Capstone Project

Group 1

2023-06-30

R Markdown

This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For more details on using R Markdown see http://rmarkdown.rstudio.com.

When you click the **Knit** button a document will be generated that includes both content as well as the output of any embedded R code chunks within the document. You can embed an R code chunk like this:

```r
shipment_tracking <- read.csv("C:/Users/AJ/Downloads/edata.csv")
# Display the first few rows of the dataset
head(shipment_tracking)
```

```
##   ID Warehouse_block Mode_of_Shipment Customer_care_calls Customer_rating
## 1  1               D           Flight                   4               2
## 2  2               F           Flight                   4               5
## 3  3               A           Flight                   2               2
## 4  4               B           Flight                   3               3
## 5  5               C           Flight                   2               2
## 6  6               F           Flight                   3               1
##   Cost_of_the_Product Prior_purchases Product_importance Gender
## 1                 177               3                low      F
## 2                 216               2                low      M
## 3                 183               4                low      M
## 4                 176               4             medium      M
## 5                 184               3             medium      F
## 6                 162               3             medium      F
##   Discount_offered Weight_in_gms Reached.on.Time_Y.N
## 1               44          1233                   1
## 2               59          3088                   1
## 3               48          3374                   1
## 4               10          1177                   1
## 5               46          2484                   1
## 6               12          1417                   1
```

```r
#Shape of the dataset
dim(shipment_tracking)
```

```
## [1] 10999    12
```

```r
#Printing the Structure of the data
str(shipment_tracking)
```

```
## 'data.frame':    10999 obs. of  12 variables:
##  $ ID                 : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ Warehouse_block    : chr  "D" "F" "A" "B" ...
##  $ Mode_of_Shipment   : chr  "Flight" "Flight" "Flight" "Flight" ...
##  $ Customer_care_calls: int  4 4 2 3 2 3 3 4 3 3 ...
##  $ Customer_rating    : int  2 5 2 3 2 1 4 1 4 2 ...
##  $ Cost_of_the_Product: int  177 216 183 176 184 162 250 233 150 164 ...
##  $ Prior_purchases    : int  3 2 4 4 3 3 3 2 3 3 ...
##  $ Product_importance : chr  "low" "low" "low" "medium" ...
##  $ Gender             : chr  "F" "M" "M" "M" ...
##  $ Discount_offered   : int  44 59 48 10 46 12 3 48 11 29 ...
##  $ Weight_in_gms      : int  1233 3088 3374 1177 2484 1417 2371 2804 1861 1187 ...
##  $ Reached.on.Time_Y.N: int  1 1 1 1 1 1 1 1 1 1 ...
```

```r
#Summary of the shipment dataset
summary(shipment_tracking)
```

```
##        ID        Warehouse_block    Mode_of_Shipment   Customer_care_calls
##  Min.   :    1   Length:10999       Length:10999       Min.   :2.000      
##  1st Qu.: 2750   Class :character   Class :character   1st Qu.:3.000      
##  Median : 5500   Mode  :character   Mode  :character   Median :4.000      
##  Mean   : 5500                                         Mean   :4.054      
##  3rd Qu.: 8250                                         3rd Qu.:5.000      
##  Max.   :10999                                         Max.   :7.000      
##  Customer_rating Cost_of_the_Product Prior_purchases  Product_importance
##  Min.   :1.000   Min.   : 96.0       Min.   : 2.000   Length:10999      
##  1st Qu.:2.000   1st Qu.:169.0       1st Qu.: 3.000   Class :character  
##  Median :3.000   Median :214.0       Median : 3.000   Mode  :character  
##  Mean   :2.991   Mean   :210.2       Mean   : 3.568                     
##  3rd Qu.:4.000   3rd Qu.:251.0       3rd Qu.: 4.000                     
##  Max.   :5.000   Max.   :310.0       Max.   :10.000                     
##     Gender          Discount_offered Weight_in_gms  Reached.on.Time_Y.N
##  Length:10999       Min.   : 1.00    Min.   :1001   Min.   :0.0000     
##  Class :character   1st Qu.: 4.00    1st Qu.:1840   1st Qu.:0.0000     
##  Mode  :character   Median : 7.00    Median :4149   Median :1.0000     
##                     Mean   :13.37    Mean   :3634   Mean   :0.5967     
##                     3rd Qu.:10.00    3rd Qu.:5050   3rd Qu.:1.0000     
##                     Max.   :65.00    Max.   :7846   Max.   :1.0000     
```

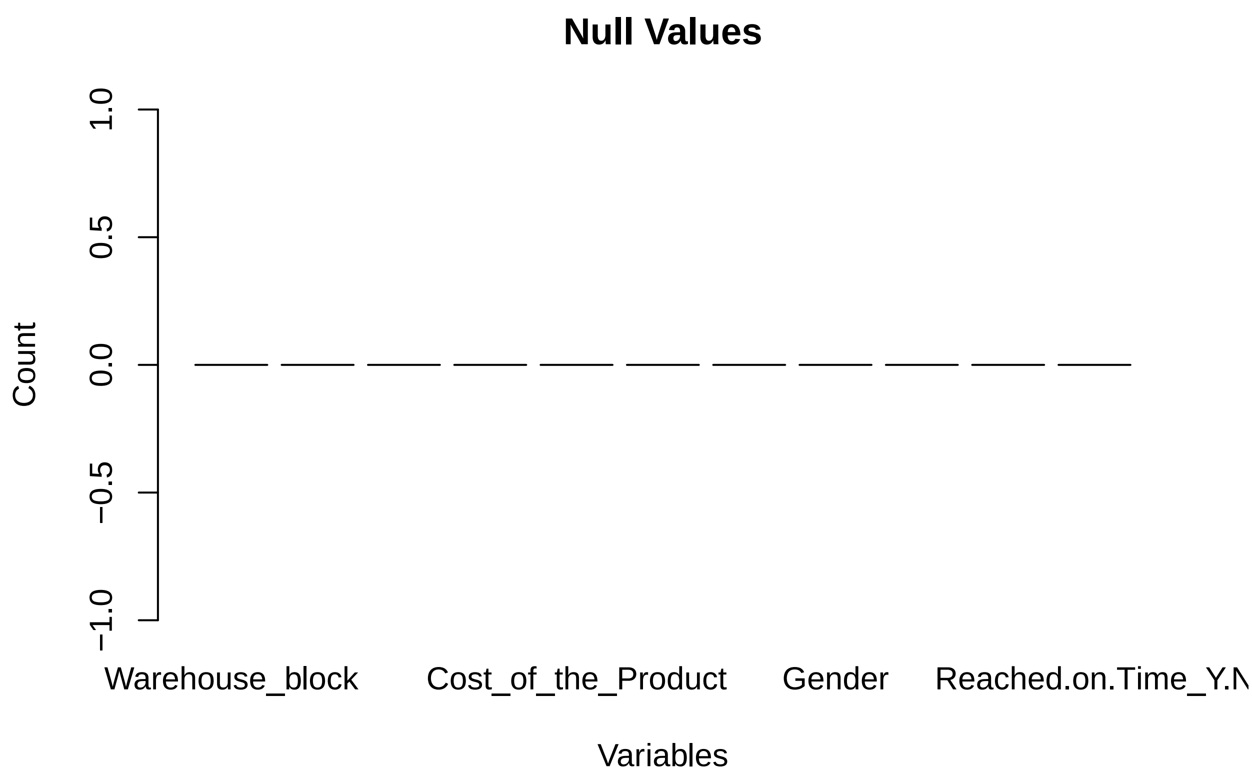
```r
#Removing ID Column
shipment_tracking <- shipment_tracking[, -which(names(shipment_tracking) == "ID")]

#Removed ID COlumn
```

```r
#Checking if there are any missing values column wise
missing_counts = colSums(is.na(shipment_tracking))
print(missing_counts)
```

```
##     Warehouse_block    Mode_of_Shipment Customer_care_calls     Customer_rating
```

