DATA605-FinalExam

John Ferrara

2025-05-05

Final Examination: Business Analytics and Data Science

Instructions:

You are required to complete this take-home final examination by the end of the last week of class. Your solutions should be uploaded in **pdf format** as a knitted document (with graphs, content, commentary, etc. in the pdf). This project will showcase your ability to apply the concepts learned throughout the course.

The dataset you will use for this examination is provided as **`retail_data.csv`**, which contains the following variables:

- **Product_ID**: Unique identifier for each product.
- **Sales**: Simulated sales numbers (in dollars).
- **Inventory_Levels**: Inventory levels for each product.
- **Lead_Time_Days**: The lead time in days for each product.
- **Price**: The price of each product.
- **Seasonality_Index**: An index representing seasonality.

```r
# Reading in the data
retail_data <- read.csv("synthetic_retail_data.csv")

## Overview stats
print(summary(retail_data))
```

```
##    Product_ID        Sales          Inventory_Levels Lead_Time_Days
##  Min.   :  1.00   Min.   :  25.57   Min.   : 67.35   Min.   : 0.491
##  1st Qu.: 50.75   1st Qu.: 284.42   1st Qu.:376.51   1st Qu.: 5.291
##  Median :100.50   Median : 533.54   Median :483.72   Median : 6.765
##  Mean   :100.50   Mean   : 636.92   Mean   :488.55   Mean   : 6.834
##  3rd Qu.:150.25   3rd Qu.: 867.58   3rd Qu.:600.42   3rd Qu.: 8.212
##  Max.   :200.00   Max.   :2447.49   Max.   :858.79   Max.   :12.722
##      Price        Seasonality_Index
##  Min.   : 5.053   Min.   :0.3305
##  1st Qu.:16.554   1st Qu.:0.8475
##  Median :19.977   Median :0.9762
##  Mean   :19.560   Mean   :0.9829
##  3rd Qu.:22.924   3rd Qu.:1.1205
##  Max.   :29.404   Max.   :1.5958
```

```r
# Data types
str(retail_data) # One int column, with the rest being numbers.
```

```
## 'data.frame':    200 obs. of  6 variables:
##  $ Product_ID       : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ Sales            : num  158 279 699 1832 460 ...
```

```
##  $ Inventory_Levels : num  367 427 408 392 448 ...
##  $ Lead_Time_Days   : num  6.31 5.8 3.07 3.53 10.8 ...
##  $ Price            : num  18.8 26.1 22.4 27.1 18.3 ...
##  $ Seasonality_Index: num  1.184 0.857 0.699 0.698 0.841 ...
```

```
# Nulls check
colSums(is.na(retail_data)) # no nulls
```

```
##        Product_ID             Sales  Inventory_Levels    Lead_Time_Days
##                 0                 0                 0                 0
##             Price Seasonality_Index
##                 0                 0
```

```
## Examining the actual data before getting started
head(retail_data)
```

```
##   Product_ID     Sales Inventory_Levels Lead_Time_Days    Price
## 1          1  158.4395         367.4421       6.314587 18.79520
## 2          2  278.9902         426.6512       5.800673 26.08964
## 3          3  698.8587         407.6394       3.071936 22.39998
## 4          4 1832.3947         392.3912       3.534253 27.09201
## 5          5  459.7039         448.3120      10.802241 18.30782
## 6          6 1692.5191         547.4102      10.147199 23.48068
##   Seasonality_Index
## 1         1.1839497
## 2         0.8573051
## 3         0.6986774
## 4         0.6975404
## 5         0.8407251
## 6         1.1319305
```

Problem 1: Business Risk and Revenue Modeling

Context:

You are a data scientist working for a retail chain that models sales, inventory levels, and the impact of pricing and seasonality on revenue. Your task is to analyze various distributions that can describe sales variability and forecast potential revenue.

Part 1: Empirical and Theoretical Analysis of Distributions (5 Points)

Task:

1. **Generate and Analyze Distributions:**
    - **X ~ Sales**: Consider the `Sales` variable from the dataset. Assume it follows a **Gamma distribution** and estimate its shape and scale parameters using the `fitdistr` function from the **MASS** package.

```
## SALES DATA

## Usingf the fitdistr package as outlined.
sales_gamma <- fitdist(retail_data$Sales, "gamma")
summary(sales_gamma)
```

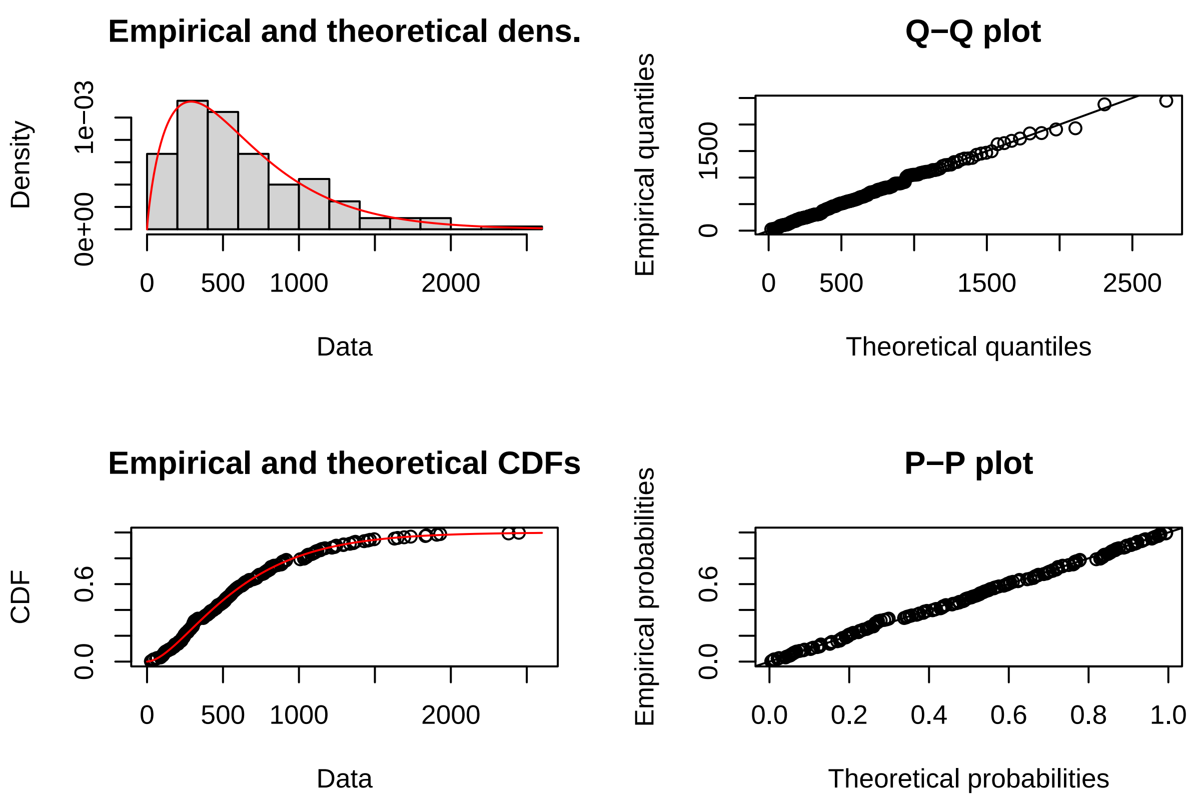
```
## Fitting of the distribution ' gamma ' by maximum likelihood
## Parameters :
##          estimate   Std. Error
```

```
## shape 1.834565518 0.1325722042
## rate  0.002880745 0.0002047032
## Loglikelihood:  -1472.939   AIC:  2949.878   BIC:  2956.475
## Correlation matrix:
##           shape      rate
## shape 1.0000000 0.7774869
## rate  0.7774869 1.0000000
```

```
## Sales Shape: 1.834 with a Std. Error of 0.1325
## Sales Rate: 0.0028 with a std. Error of 0.000204

## Calculating the scale of the data with the results from above.
scale <- 1/ sales_gamma$estimate['rate']
print(scale) ## 347.1325
```

```
##     rate
## 347.1325
```

```
## The scale of the sales data is 347.13

## Plotting for good measure.
plot(sales_gamma)
```



