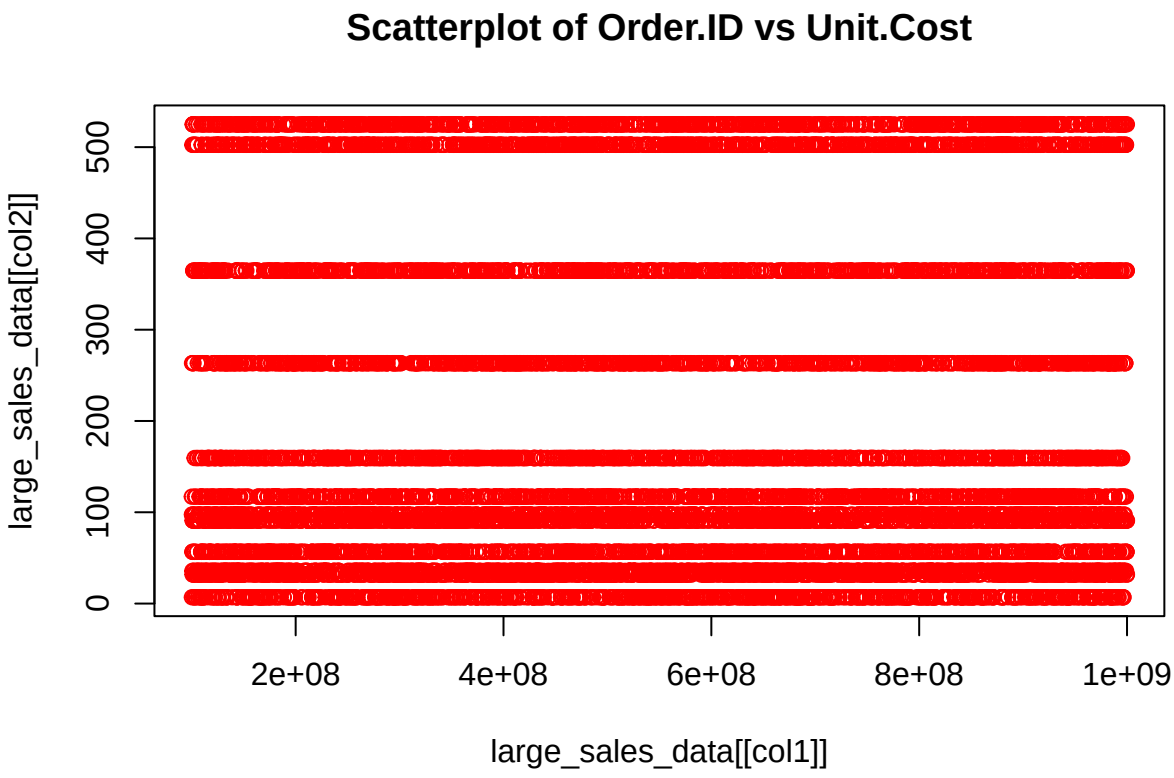
```
# Identify numeric columns in the 'large_sales_data' dataset
numeric_data_cols <- sapply(large_sales_data, is.numeric)

# Generate scatterplots for each pair of numeric columns
for (col1 in names(large_sales_data)[numeric_data_cols]) {
  for (col2 in names(large_sales_data)[numeric_data_cols]) {
    # Ensure we only plot different column pairs
    if (col1 != col2) {
      # Create a scatterplot with red-colored points for each pair
```

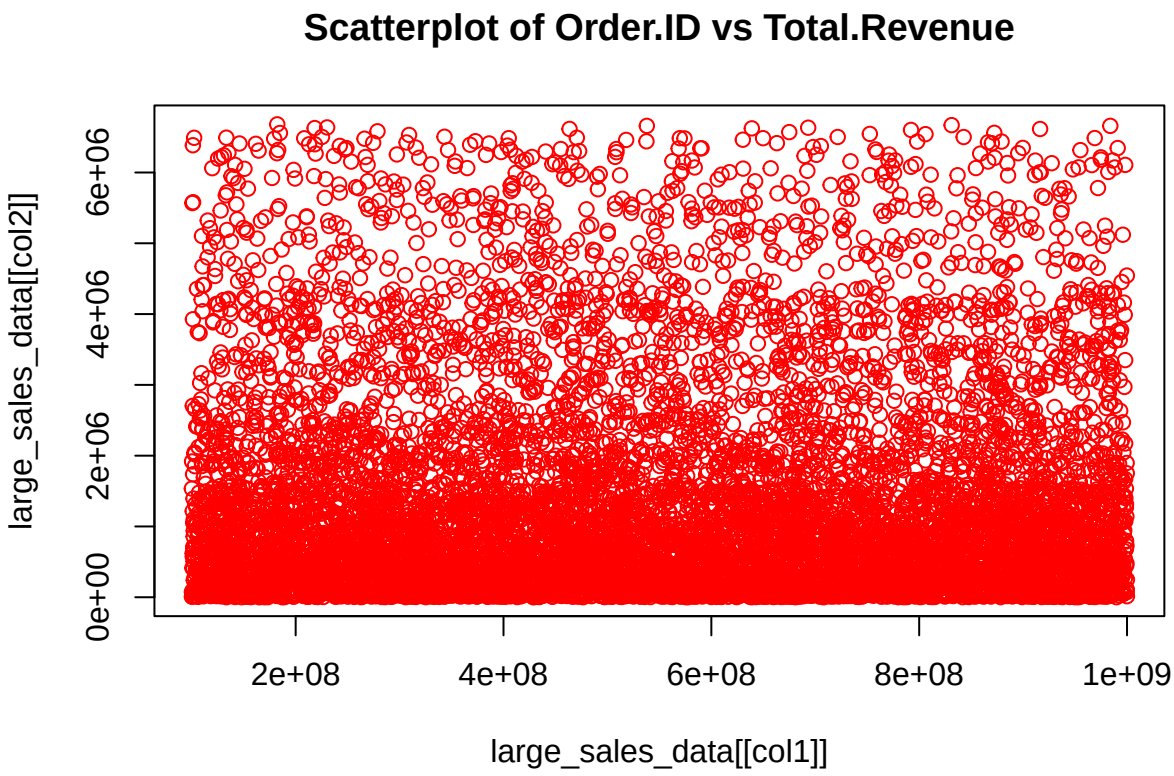
```
      plot(large_sales_data[[col1]], large_sales_data[[col2]],
           main = paste("Scatterplot of", col1, "vs", col2),
           col = "red")
    }
  }
}
```

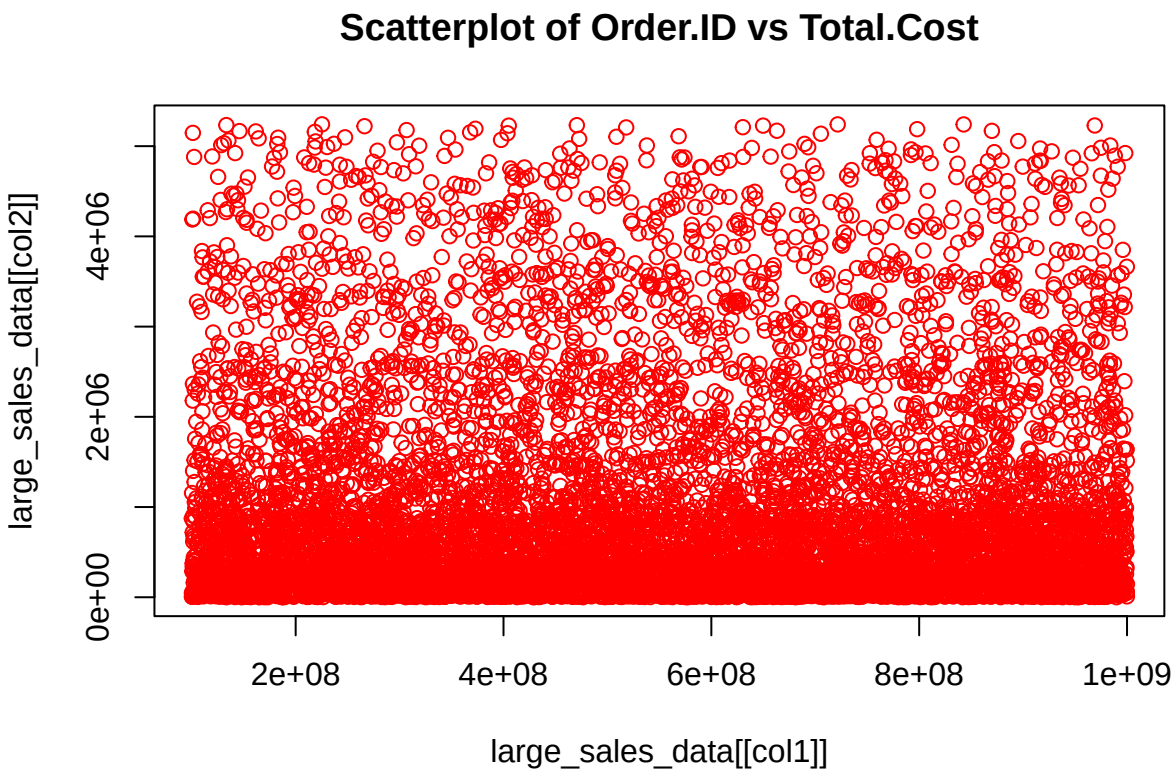


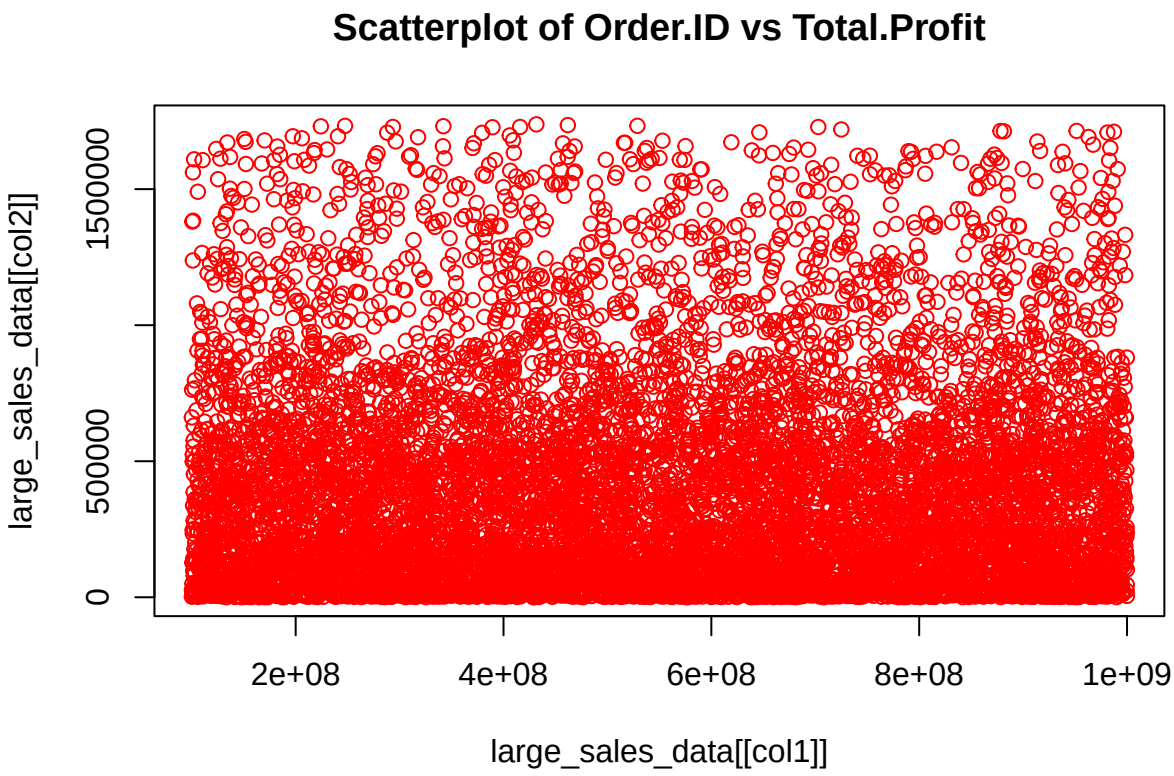
Individual Scatter Plots to explore relationships for each pair of variables:

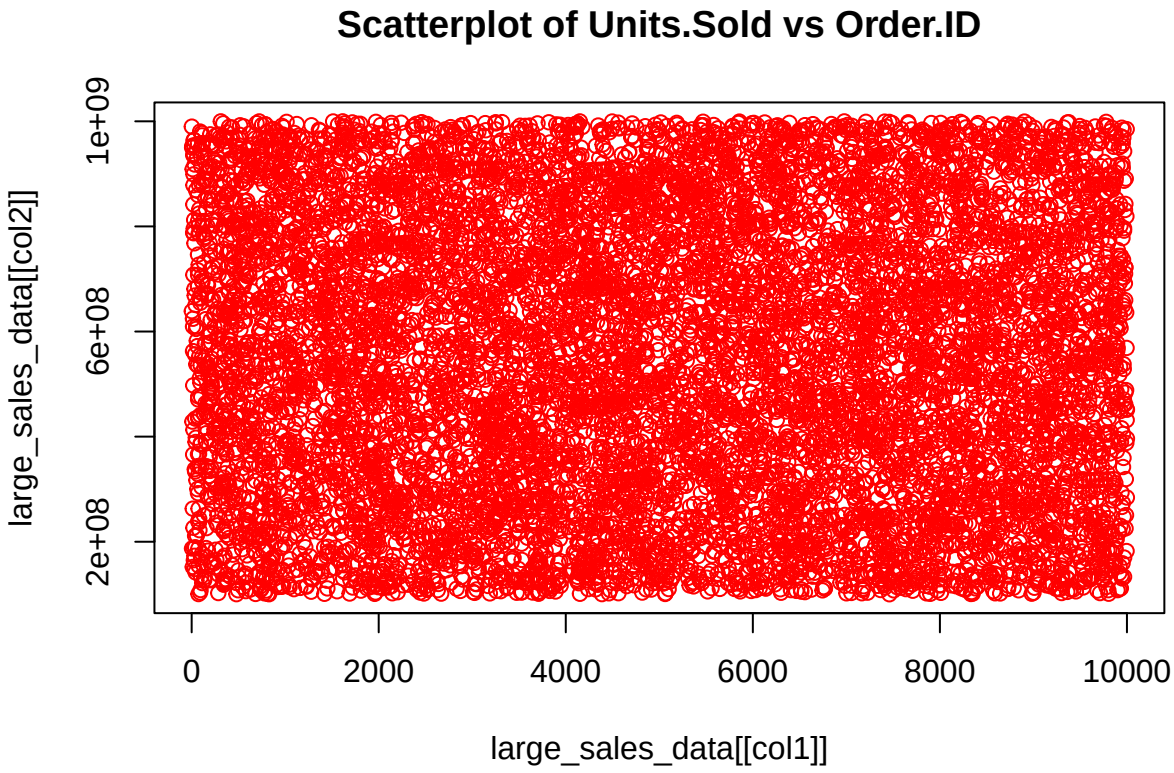


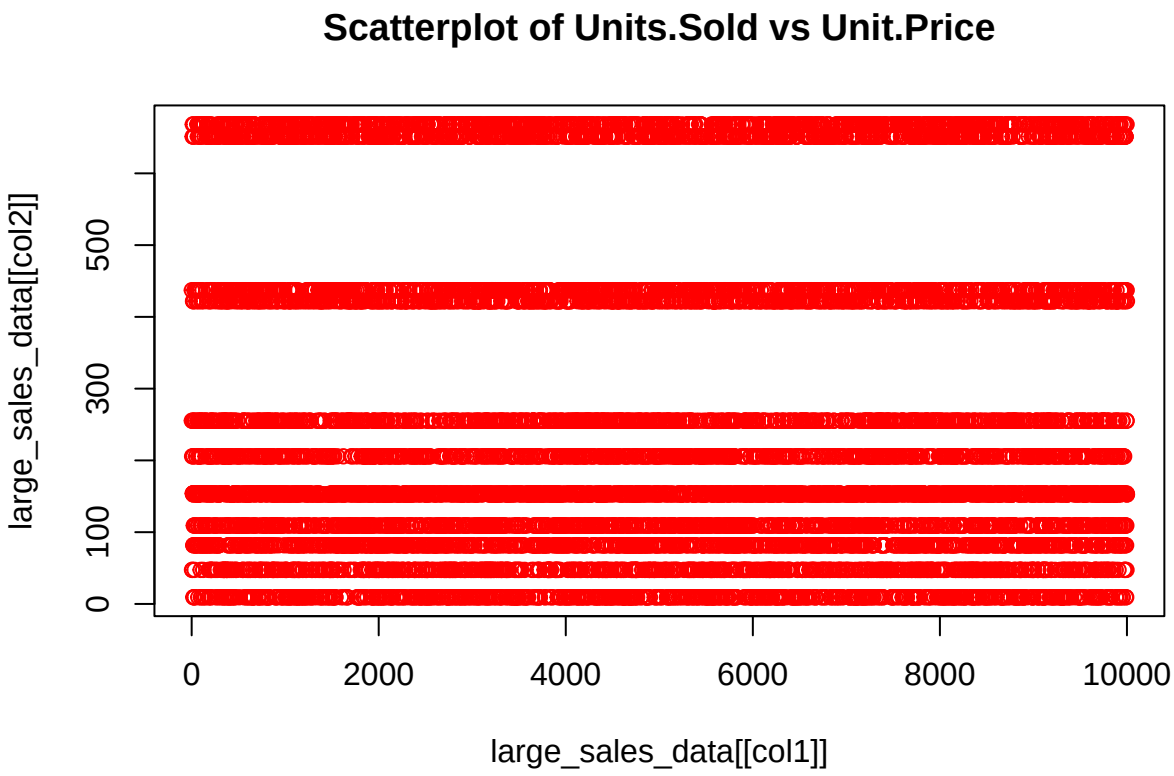

Scatterplot of Order.ID vs Unit.Cost
large_sales_data[[col2]]
0
100
200
300
400
500
2e+08
4e+08
6e+08
8e+08
1e+09
large_sales_data[[col1]]

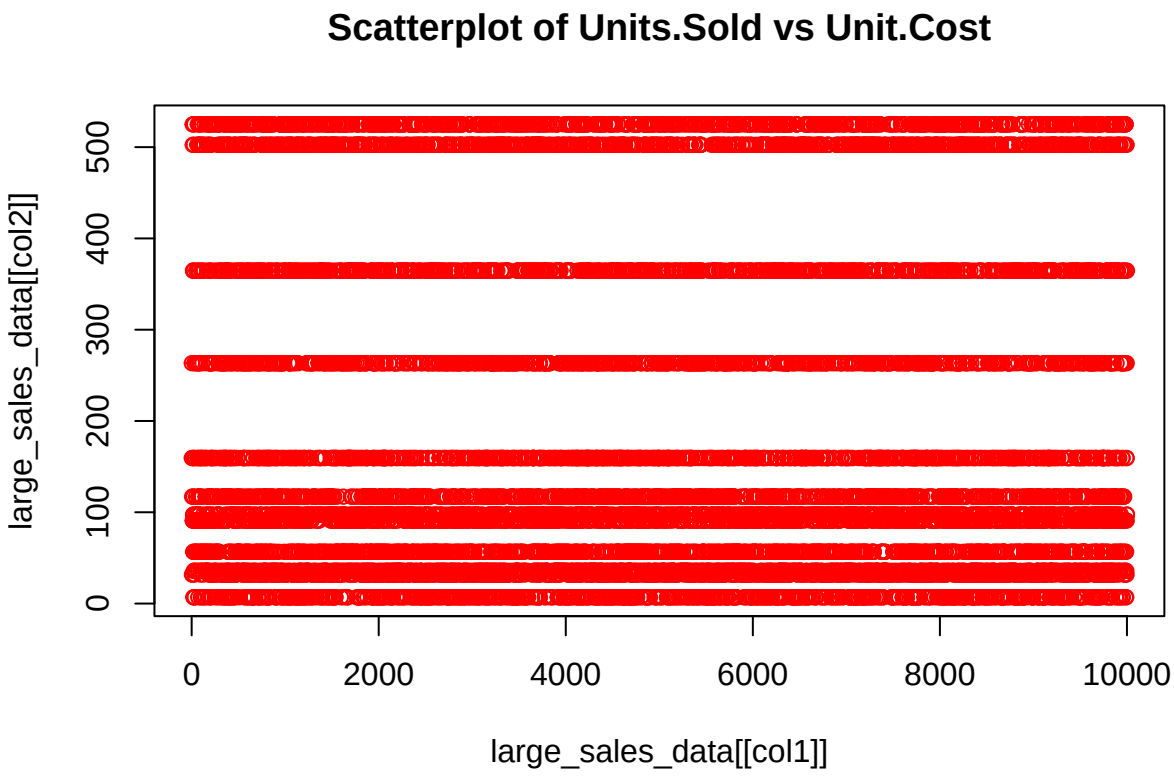

Scatterplot of Order.ID vs Total.Revenue
large_sales_data[[col2]]
0e+00
2e+06
4e+06
6e+06
2e+08
4e+08
6e+08
8e+08
1e+09
large_sales_data[[col1]]