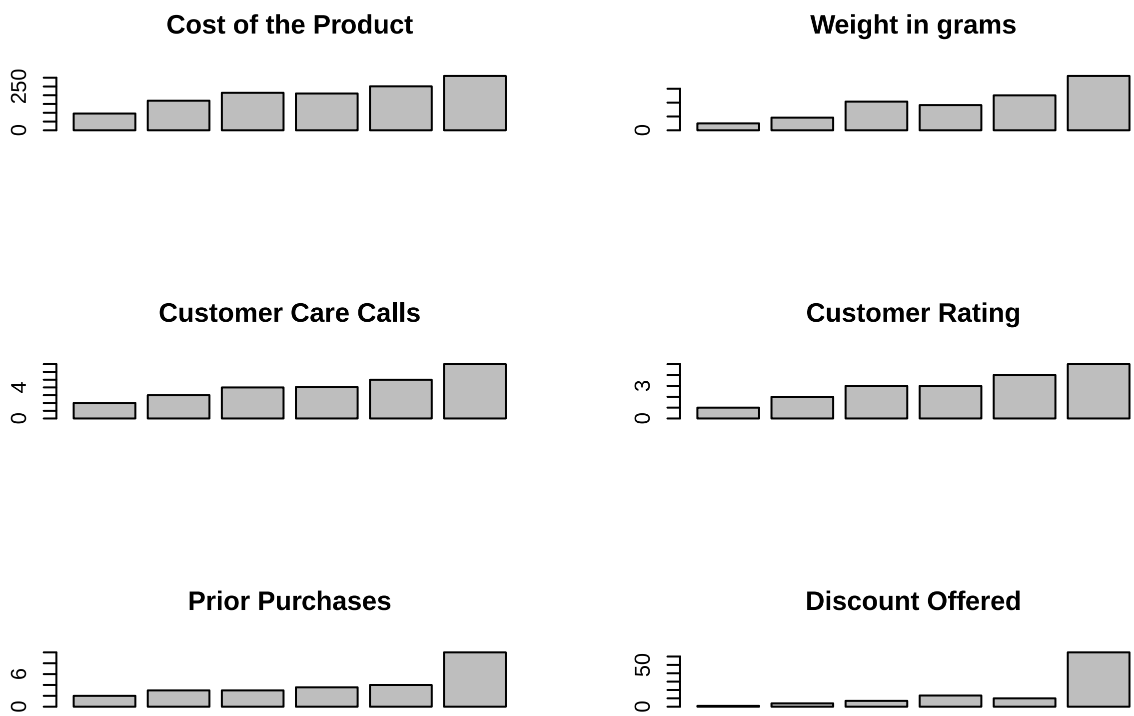
```
##                    0                    0                    0                    0
## Cost_of_the_Product     Prior_purchases  Product_importance               Gender
##                    0                    0                    0                    0
##    Discount_offered       Weight_in_gms Reached.on.Time_Y.N
##                    0                    0                    0
```

```r
# Create a bar plot to visualize the missing values
barplot(missing_counts, main = "Null Values", xlab = "Variables", ylab = "Count")
```

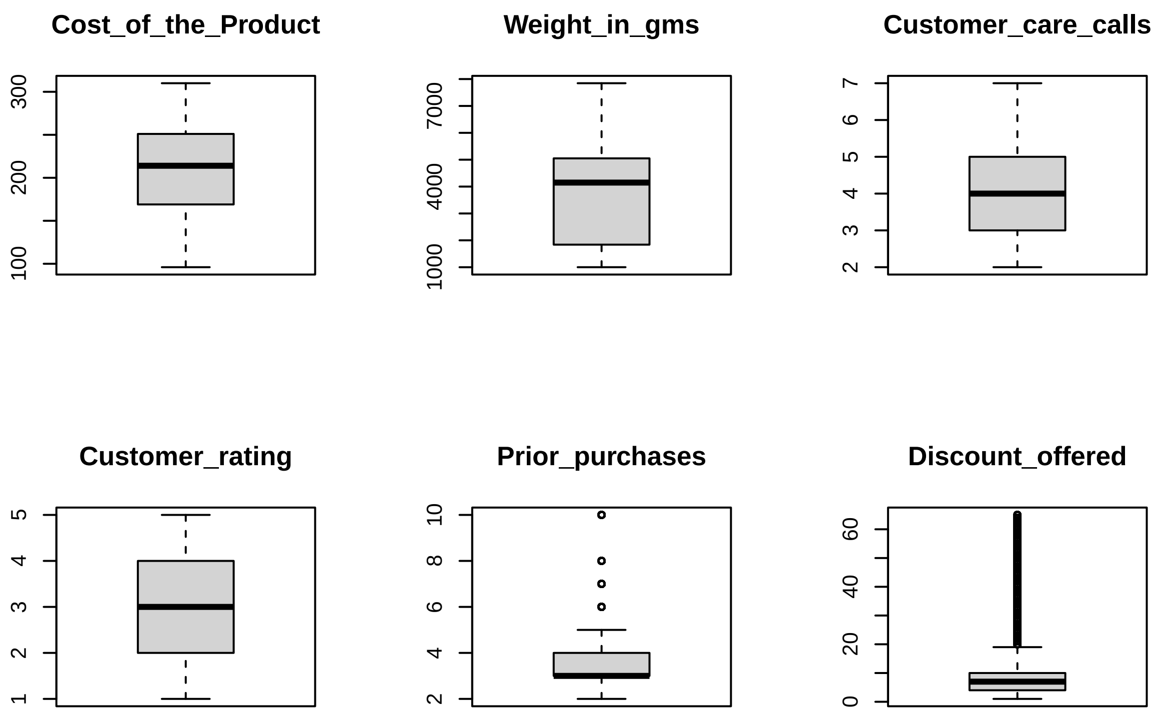



```r
#Separating Categorical and Numerical Columns
# Function to analyze and display frequency distribution of a categorical variable
analyze_categorical <- function(variable) {
  freq <- table(variable)
  freq_df <- as.data.frame(freq)
  colnames(freq_df) <- c("Category", "Count")
  freq_df <- freq_df[order(freq_df$Count, decreasing = TRUE), ]
  print(freq_df)
}

# Extract categorical variables
categorical_shipment <- shipment_tracking[, sapply(shipment_tracking, is.factor) | sapply(shipment_track

# Extract numerical variables
numerical_shipment <- shipment_tracking[, sapply(shipment_tracking, is.numeric) | sapply(shipment_track
```

```r
# Descriptive analysis of numeric variables
cost_summary <- summary(shipment_tracking$Cost_of_the_Product)
weight_summary <- summary(shipment_tracking$Weight_in_gms)
care_calls_summary <- summary(shipment_tracking$Customer_care_calls)
rating_summary <- summary(shipment_tracking$Customer_rating)
prior_purchases_summary <- summary(shipment_tracking$Prior_purchases)
discount_summary <- summary(shipment_tracking$Discount_offered)


# Convert summary statistics into numeric vectors
cost_vals <- as.numeric(cost_summary)
weight_vals <- as.numeric(weight_summary)
care_calls_vals <- as.numeric(care_calls_summary)
rating_vals <- as.numeric(rating_summary)
prior_purchases_vals <- as.numeric(prior_purchases_summary)
discount_vals <- as.numeric(discount_summary)


# Set the layout for the plot grid
par(mfrow = c(3, 2))
# Create individual bar plots for each numeric variable
barplot(cost_vals, main = "Cost of the Product")
barplot(weight_vals, main = "Weight in grams")
barplot(care_calls_vals, main = "Customer Care Calls")
barplot(rating_vals, main = "Customer Rating")
barplot(prior_purchases_vals, main = "Prior Purchases")
barplot(discount_vals, main = "Discount Offered")
```