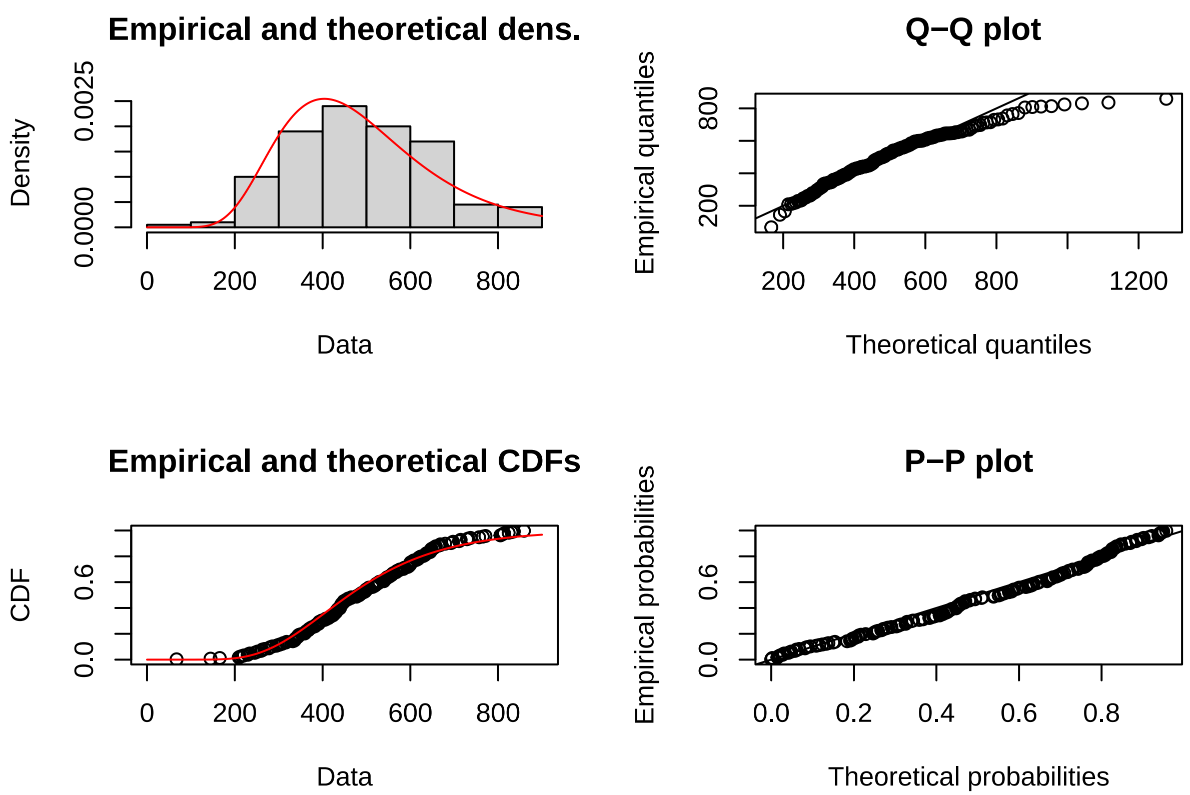
Sales Shape: 1.834 with a Std. Error of 0.1325. Sales Rate: 0.0028 with a std. Error of 0.000204

- **Y** ~ **Inventory Levels**: Assume that the sum of inventory levels across similar products follows a **Lognormal distribution**. Estimate the parameters for this distribution.

```r
## Inventory Levels
inventory_lognorm <- fitdist(retail_data$Inventory_Levels, "lnorm")
summary(inventory_lognorm)
```

```
## Fitting of the distribution ' lnorm ' by maximum likelihood
## Parameters :
##          estimate Std. Error
## meanlog 6.1330368 0.02569112
## sdlog   0.3633273 0.01816574
## Loglikelihood:  -1307.905   AIC:  2619.81   BIC:  2626.406
## Correlation matrix:
##              meanlog        sdlog
## meanlog 1.000000e+00 2.072557e-11
## sdlog   2.072557e-11 1.000000e+00
```

```r
## Est. PARAMETERS:
## Inventory meanlog 6.13 with std. error 0.0256
## Inventory sdlog 0.3633 with std. error 0.01816


# Plotting for Good Measure
plot(inventory_lognorm)
```



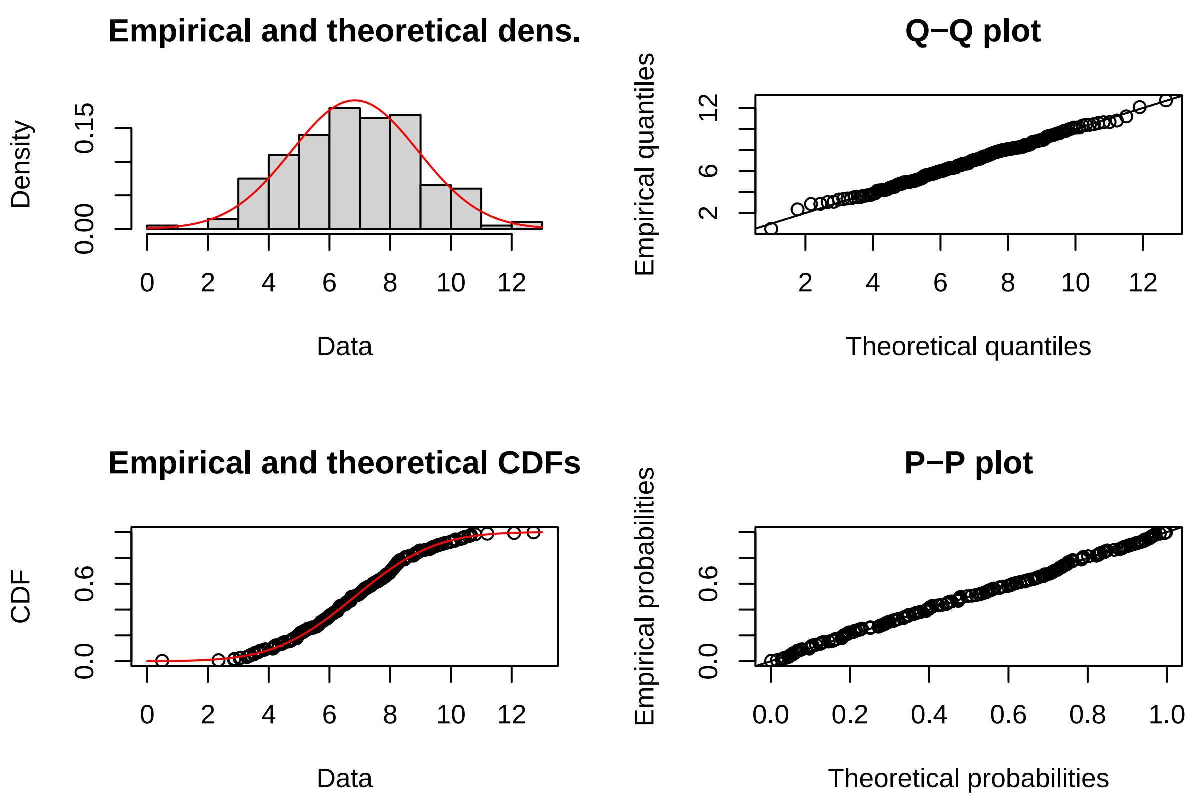
Inventory meanlog 6.13 with std. error 0.0256. Inventory sdlog 0.3633 with std. error 0.01816.

- **Z ~ Lead Time**: Assume that `Lead_Time_Days` follows a **Normal distribution**. Estimate the mean and standard deviation.

```r
##Lead_Time_Days
leadtime_norm <- fitdist(retail_data$Lead_Time_Days, "norm")
# View estimated mean and standard deviation
summary(leadtime_norm)
```

```
## Fitting of the distribution ' norm ' by maximum likelihood
## Parameters :
##      estimate Std. Error
## mean 6.834298  0.1473055
## sd   2.083214  0.1041606
## Loglikelihood:  -430.5701   AIC:  865.1401   BIC:  871.7367
## Correlation matrix:
##      mean sd
## mean    1  0
## sd      0  1
```

```r
## Lead Time mean est. 6.834 with std. error of 0.1473
## Lead time sd est. 2.0832 with std. error of 0.10416

# Plotting for good measure
plot(leadtime_norm)
```



Lead Time mean est. 6.834 with std. error of 0.1473. Lead time sd est. 2.0832 with std. error of 0.10416.

2. **Calculate Empirical Expected Value and Variance:**
   - Calculate the **empirical mean and variance** for all three variables.
   - Compare these empirical values with the **theoretical values** derived from the estimated distribu-