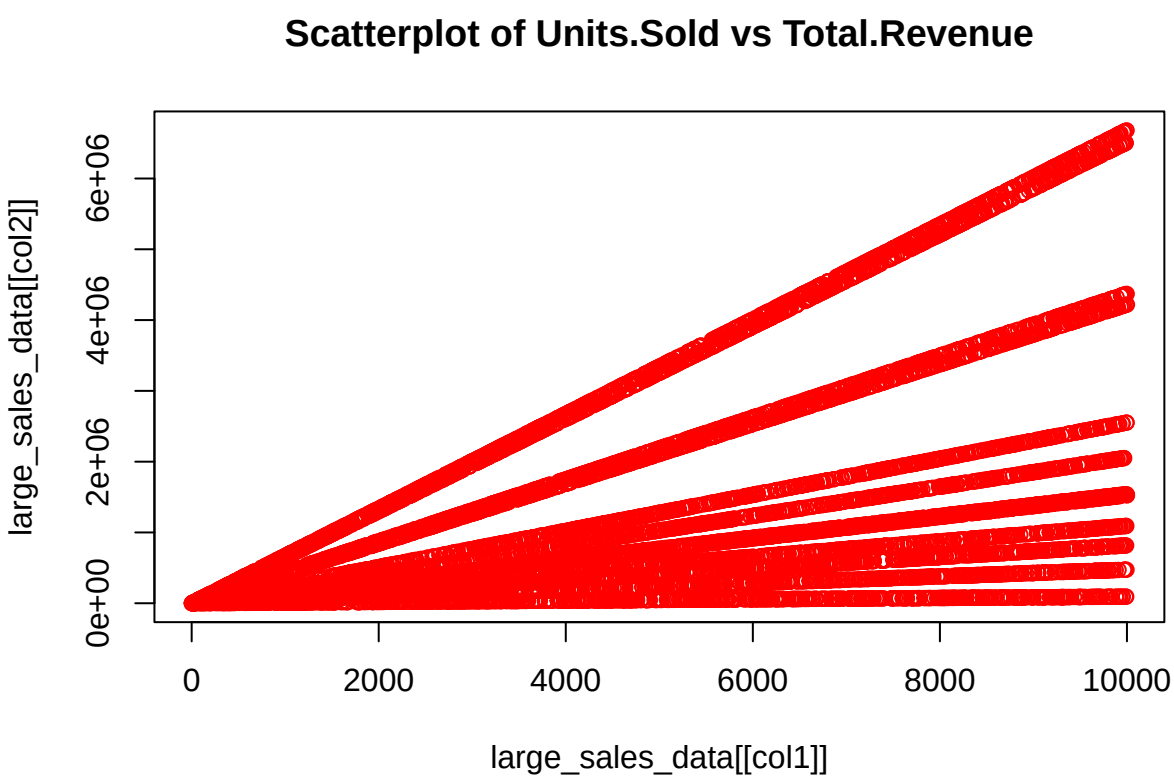

Scatterplot of Order.ID vs Total.Cost
large_sales_data[[col2]]
0e+00
2e+06
4e+06
2e+08
4e+08
6e+08
8e+08
1e+09
large_sales_data[[col1]]

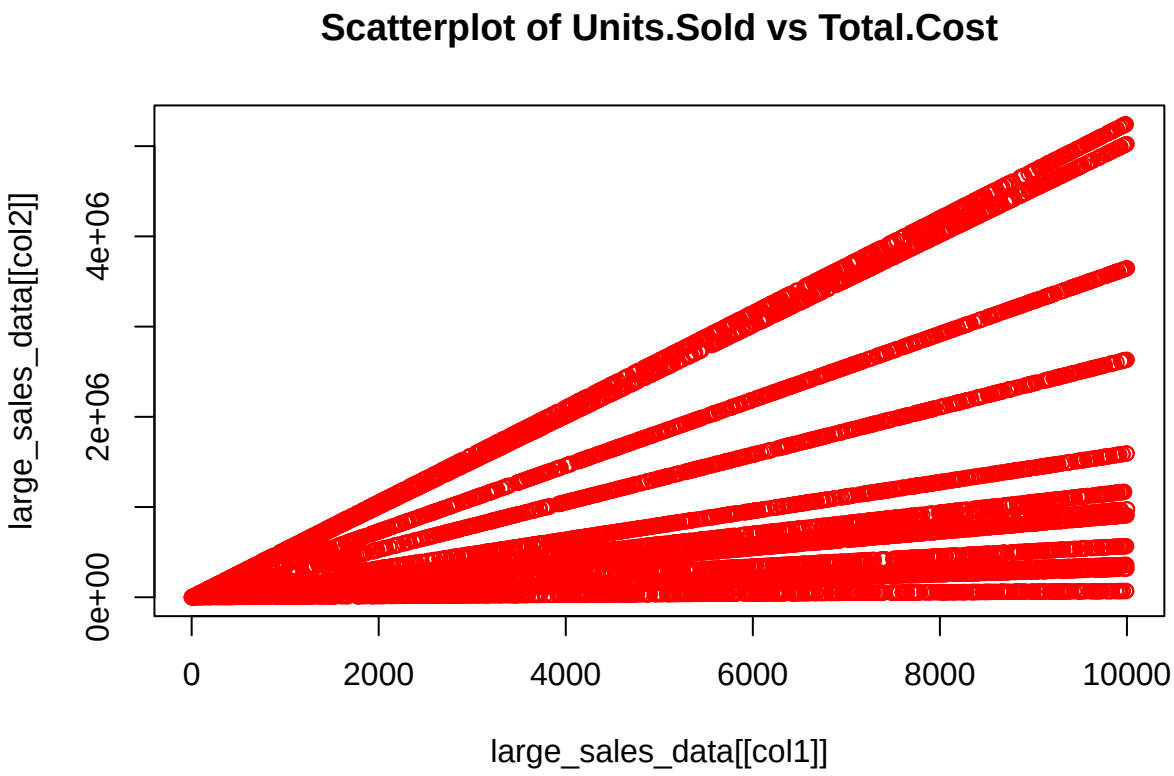

Scatterplot of Order.ID vs Total.Profit
large_sales_data[[col2]]
0
500000
1500000
2e+08
4e+08
6e+08
8e+08
1e+09
large_sales_data[[col1]]

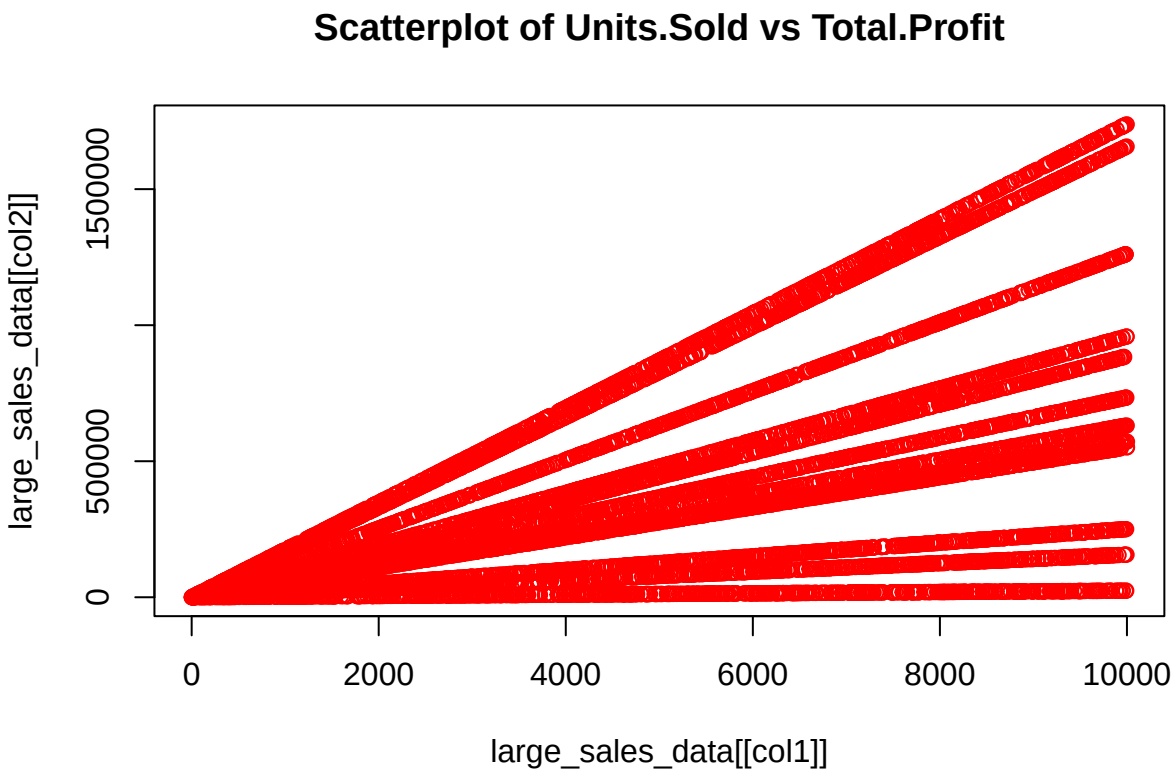

Scatterplot of Units.Sold vs Order.ID
large_sales_data[[col2]]
1e+09
6e+08
2e+08
0
2000
4000
6000
8000
10000
large_sales_data[[col1]]

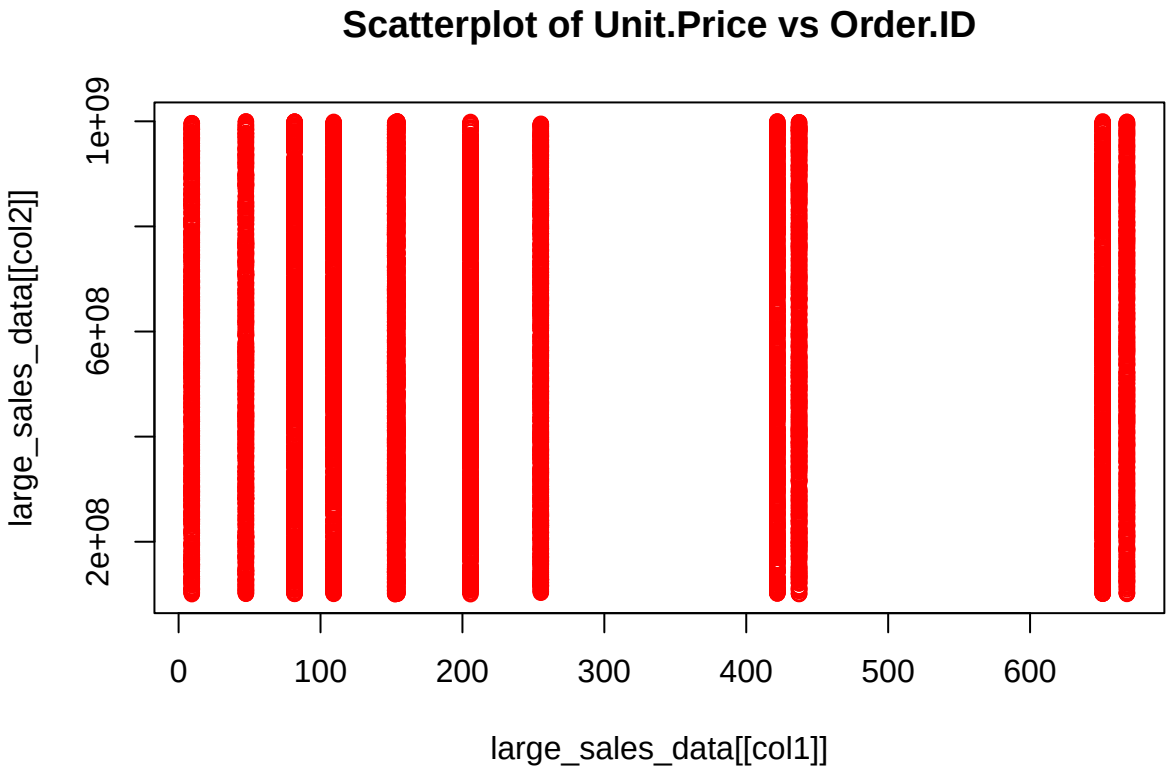

Scatterplot of Units.Sold vs Unit.Price
large_sales_data[[col2]]
0
100
300
500
0
2000
4000
6000
8000
10000
large_sales_data[[col1]]

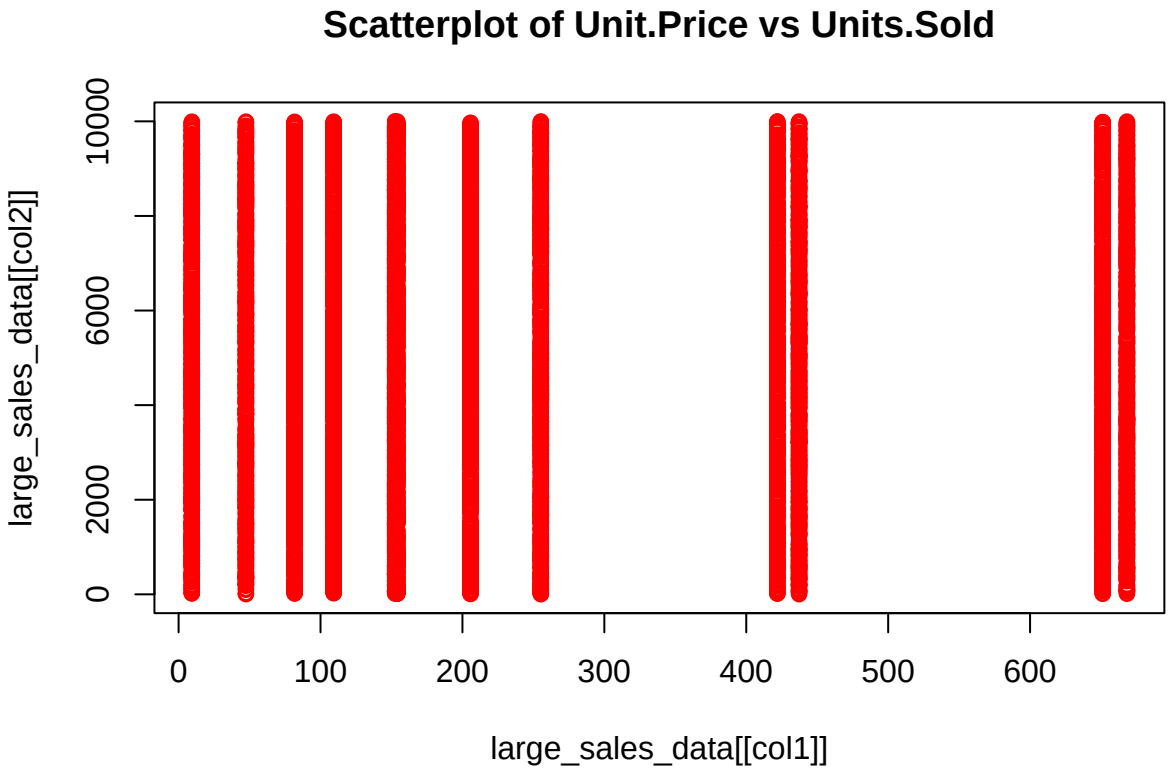

Scatterplot of Units.Sold vs Unit.Cost
large_sales_data[[col2]]
0
100
200
300
400
500
0
2000
4000
6000
8000
10000
large_sales_data[[col1]]

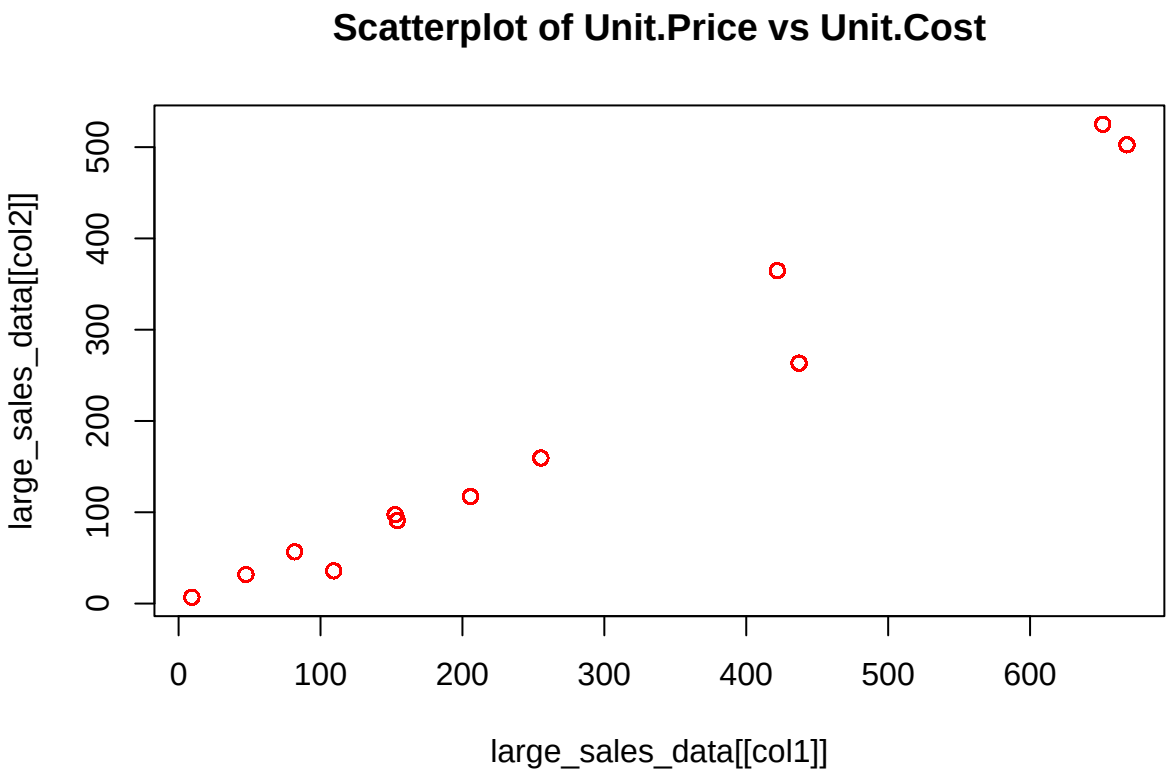

Scatterplot of Units.Sold vs Total.Revenue
large_sales_data[[col2]]
0e+00
2e+06
4e+06
6e+06
0
2000
4000
6000
8000
10000
large_sales_data[[col1]]

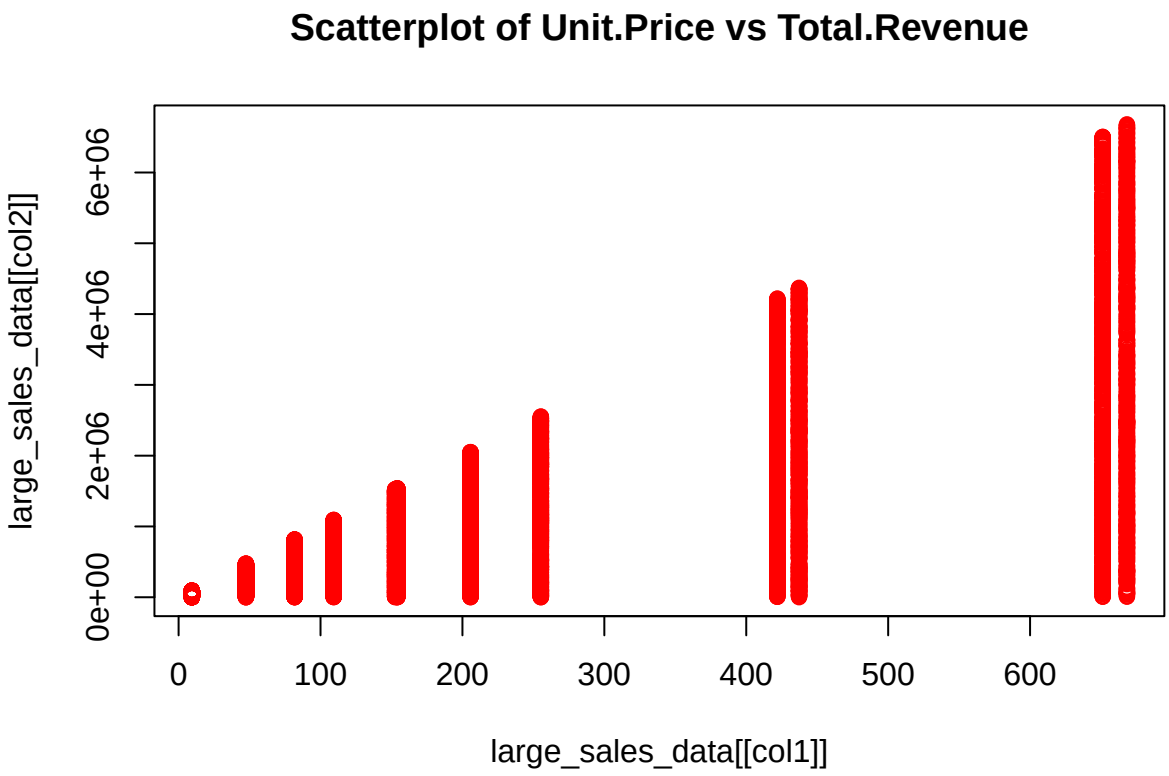

Scatterplot of Units.Sold vs Total.Cost
large_sales_data[[col2]]
0e+00
2e+06
4e+06
0
2000
4000
6000
8000
10000
large_sales_data[[col1]]