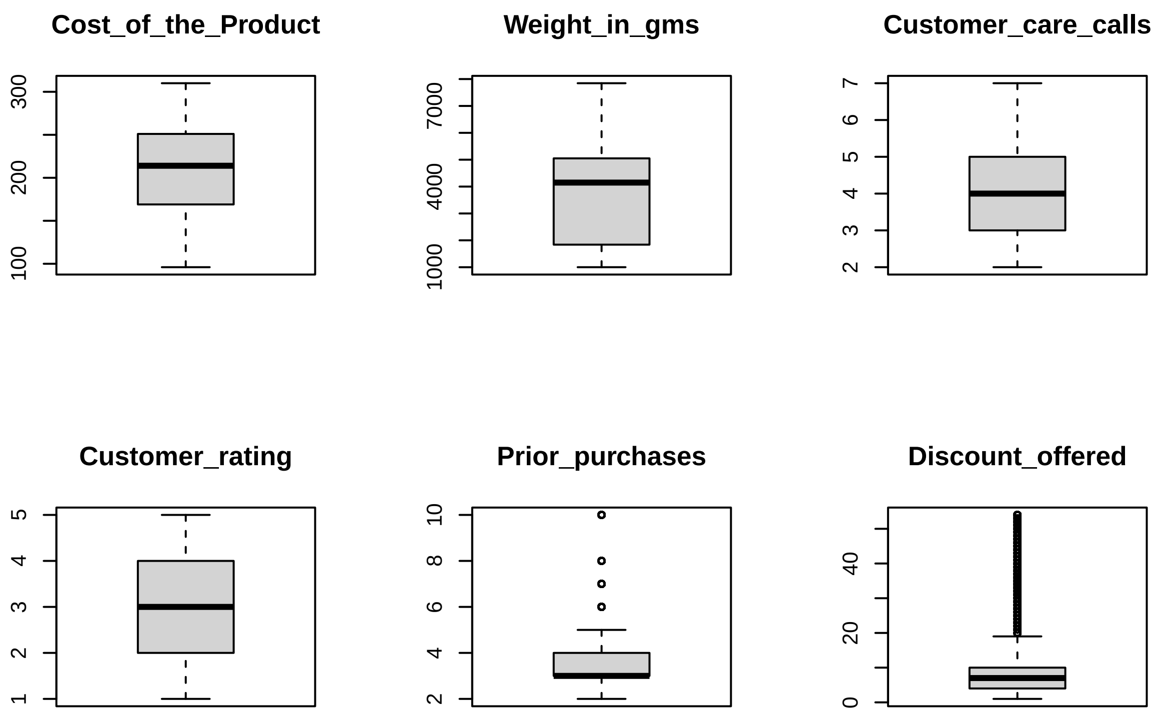



```r
numeric_vars <- c("Cost_of_the_Product", "Weight_in_gms", "Customer_care_calls", "Customer_rating", "Pri

# Set the layout for the plot grid
par(mfrow = c(2, 3))

# Create boxplots for each numerical variable
for (var in numeric_vars) {
  # Plot the boxplot
  boxplot(shipment_tracking[[var]], main = var)
}
```




```
# #Replacing extreme values with less extreme values using winsorize by introducing percentiles
#
# # Winsorize outliers in "Prior_purchases" variable
q <- quantile(shipment_tracking$Prior_purchases, c(0.05, 0.95), na.rm = TRUE)  # Set the threshold perc
# shipment_tracking$Prior_purchases <- pmin(pmax(shipment_tracking$Prior_purchases, q[1]), q[2])
#
# # Winsorize outliers in "Discount_offered" variable
 q <- quantile(shipment_tracking$Discount_offered, c(0.05, 0.95), na.rm = TRUE)  # Set the threshold pe
shipment_tracking$Discount_offered <- pmin(pmax(shipment_tracking$Discount_offered, q[1]), q[2])
```

```
#Checking box plots after performing winsorize method
numeric_vars <- c("Cost_of_the_Product", "Weight_in_gms", "Customer_care_calls", "Customer_rating", "Pr

# Set the layout for the plot grid
par(mfrow = c(2, 3))

# Create boxplots for each numerical variable
for (var in numeric_vars) {
  # Plot the boxplot
  boxplot(shipment_tracking[[var]], main = var)
}
```

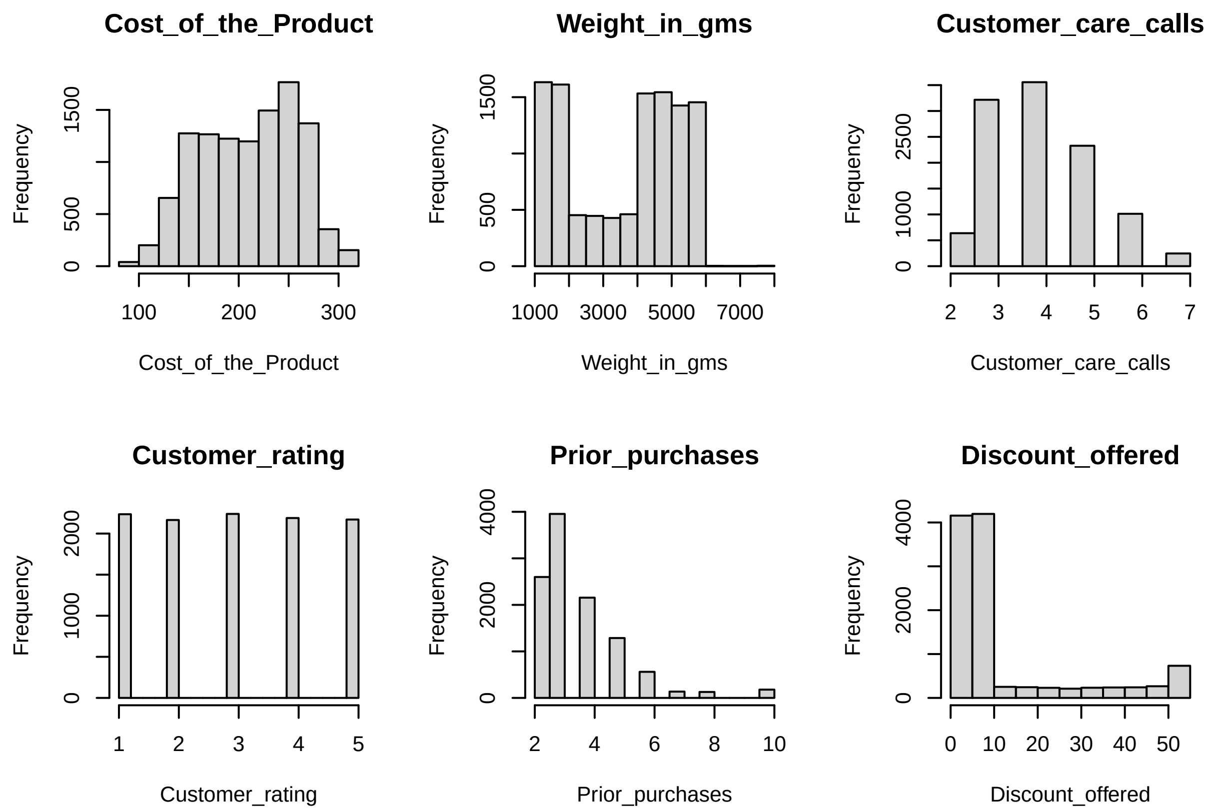



```r
#Summary of All Numerical Variables
summary(numerical_shipment)
```

```
##  Customer_care_calls Customer_rating Cost_of_the_Product Prior_purchases
##  Min.   :2.000       Min.   :1.000   Min.   : 96.0       Min.   : 2.000
##  1st Qu.:3.000       1st Qu.:2.000   1st Qu.:169.0       1st Qu.: 3.000
##  Median :4.000       Median :3.000   Median :214.0       Median : 3.000
##  Mean   :4.054       Mean   :2.991   Mean   :210.2       Mean   : 3.568
##  3rd Qu.:5.000       3rd Qu.:4.000   3rd Qu.:251.0       3rd Qu.: 4.000
##  Max.   :7.000       Max.   :5.000   Max.   :310.0       Max.   :10.000
##  Discount_offered Weight_in_gms  Reached.on.Time_Y.N
##  Min.   : 1.00    Min.   :1001   Min.   :0.0000
##  1st Qu.: 4.00    1st Qu.:1840   1st Qu.:0.0000
##  Median : 7.00    Median :4149   Median :1.0000
##  Mean   :13.37    Mean   :3634   Mean   :0.5967
##  3rd Qu.:10.00    3rd Qu.:5050   3rd Qu.:1.0000
##  Max.   :65.00    Max.   :7846   Max.   :1.0000
```

```r
#Visualizing the Numerical Variables
# Select the numerical variables for histogram plots
numeric_vars <- c("Cost_of_the_Product", "Weight_in_gms", "Customer_care_calls", "Customer_rating", "Pr

# Create histogram plots for each numerical variable
par(mfrow = c(2, 3)) # Set the layout for the plot grid
for (var in numeric_vars) {
```