tion parameters.

```r
## Taking the raw data for all three variables for empirical values

# Sales
emp_mean_sales <- mean(retail_data$Sales)
emp_var_sales <- var(retail_data$Sales)
print(paste0("The empirical mean of sales and the empirical variance of sales are ",emp_mean_sales," and
```

```
## [1] "The empirical mean of sales and the empirical variance of sales are 636.91621029515 and 214831.7
```

```r
## Comparison
gamma_shape <- sales_gamma$estimate["shape"]
gamma_rate <- sales_gamma$estimate["rate"]
theo_mean_sales <- gamma_shape / gamma_rate
theo_var_sales <- gamma_shape / (gamma_rate^2)
print(paste0("The estimated via distribution mean of sales and the estimated via distribution variance o
```

```
## [1] "The estimated via distribution mean of sales and the estimated via distribution variance of sale
```

```r
#Inventory
emp_mean_inventory <- mean(retail_data$Inventory_Levels)
emp_var_inventory <- var(retail_data$Inventory_Levels)
print(paste0("The empirical mean of inventory levels and the empirical variance of lnventory levels are
```

```
## [1] "The empirical mean of inventory levels and the empirical variance of lnventory levels are 488.54
```

```r
## Comparison
lnorm_meanlog <- inventory_lognorm$estimate["meanlog"]
lnorm_sdlog <- inventory_lognorm$estimate["sdlog"]
theo_mean_inventory <- exp(lnorm_meanlog + (lnorm_sdlog^2)/2)
theo_var_inventory <- (exp(lnorm_sdlog^2) - 1) * exp(2 * lnorm_meanlog + lnorm_sdlog^2)
print(paste0("The estimated via distribution mean of inventory levels and the estimated via distribution
```

```
## [1] "The estimated via distribution mean of inventory levels and the estimated via distribution varia
```

```r
# Lead Time
emp_mean_lead_time <- mean(retail_data$Lead_Time_Days)
emp_var_lead_time <- var(retail_data$Lead_Time_Days)
print(paste0("The empirical mean of lead time and the empirical variance of lead time are ",emp_mean_lea
```

```
## [1] "The empirical mean of lead time and the empirical variance of lead time are 6.83429809368 and 4
```

```r
## Comparison
theo_mean_lead <- leadtime_norm$estimate["mean"]
theo_var_lead <- leadtime_norm$estimate["sd"]^2
print(paste0("The estimated via distribution mean of lead time and estimated via distribution variance o
```

```
## [1] "The estimated via distribution mean of lead time and estimated via distribution variance of lead
```

Overall, the theoretical estimates from the fitted distributions are pretty close to the empirical calculations. For sales, the Gamma distribution provides a strong fit, with minimal difference in mean and variance. For inventory levels, the distribution shows a slightly higher theoretical variance. Overall it still seems to support the Lognormal distribution assumption. Finally, lead time aligns very closely with the Normal distribution assumption.

Part 2: Probability Analysis and Independence Testing (5 Points)

Task:

1. **Empirical Probabilities**
   For the `Lead_Time_Days` variable (assumed to be normally distributed), calculate the following empirical probabilities:

   - $$(P(Z > \mu \mid Z > \mu - \sigma))$$

```
#Leat TIme variables from above
mu <- theo_mean_lead
sigma <- leadtime_norm$estimate["sd"]
probability <- (sum(retail_data$Lead_Time_Days > mu)) / (sum(retail_data$Lead_Time_Days > (mu - sigma))
print(paste0("Probability is ",round(probability,2)))
```

```
## [1] "Probability is 0.59"
```

- $$(P(Z > \mu + \sigma \mid Z > \mu))$$

```
probability <- (sum(retail_data$Lead_Time_Days > (mu + sigma))) / (sum(retail_data$Lead_Time_Days > mu)
print(paste0("Probability is ",round(probability,2)))
```

```
## [1] "Probability is 0.31"
```

- $$(P(Z > \mu + 2\sigma \mid Z > \mu))$$

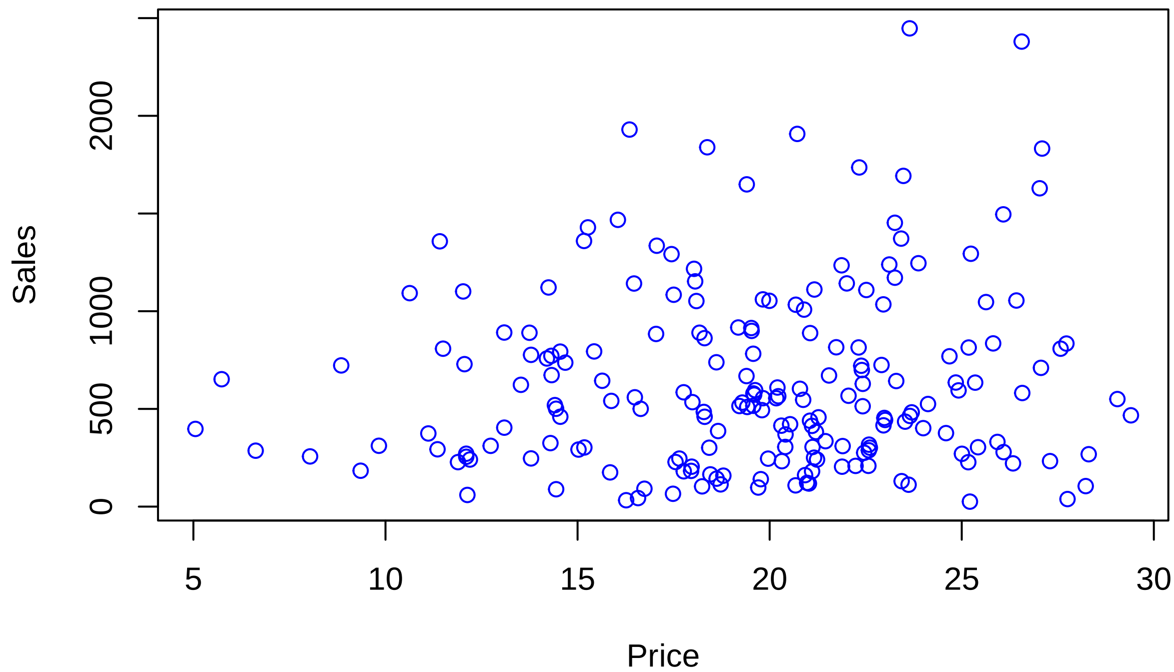
```
probability <- sum(retail_data$Lead_Time_Days > (mu + 2*sigma)) / sum(retail_data$Lead_Time_Days > mu)
print(paste0("Probability is ",round(probability,2)))
```

```
## [1] "Probability is 0.03"
```

2. **Correlation and Independence**

   - Investigate the **correlation** between `Sales` and `Price`.
     Create a **contingency table** using **quartiles** of `Sales` and `Price`, and then evaluate the **marginal** and **joint probabilities**.

```
## Firstly plotting these two variables
plot(retail_data$Price, retail_data$Sales,
     xlab = "Price",
     ylab = "Sales",
     col = "blue")
```




```
## Getting actual correlation
correlation <- cor(retail_data$Sales, retail_data$Price)
print(paste0("COrrelation between slaes and price is: ", correlation))
```

```
## [1] "COrrelation between slaes and price is: 0.102727304237703"
```

```
## Creating the contingency table w/ quartiles
retail_data_w_qrtl <- retail_data %>%  mutate(Sales_Quartile = ntile(Sales, 4),  Price_Quartile = ntile
## Checking results
print(head(retail_data_w_qrtl))
```

```
##   Product_ID     Sales Inventory_Levels Lead_Time_Days    Price
## 1          1  158.4395         367.4421       6.314587 18.79520
## 2          2  278.9902         426.6512       5.800673 26.08964
## 3          3  698.8587         407.6394       3.071936 22.39998
## 4          4 1832.3947         392.3912       3.534253 27.09201
## 5          5  459.7039         448.3120      10.802241 18.30782
## 6          6 1692.5191         547.4102      10.147199 23.48068
##   Seasonality_Index Sales_Quartile Price_Quartile
## 1         1.1839497              1              2
## 2         0.8573051              1              4
## 3         0.6986774              3              3
## 4         0.6975404              4              4
## 5         0.8407251              2              2
## 6         1.1319305              4              4
```

```
## making table
contingency_table <- table(retail_data_w_qrtl$Sales_Quartile, retail_data_w_qrtl$Price_Quartile)
```

```r
print(contingency_table)
```

```
##
##      1  2  3  4
##   1 11 16 12 11
##   2 13 10 15 12
##   3 15 10 13 12
##   4 11 14 10 15
```

```r
## Evaluation of the Marginal and Joing Probs.
joint_prob <- prop.table(contingency_table)
print("Here is the join probabilities table, displaying the chances that a product lands in the respecti
```

```
## [1] "Here is the join probabilities table, displaying the chances that a product lands in the respect
```

```r
print(joint_prob)
```

```
##
##         1     2     3     4
##   1 0.055 0.080 0.060 0.055
##   2 0.065 0.050 0.075 0.060
##   3 0.075 0.050 0.065 0.060
##   4 0.055 0.070 0.050 0.075
```

```r
## Mariginal probablities
sales_marginals <- prop.table(rowSums(contingency_table))
print("The following is the marginal probabiltiy for the Sales quartiles and as it should be is 25%")
```

```
## [1] "The following is the marginal probabiltiy for the Sales quartiles and as it should be is 25%"
```

```r
print(round(sales_marginals, 4))
```

```
##    1    2    3    4
## 0.25 0.25 0.25 0.25
```

```r
price_marginals <- prop.table(colSums(contingency_table))
print("The following is the marginal probabiltiy for the Price quartiles and as it should be is 25%")
```

```
## [1] "The following is the marginal probabiltiy for the Price quartiles and as it should be is 25%"
```

```r
print(round(price_marginals, 4))
```

```
##    1    2    3    4
## 0.25 0.25 0.25 0.25
```

- Use **Fisher's Exact Test** and the **Chi-Square Test** to check for **independence** between `Sales` and `Price`.
  Discuss which test is most appropriate and why.

```r
### Independence Tests
chi_sq_test <- chisq.test(contingency_table)
print(chi_sq_test)
```

```
##
##  Pearson's Chi-squared test
##
## data:  contingency_table
## X-squared = 4.8, df = 9, p-value = 0.8514
```

```r
print(" THe above chi squared text yielded a p value of 0.8514 which is much above the 0.05 limit for s
```

```
## [1] " THe above chi squared text yielded a p value of 0.8514 which is much above the 0.05 limit for s
```

```r
#fisher_test <- fisher.test(contingency_table)
print("Lastly, the Fisher test failed. This is because of the size of the contingency table itself. Itr
```

```
## [1] "Lastly, the Fisher test failed. This is because of the size of the contingency table itself. Itr
```

Problem 2: Advanced Forecasting and Optimization (Calculus) in Retail

Context:

You are working for a large retail chain that wants to optimize pricing, inventory management, and sales forecasting using data-driven strategies. Your task is to use regression, statistical modeling, and calculus-based methods to make informed decisions.