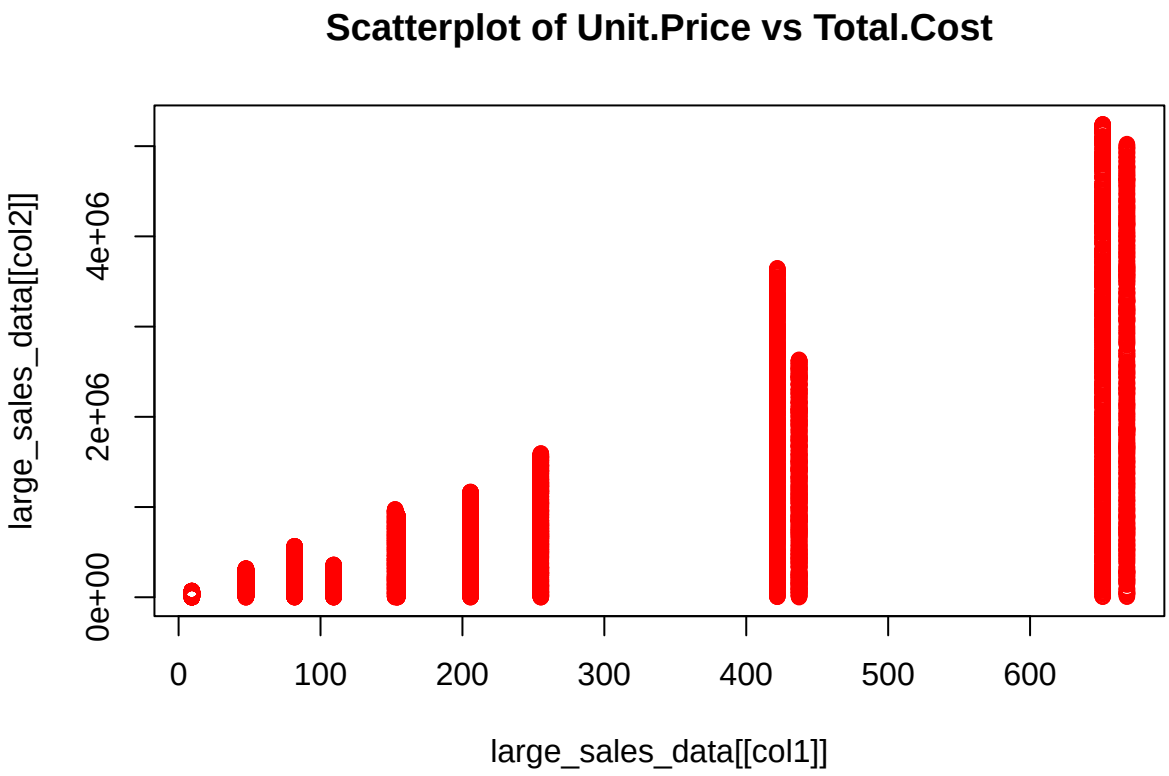

Scatterplot of Units.Sold vs Total.Profit
large_sales_data[[col2]]
0
500000
1500000
0
2000
4000
6000
8000
10000
large_sales_data[[col1]]

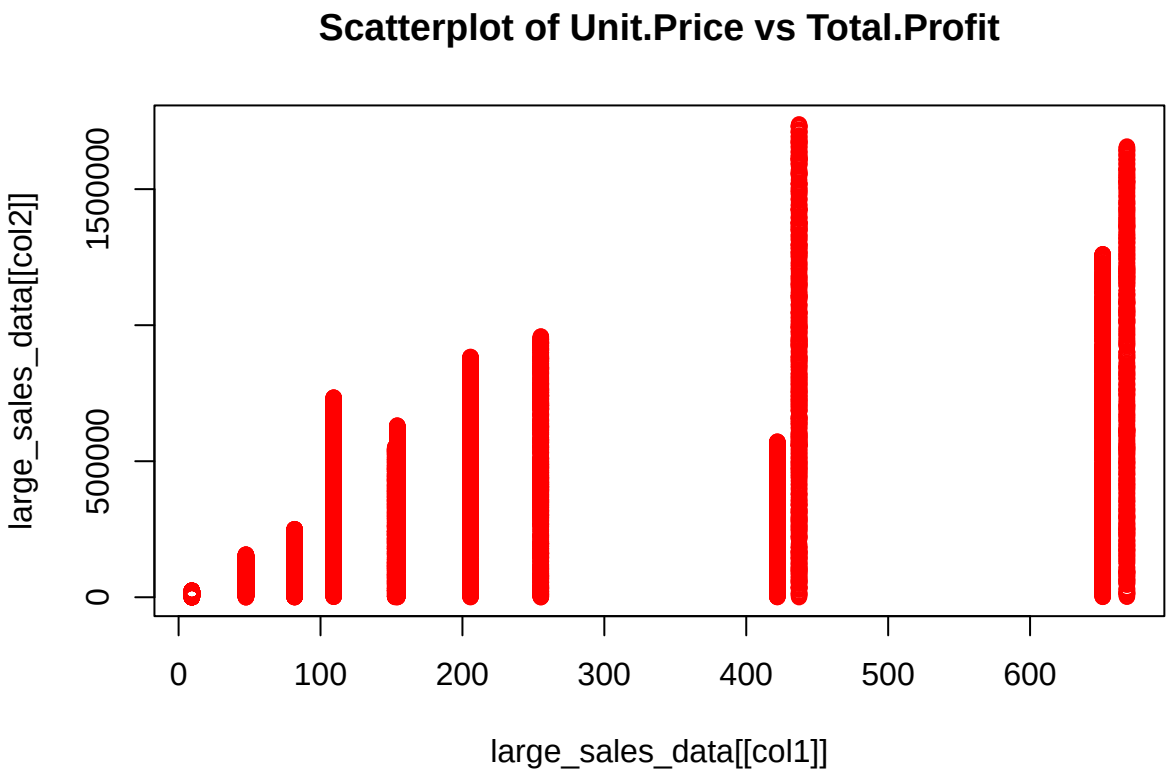

Scatterplot of Unit.Price vs Order.ID
large_sales_data[[col2]]
2e+08
6e+08
1e+09
0
100
200
300
400
500
600
large_sales_data[[col1]]

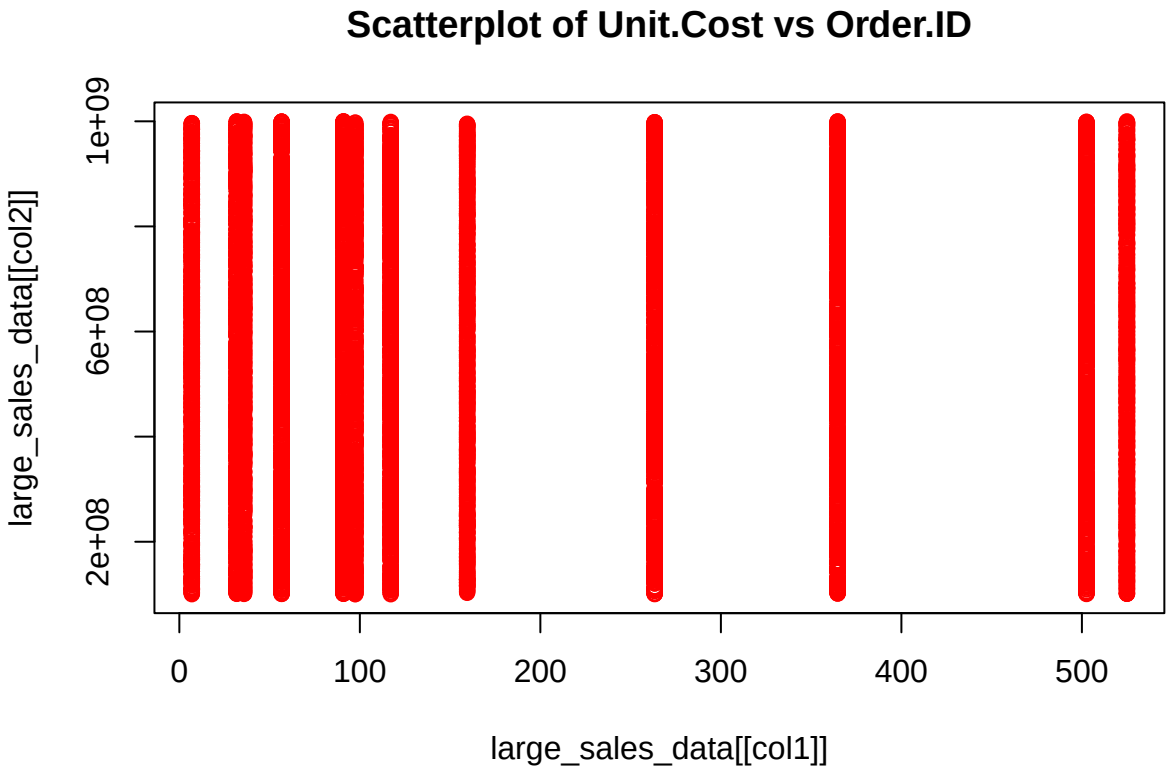

Scatterplot of Unit.Price vs Units.Sold
large_sales_data[[col2]]
0
2000
6000
10000
0
100
200
300
400
500
600
large_sales_data[[col1]]

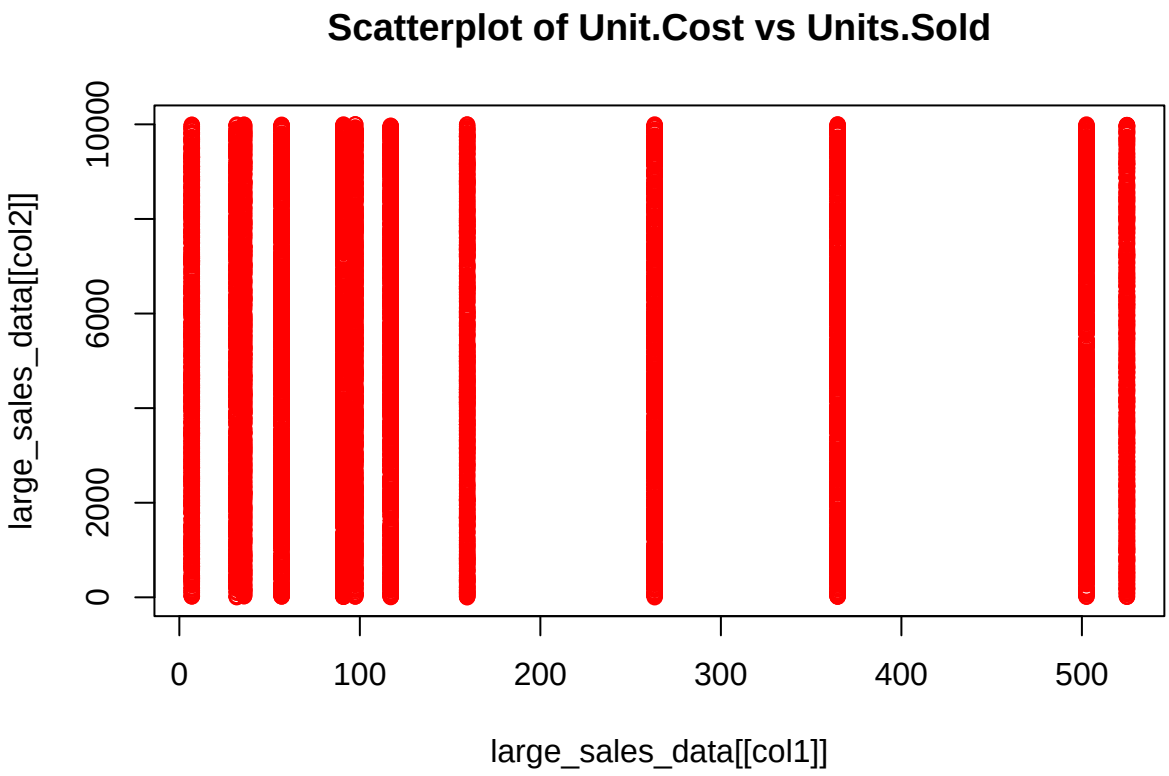

Scatterplot of Unit.Price vs Unit.Cost
large_sales_data[[col2]]
0
100
200
300
400
500
large_sales_data[[col1]]
0
100
200
300
400
500
600

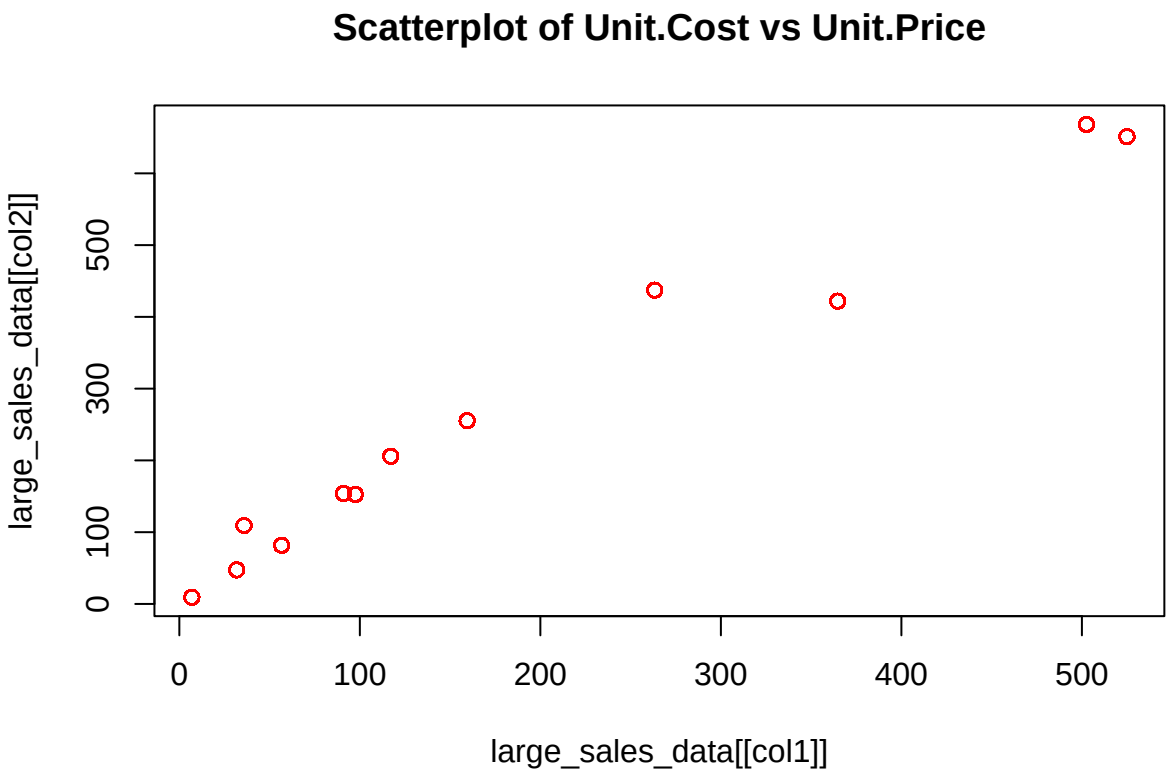

Scatterplot of Unit.Price vs Total.Revenue
large_sales_data[[col2]]
0e+00
2e+06
4e+06
6e+06
0
100
200
300
400
500
600
large_sales_data[[col1]]

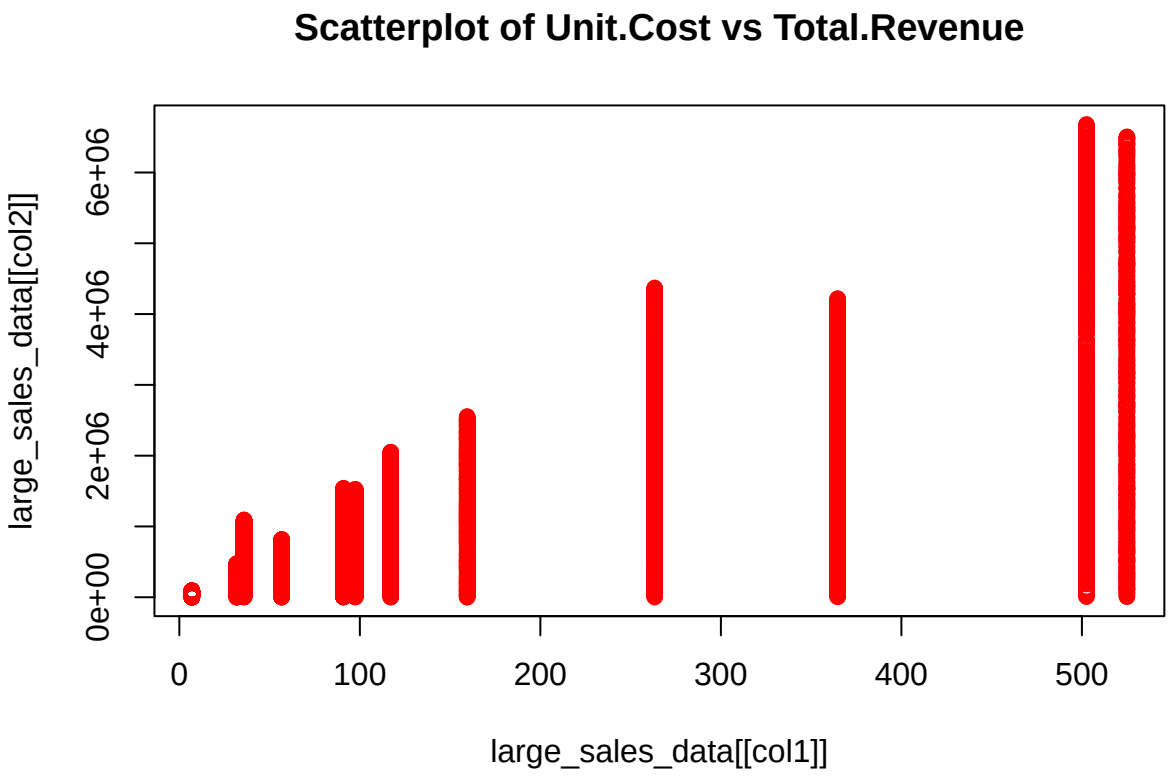

Scatterplot of Unit.Price vs Total.Cost
large_sales_data[[col2]]
0e+00
2e+06
4e+06
0
100
200
300
400
500
600
large_sales_data[[col1]]

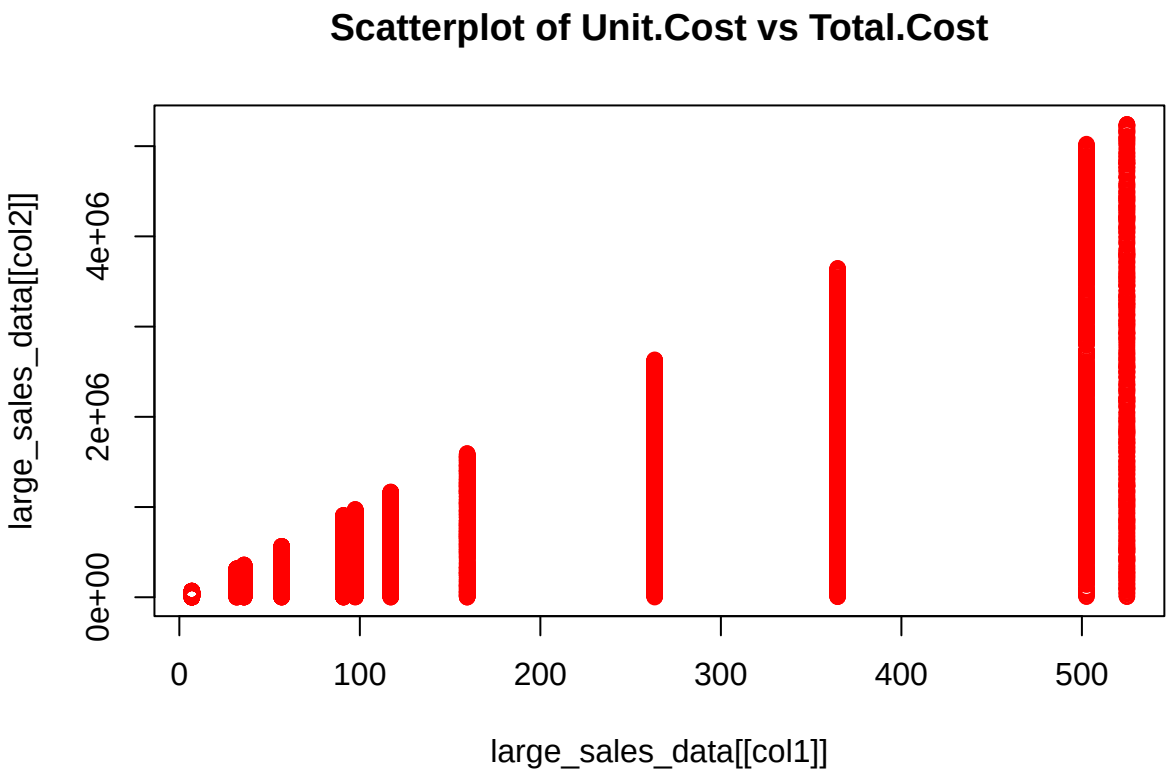

Scatterplot of Unit.Price vs Total.Profit
large_sales_data[[col2]]
0
500000
1500000
0
100
200
300
400
500
600
large_sales_data[[col1]]