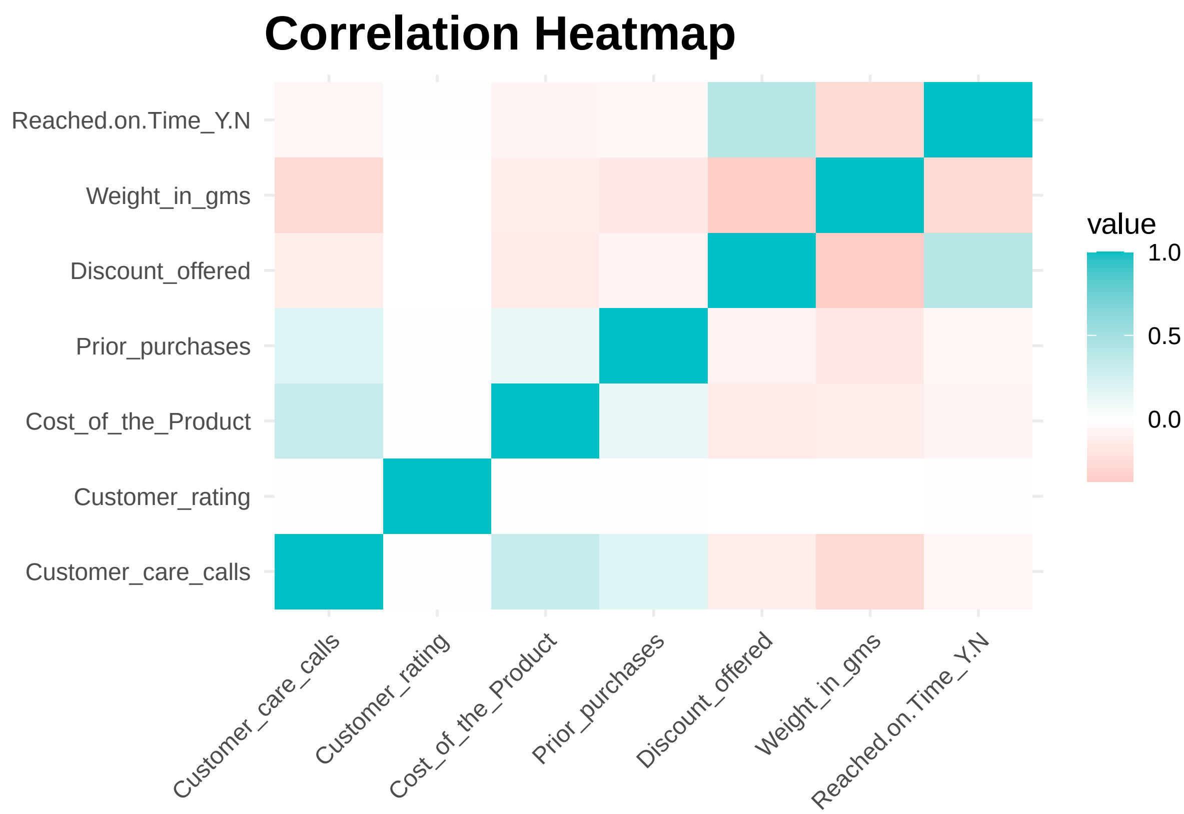
```r
  hist(shipment_tracking[[var]], main = var, xlab = var)
}
```




```r
#Correlation Analysis
# Install and load required packages
#install.packages("ggplot2")  # Install ggplot2 package
library(ggplot2)  # Load ggplot2 package
```

```
## Warning: package 'ggplot2' was built under R version 4.3.1
```

```r
# Compute correlation matrix
cor_matrix <- cor(numerical_shipment)

# Create a long format of the correlation matrix
cor_df <- reshape2::melt(cor_matrix)

# Create correlation heatmap
ggplot(cor_df, aes(Var1, Var2, fill = value)) +
  geom_tile() +
  scale_fill_gradient2(low = "#F8766D", mid = "white", high = "#00BFC4", midpoint = 0, na.value = "grey5
  labs(title = "Correlation Heatmap", x = "", y = "") +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1),
        plot.title = element_text(size = 18, face = "bold"))
```




```r
# Install and load required packages
#install.packages("corrplot")  # Install corrplot package
library(corrplot)  # Load corrplot package
```

```
## Warning: package 'corrplot' was built under R version 4.3.1
```

```
## corrplot 0.92 loaded
```

```r
# Compute correlation matrix
cor_matrix <- cor(numerical_shipment)

print(cor_matrix)
```

```
##                     Customer_care_calls Customer_rating Cost_of_the_Product
## Customer_care_calls          1.00000000     0.012208804         0.323181830
## Customer_rating              0.01220880     1.000000000         0.009269525
## Cost_of_the_Product          0.32318183     0.009269525         1.000000000
## Prior_purchases              0.18077119     0.013179389         0.123675999
## Discount_offered            -0.13075005    -0.003124444        -0.138311686
## Weight_in_gms               -0.27661519    -0.001896852        -0.132604048
## Reached.on.Time_Y.N         -0.06712586     0.013118604        -0.073587211
##                     Prior_purchases Discount_offered Weight_in_gms
## Customer_care_calls      0.18077119     -0.130750050  -0.276615190
## Customer_rating          0.01317939     -0.003124444  -0.001896852
```