Part 1: Descriptive and Inferential Statistics for Inventory Data (5 Points)

Task:

1. **Inventory Data Analysis:**
    - Generate **univariate descriptive statistics** for the `Inventory_Levels` and `Sales` variables.

```r
## Descriptive Stats for Two variables ofi interest
print("INventory Levels")
```

```
## [1] "INventory Levels"
```

```r
print(summary(retail_data$Inventory_Levels))
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   67.35  376.51  483.72  488.55  600.42  858.79
```

```r
print(paste0("standard dev: ", sd(retail_data$Inventory_Levels)))
```

```
## [1] "standard dev: 155.04659439169"
```

```r
print("Sales")
```

```
## [1] "Sales"
```

```r
print(summary(retail_data$Sales))
```

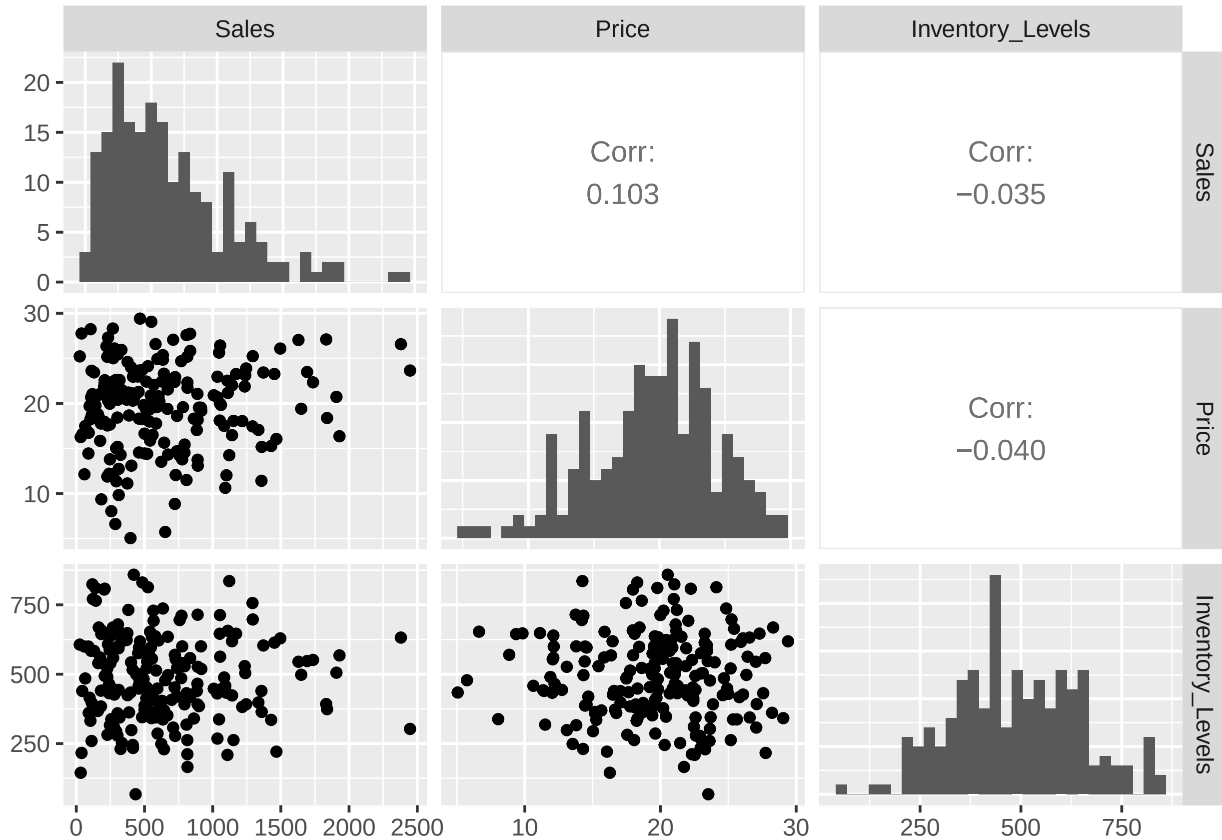
```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   25.57  284.42  533.54  636.92  867.58 2447.49
```

```r
print(paste0("standard dev: ", sd(retail_data$Sales)))
```

```
## [1] "standard dev: 463.499461630208"
```

- Create appropriate **visualizations** such as **histograms** and **scatterplots** for `Inventory_Levels`, `Sales`, and `Price`.

```r
## Creating Sub df with sales price and inv. lvls
retail_subset <- retail_data[, c("Sales", "Price", "Inventory_Levels")]

# Plot ggpairs with histograms
print(ggpairs(retail_subset, diag = list(continuous = wrap("barDiag"))))
```



- Compute a **correlation matrix** for `Sales`, `Price`, and `Inventory_Levels`.

```
## USing the same subset df as before
cor_matrix <- cor(retail_subset,)# method = "pearson")
print("Correlation Matrix")
```

```
## [1] "Correlation Matrix"
```

```
print(cor_matrix)
```

```
##                        Sales       Price Inventory_Levels
## Sales             1.00000000  0.10272730      -0.03529619
## Price             0.10272730  1.00000000      -0.04025941
## Inventory_Levels -0.03529619 -0.04025941       1.00000000
```

- Test the hypotheses that the **correlations between the variables are zero** and provide a **95% confidence interval**.

```
## Sales v Price
print(cor.test(retail_data$Sales, retail_data$Price))
```

```
##
##  Pearson's product-moment correlation
##
## data:  retail_data$Sales and retail_data$Price
## t = 1.4532, df = 198, p-value = 0.1478
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  -0.03653442  0.23807516
```

```
## sample estimates:
##       cor
## 0.1027273
```

```r
## Price v Inventory Levels
print(cor.test(retail_data$Price, retail_data$Inventory_Levels))
```

```
##
##  Pearson's product-moment correlation
##
## data:  retail_data$Price and retail_data$Inventory_Levels
## t = -0.56696, df = 198, p-value = 0.5714
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  -0.17800614  0.09903478
## sample estimates:
##         cor
## -0.04025941
```

```r
## Inventory Levels v Sales
print(cor.test(retail_data$Inventory_Levels, retail_data$Sales))
```

```
##
##  Pearson's product-moment correlation
##
## data:  retail_data$Inventory_Levels and retail_data$Sales
## t = -0.49697, df = 198, p-value = 0.6198
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  -0.1731891  0.1039539
## sample estimates:
##         cor
## -0.03529619
```

```r
print("For each of these tests the p-values are above 0.05, meaning none of them are statistically signi
```

```
## [1] "For each of these tests the p-values are above 0.05, meaning none of them are statistically sign
```

2. **Discussion:**
   - Explain the meaning of your findings and discuss the **implications of the correlations for inventory management**.
   - Would you be concerned about **multicollinearity** in a potential regression model? Why or why not?

These tets and analyses demonstrated that there is no strong linear relatinoships between any of these three variables in this retail dataset. In practical terms, increasing sales will not have a strong impact on either of the other two variables, and similarly, changing inventroy levels or price would not have much impact on the other ones. With this being said, that implies that one should not be worried about multicolinearity between these variables, as they are not showing strong relationships wioth one another.

Part 2: Linear Algebra and Pricing Strategy (5 Points)

Task:

1. **Price Elasticity of Demand:**
   - Use **linear regression** to model the relationship between `Sales` and `Price` (assuming `Sales` as the dependent variable).

```r
# Linear Regrssion Model
# Linear regression: Sales ~ Price
lm_model <- lm(Sales ~ Price, data = retail_data)
print(summary(lm_model))
```

```
##
## Call:
## lm(formula = Sales ~ Price, data = retail_data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -679.54 -347.85  -98.63  241.12 1770.08
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  442.951    137.419   3.223  0.00148 **
## Price          9.916      6.824   1.453  0.14775
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 462.2 on 198 degrees of freedom
## Multiple R-squared:  0.01055,    Adjusted R-squared:  0.005556
## F-statistic: 2.112 on 1 and 198 DF,  p-value: 0.1478
```

- **Invert the correlation matrix** from your model, and calculate the **precision matrix**.

```r
## Correlation and Precision Matrix
cor_matrix <- cor(retail_data[, c("Sales", "Price")])
print("correlation Matrix:")
```

```
## [1] "correlation Matrix:"
```

```r
print(cor_matrix)
```

```
##           Sales     Price
## Sales 1.0000000 0.1027273
## Price 0.1027273 1.0000000
```

```r
precision_matrix <- solve(cor_matrix)
print("Precision Matrix:")
```

```
## [1] "Precision Matrix:"
```

```r
print(precision_matrix)
```

```
##            Sales      Price
## Sales  1.0106655 -0.1038229
## Price -0.1038229  1.0106655
```

- Discuss the implications of the **diagonal elements of the precision matrix** (which are **variance inflation factors**).