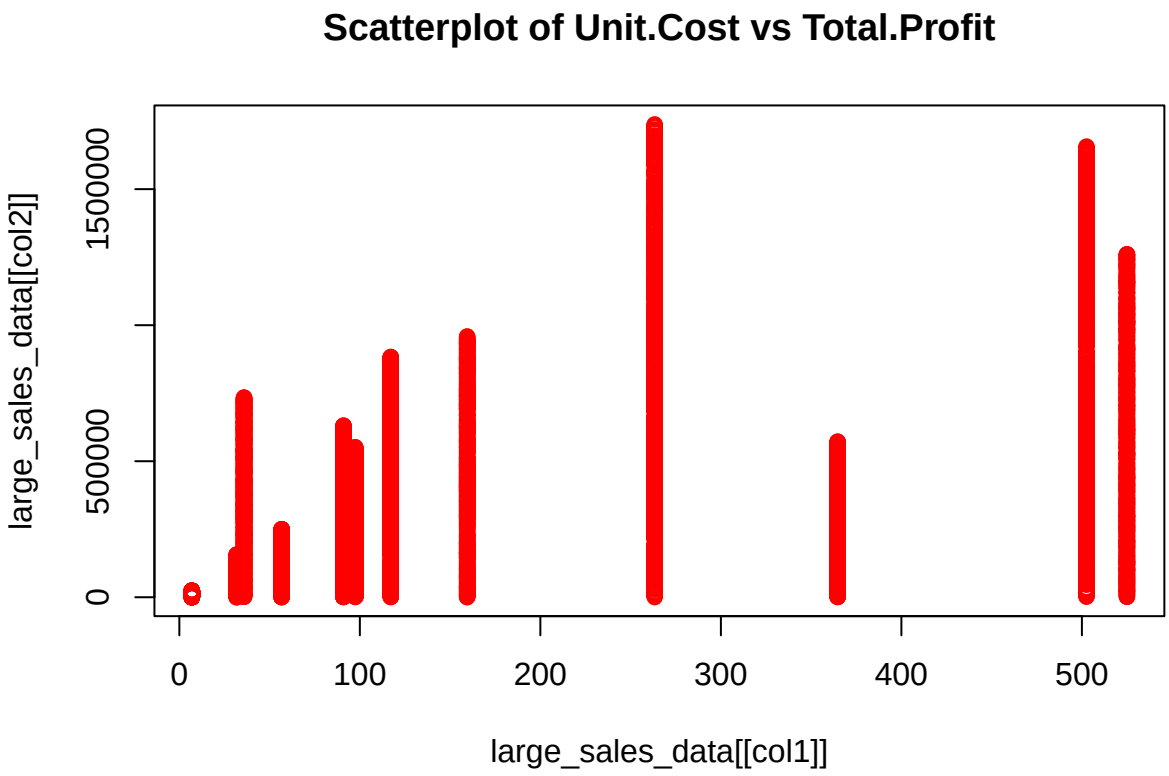

Scatterplot of Unit.Cost vs Order.ID
large_sales_data[[col2]]
1e+09
6e+08
2e+08
0
100
200
300
400
500
large_sales_data[[col1]]

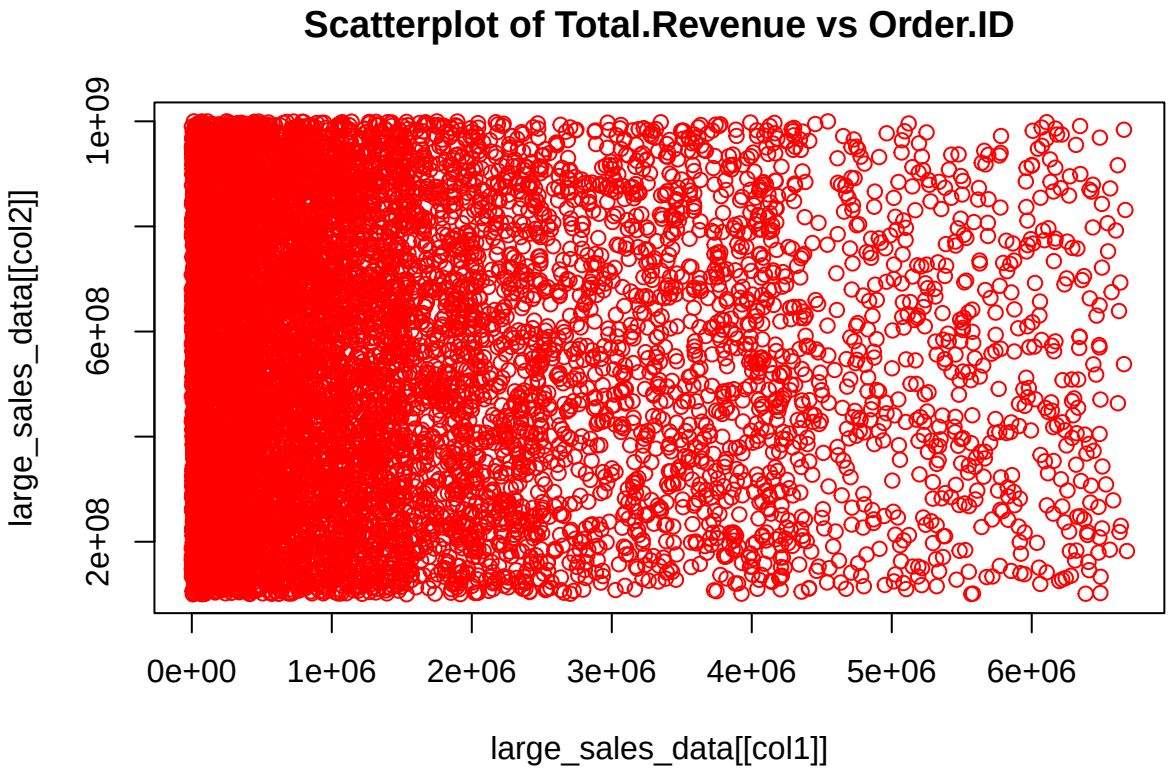

Scatterplot of Unit.Cost vs Units.Sold
large_sales_data[[col2]]
0
2000
6000
10000
0
100
200
300
400
500
large_sales_data[[col1]]

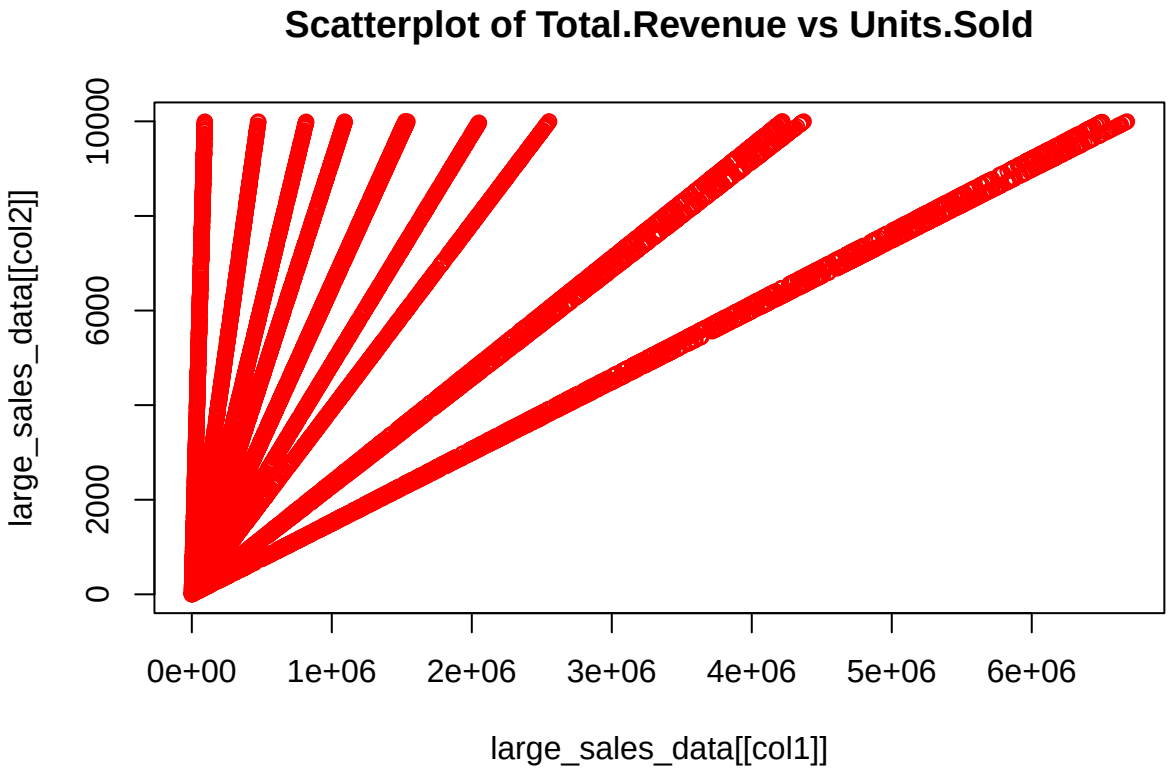

Scatterplot of Unit.Cost vs Unit.Price
large_sales_data[[col2]]
0
100
300
500
0
100
200
300
400
500
large_sales_data[[col1]]

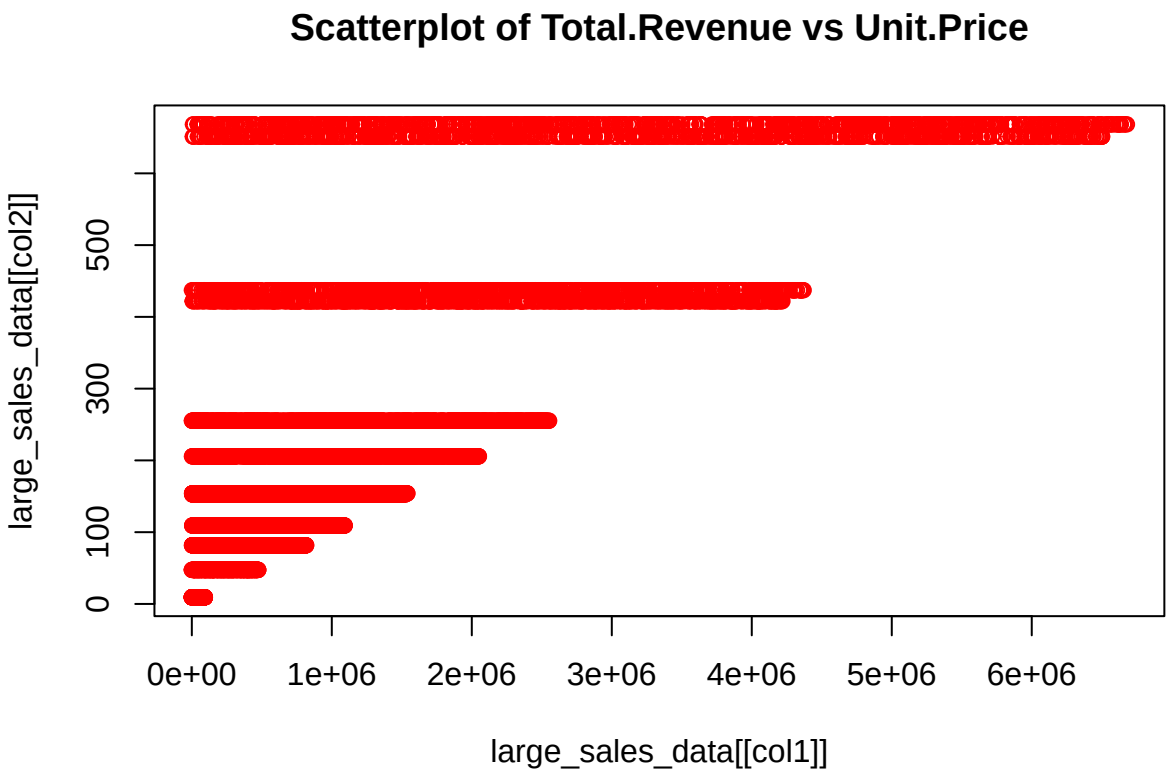

Scatterplot of Unit.Cost vs Total.Revenue
large_sales_data[[col2]]
0e+00
2e+06
4e+06
6e+06
0
100
200
300
400
500
large_sales_data[[col1]]

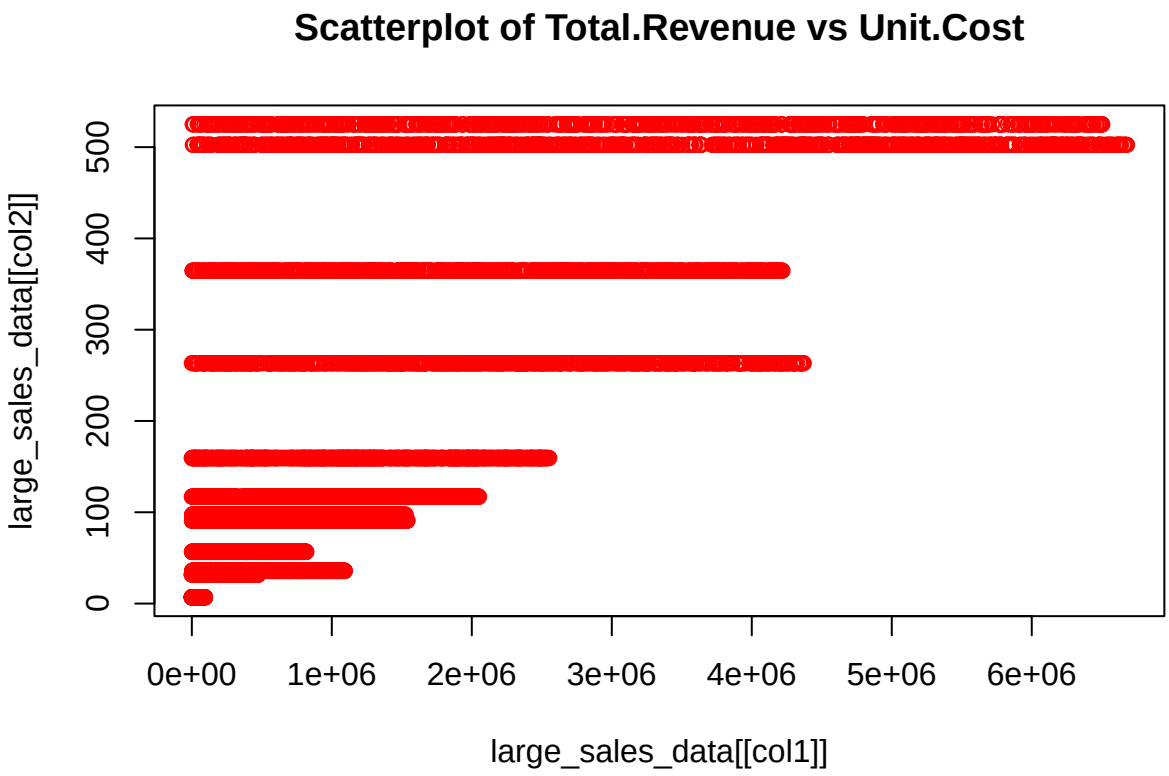

Scatterplot of Unit.Cost vs Total.Cost
large_sales_data[[col2]]
0e+00
2e+06
4e+06
0
100
200
300
400
500
large_sales_data[[col1]]

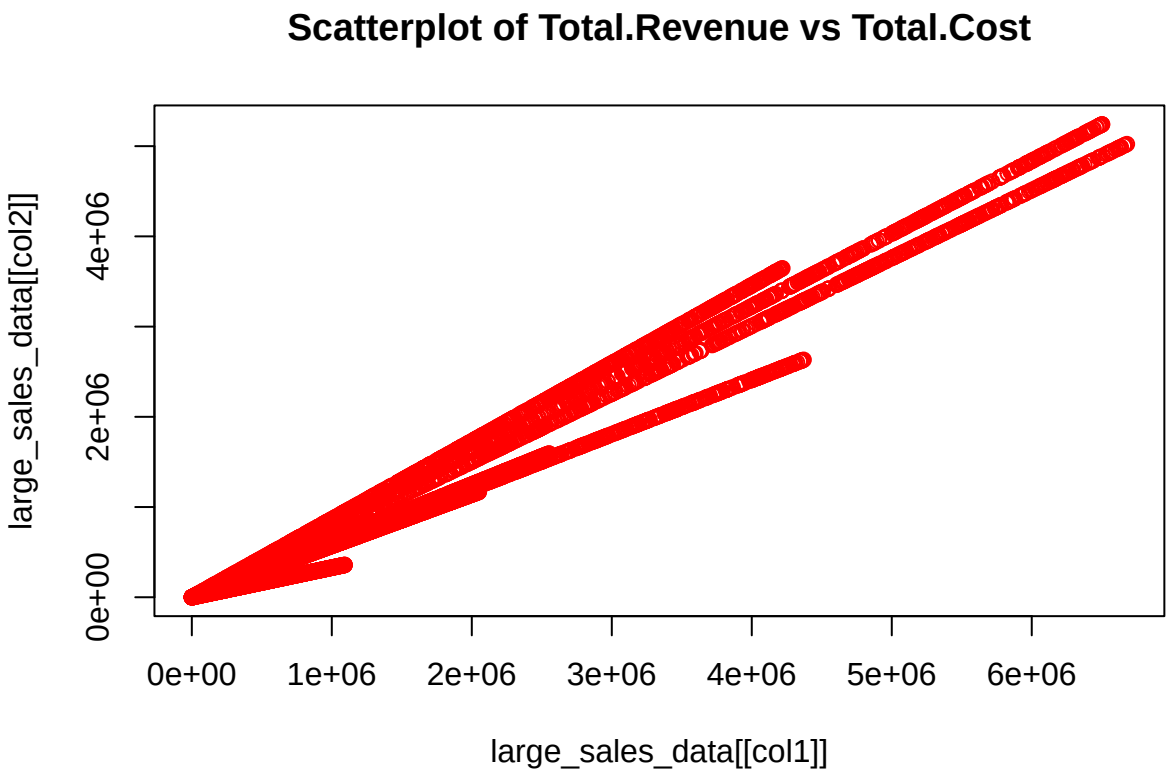

Scatterplot of Unit.Cost vs Total.Profit
large_sales_data[[col2]]
0
500000
1500000
0
100
200
300
400
500
large_sales_data[[col1]]

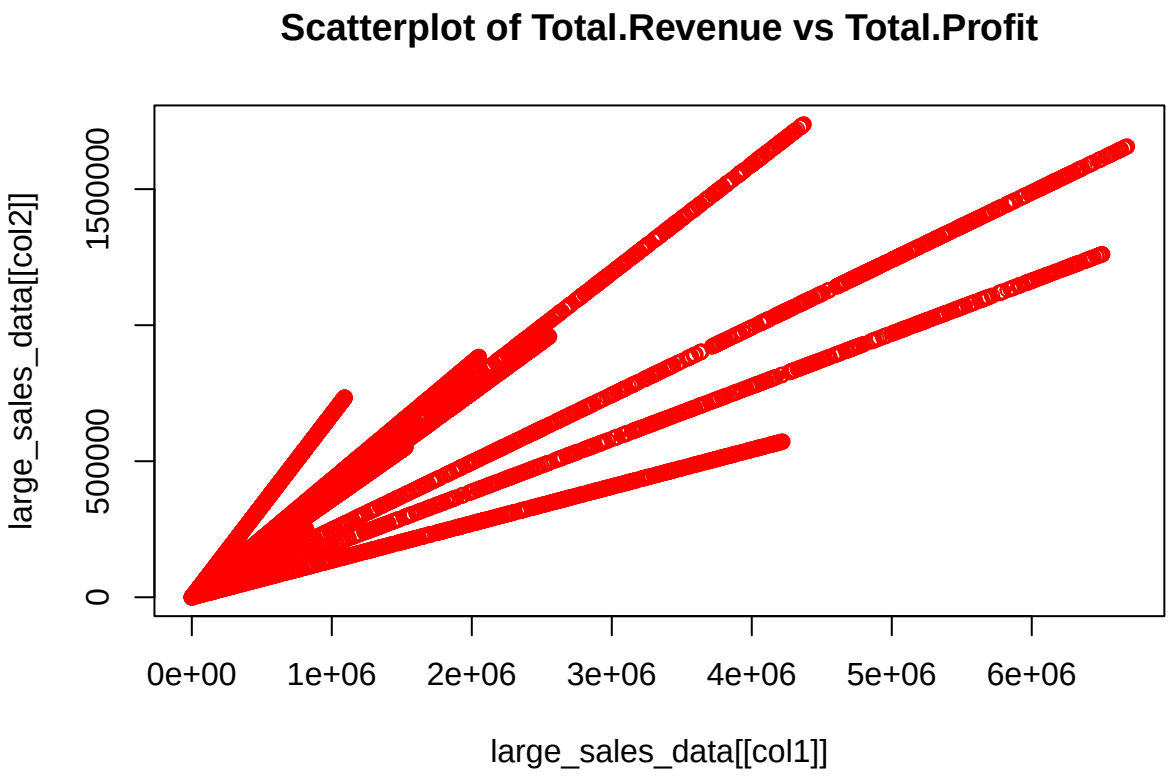

Scatterplot of Total.Revenue vs Order.ID
large_sales_data[[col2]]
1e+09
6e+08
2e+08
0e+00
1e+06
2e+06
3e+06
4e+06
5e+06
6e+06
large_sales_data[[col1]]