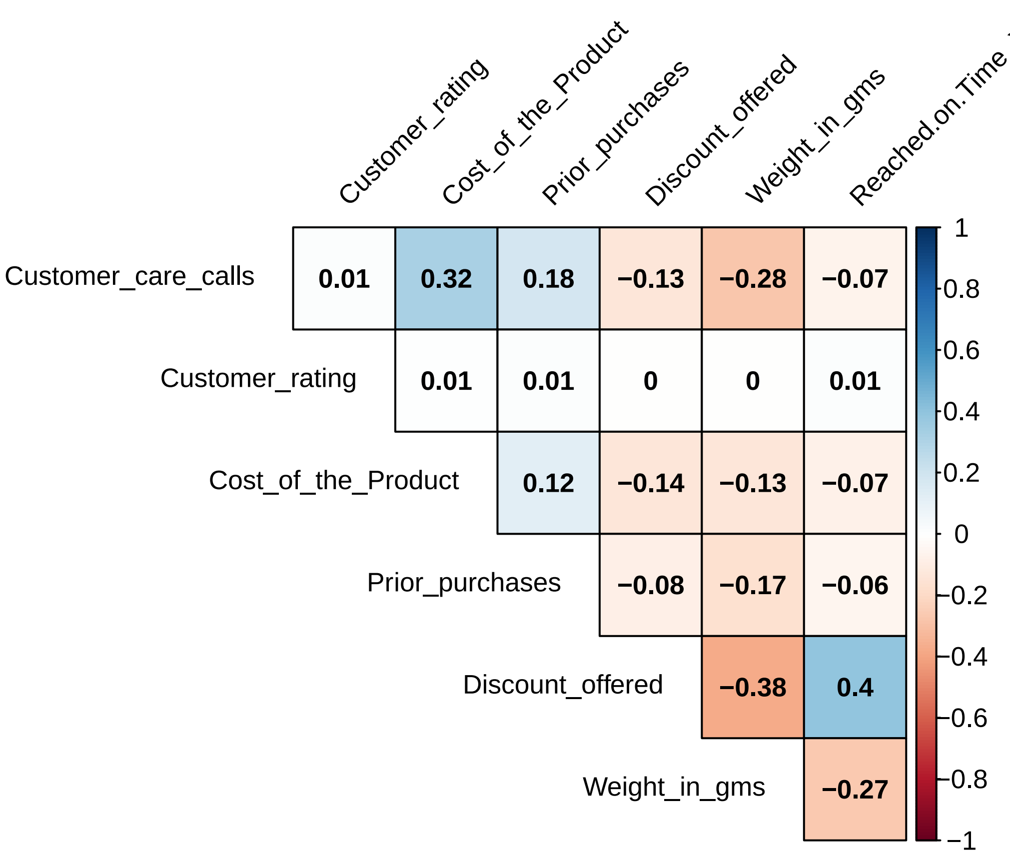
```
## Cost_of_the_Product      0.12367600     -0.138311686  -0.132604048
## Prior_purchases          1.00000000     -0.082769300  -0.168213085
## Discount_offered        -0.08276930      1.000000000  -0.376066715
## Weight_in_gms           -0.16821308     -0.376066715   1.000000000
## Reached.on.Time_Y.N     -0.05551501      0.397108474  -0.268792584
##                       Reached.on.Time_Y.N
## Customer_care_calls          -0.06712586
## Customer_rating               0.01311860
## Cost_of_the_Product          -0.07358721
## Prior_purchases              -0.05551501
## Discount_offered              0.39710847
## Weight_in_gms                -0.26879258
## Reached.on.Time_Y.N           1.00000000
```

```r
# Create correlation heatmap
corrplot(cor_matrix, method = "color", type = "upper", tl.col = "black",
         tl.srt = 45, tl.cex = 0.8, tl.offset = 1, cl.lim = c(-1, 1),
         addCoef.col = "black", number.cex = 0.8, number.digits = 2,
         diag = FALSE, outline = TRUE)
```

```
## Warning in text.default(pos.xlabel[, 1], pos.xlabel[, 2], newcolnames, srt =
## tl.srt, : "cl.lim" is not a graphical parameter
```

```
## Warning in text.default(pos.ylabel[, 1], pos.ylabel[, 2], newrownames, col =
## tl.col, : "cl.lim" is not a graphical parameter
```

```
## Warning in title(title, ...): "cl.lim" is not a graphical parameter
```




```
#points to note from Correlation Analysis

# Discount_offered has a moderate positive correlation coefficient of 0.397 with Reached.on.Time_Y.N. T
#
# Cost_of_the_Product and Customer_care_calls both have negative correlation coefficients with Reached.
#
# Other features such as Customer_rating, Prior_purchases, and Weight_in_gms do not show strong correla
```

```
#Technique-2 Anova

# Assuming you have a data frame called 'df' with the relevant variables

# Fit the ANOVA model
model <- aov(Reached.on.Time_Y.N ~ Customer_care_calls + Customer_rating + Cost_of_the_Product + Prior_

# Perform ANOVA analysis
anova_result <- anova(model)

# View the ANOVA table
print(anova_result)
```

```
## Analysis of Variance Table
##
## Response: Reached.on.Time_Y.N
##                          Df  Sum Sq Mean Sq   F value    Pr(>F)
```

```
## Customer_care_calls     1   11.93   11.93   60.5200 7.936e-15 ***
## Customer_rating         1    0.51    0.51    2.6097    0.1062    
## Cost_of_the_Product     1    7.98    7.98   40.5048 2.039e-10 ***
## Prior_purchases         1    4.35    4.35   22.0505 2.688e-06 ***
## Discount_offered        1  395.43  395.43 2006.5272 < 2.2e-16 ***
## Weight_in_gms           1   60.52   60.52  307.1162 < 2.2e-16 ***
## Residuals           10992 2166.20    0.20                        
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
#Points to note from Anova Analysis

# In summary, the ANOVA analysis indicates that Customer_care_calls, Cost_of_the_Product, Prior_purchas
# Discount_offered, and Weight_in_gms are statistically significant predictors of the target variable
# Reached.on.Time_Y.N. However, Customer_rating does not have a statistically significant relationship
# with the target variable based on the given significance level
```

```r
# Install and load the 'MASS' package for stepwise regression
#install.packages("MASS")
library(MASS)
```

```
## Warning: package 'MASS' was built under R version 4.3.1
```

```r
# Perform stepwise regression using the 'stepAIC' function
model <- lm(Reached.on.Time_Y.N ~ Customer_care_calls + Customer_rating + Cost_of_the_Product + Prior_pu
step_model <- stepAIC(model, direction = "both")  # Stepwise regression with both forward and backward 
```

```
## Start:  AIC=-17857.51
## Reached.on.Time_Y.N ~ Customer_care_calls + Customer_rating + 
##     Cost_of_the_Product + Prior_purchases + Discount_offered + 
##     Weight_in_gms
## 
##                       Df Sum of Sq    RSS    AIC
## <none>                             2166.2 -17858
## - Customer_rating      1     0.623 2166.8 -17856
## - Cost_of_the_Product  1     2.018 2168.2 -17849
## - Prior_purchases      1     5.122 2171.3 -17834
## - Customer_care_calls  1     6.966 2173.2 -17824
## - Weight_in_gms        1    60.524 2226.7 -17556
## - Discount_offered     1   202.778 2369.0 -16875
```

```r
# Print the summary of the final stepwise regression model
summary(step_model)
```

```
## 
## Call:
## lm(formula = Reached.on.Time_Y.N ~ Customer_care_calls + Customer_rating + 
##     Cost_of_the_Product + Prior_purchases + Discount_offered + 
##     Weight_in_gms, data = numerical_shipment)
## 
## Residuals:
##      Min       1Q   Median       3Q      Max
```

```
## -0.73576 -0.46137  0.02599  0.45664  0.77808
##
## Coefficients:
##                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)          8.631e-01  3.278e-02  26.331  < 2e-16 ***
## Customer_care_calls -2.474e-02  4.161e-03  -5.945 2.84e-09 ***
## Customer_rating      5.324e-03  2.995e-03   1.778  0.07548 .
## Cost_of_the_Product -3.014e-04  9.416e-05  -3.200  0.00138 **
## Prior_purchases     -1.466e-02  2.876e-03  -5.098 3.49e-07 ***
## Discount_offered     9.533e-03  2.972e-04  32.077  < 2e-16 ***
## Weight_in_gms       -5.335e-05  3.044e-06 -17.525  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.4439 on 10992 degrees of freedom
## Multiple R-squared:  0.1816, Adjusted R-squared:  0.1812
## F-statistic: 406.6 on 6 and 10992 DF,  p-value: < 2.2e-16
```

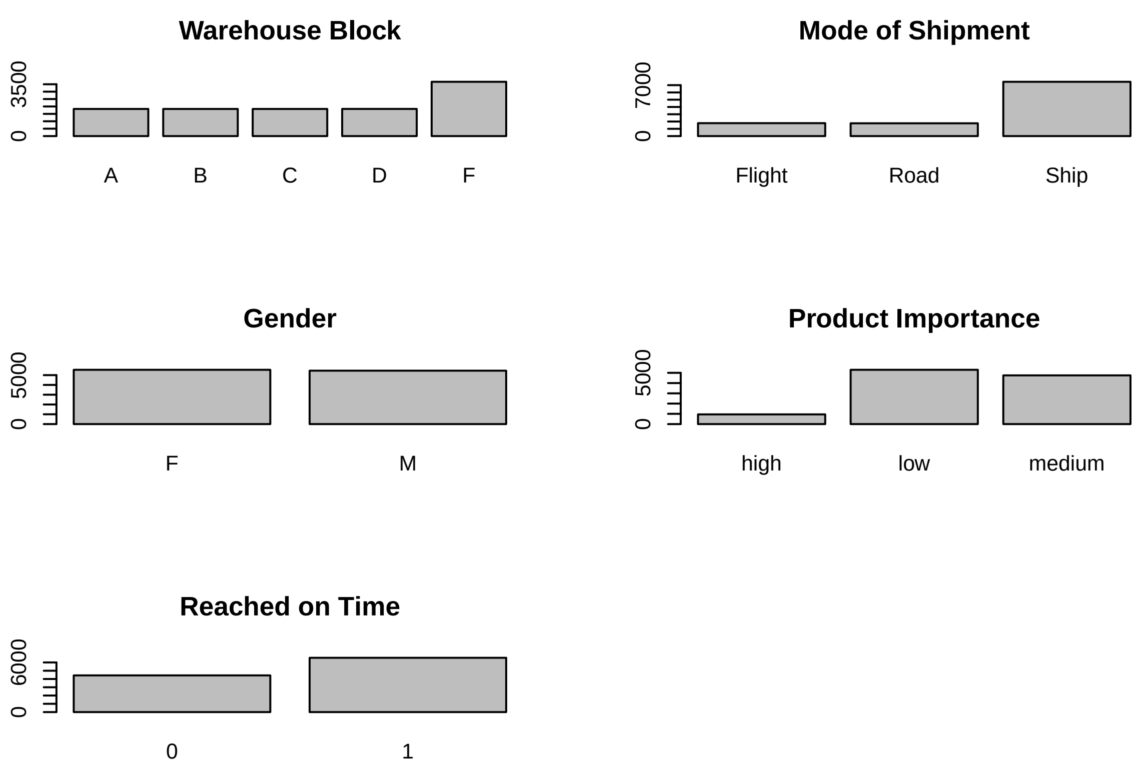
```r
# Based on the stepwise regression analysis, the important variables for
# predicting the target variable "Reached.on.Time_Y.N" are:
#
# 1.Customer_care_calls
# 2.Cost_of_the_Product
# 3.Prior_purchases
# 4.Discount_offered
# 5.Weight_in_gms
# These variables have been selected based on their contribution to the model's fit,
# as indicated by the p-values and the AIC (Akaike Information Criterion) values.
# These variables have statistically significant coefficients and are considered
# important in predicting the target variable.
```

```r
# Now Analyzing Categorical variables
for (col in names(categorical_shipment)) {
  cat("\n")
  cat("Variable:", col, "\n")
  analyze_categorical(shipment_tracking[[col]])
}
```

```
##
## Variable: Warehouse_block
##   Category Count
## 5        F  3666
## 4        D  1834
## 1        A  1833
## 2        B  1833
## 3        C  1833
##
## Variable: Mode_of_Shipment
##   Category Count
## 3     Ship  7462
## 1   Flight  1777
## 2     Road  1760
##
```

```
## Variable: Product_importance
##   Category Count
## 2      low  5297
## 3   medium  4754
## 1     high   948
##
## Variable: Gender
##   Category Count
## 1        F  5545
## 2        M  5454
```

```r
# Set the layout for the plot grid
par(mfrow = c(3, 2))
# Create individual bar plots for each categorical variable
barplot(table(shipment_tracking$Warehouse_block), main = "Warehouse Block")
barplot(table(shipment_tracking$Mode_of_Shipment), main = "Mode of Shipment")
barplot(table(shipment_tracking$Gender), main = "Gender")
barplot(table(shipment_tracking$Product_importance), main = "Product Importance")
barplot(table(shipment_tracking$Reached.on.Time_Y.N), main = "Reached on Time")
# Reset the layout
par(mfrow = c(1, 1))
```




```r
#Performing Chi Square test

# Create a vector of categorical variable names
```

```r
categorical_vars <- c("Warehouse_block", "Mode_of_Shipment","Gender", "Product_importance", "Reached.on

# Initialize an empty list to store the results
chi_square_results <- list()


for (var in categorical_vars) {
  print(var)

  table_data <- table(shipment_tracking[[var]], shipment_tracking$Reached.on.Time_Y.N)
  print(table_data)


  # Perform chi-square test
  chi_result <- chisq.test(table_data)

  # Print the results
  print(chi_result)
}
```

```
## [1] "Warehouse_block"
##
##        0    1
##   A  758 1075
##   B  729 1104
##   C  739 1094
##   D  738 1096
##   F 1472 2194
##
##  Pearson's Chi-squared test
##
## data:  table_data
## X-squared = 1.0894, df = 4, p-value = 0.896
##
## [1] "Mode_of_Shipment"
##
##             0    1
##   Flight  708 1069
##   Road    725 1035
##   Ship   3003 4459
##
##  Pearson's Chi-squared test
##
## data:  table_data
## X-squared = 0.74344, df = 2, p-value = 0.6895
##
## [1] "Gender"
##
##        0    1
##   F 2249 3296
##   M 2187 3267
##
##  Pearson's Chi-squared test with Yates' continuity correction
```

```
##
## data:  table_data
## X-squared = 0.22308, df = 1, p-value = 0.6367
##
## [1] "Product_importance"
##
##             0    1
##   high    332  616
##   low    2157 3140
##   medium 1947 2807
##
##  Pearson's Chi-squared test
##
## data:  table_data
## X-squared = 12.211, df = 2, p-value = 0.00223
##
## [1] "Reached.on.Time_Y.N"
##
##        0    1
##   0 4436    0
##   1    0 6563
##
##  Pearson's Chi-squared test with Yates' continuity correction
##
## data:  table_data
## X-squared = 10995, df = 1, p-value < 2.2e-16
```