The implications of the diagonal elements of the precision matrix sugest no multicolinearity between the two variables. The values at aore extremely close to 1 imply this. Sales and Price are NOT linearly related to one another. ‘

- Perform **LU decomposition** on the correlation matrix and interpret the results in the context of **price elasticity**.

```r
mtx_cor <- Matrix(cor_matrix)
lu_decomp <- lu(mtx_cor)
print("------------LU Decomp")
```

```
## [1] "------------LU Decomp"
```

```r
print(lu_decomp)
```

```
## LU factorization of Formal class 'denseLU' [package "Matrix"] with 4 slots
##   ..@ x       : num [1:4] 1 0.103 0.103 0.989
##   ..@ perm    : int [1:2] 1 2
##   ..@ Dim     : int [1:2] 2 2
##   ..@ Dimnames:List of 2
##   .. ..$ : chr [1:2] "Sales" "Price"
##   .. ..$ : chr [1:2] "Sales" "Price"
```

```r
lu_expanded <- expand(lu_decomp)
print("-----------LU Expanded")
```

```
## [1] "-----------LU Expanded"
```

```r
print(lu_expanded)
```

```
## $L
## 2 x 2 Matrix of class "dtrMatrix" (unitriangular)
##      [,1]      [,2]
## [1,] 1.0000000         .
## [2,] 0.1027273 1.0000000
##
## $U
## 2 x 2 Matrix of class "dtrMatrix"
##      [,1]      [,2]
## [1,] 1.0000000 0.1027273
## [2,]         . 0.9894471
##
## $P
## 2 x 2 sparse Matrix of class "pMatrix"
##
## [1,] | .
## [2,] . |
```

Overall the non diagonal values for the lower and upper matrices are low values. THis implies a lack of a correlation or relationship between the two vairables: Sales and Price. Due to this weak correlation, it means that price does not really impact the number of sales. Thus, if we assume sales is a proxy for demand, the demand is not indicative of elasticity in price.

Part 3: Calculus-Based Probability & Statistics for Sales Forecasting (5 Points)

Task:

1. **Sales Forecasting Using Exponential Distribution:**
    - Identify a variable in the dataset that is skewed to the right (e.g., `Sales` or `Price`) and fit an **exponential distribution** to this data using the `fitdistr` function.

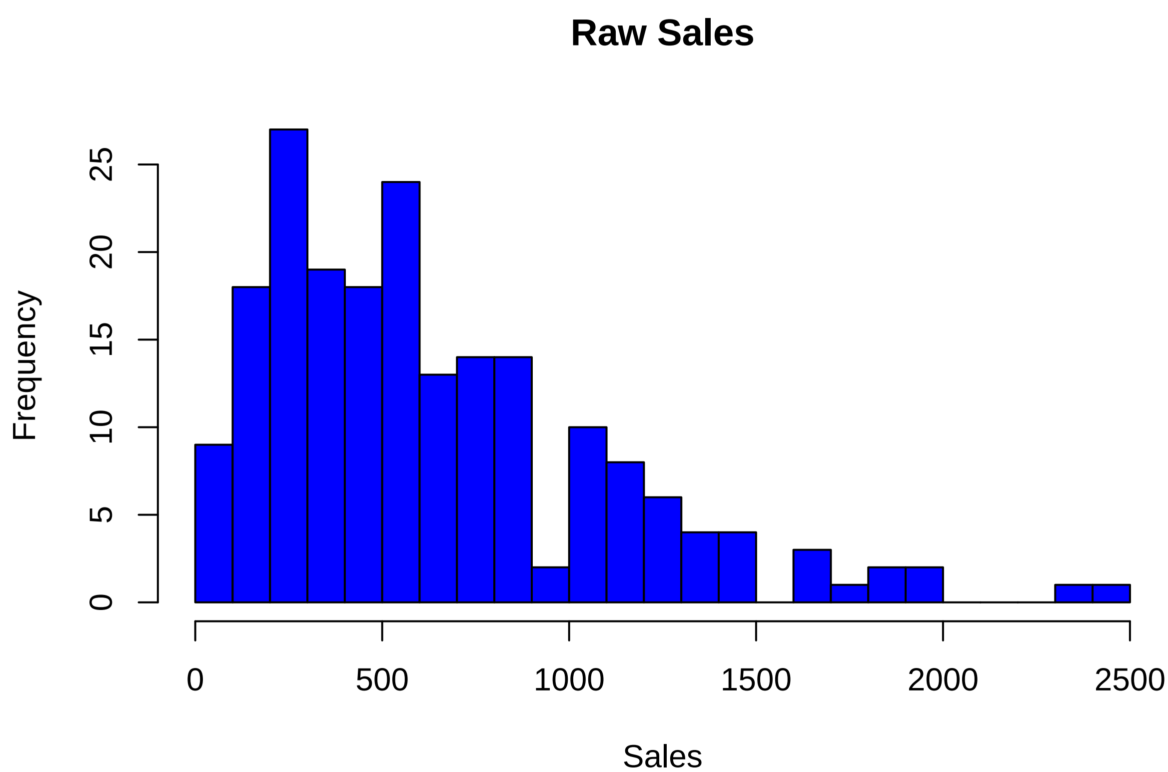
```r
# Going by the GGPairs plot earlier Sales is right skewed.
exp_fit <- fitdistr(retail_data$Sales, "exponential")
print(exp_fit)
```

```
##          rate
##    0.0015700652
##   (0.0001110204)
```

```r
# 0.0015700652
```

- Generate **1,000 samples** from the fitted exponential distribution and compare a **histogram** of these samples with the original data's histogram.