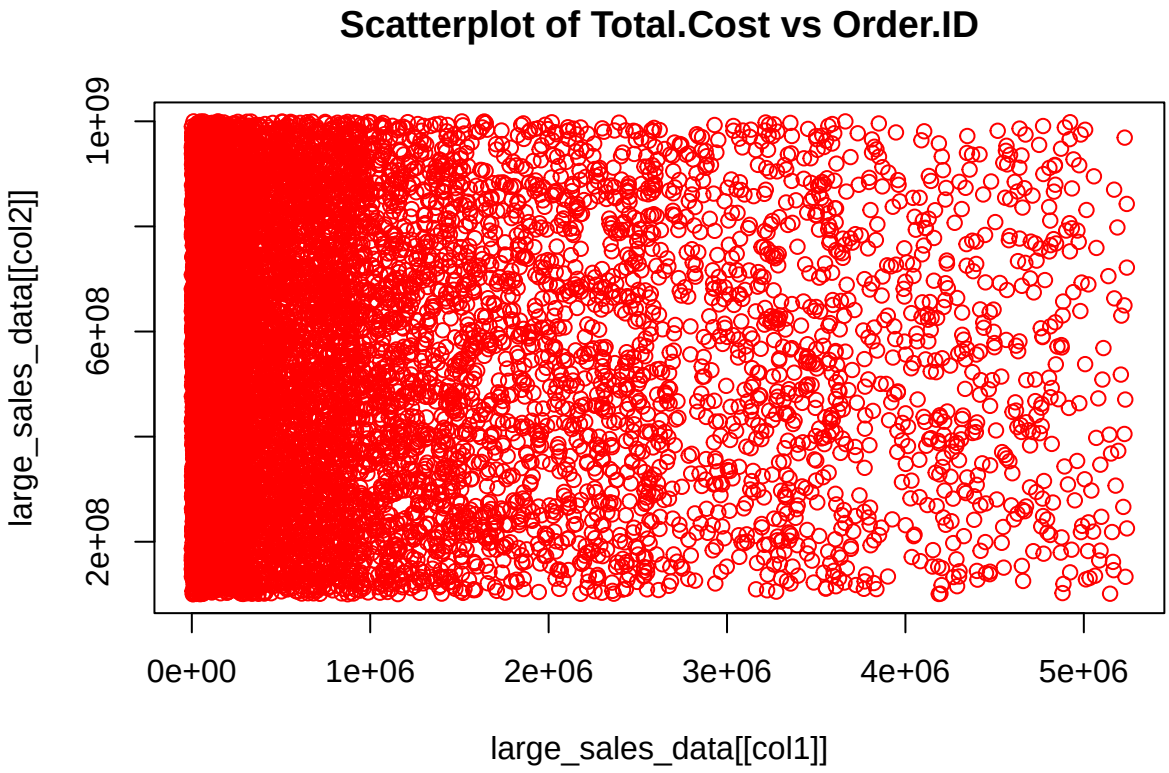

Scatterplot of Total.Revenue vs Units.Sold
large_sales_data[[col2]]
0
2000
6000
10000
0e+00
1e+06
2e+06
3e+06
4e+06
5e+06
6e+06
large_sales_data[[col1]]

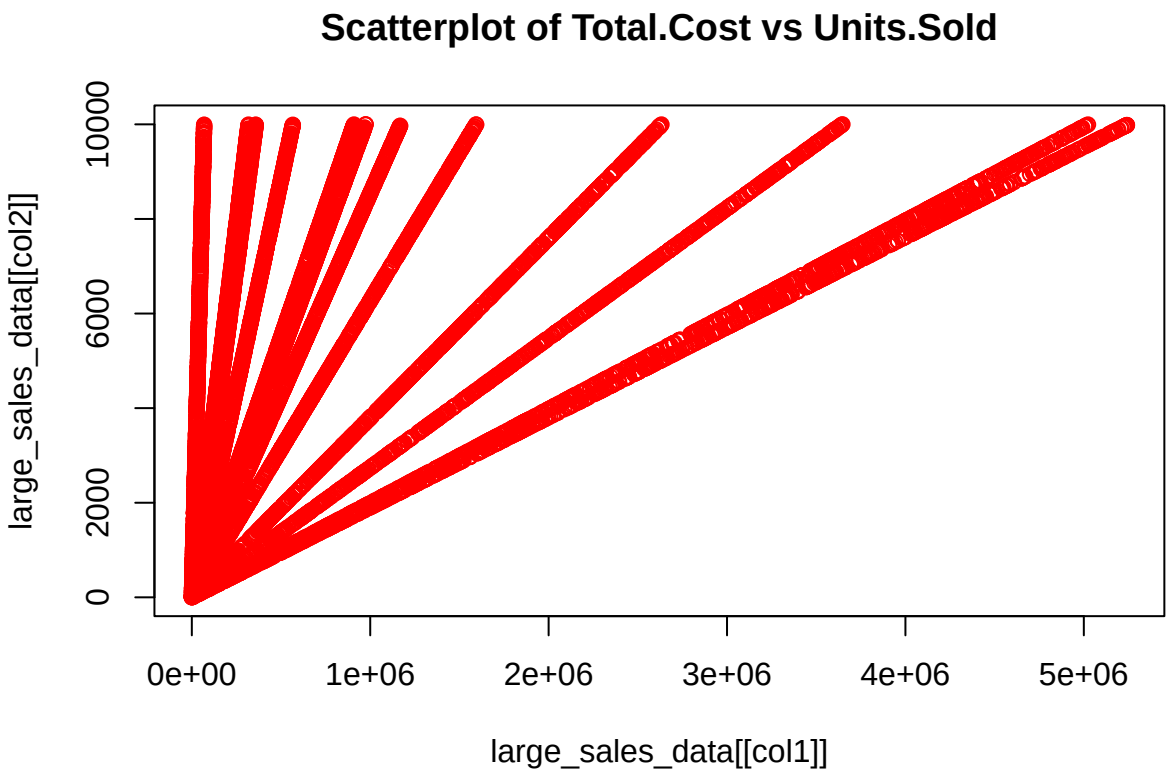

Scatterplot of Total.Revenue vs Unit.Price
large_sales_data[[col2]]
0
100
300
500
0e+00
1e+06
2e+06
3e+06
4e+06
5e+06
6e+06
large_sales_data[[col1]]

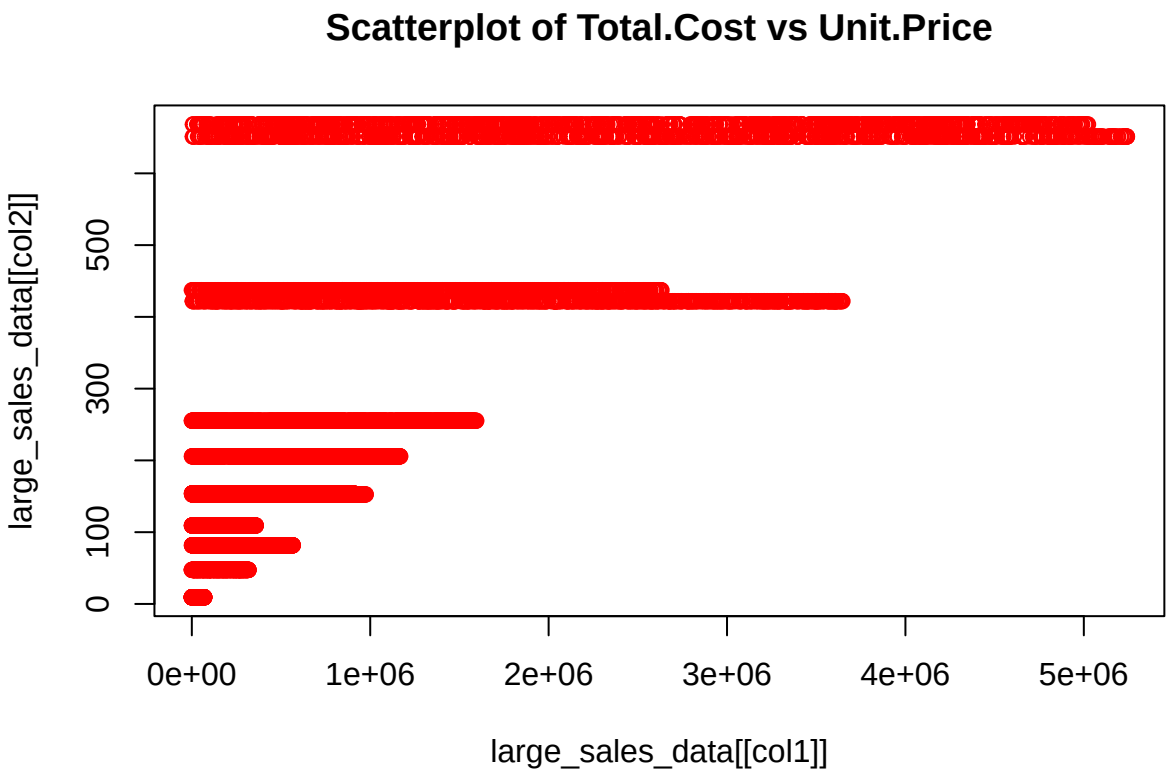

Scatterplot of Total.Revenue vs Unit.Cost
large_sales_data[[col2]]
0
100
200
300
400
500
0e+00
1e+06
2e+06
3e+06
4e+06
5e+06
6e+06
large_sales_data[[col1]]

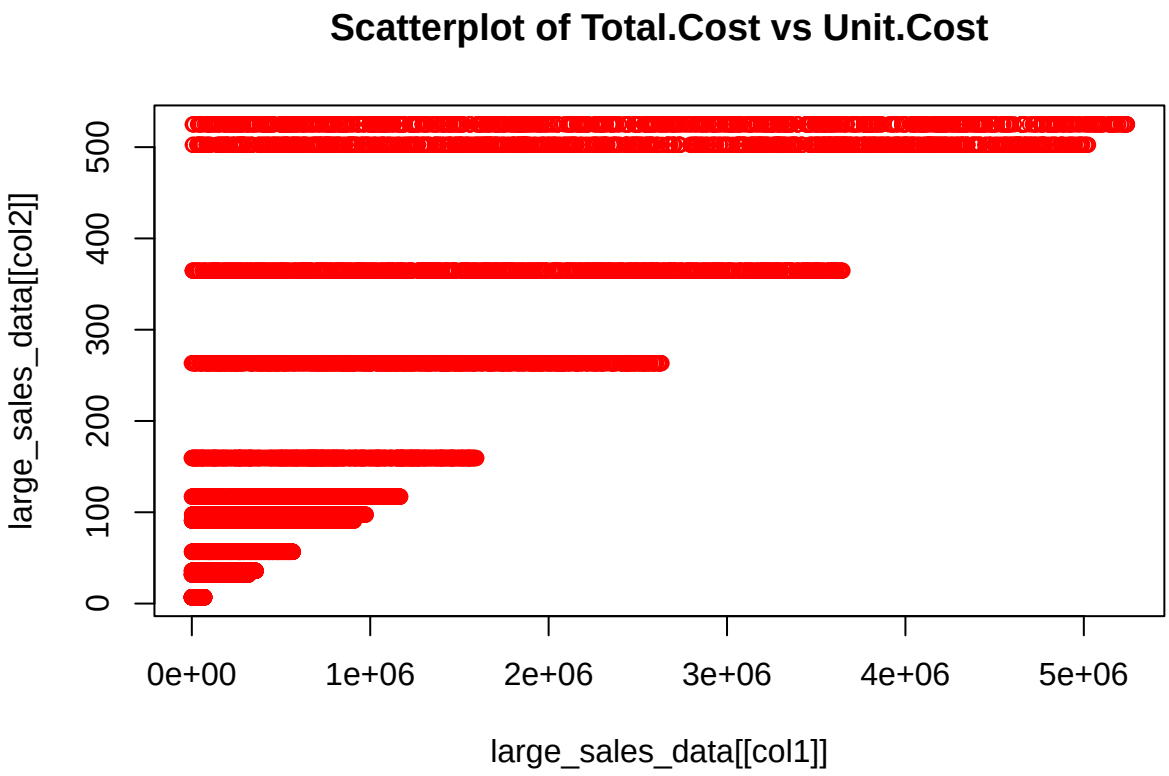

Scatterplot of Total.Revenue vs Total.Cost
large_sales_data[[col2]]
0e+00
2e+06
4e+06
0e+00
1e+06
2e+06
3e+06
4e+06
5e+06
6e+06
large_sales_data[[col1]]

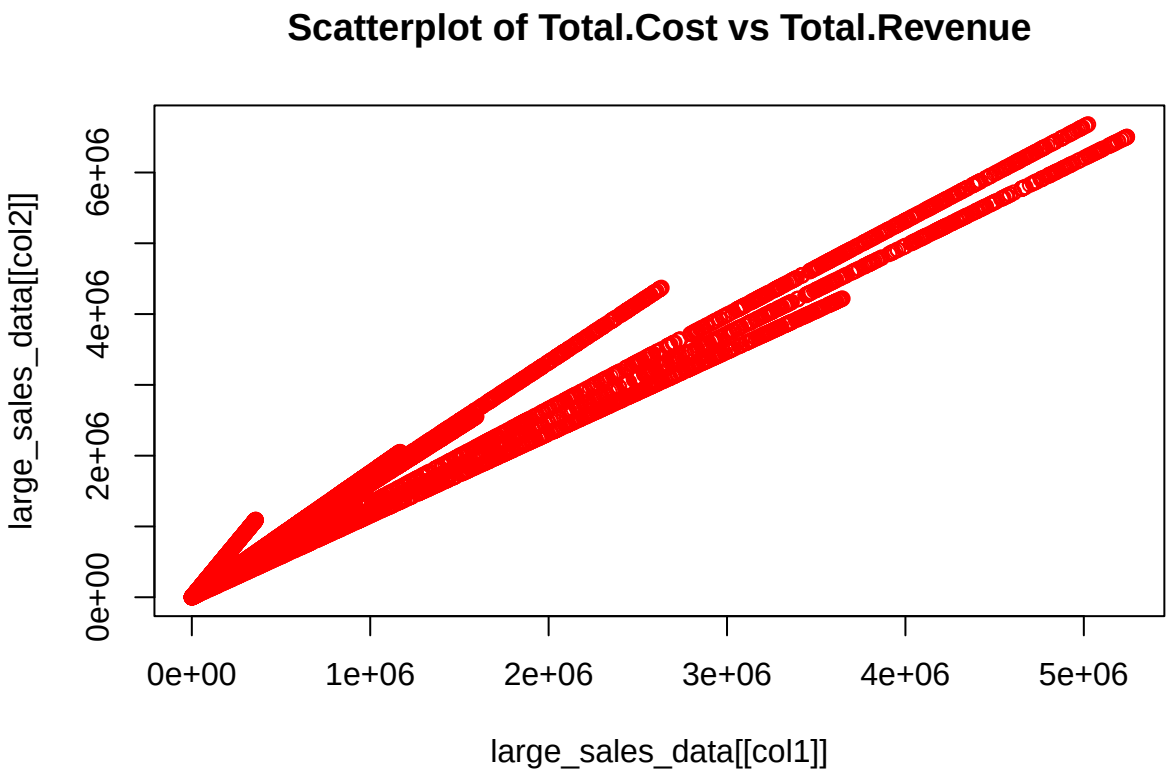

Scatterplot of Total.Revenue vs Total.Profit
large_sales_data[[col2]]
0
500000
1500000
0e+00
1e+06
2e+06
3e+06
4e+06
5e+06
6e+06
large_sales_data[[col1]]

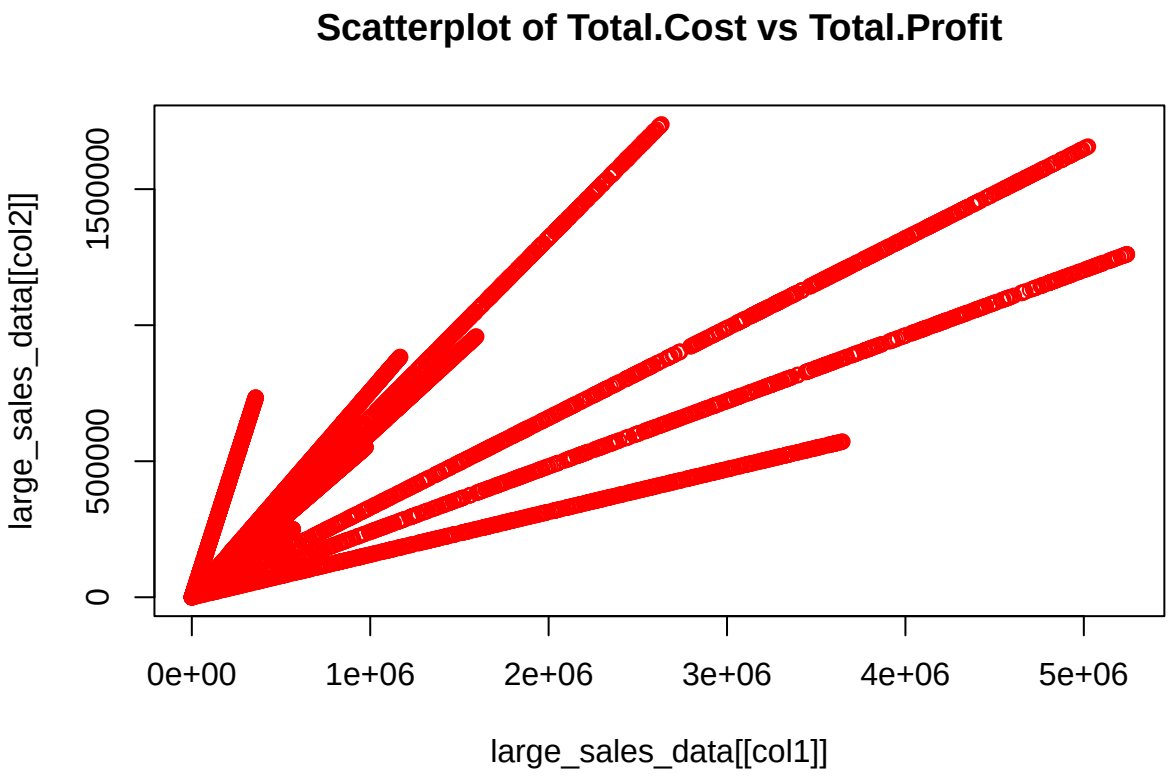

Scatterplot of Total.Cost vs Order.ID
large_sales_data[[col2]]
1e+09
6e+08
2e+08
0e+00
1e+06
2e+06
3e+06
4e+06
5e+06
large_sales_data[[col1]]

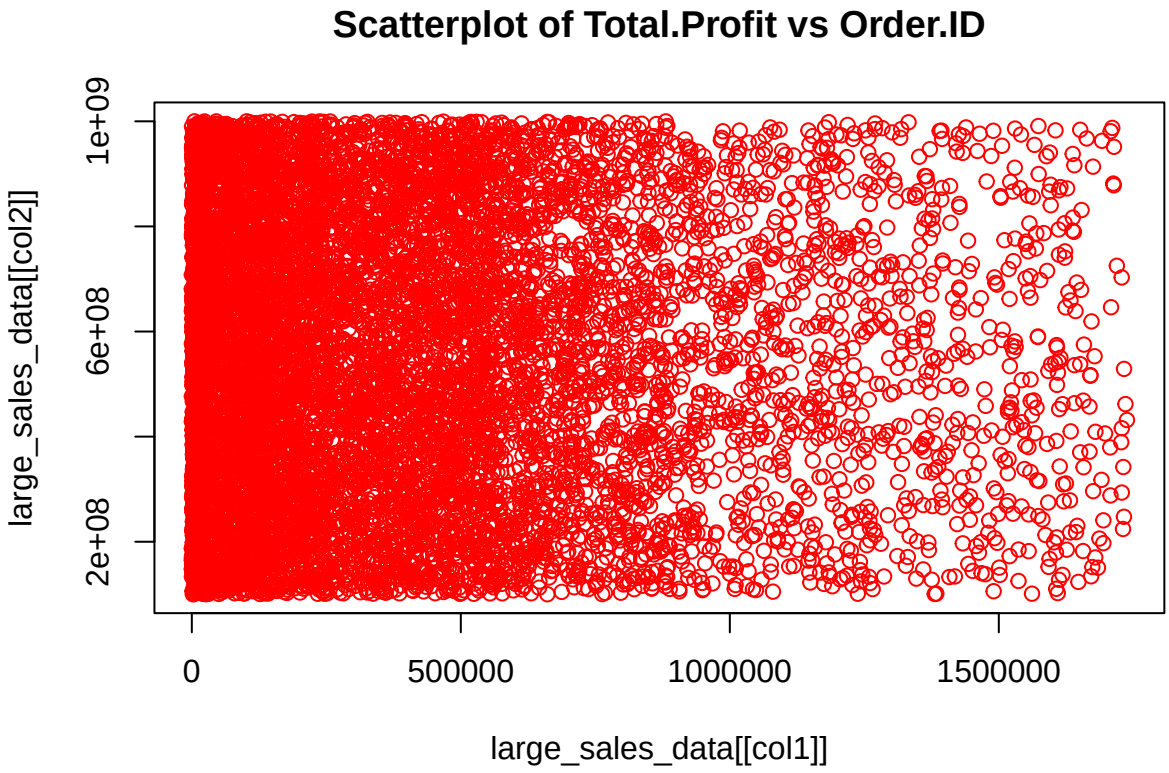

Scatterplot of Total.Cost vs Units.Sold
large_sales_data[[col2]]
0
2000
6000
10000
0e+00
1e+06
2e+06
3e+06
4e+06
5e+06
large_sales_data[[col1]]