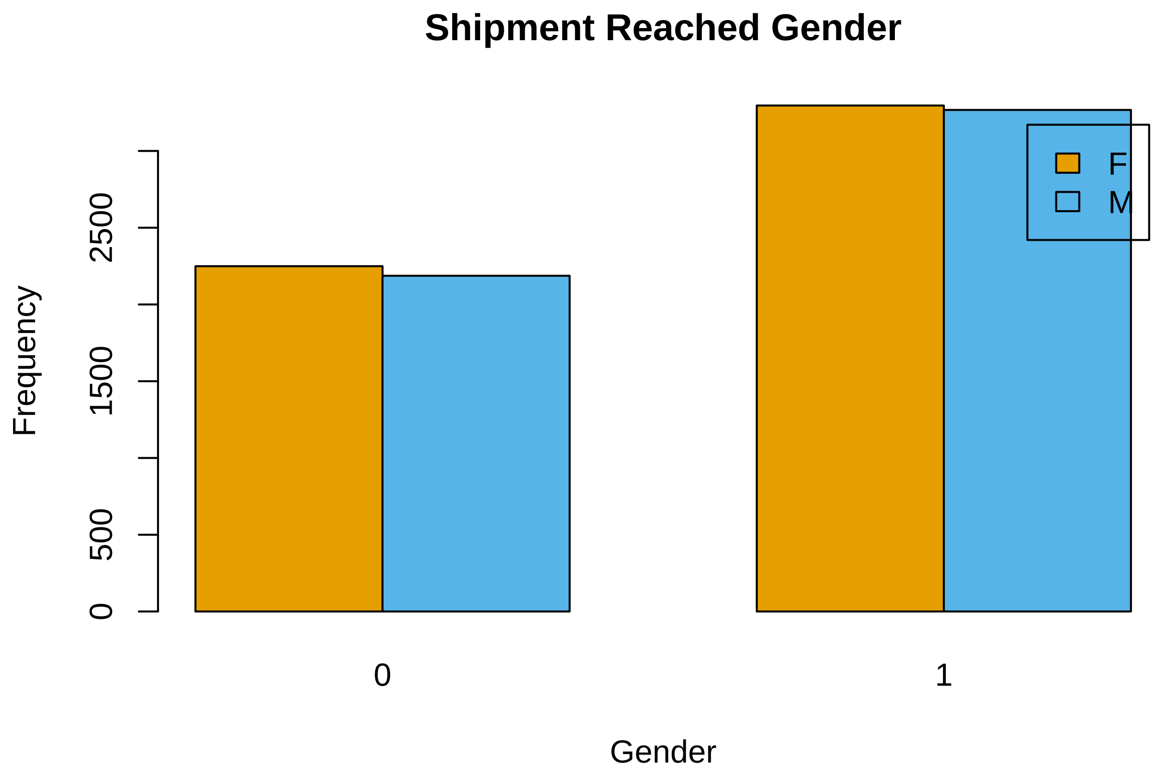
```r
#Performing Chi Square test

# Create a vector of categorical variable names
categorical_vars <- c("Warehouse_block", "Mode_of_Shipment","Gender", "Product_importance", "Reached.on

# Initialize an empty list to store the results
chi_square_results <- list()


for (var in categorical_vars) {
  print(var)

  table_data <- table(shipment_tracking[[var]], shipment_tracking$Reached.on.Time_Y.N)
  print(table_data)

  # Optionally, create a bar plot to visualize the relationship
  barplot(table_data, beside = TRUE, legend = TRUE, col = c("#E69F00", "#56B4E9"),
          main = paste("Shipment Reached", var), xlab = var, ylab = "Frequency")

  # Perform chi-square test
  chi_result <- chisq.test(table_data)

  # Print the results
  print(chi_result)
}
```

```
## [1] "Warehouse_block"
```