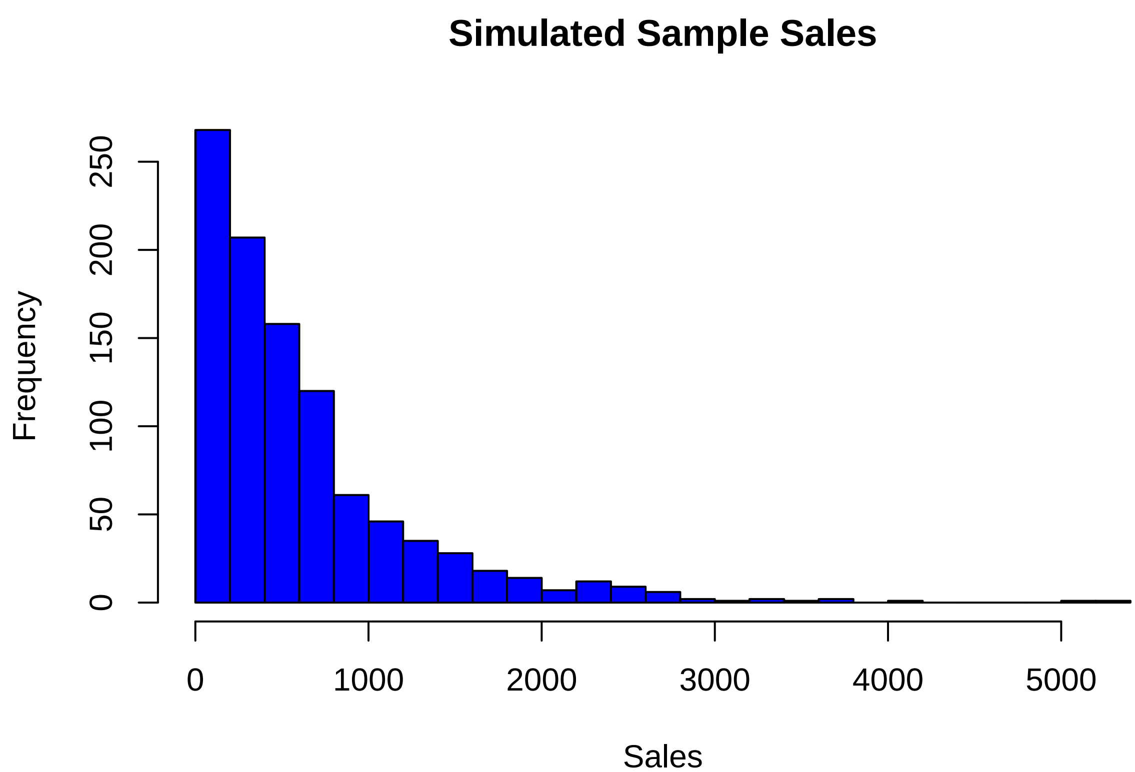
```r
set.seed(100)

# Random Samples from dist
samples <- rexp(1000, rate = exp_fit$estimate)


print(hist(retail_data$Sales, breaks = 30, col = "blue",
     main = "Raw Sales", xlab = "Sales"))
```




```
## $breaks
##  [1]    0  100  200  300  400  500  600  700  800  900 1000 1100 1200 1300 1400
## [16] 1500 1600 1700 1800 1900 2000 2100 2200 2300 2400 2500
##
## $counts
##  [1]  9 18 27 19 18 24 13 14 14  2 10  8  6  4  4  0  3  1  2  2  0  0  0  1  1
##
## $density
##  [1] 0.00045 0.00090 0.00135 0.00095 0.00090 0.00120 0.00065 0.00070 0.00070
## [10] 0.00010 0.00050 0.00040 0.00030 0.00020 0.00020 0.00000 0.00015 0.00005
```

```
## [19] 0.00010 0.00010 0.00000 0.00000 0.00000 0.00005 0.00005
##
## $mids
##  [1]   50  150  250  350  450  550  650  750  850  950 1050 1150 1250 1350 1450
## [16] 1550 1650 1750 1850 1950 2050 2150 2250 2350 2450
##
## $xname
## [1] "retail_data$Sales"
##
## $equidist
## [1] TRUE
##
## attr(,"class")
## [1] "histogram"
```

```r
print(hist(samples, breaks = 30, col = "blue",
     main = "Simulated Sample Sales", xlab = "Sales"))
```




```
## $breaks
##  [1]    0  200  400  600  800 1000 1200 1400 1600 1800 2000 2200 2400 2600 2800
## [16] 3000 3200 3400 3600 3800 4000 4200 4400 4600 4800 5000 5200 5400
##
## $counts
##  [1] 268 207 158 120  61  46  35  28  18  14   7  12   9   6   2   1   2   1   2
## [20]   0   1   0   0   0   0   1   1
##
## $density
```

```
##  [1] 0.001340 0.001035 0.000790 0.000600 0.000305 0.000230 0.000175 0.000140
##  [9] 0.000090 0.000070 0.000035 0.000060 0.000045 0.000030 0.000010 0.000005
## [17] 0.000010 0.000005 0.000010 0.000000 0.000005 0.000000 0.000000 0.000000
## [25] 0.000000 0.000005 0.000005
##
## $mids
##  [1]  100  300  500  700  900 1100 1300 1500 1700 1900 2100 2300 2500 2700 2900
## [16] 3100 3300 3500 3700 3900 4100 4300 4500 4700 4900 5100 5300
##
## $xname
## [1] "samples"
##
## $equidist
## [1] TRUE
##
## attr(,"class")
## [1] "histogram"
```

The original plot with the raw data shows the right skewed data break down, when looking at the exponentiual distribution samples histogram, the right skew is more extreme. This is expected with an exponential distribution. However, It may over estimate the frequency of the lower values when looked at in context with the original data.

- Calculate the **5th and 95th percentiles** using the **cumulative distribution function (CDF)** of the exponential distribution.

```
exp_fit_5th_pctnl <- qexp(0.05, rate = exp_fit$estimate)
exp_fit_95th_pctnl <- qexp(0.95, rate = exp_fit$estimate)

print(paste0("5th Percentile: ", exp_fit_5th_pctnl))
```

```
## [1] "5th Percentile: 32.6695306748722"
```

```
print(paste0("95th Percentile: ", exp_fit_95th_pctnl))
```

```
## [1] "95th Percentile: 1908.03044673088"
```

- Compute a **95% confidence interval** for the original data assuming normality and compare it with the empirical percentiles.

```
sales_sd <- sd(retail_data$Sales)
## variable from before
#print(emp_mean_sales)

n <- length(retail_data$Sales)
error_margin <- 1.96 * (sales_sd / sqrt(n))

ci_lower <- emp_mean_sales - error_margin
ci_upper <- emp_mean_sales + error_margin

print(paste("95% CI Lower Bound:", ci_lower))
```

```
## [1] "95% CI Lower Bound: 572.678462265723"
```

```
print(paste("95% CI Upper Bound:", ci_upper))
```

```
## [1] "95% CI Upper Bound: 701.153958324577"
```

```r
### Emprical Values from befdore
print("Empiriocal Values from before with Sales")
```

```
## [1] "Empiriocal Values from before with Sales"
```

```r
print(summary(retail_data$Sales))
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   25.57  284.42  533.54  636.92  867.58 2447.49
```

```r
print(paste0("standard dev: ", sd(retail_data$Sales)))
```

```
## [1] "standard dev: 463.499461630208"
```

```r
empirical_ci <- quantile(retail_data$Sales, probs = c(0.025, 0.975))
print(empirical_ci)
```

```
##       2.5%      97.5%
##   65.48397 1832.55772
```

Looking at the numbers generared for the confidence interval via the normality distribution and comparing it to the empircal values obtained before, one can see the "normality" assumption is likely incorrect. The 95% lower and uppwer bounds are ~572 and ~701 from the most recent calculation assuming normality, but the empirical values for this are different. Using the empirical data we get a lower bound of ~65 and upper bound of ~1832.

2. **Discussion:**
    - Discuss how well the exponential distribution models the data and what this implies for forecasting future sales or pricing.
    - Consider whether a different distribution might be more appropriate.

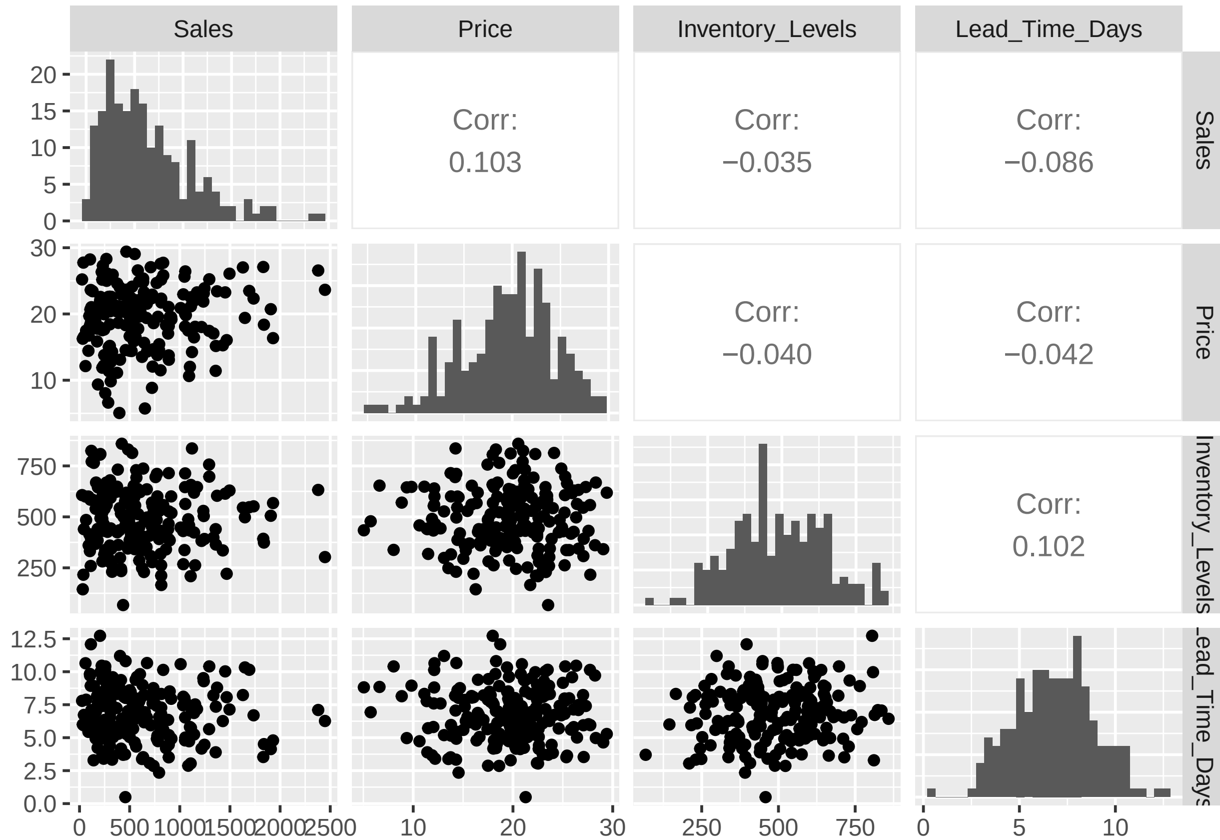
Overall the exponential distribution model of the data help deal with the right skewed nature of the raw sales data. However, the simulated data showed that the estimation of the lower value sales may be overestimated by this model, with the model having an even more extreme skew than the original data. The flip side of this reality is that the higher values may also be underestimated by this model. A potential alternative to the exponential fit model would be a log normal distribution as this is also decent for right-skewed data. This model may accommodate the larger number of higher values, aka the longer right side tail.