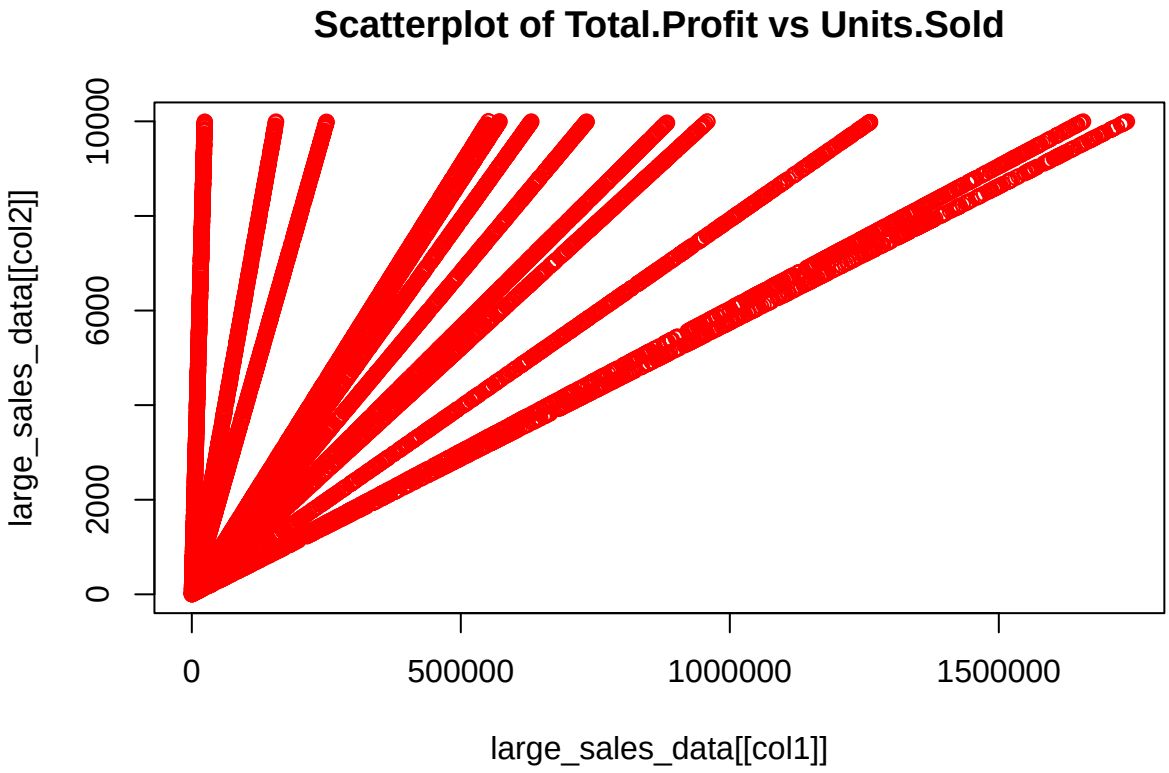

Scatterplot of Total.Cost vs Unit.Price
large_sales_data[[col2]]
0
100
300
500
0e+00
1e+06
2e+06
3e+06
4e+06
5e+06
large_sales_data[[col1]]

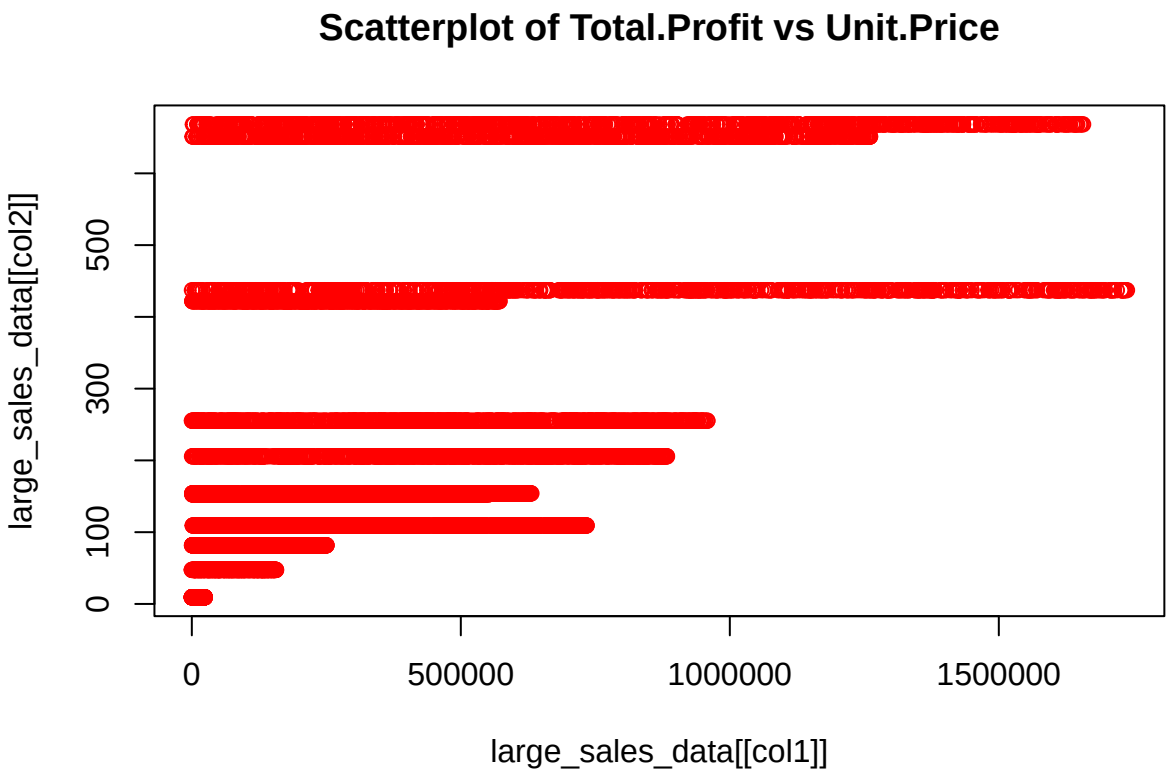

Scatterplot of Total.Cost vs Unit.Cost
large_sales_data[[col2]]
0
100
200
300
400
500
0e+00
1e+06
2e+06
3e+06
4e+06
5e+06
large_sales_data[[col1]]

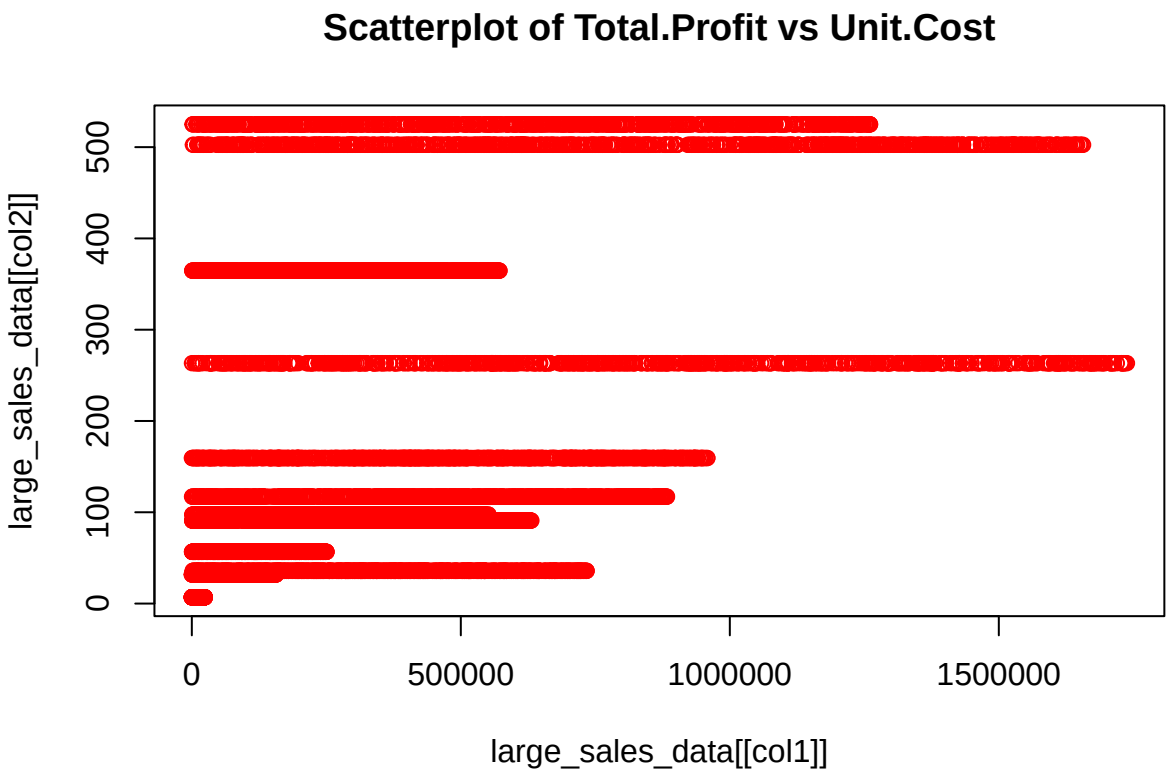

Scatterplot of Total.Cost vs Total.Revenue
large_sales_data[[col2]]
0e+00
2e+06
4e+06
6e+06
0e+00
1e+06
2e+06
3e+06
4e+06
5e+06
large_sales_data[[col1]]

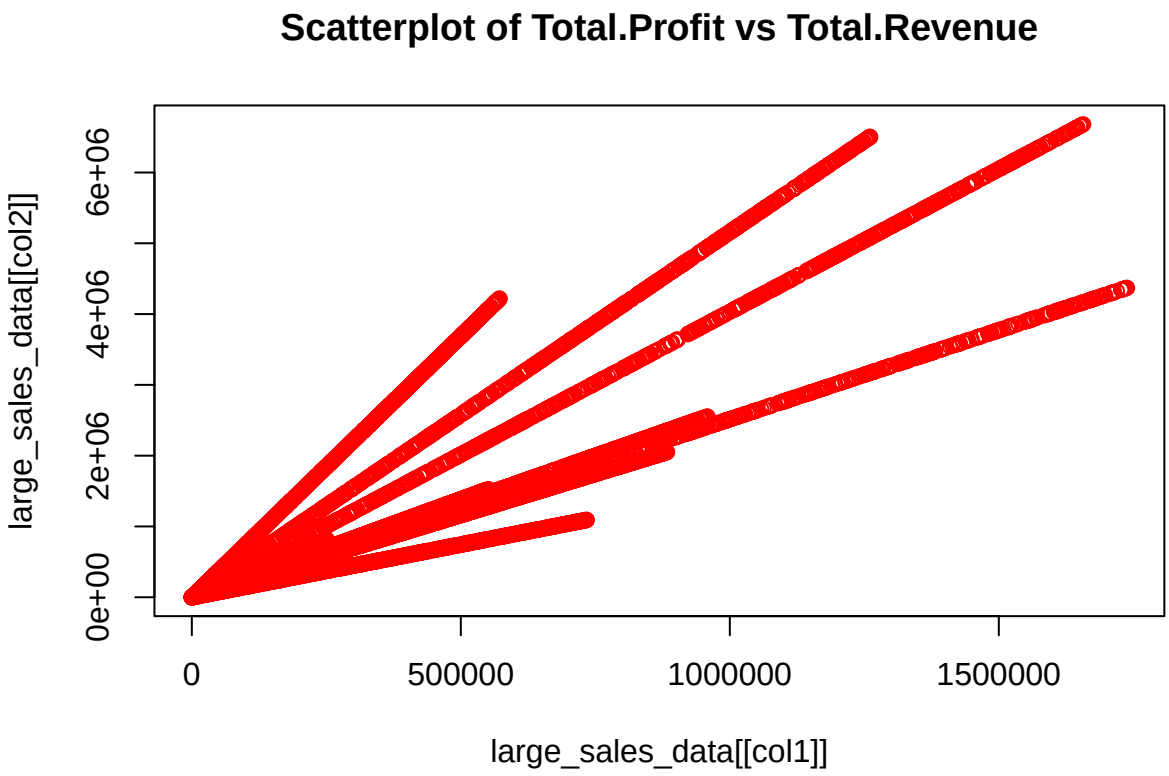

Scatterplot of Total.Cost vs Total.Profit
large_sales_data[[col2]]
0
500000
1500000
0e+00
1e+06
2e+06
3e+06
4e+06
5e+06
large_sales_data[[col1]]

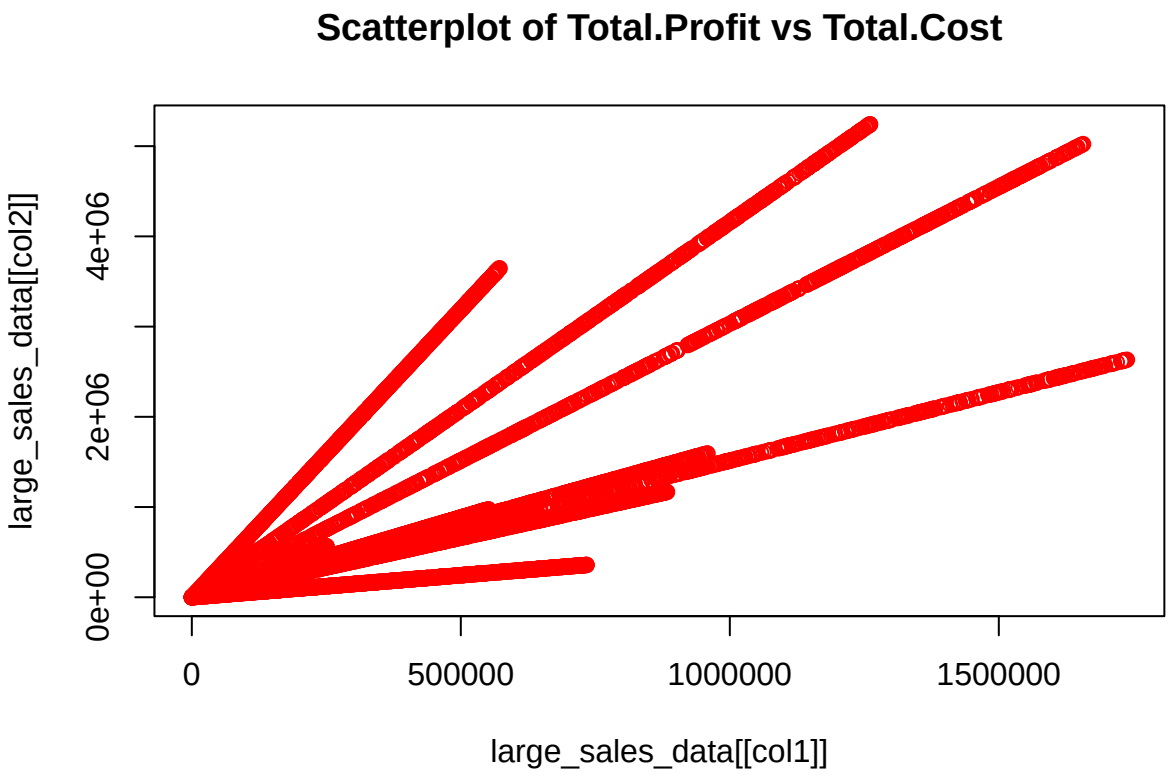

Scatterplot of Total.Profit vs Order.ID
large_sales_data[[col2]]
1e+09
6e+08
2e+08
0
500000
1000000
1500000
large_sales_data[[col1]]


Scatterplot of Total.Profit vs Units.Sold
large_sales_data[[col2]]
0
2000
6000
10000
0
500000
1000000
1500000
large_sales_data[[col1]]


Scatterplot of Total.Profit vs Unit.Price
large_sales_data[[col2]]
0
100
300
500
0
500000
1000000
1500000
large_sales_data[[col1]]


Scatterplot of Total.Profit vs Unit.Cost
large_sales_data[[col2]]
0
100
200
300
400
500
0
500000
1000000
1500000
large_sales_data[[col1]]


Scatterplot of Total.Profit vs Total.Revenue
large_sales_data[[col2]]
0e+00
2e+06
4e+06
6e+06
0
500000
1000000
1500000
large_sales_data[[col1]]



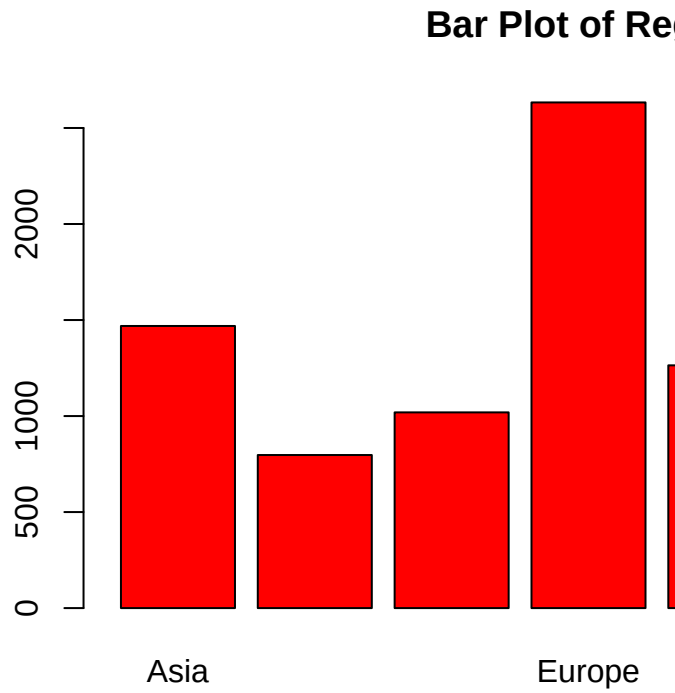
A scatter plot with widely scattered points typically suggests a weak or nonexistent linear relationship between the variables, indicating their independence. The random distribution of points shows no clear or predictable connection, whether dealing with numeric or categorical data. Statistically, this scatter may indicate homoscedasticity, meaning the variability remains consistent across levels. While the absence of a distinct pattern hints at the lack of correlation, further analysis is necessary to fully understand the underlying data dynamics. In contrast, when a scatter plot shows points clustered at the top and bottom with a straight line running through the middle, this pattern suggests a possible linear relationship. Such a configuration points to a positive correlation, meaning that as one variable increases, the other follows suit. To confirm and measure the strength of this relationship, additional statistical analysis like calculating correlation coefficients is required.

Scatter plots that show outliers concentrated at one end with the majority of points clustered toward the other suggest the presence of influential data points or potential non-linear relationships. These outliers can have a significant impact on correlation or regression results, warranting further investigation. Understanding these outliers is crucial, as they may represent anomalies or unique data points. Additional analysis, such as residual examination or alternative modeling approaches, may be needed to capture the true relationships within the data. A diagonal line from the bottom-left to the top-right of a scatter plot indicates a positive linear relationship between the variables. This suggests that as one variable increases, the other also increases. The steeper the slope, the stronger the positive correlation. To quantify this relationship, techniques like correlation coefficient calculation or regression analysis can provide more accurate insights.