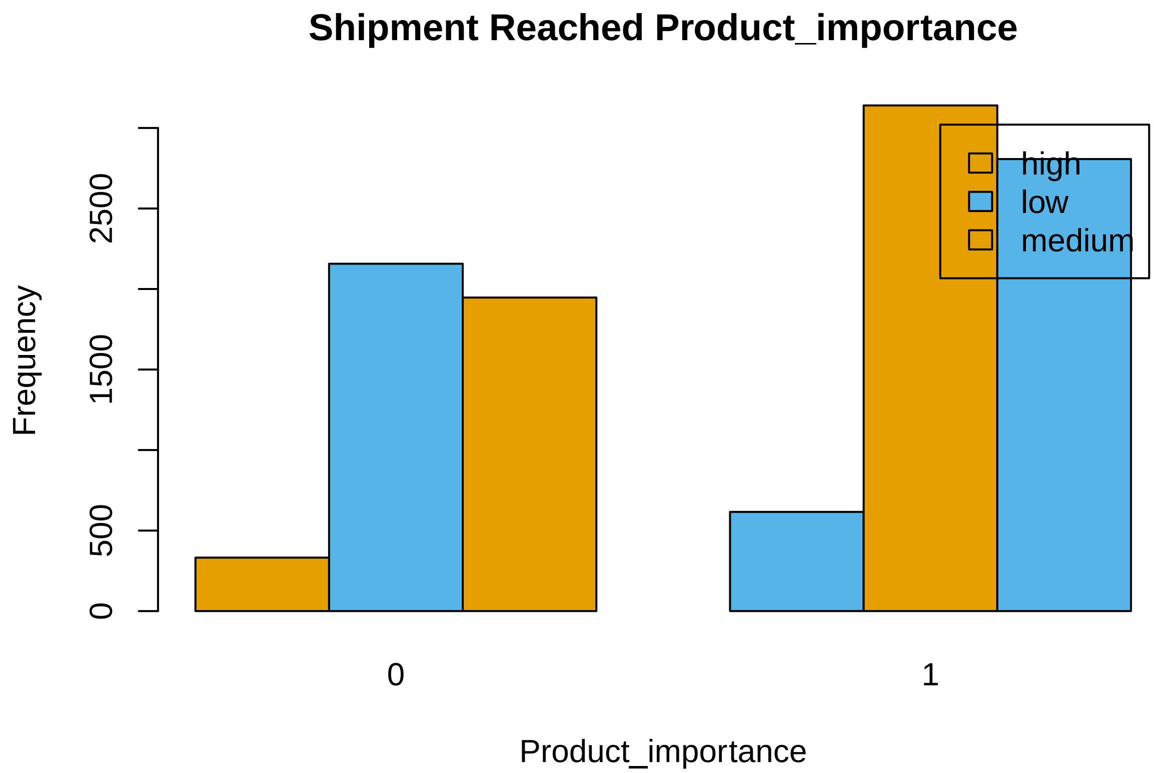
```
##
##         0    1
##   A  758 1075
##   B  729 1104
##   C  739 1094
##   D  738 1096
##   F 1472 2194
```


```
##
##  Pearson's Chi-squared test
##
## data:  table_data
## X-squared = 1.0894, df = 4, p-value = 0.896
##
## [1] "Mode_of_Shipment"
##
##              0    1
##   Flight  708 1069
##   Road    725 1035
##   Ship   3003 4459
```


```
##
##  Pearson's Chi-squared test
##
## data:  table_data
## X-squared = 0.74344, df = 2, p-value = 0.6895
##
## [1] "Gender"
##
##        0    1
##   F 2249 3296
##   M 2187 3267
```




```
##
##  Pearson's Chi-squared test with Yates' continuity correction
##
## data:  table_data
## X-squared = 0.22308, df = 1, p-value = 0.6367
##
## [1] "Product_importance"
##
##             0    1
##   high    332  616
##   low    2157 3140
##   medium 1947 2807
```




```
##
##  Pearson's Chi-squared test
##
## data:  table_data
## X-squared = 12.211, df = 2, p-value = 0.00223
##
## [1] "Reached.on.Time_Y.N"
##
##         0    1
##    0 4436    0
##    1    0 6563
```


```
##
##  Pearson's Chi-squared test with Yates' continuity correction
##
## data:  table_data
## X-squared = 10995, df = 1, p-value < 2.2e-16
```

```
#Important variable from Chi Square test
#Product Importance
```

```
# Identify categorical variables
categorical_vars <- sapply(shipment_tracking, is.factor)

# Convert categorical variables to dummy variables
one_hot_encoded <- model.matrix(~ 0 + Warehouse_block + Mode_of_Shipment + Gender + Product_importance,

# Combine dummy variables with numerical variables
encoded_data <- cbind(numerical_shipment, one_hot_encoded)

#Removing Encoded data
#encoded_data <- select(encoded_data,-"Customer_rating")

# Perform PCA on the encoded data
pca_result <- prcomp(encoded_data, scale. = TRUE)

# Extract the principal components
```

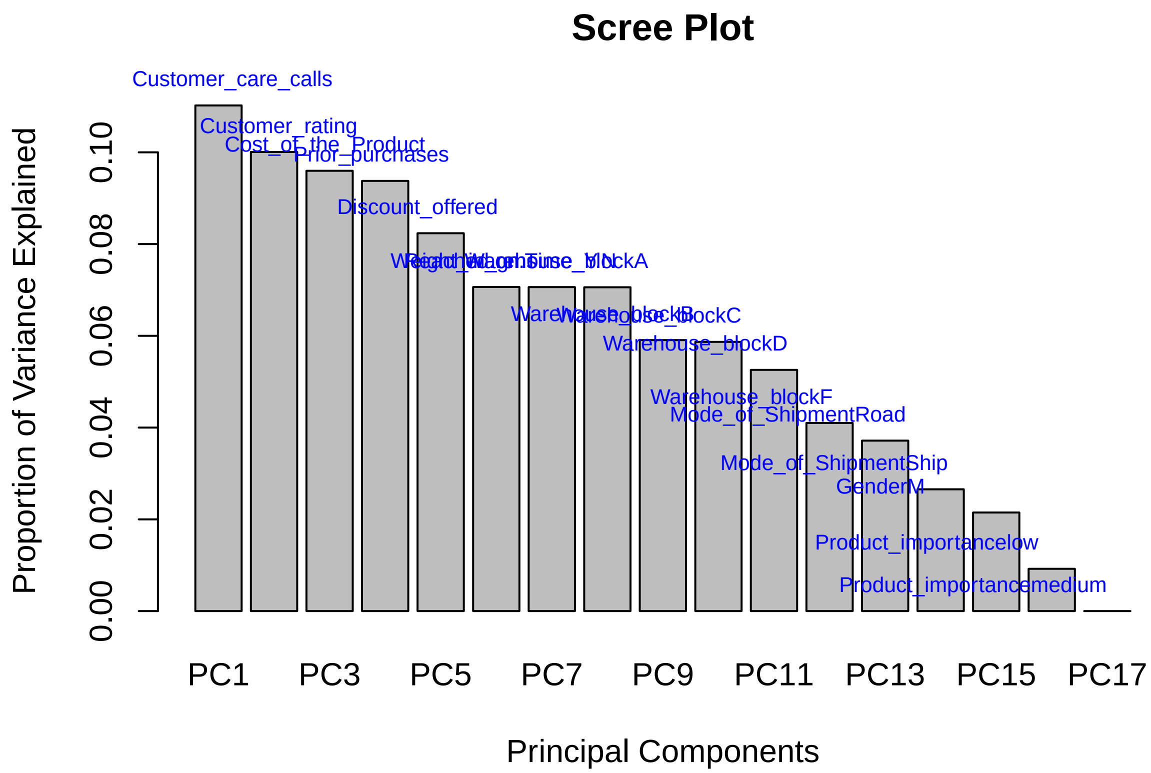
```r
pcs <- pca_result$x

# Extract the standard deviations (square roots of the eigenvalues)
std_dev <- sqrt(pca_result$sdev)

# Extract the proportion of variance explained by each principal component
prop_var <- pca_result$sdev^2 / sum(pca_result$sdev^2)

cumulative_var <- cumsum(prop_var)

selected_pcs <- which(cumulative_var >= 0.7)  # Adjust the threshold as desired

selected_variables <- encoded_data[apply(pcs[, selected_pcs, drop = FALSE], 2, function(x) any(abs(x) >

# Get the variable names
variable_names <- colnames(selected_variables)
print("Selected Features:")
```

```
## [1] "Selected Features:"
```

```r
print(variable_names)
```

```
##  [1] "Customer_care_calls"      "Customer_rating"
##  [3] "Cost_of_the_Product"      "Prior_purchases"
##  [5] "Discount_offered"         "Weight_in_gms"
##  [7] "Reached.on.Time_Y.N"      "Warehouse_blockA"
##  [9] "Warehouse_blockC"         "Warehouse_blockD"
## [11] "Warehouse_blockF"         "Mode_of_ShipmentRoad"
## [13] "Mode_of_ShipmentShip"     "GenderM"
## [15] "Product_importancelow"    "Product_importancemedium"
```

```r
# Get the variable names
variable_names <- colnames(encoded_data)

# Create a scree plot with labeled bars
barplot(prop_var, names.arg = paste0("PC", 1:length(prop_var)), xlab = "Principal Components", ylab = "

# Add variable names to the bars
text(x = 1:length(prop_var), y = prop_var, labels = variable_names, pos = 3, cex = 0.7, xpd = TRUE, col
```