Part 4: Regression Modeling for Inventory Optimization (10 Points)

Task:

1. **Multiple Regression Model:**
    - Build a **multiple regression model** to predict `Inventory_Levels` based on `Sales`, `Lead_Time_Days`, and `Price`.

```r
retail_sub_multi <- retail_data[, c("Sales", "Price", "Inventory_Levels", "Lead_Time_Days")]
print(ggpairs(retail_sub_multi, diag = list(continuous = wrap("barDiag"))))
```




```r
## Looking at the ggpairs results and transforming the right skewed variables
retail_sub_multi$log_Sales <- log(retail_sub_multi$Sales + 1)
retail_sub_multi$log_LeadTime <- log(retail_sub_multi$Lead_Time_Days + 1)
retail_sub_multi$log_Inventory <- log(retail_sub_multi$Inventory_Levels + 1)
retail_sub_multi$log_price <- log(retail_sub_multi$Price+1)


multi_model <- lm(log_Inventory ~ log_Sales + log_LeadTime + log_price, data = retail_sub_multi)
print(summary(multi_model))
```

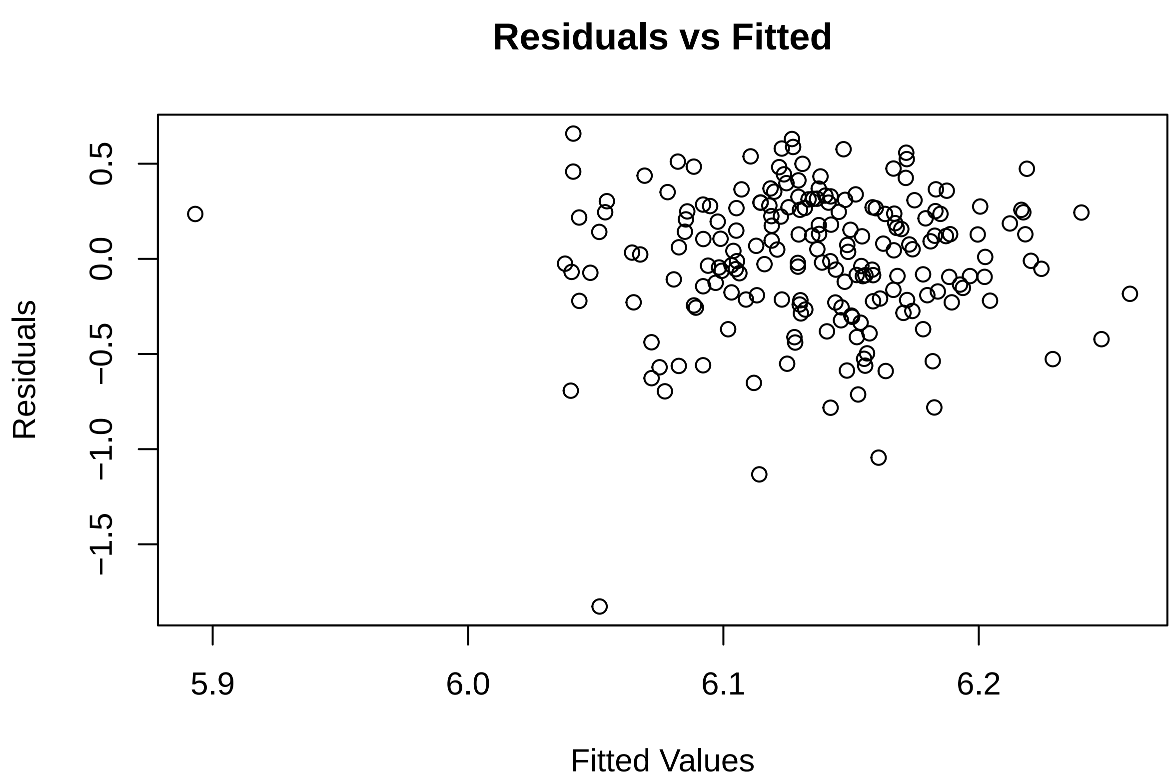
```
##
## Call:
## lm(formula = log_Inventory ~ log_Sales + log_LeadTime + log_price,
##     data = retail_sub_multi)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.82687 -0.21438  0.04694  0.25999  0.65835
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)   6.012962   0.390975  15.379   <2e-16 ***
## log_Sales     0.008023   0.030653   0.262    0.794
## log_LeadTime  0.144350   0.086812   1.663    0.098 .
## log_price    -0.073020   0.095074  -0.768    0.443
## ---
```

```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.3627 on 196 degrees of freedom
## Multiple R-squared:  0.0174, Adjusted R-squared:  0.002355
## F-statistic: 1.157 on 3 and 196 DF,  p-value: 0.3275
```

- Provide a **full summary** of your model, including **coefficients**, **R-squared value**, and **residual analysis**.

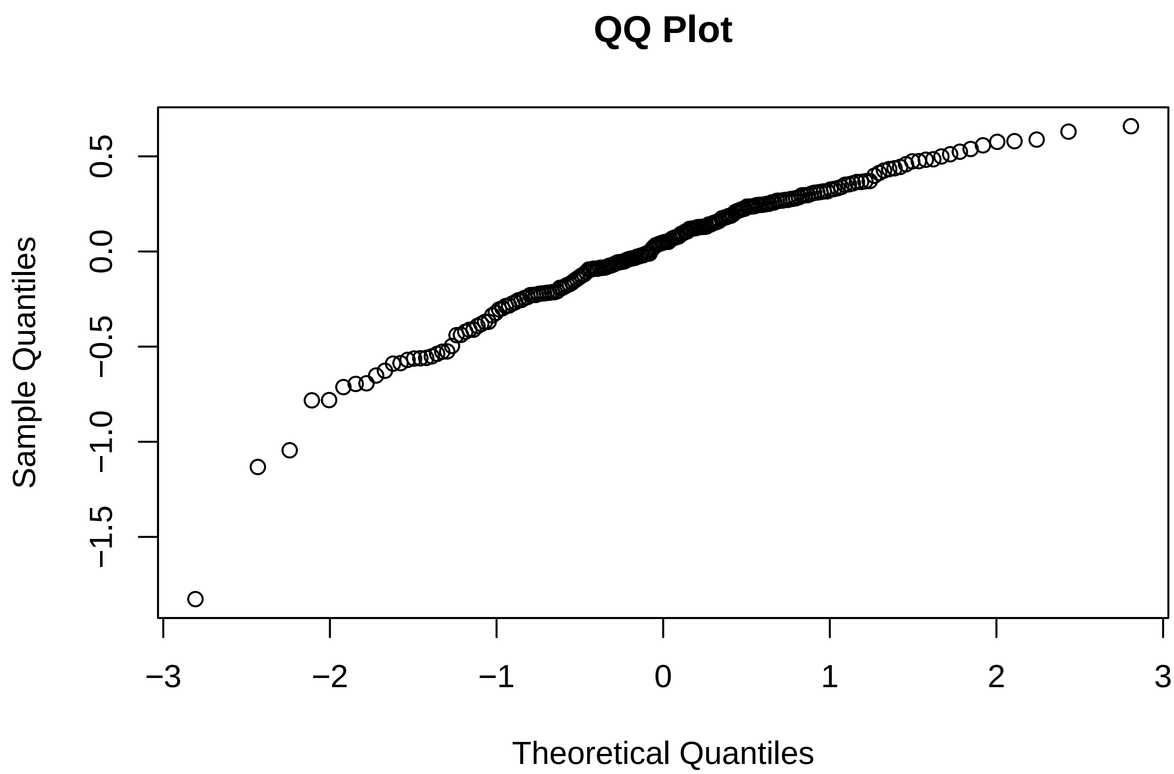
I tried several different transformations on the data in order to attempt ot generate decent model. Mainly, after working through this assignment, a log transformation on the right skewed vairables. Then a log transformation on all of them. Additionally, I tried a square root transformation in a similar manner, but the model still as a super low r^2 (0.0023). In addition to that the p values for all of the variables are not statistically significant. Respectively, the p values were 0.79, 0.098, and 0.443 for the logged sales, logged Lead Time, and logged price predictors. While not statistically significant, the coefficients were -0.073, 0.144, and 0.008 for the logged price, the logged lead time, and the logges sales, repectively. Overall this model is not good. .

```r
## Residual Plotting and overview
residuals <- residuals(multi_model)
fitted_vals <- fitted(multi_model)

print(plot(fitted_vals, residuals,
     main = "Residuals vs Fitted",
     xlab = "Fitted Values",
     ylab = "Residuals"))
```




```
## NULL
```

```r
print(qqnorm(residuals, main = "QQ Plot"))
```