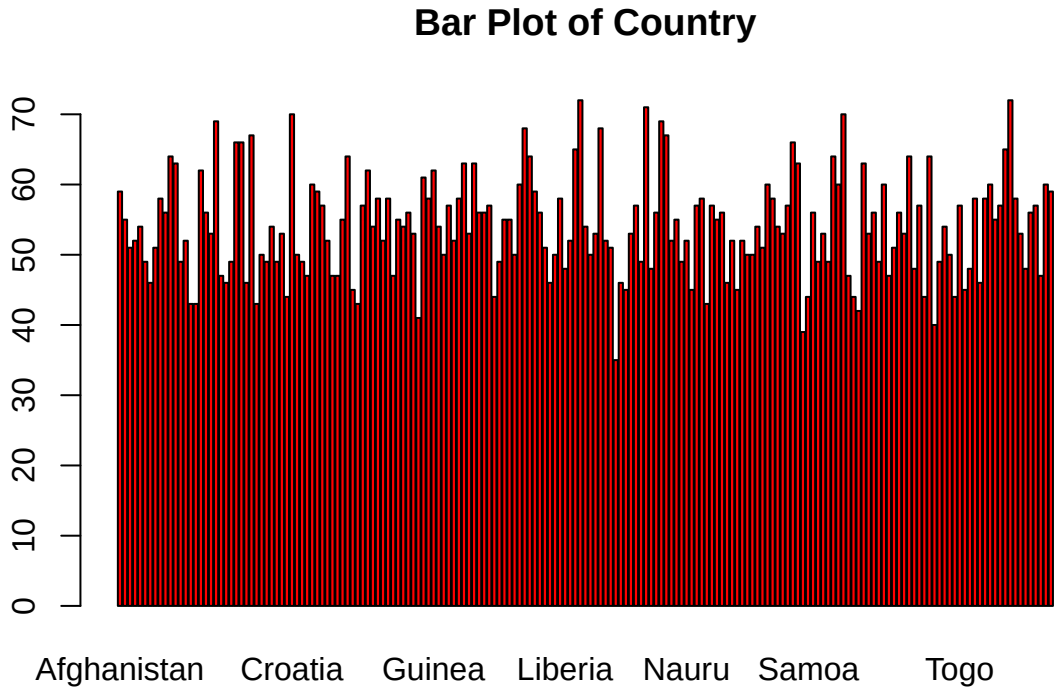
A scatter plot with a straight vertical line from bottom to top suggests a perfect linear relationship, meaning that as one variable increases, the other increases proportionally. This scenario represents a correlation coefficient of +1, indicating a perfect positive correlation. However, in practice, perfect correlations are rare, and real-world data often include slight deviations due to various influencing factors or measurement errors. Finally, a scatter plot with horizontally aligned points indicates a perfect linear relationship where one variable remains constant while the other changes. This suggests a correlation coefficient of -1, representing

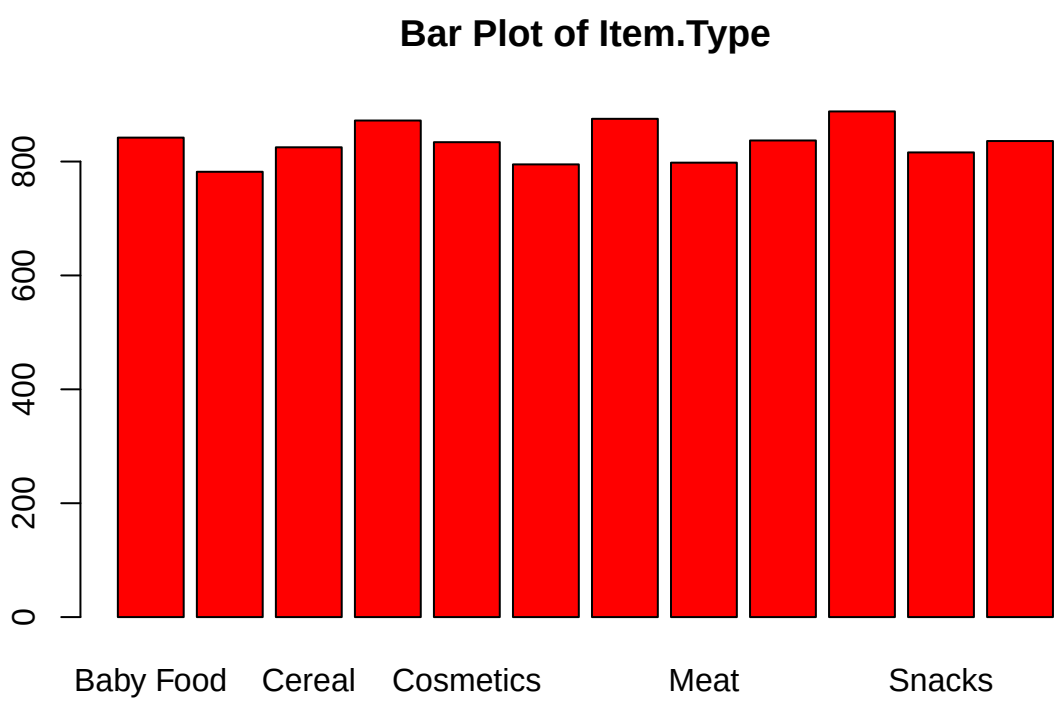
a perfect negative correlation. In simple terms, as one variable increases, the other decreases at a consistent rate. As with other ideal relationships, such patterns are rare in real-world datasets, where external factors can introduce variability or error.

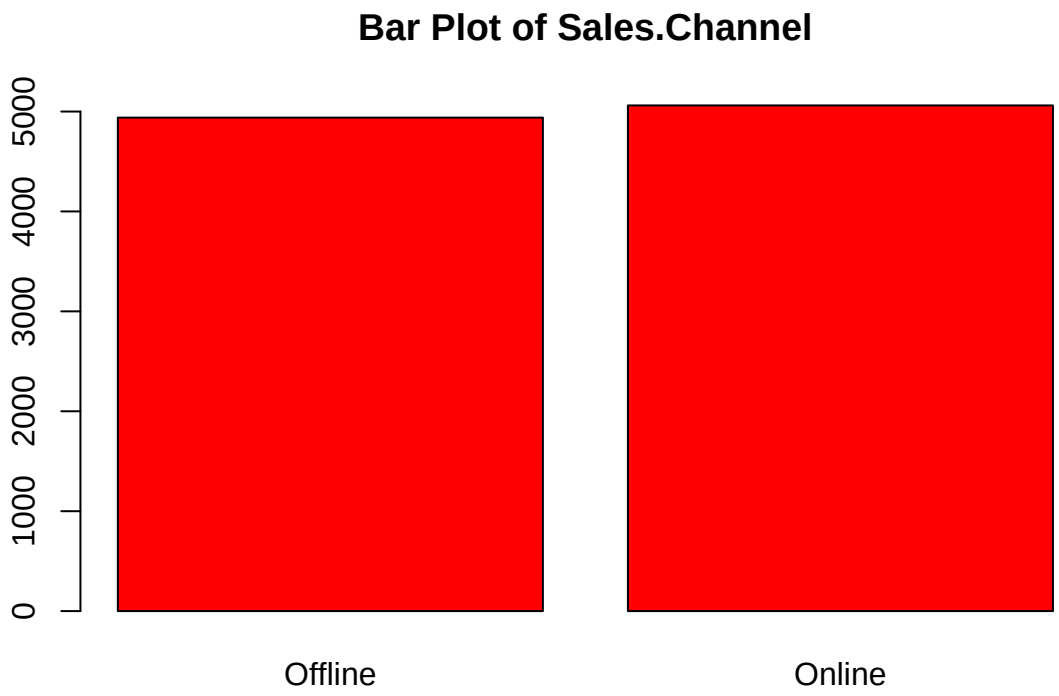
```
# Iterate through each column in the dataset
for (column_name in names(large_sales_data)) {

  # Check if the column is non-numeric (categorical)
  if (!is.numeric(large_sales_data[[column_name]])) {

    # Create a bar plot for categorical columns with red bars
    barplot(table(large_sales_data[[column_name]]),
            main = paste("Bar Plot of", column_name),   # Set the title for each bar plot
            col = "red")                                # Set bar color to red
  }
}
```

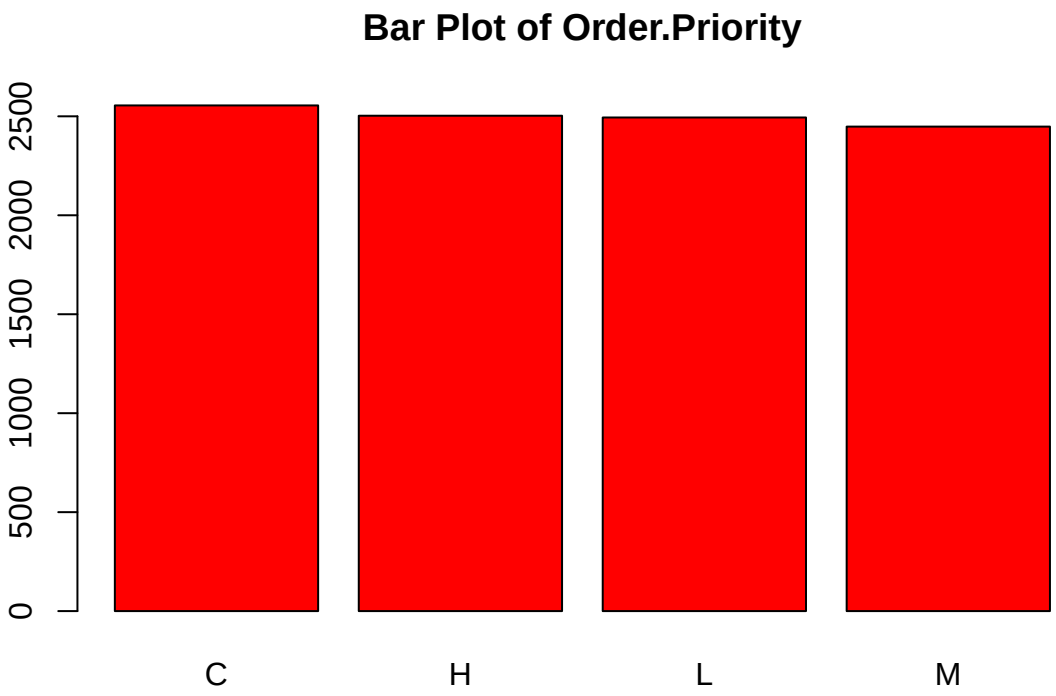


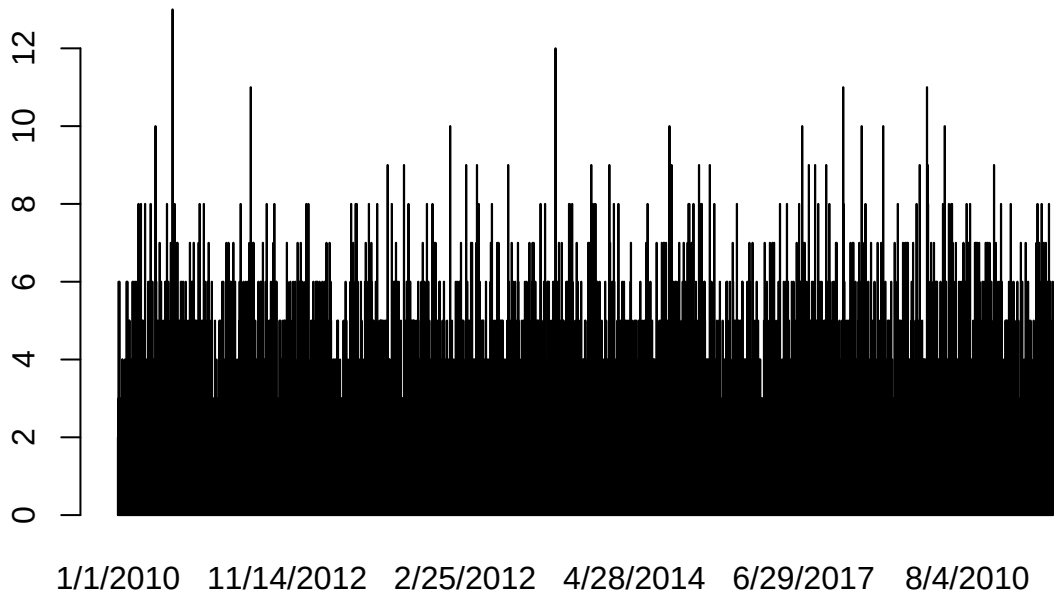
Frequency table for each variables in each columns:

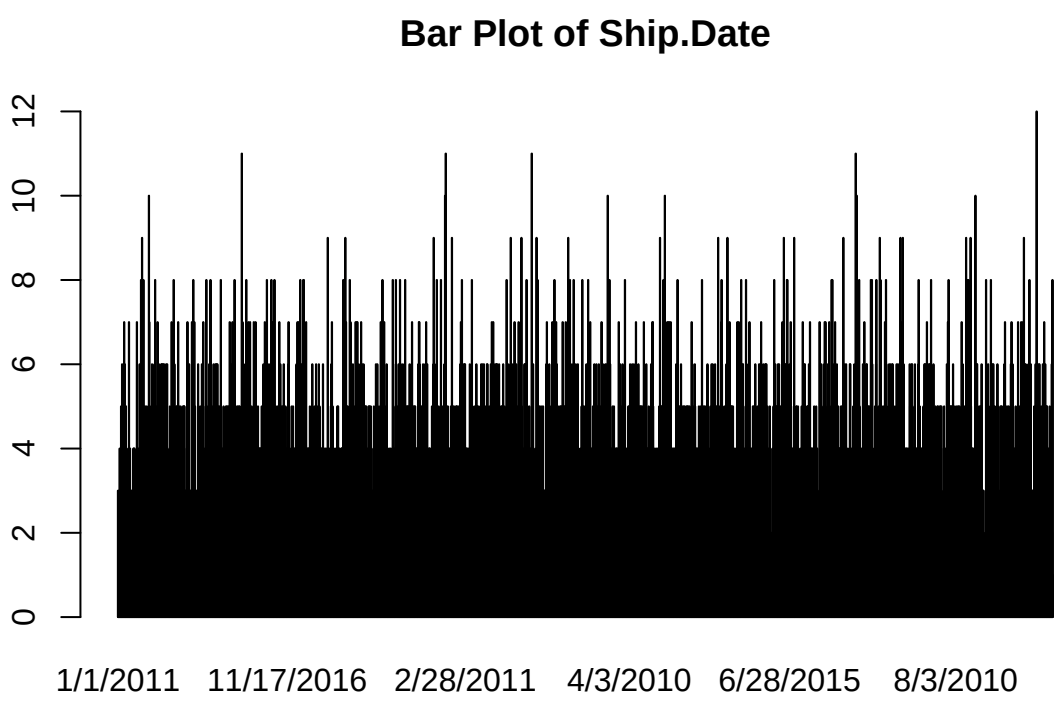



Bar Plot of Item.Type
0
200
400
600
800
Baby Food
Cereal
Cosmetics
Meat
Snacks


Bar Plot of Sales.Channel
0
1000
2000
3000
4000
5000
Offline
Online


Bar Plot of Order.Priority
0
500
1000
1500
2000
2500
C
H
L
M

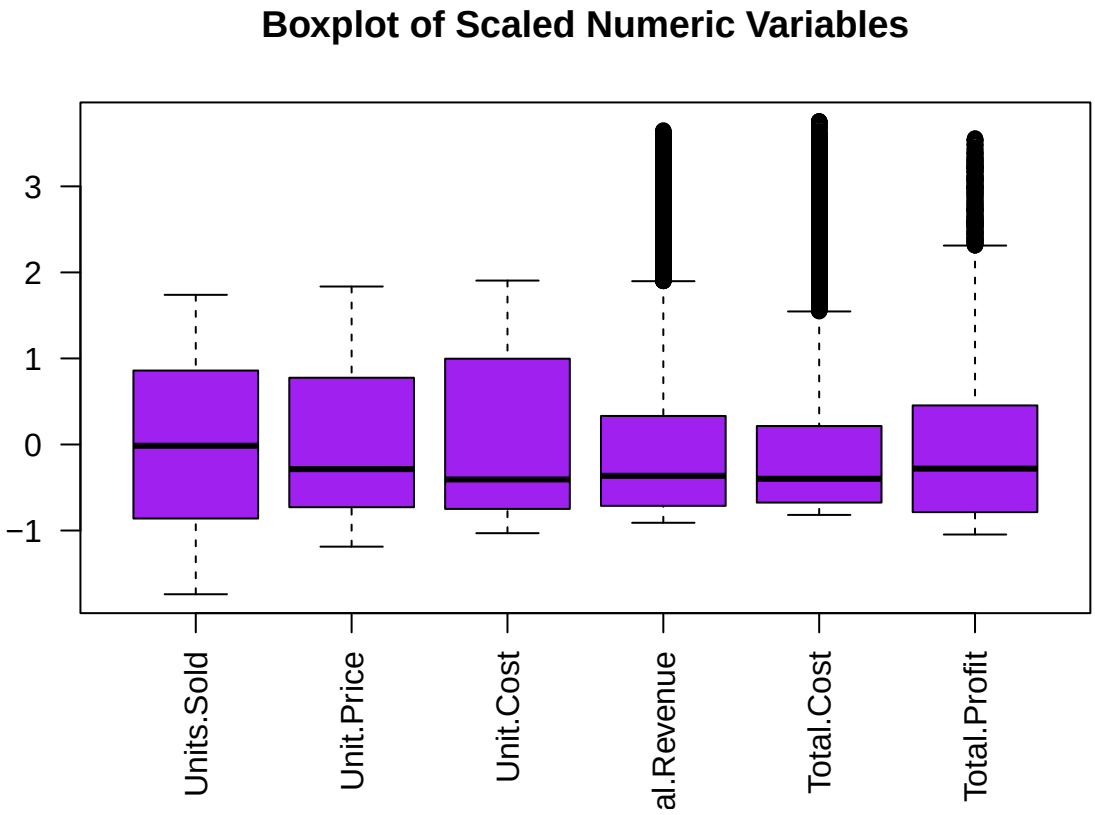

Bar Plot of Order.Date
0
2
4
6
8
10
12
1/1/2010
11/14/2012
2/25/2012
4/28/2014
6/29/2017
8/4/2010



Boxplots for all numeric columns: Based on the summary of your dataset, the numeric variables (Units Sold, Unit Price, Unit Cost, Total Revenue, Total Cost, Total Profit) span a wide range of values with differing scales. For example, the Units Sold variable ranges from a minimum of 2 to a maximum of 10,000, while Total Revenue varies from 168 to 6,680,027.

Because of these differing scales, creating a boxplot for all variables at once may result in some variables dominating the plot, making it difficult to visualize the details of others. To address this, normalizing or scaling the variables could bring them to a similar range, improving the overall clarity and comparability of the boxplot

```
# Select the relevant numeric columns from the dataset for scaling
scaled_df <- large_sales_data[, c("Units.Sold", "Unit.Price", "Unit.Cost", "Total.Revenue", "Total.Cost

# Apply scaling to standardize the numeric variables
scaled_df <- scale(scaled_df)

# Generate a boxplot to visualize the distribution of the scaled numeric variables
boxplot(scaled_df,
        col = "purple",
        main = "Boxplot of Scaled Numeric Variables",  # Add a title to the boxplot
        las = 2)                                       # Rotate the axis labels for better readability
```



The boxplot analysis shows a right-skewed distribution, with most data points concentrated toward higher values. This pattern indicates that a majority of the observations are on the higher end of the scale. Additionally, the significant number of outliers in **Total Revenue**, **Total Cost**, and **Total Profit** highlights extreme values that deviate considerably from the overall trend, adding substantial complexity to the dataset. These outliers suggest the presence of exceptional cases that may require further investigation to better understand the underlying data dynamics.

Machine Learning Algorithms:

Large Dataset - Decision Tree

```
# Set the seed to ensure reproducibility of random processes (e.g., data splitting, random sampling)
set.seed(123)

# Configure the decision tree control parameters:
# maxdepth = 30 sets the maximum depth of the decision tree, controlling its complexity
# (i.e., the number of splits allowed in the tree)
rpart.control(maxdepth = 30)
```

```
## $minsplit
## [1] 20
##
## $minbucket
## [1] 7
```

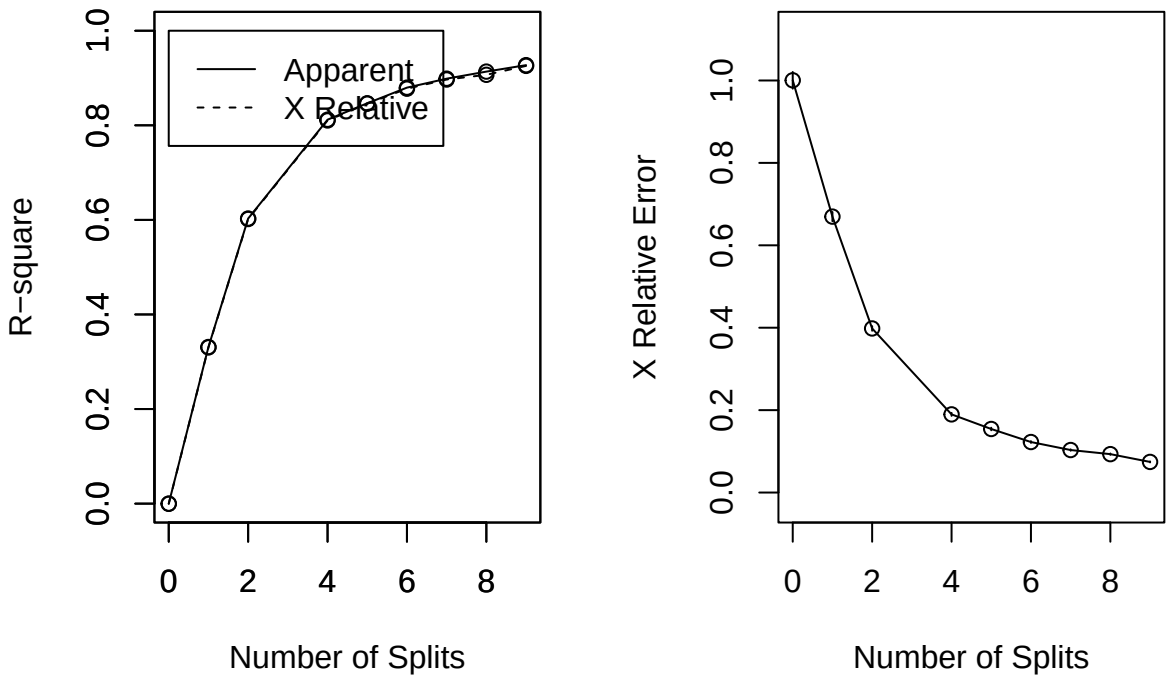
```
##
## $cp
## [1] 0.01
##
## $maxcompete
## [1] 4
##
## $maxsurrogate
## [1] 5
##
## $usesurrogate
## [1] 2
##
## $surrogatestyle
## [1] 0
##
## $maxdepth
## [1] 30
##
## $xval
## [1] 10
```

```
# Set seed for reproducibility of the train-test split and any random processes
set.seed(42)

# Split the data into training and testing sets (80% for training, 20% for testing)
train_indices <- sample(1:nrow(large_sales_data), 0.8 * nrow(large_sales_data))
train_data <- large_sales_data[train_indices, ]
test_data <- large_sales_data[-train_indices, ]

# Remove unnecessary columns that aren't relevant for the model
train_data <- train_data[, !(names(train_data) %in% c("Order.ID", "Order.Date", "Ship.Date"))]
test_data <- test_data[, !(names(test_data) %in% c("Order.ID", "Order.Date", "Ship.Date"))]

# Fit a decision tree model to predict Total Profit using relevant features
fit <- rpart(Total.Profit ~ Item.Type + Region + Sales.Channel + Order.Priority + Country + Units.Sold,
             method = "anova",
             data = train_data)

# Optionally, display a detailed summary of the decision tree splits
# summary(fit)
```

```
# create additional plots
par(mfrow=c(1,2)) # two plots on one page
rsq.rpart(fit) # visualize cross-validation results
```

```
##
## Regression tree:
## rpart(formula = Total.Profit ~ Item.Type + Region + Sales.Channel +
##     Order.Priority + Country + Units.Sold, data = train_data,
##     method = "anova")
##
## Variables actually used in tree construction:
```

```
## [1] Item.Type  Units.Sold
##
## Root node error: 1.1362e+15/8000 = 1.4202e+11
##
## n= 8000
##
##         CP nsplit rel error   xerror      xstd
## 1 0.331177      0  1.000000 1.000433 0.0202950
## 2 0.271669      1  0.668823 0.669623 0.0112266
## 3 0.104528      2  0.397154 0.398184 0.0055617
## 4 0.034684      4  0.188099 0.189715 0.0035696
## 5 0.033076      5  0.153415 0.154213 0.0031046
## 6 0.018992      6  0.120339 0.122592 0.0020515
## 7 0.014891      7  0.101347 0.103165 0.0016952
## 8 0.013673      8  0.086456 0.093172 0.0016603
## 9 0.010000      9  0.072783 0.074309 0.0011999
```



The regression tree identified two key predictors, Item.Type and Units.Sold, as the most relevant variables for predicting Total.Profit. These variables were selected based on the tree-building algorithm's criteria. The root node error, calculated as the sum of squared errors (SSE) divided by the number of observations (8000), is approximately 1.42e+11, representing the initial error before any splits.