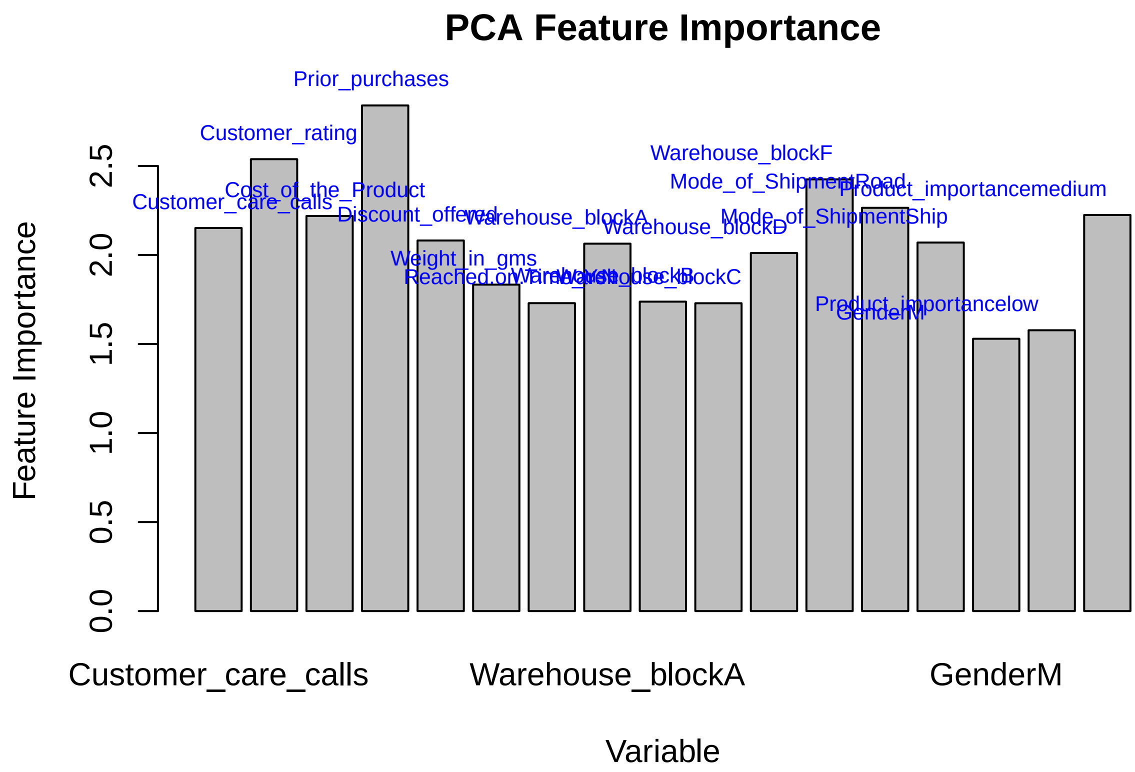

```r
# Rotate x-axis labels if needed
par(las = 2)
```

```r
# Perform PCA
pca_result <- prcomp(encoded_data, scale. = TRUE)

# Extract the principal components
pcs <- pca_result$x

# Extract the loadings (correlation between original features and principal components)
loadings <- pca_result$rotation

# Calculate the feature importance as the absolute sum of loadings across all principal components
feature_importance <- colSums(abs(loadings))

# Create a data frame with variable names and their importance
importance_df <- data.frame(Variable = colnames(encoded_data), Importance = feature_importance)

# Sort the features based on their importance
sorted_importance <- importance_df[order(importance_df$Importance, decreasing = TRUE), ]

# Print the variable names and their importance
print(sorted_importance)
```

```
##                         Variable Importance
```

```
## PC4           Prior_purchases   2.840158
## PC2           Customer_rating   2.538250
## PC12         Warehouse_blockF   2.425027
## PC13     Mode_of_ShipmentRoad   2.264932
## PC17 Product_importancemedium   2.224835
## PC3       Cost_of_the_Product   2.219239
## PC1       Customer_care_calls   2.151928
## PC5          Discount_offered   2.081472
## PC14     Mode_of_ShipmentShip   2.069986
## PC8          Warehouse_blockA   2.063621
## PC11         Warehouse_blockD   2.011003
## PC6             Weight_in_gms   1.833654
## PC9          Warehouse_blockB   1.738084
## PC7       Reached.on.Time_Y.N   1.729600
## PC10         Warehouse_blockC   1.729141
## PC16    Product_importancelow   1.577414
## PC15                  GenderM   1.529469
```

```r
# Perform PCA
pca_result <- prcomp(encoded_data, scale. = TRUE)

# Extract the principal components
pcs <- pca_result$x

# Extract the loadings (correlation between original features and principal components)
loadings <- pca_result$rotation

# Calculate the feature importance as the absolute sum of loadings across all principal components
feature_importance <- colSums(abs(loadings))

# Create a bar plot of feature importance
barplot(feature_importance, names.arg = colnames(encoded_data), xlab = "Variable", ylab = "Feature Impo

# Add text labels for variable names and their positions
text(x = 1:length(feature_importance), y = feature_importance, labels = colnames(encoded_data), pos = 3
```




```r
# Rotate x-axis labels if needed
par(las = 2)
```

```r
library(randomForest)
```

```
## Warning: package 'randomForest' was built under R version 4.3.1

## randomForest 4.7-1.1

## Type rfNews() to see new features/changes/bug fixes.

##
## Attaching package: 'randomForest'

## The following object is masked from 'package:ggplot2':
##
##     margin
```

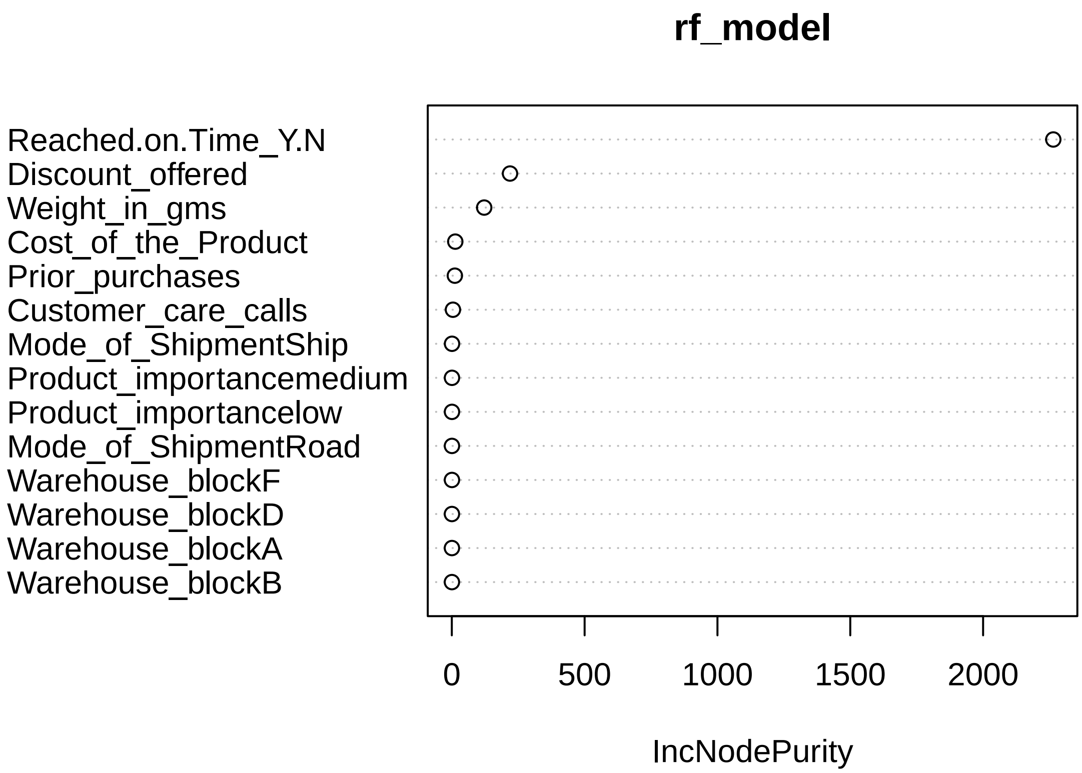
```r
# Split the data into training and testing sets
set.seed(123)  # For reproducibility
train_indices <- sample(1:nrow(encoded_data), 0.7*nrow(encoded_data))
train_data <- shipment_tracking[train_indices, ]
test_data <- shipment_tracking[-train_indices, ]

# Train a random forest model
rf_model <- randomForest(Reached.on.Time_Y.N ~ Customer_care_calls + Cost_of_the_Product + Prior_purcha
```

```
## Warning in randomForest.default(m, y, ...): The response has five or fewer
## unique values.  Are you sure you want to do regression?
```

```r
# Extract feature importance scores
importance_scores <- importance(rf_model)

# Print the feature importance scores
print(importance_scores)
```

```
##                           IncNodePurity
## Customer_care_calls           4.2255907
## Cost_of_the_Product          13.0907002
## Prior_purchases              11.4688948
## Discount_offered            219.1238325
## Weight_in_gms               121.8827396
## Reached.on.Time_Y.N        2264.2645171
## Warehouse_blockA              0.4542272
## Warehouse_blockB              0.4354293
## Warehouse_blockD              0.4701353
## Warehouse_blockF              0.5211670
## Mode_of_ShipmentRoad          0.5524031
## Mode_of_ShipmentShip          0.8161598
## Product_importancelow         0.6144687
## Product_importancemedium      0.6398738
```

```r
# Visualize the feature importance
varImpPlot(rf_model)
```


rf_model
Reached.on.Time_Y.N
Discount_offered
Weight_in_gms
Cost_of_the_Product
Prior_purchases
Customer_care_calls
Mode_of_ShipmentShip
Product_importancemedium
Product_importancelow
Mode_of_ShipmentRoad
Warehouse_blockF
Warehouse_blockD
Warehouse_blockA
Warehouse_blockB
0
500
1000
1500
2000
IncNodePurity