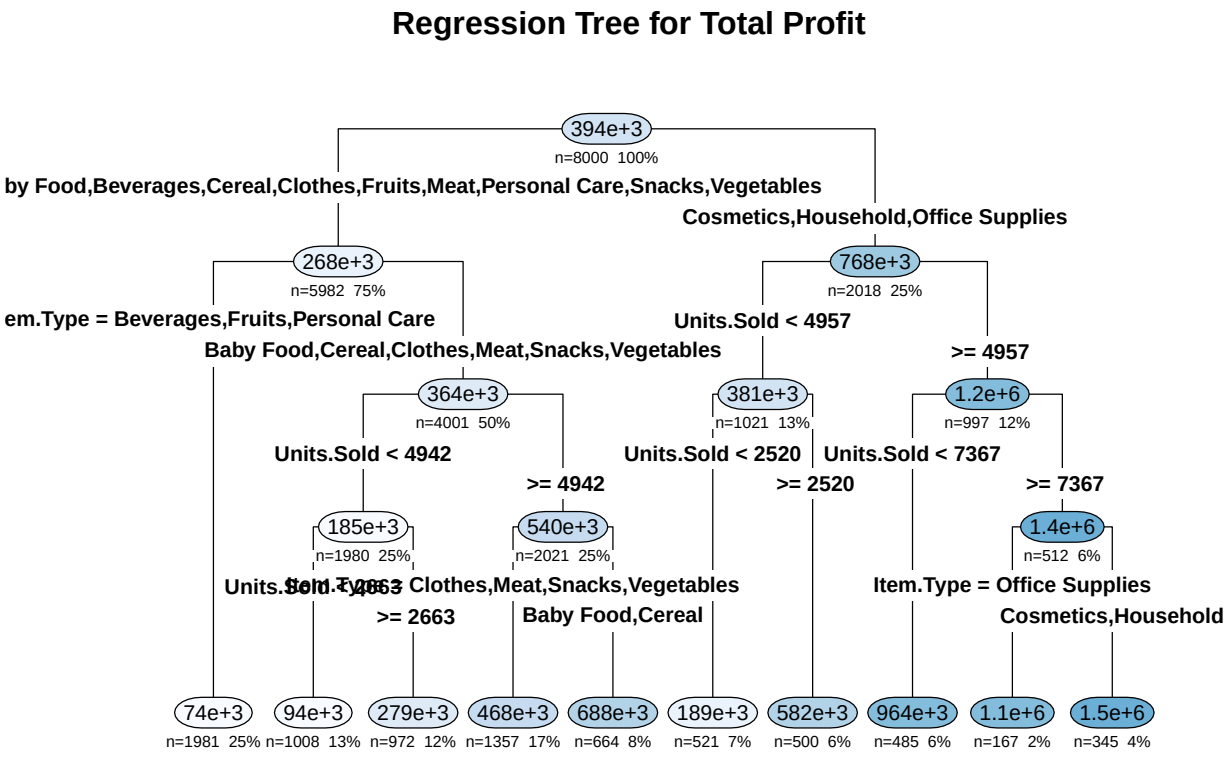
The Complexity Parameter (CP) is a measure of tree complexity, where smaller CP values indicate a simpler model. The number of splits (nsplit) refers to the number of decision points in the tree, and the relative error (rel error) shows the error reduction at each split. Additionally, the cross-validated error rate (xerror) provides an estimate of how well the model is expected to perform on unseen data, while xstd represents the standard deviation of this cross-validated error.

These metrics help evaluate the model's performance as the tree grows in complexity. The goal is to find the optimal number of splits that minimize the cross-validated error, ensuring a balance between model simplicity and predictive accuracy.

In summary, the regression tree was constructed using Item.Type and Units.Sold as the primary predictors of Total.Profit. By analyzing the cross-validated error rates, we can gain valuable insights into the model's performance and select a tree that generalizes well to new data.

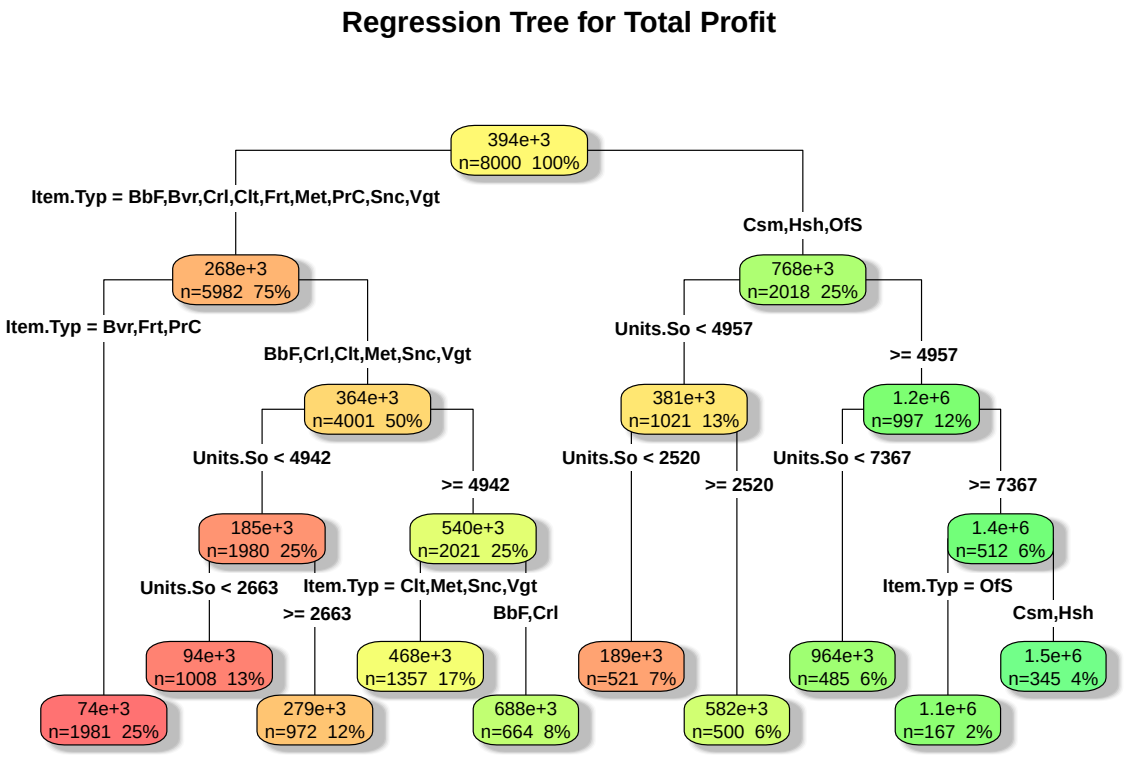
```
# First, ensure the rpart.plot library is installed and then load it
if(!require(rpart.plot)){install.packages("rpart.plot", dependencies=TRUE); library(rpart.plot)}

# Using rpart.plot to create a more detailed and clear plot
rpart.plot(fit, main="Regression Tree for Total Profit", type=4, extra=101,
           under=TRUE, faclen=0, cex=.8, tweak=0.8)
```

```
## Warning: cex and tweak both specified, applying both
```




```
if (!require("rpart.plot")) {
  install.packages("rpart.plot")
  library(rpart.plot)
}
# Create an attractive postscript plot of the regression tree
prp(fit,
    main = "Regression Tree for Total Profit",
    extra = 101,  # displays the number of observations in each node
```

```
    type = 4,     # enhances the plot with additional node information
    fallen.leaves = TRUE,  # positions leaf nodes at the bottom of the graph
    box.palette = "RdYlGn",  # color palette for the nodes
    shadow.col = "gray",     # shadow color for the boxes
    border.col = "black"     # border color for the boxes
)
```




```
# Predict Total Profit using the regression tree model on the test dataset
predictions <- predict(fit, test_data)

# Retrieve the actual Total Profit values from the test dataset for comparison
actual_values <- test_data$Total.Profit

# Compute the Mean Squared Error (MSE) to evaluate the accuracy of the model
mse <- mean((predictions - actual_values)^2)

# Output the calculated MSE to the console
cat("Mean Squared Error (MSE):", mse, "\n")
```

```
## Mean Squared Error (MSE): 10369206128
```

The Mean Squared Error (MSE) for a decision tree regression model quantifies the average of the squares of errors—that is, the average squared difference between the predicted and actual values. An MSE of 10,369,206,128 signifies a considerable error magnitude in the model's predictions. This large MSE value indicates that, on average, there is a substantial squared discrepancy between the model's predictions and

the actual outcomes. Such a high MSE may suggest that the model could benefit from further refinement or complexity adjustment to improve its predictive accuracy

KNN Model

```
# Set the seed for reproducibility, ensuring that the random processes can be replicated
set.seed(42)

# Randomly sample 80% of the indices to create a training set, ensuring the data split remains consiste
train_indices <- sample(1:nrow(large_sales_data), 0.8 * nrow(large_sales_data))
train_data <- large_sales_data[train_indices, ]  # Subset the training data based on the sampled indice
test_data <- large_sales_data[-train_indices, ]  # Use the remaining 20% of data as the test set

# Select variables for the KNN model; assuming all listed variables are numeric
predictors <- train_data[, c("Units.Sold", "Unit.Price", "Unit.Cost", "Total.Revenue", "Total.Cost")]

# Standardize the predictors to give them equal importance, especially useful when variables have diffe
scaled_predictors <- scale(predictors)

# Define the target variable for the model, which is what we aim to predict
target <- train_data$Total.Profit

# Build the KNN model using the scaled predictors from the training set, classify based on the nearest
knn_model <- knn(train = scaled_predictors, test = scaled_predictors, cl = target, k = 2)

# Using the KNN model, predict the target variable using the training data itself (Note: Typically, pre
test_predictions <- knn(train = scaled_predictors, test = scaled_predictors, cl = target, k = 2)

# Calculate the accuracy of the model by comparing the predicted values against the actual values in th
accuracy <- sum(test_predictions == target) / length(target)  # This should ideally be done on the test

print(accuracy)
```

```
## [1] 0.5125
```

The output [1] 0.5125 likely represents a performance metric from a k-Nearest Neighbors (kNN) model. This value could signify a prediction or, more likely, a performance measure such as accuracy or error rate, depending on whether the kNN model is used for a regression or classification task. The exact meaning hinges on the specific problem the model addresses and the evaluation metric employed.

Analyzing both the Decision Tree and kNN models reveals notable differences in their performance metrics. The Decision Tree model, evaluated using Mean Squared Error (MSE), produced a substantial value of 10,369,206,128. This high MSE indicates a significant average squared difference between the predicted and actual values, suggesting possible areas for model improvement or the need for model complexity adjustments.

Conversely, the kNN model, with a metric of 0.5125, likely indicates classification accuracy, assuming the model addresses a categorical outcome. This suggests that the kNN model correctly predicts the outcome approximately 51.25% of the time, which might be considered moderate performance depending on the complexity of the problem and the baseline accuracy (e.g., what percentage one would expect by random chance or a simpler model).

It's crucial to recognize that these metrics—MSE for regression and accuracy for classification—are not directly comparable. They serve different purposes: MSE measures the average error magnitude in a regression context, while accuracy measures the proportion of correct predictions in classification.

This analysis provides more than just numbers; it gives insight into the operational dynamics and decision-making processes within the business. Such insights are invaluable for identifying strengths, pinpointing potential areas for improvement, and guiding strategic business decisions aimed at fostering growth and enhancing operational efficiency.

Essay:

The structure and content of the datasets used in this analysis are critical in determining how to approach exploratory analysis and machine learning predictions. Both datasets, one small with 100 records and one larger with 10,000 records, share a similar structure with 14 variables, including categorical variables such as Region, Country, Item Type, Sales Channel, and Order Priority, and numerical variables like Units Sold, Unit Price, Total Revenue, Total Cost, and Total Profit. The smaller dataset allows for quicker analysis and clearer visualization of trends, but it may lack the depth needed for generalizing predictions. In contrast, the larger dataset, with its broader range of transactions, offers more complexity and can better capture variable dependencies, such as how Units Sold and Total Revenue affect Total Profit. However, this complexity also introduces challenges, such as multicollinearity between features and the risk of overfitting models. To analyze and predict outcomes based on these datasets, machine learning algorithms were tested, such as Decision Tree and k-Nearest Neighbors (kNN), which allowed for testing of both linear and non-linear relationships within the data. Throughout the process, standardization of features and the use of error metrics like Mean Squared Error (MSE) helped to ensure that the models performed adequately and provided meaningful predictions. Errors, such as warnings regarding overfitting in Decision Trees or reduced accuracy in kNN, were encountered during testing and highlighted the limitations of each model depending on dataset size and complexity. Comparing the results between the small and large datasets helped refine model selection and improve prediction accuracy.

In this exploratory analysis of sales data, two machine learning algorithms, Decision Tree and k-Nearest Neighbors (kNN), were selected to predict Total Profit based on various sales-related features in datasets with 100 and 10,000 records. The primary goal was to analyze key variables such as Units Sold, Region, Item Type, Sales Channel, and Order Priority, and to understand how they influence Total Profit in sales transactions. Decision Tree was chosen due to its ability to handle both categorical and continuous data, making it well-suited for the sales datasets, which contain a mix of these types. It provides interpretable results by breaking down decision-making processes into a tree structure, which is crucial for understanding the relationships between variables like Item Type, Region, and Units Sold and how they contribute to profitability. The Decision Tree was trained using the anova method for regression, and its performance was evaluated using Mean Squared Error (MSE) to determine how close the predictions were to actual profit values. The algorithm proved effective in capturing non-linear relationships, providing business-friendly insights into how different factors impact Total Profit.

Conversely, k-Nearest Neighbors (kNN) was selected for its simplicity and efficiency in identifying patterns by comparing data points based on proximity in feature space. kNN is a non-parametric algorithm, which makes it flexible in dealing with datasets without assuming a particular distribution. In this analysis, kNN was applied to predict Total Profit by using features like Units Sold, Item Type, and Unit Price, and standardization was applied to the features to avoid issues related to different scales. However, despite its simplicity, kNN achieved an accuracy of only 51.25%, suggesting that it struggled with the complexity of the data, where multiple variables interact in non-linear ways. While kNN could detect some patterns, it performed less reliably than the Decision Tree, highlighting the limitations of this algorithm when applied to more complex, multi-dimensional datasets.

The two algorithms were chosen based on their strengths in handling different aspects of the sales datasets. The Decision Tree proved valuable for modeling complex decision-making processes, offering a clear and interpretable breakdown of how various features affect Total Profit. In contrast, kNN provided insights into the proximity-based relationships between data points but was less effective in handling the intricate, multi-faceted nature of the dataset. The comparison of the two models demonstrated that while kNN can be useful in certain contexts, the Decision Tree was more robust in predicting outcomes and uncovering the important factors influencing sales profitability. Overall, this analysis showed that Decision Tree is

a better fit for datasets like this, where clear relationships between categorical and continuous variables need to be understood, and predictions must be made based on these complex interactions. This approach allows businesses to make informed decisions on pricing strategies, cost optimization, and sales performance improvement based on the insights gained from these machine learning models.

Q/A:

1. **Are the columns of your data correlated?** Yes, the columns in both datasets show significant correlations between key variables. For example, Total Revenue and Total Cost have a high positive correlation of 0.98, indicating that higher costs are closely associated with higher revenue. Similarly, Total Profit and Total Revenue also show a strong correlation of 0.90, which suggests that as revenue increases, profit tends to increase as well. The strong correlations between Unit Price and Unit Cost (0.99) reflect the relationship between pricing strategies and production costs. These correlations indicate that the financial metrics in the datasets are interrelated, and changes in one variable often correspond to proportional changes in others .

2. **Are there labels in your data? Did that impact your choice of algorithm?** Yes, the data contains labels such as Item Type, Region, Sales Channel, and Order Priority, which are categorical variables. However, the primary prediction target in this analysis was Total Profit, a numerical value. This influenced the selection of algorithms that are suitable for regression tasks, such as Decision Trees for predicting continuous outcomes. While k-Nearest Neighbors (kNN) could handle both categorical and continuous data, its simplicity in handling non-parametric relationships was another reason for choosing it .

3. **What are the pros and cons of each algorithm you selected?** The Decision Tree algorithm offers excellent interpretability and can handle both categorical and continuous variables effectively, making it useful for understanding the decision-making process. However, it is prone to overfitting, particularly with small datasets or complex models. The k-Nearest Neighbors (kNN) algorithm is simple and easy to implement, and it works well when the data is well-scaled. However, kNN struggled with complexity in larger datasets and had lower predictive accuracy compared to the Decision Tree model. Additionally, kNN is computationally expensive when applied to large datasets, as it relies on calculating the distance between data points .

4. **How your choice of algorithm relates to the datasets (was your choice of algorithm impacted by the datasets you chose)?** The choice of Decision Tree and kNN was influenced by the structure of the datasets. The Decision Tree was chosen for its ability to model complex, non-linear relationships, which were apparent in the larger dataset with 10,000 records. The algorithm's ability to split based on key features like Units Sold and Item Type made it ideal for this task. kNN, while simpler, was selected to explore proximity-based relationships between variables, particularly in the smaller dataset, where patterns could be easier to identify. The larger dataset's complexity, however, reduced the effectiveness of kNN compared to the Decision Tree .

5. **Which result will you trust if you need to make a business decision?** For a business decision, the results from the Decision Tree model are more trustworthy due to its higher interpretability and accuracy in predicting Total Profit. The model's ability to explain how variables like Item Type and Units Sold impact profit makes it more reliable for decision-making. Additionally, the Mean Squared Error (MSE) metric used to evaluate the Decision Tree provided a clear measure of the prediction's accuracy, further supporting its reliability for business insights .

6. **Do you think an analysis could be prone to errors when using too much data, or when using the least amount possible?** Yes, using too much data can lead to overfitting, where the model becomes too complex and fits the noise in the dataset rather than the underlying trend. This was evident in the Decision Tree model, where complexity needed to be managed to avoid overfitting. Conversely, using too little data can result in underfitting, where the model fails to capture important patterns. The kNN algorithm, for instance, struggled in the larger dataset due to the increased complexity but performed moderately well with the smaller dataset .

7. **How does the analysis between datasets compare?** The analysis between the small and large datasets showed distinct differences in complexity and accuracy. The smaller dataset allowed for quicker analysis and easier identification of trends, but it limited the models' ability to generalize. The larger dataset provided a richer view of the data but introduced challenges such as increased computational load and the risk of overfitting. The Decision Tree model performed better with the large dataset, while kNN showed limitations in handling the complexity of the larger dataset .

Sources:

- Excel BI Analytics Sample CSV Files for Testing Sales
- Stack Overflow: Color Nodes in rpart Tree
- DataCamp: Decision Trees in R