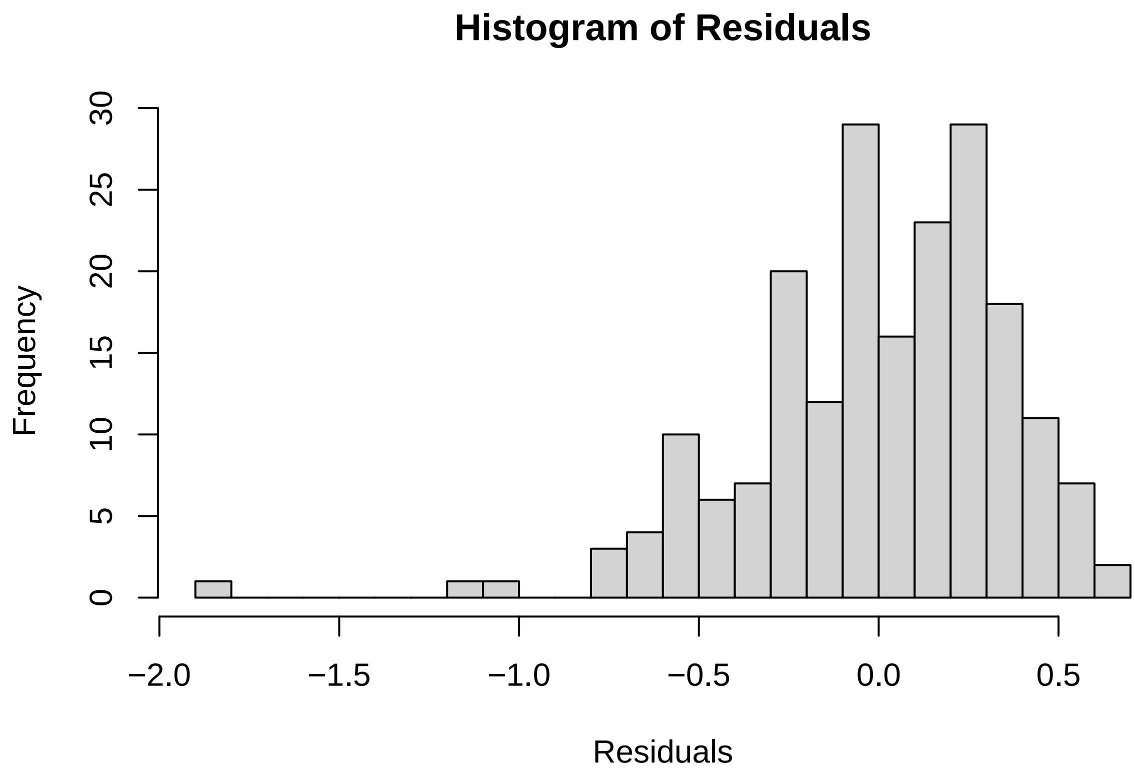



```
## $x
##   [1] -0.651072016 -0.182742585 -0.144633812 -0.312053322 -0.433020331
##   [6]  0.170012889 -0.325239256 -0.590284394  0.446826965 -1.295928846
##  [11] -0.043880072 -1.494672250  0.081555738 -1.780464342  0.984234960
##  [16] -1.162579875 -1.722383890 -0.272809053  1.669592577  0.832724719
##  [21] -2.108358399 -1.325516200 -0.056429069  1.918876226 -0.924934461
##  [26] -0.815126333  1.722383890  0.106734011  1.069154627 -0.488776411
##  [31] -0.031337982  1.187577263 -0.233980651  1.213339622  0.325239256
##  [36] -0.832724719 -0.006266612  0.651072016  0.714367440  0.144633812
##  [41] -1.845258117 -1.669592577 -1.621082251 -0.246881415 -0.338481986
##  [46] -1.213339622  0.560703032 -1.138288582 -1.239933478  0.094137414
##  [51]  1.047215930  1.780464342  0.815126333  1.576111974 -1.114651015
##  [56]  2.241402728 -2.004654462 -0.221118713  0.698283366 -1.576111974
##  [61] -0.850584865 -1.004785806 -0.119347567  1.356311745  0.068986959
##  [66] -0.351784345 -2.807033768  0.233980651  1.091620367 -1.534120544
##  [71]  0.131980140  0.157310685  1.025770021 -0.298921424 -0.698283366
##  [76] -0.170012889  1.295928846  1.239933478 -0.081555738  0.905878812
##  [81] -2.432379059  0.730638483 -1.047215930  1.534120544 -0.868720547
##  [86] -0.620391602  0.780664237  0.747105302 -0.365149249  0.378579699
##  [91]  0.405649708  0.208293252 -0.259823400  0.887146559  1.422090432
##  [96] -1.091620367 -0.575430769  0.031337982 -0.763777244 -1.918876226
## [101]  1.457421739 -2.241402728 -0.208293252  0.531604424  0.419295753
## [106] -0.094137414 -1.025770021  0.365149249 -0.887146559  0.351784345
## [111]  0.221118713  0.460719309 -0.546095926  1.494672250  0.246881415
```

```
## [116] -0.106734011  0.433020331 -0.635657014 -0.446826965  2.432379059
## [121] -0.682377942  0.272809053 -0.378579699 -0.944332036  0.620391602
## [126]  0.285840875  0.924934461  0.635657014  1.004785806 -0.666643306
## [131]  0.605269415  0.392078788 -0.964091607 -0.195501964 -1.356311745
## [136]  0.517223714  0.797776846  1.621082251 -0.018800820  0.488776411
## [141]  0.763777244 -0.780664237 -0.905878812 -0.460719309 -0.405649708
## [146] -0.797776846 -0.605269415  0.964091607  0.018800820  0.182742585
## [151] -0.285840875 -0.560703032  0.590284394  0.298921424  0.682377942
## [156]  1.138288582 -0.714367440  0.944332036 -1.187577263  2.807033768
## [161] -0.517223714  0.546095926 -0.730638483  0.006266612  1.325516200
## [166] -0.747105302  2.004654462  0.119347567  1.114651015  0.338481986
## [171]  0.474701147 -1.267434417 -0.419295753  1.267434417 -0.131980140
## [176]  0.195501964 -1.457421739 -0.531604424  1.388450197 -1.069154627
## [181]  0.868720547  0.259823400  0.043880072  0.502949184 -0.502949184
## [186] -0.157310685  0.575430769 -0.392078788  1.845258117  0.312053322
## [191] -0.474701147  1.162579875  0.666643306 -1.422090432 -0.984234960
## [196]  0.056429069 -1.388450197  2.108358399  0.850584865 -0.068986959
##
## $y
##           1           2           3           4           5           6
## -0.21360743 -0.03567780 -0.02515628 -0.07309035 -0.09456633  0.11987123
##           7           8           9          10          11          12
## -0.07534145 -0.18373095  0.21307039 -0.52520486  0.03296346 -0.56264504
##          13          14          15          16          17          18
##  0.07572783 -0.69270573  0.31615849 -0.41143317 -0.65187199 -0.05690797
##          19          20          21          22          23          24
##  0.49926581  0.29508670 -0.78230734 -0.52677894  0.02310870  0.55792517
##          25          26          27          28          29          30
## -0.28396950 -0.23999392  0.51104077  0.09217455  0.33884270 -0.12587930
##          31          32          33          34          35          36
##  0.03685648  0.36549731 -0.05226506  0.36914410  0.15664658 -0.24475486
##          37          38          39          40          41          42
##  0.04496473  0.25751847  0.26738276  0.10559525 -0.69607341 -0.62686307
##          43          44          45          46          47          48
## -0.58949458 -0.05407332 -0.08140909 -0.43817523  0.24347517 -0.41107224
##          49          50          51          52          53          54
## -0.43972265  0.08007386  0.33253888  0.52427435  0.28533223  0.48206511
##          55          56          57          58          59          60
## -0.39143734  0.58803034 -0.78093037 -0.04521464  0.26666970 -0.58685619
##          61          62          63          64          65          66
## -0.25478011 -0.32268040 -0.01952158  0.43319954  0.07312754 -0.08480989
##          67          68          69          70          71          72
## -1.82686605  0.12917429  0.35075868 -0.56968957  0.10420796  0.11848843
##          73          74          75          76          77          78
##  0.32831688 -0.06857511 -0.21747773 -0.03352362  0.41261142  0.37010459
##          79          80          81          82          83          84
## -0.01024681  0.30821955 -1.13253079  0.27087965 -0.36952622  0.47464291
##          85          86          87          88          89          90
## -0.25628306 -0.19131303  0.27743432  0.27115288 -0.08548561  0.17989413
##          91          92          93          94          95          96
##  0.18762415  0.12827547 -0.05653623  0.30309319  0.44429458 -0.38109204
##          97          98          99         100         101         102
## -0.17627878  0.05064778 -0.22838991 -0.71256438  0.45855658 -1.04469036
##         103         104         105         106         107         108
```

```
## -0.04096436  0.23636297  0.19564824 -0.01194582 -0.33546014  0.17723383
##         109         110         111         112         113         114
## -0.26634690  0.17426617  0.12843615  0.21731766 -0.16279536  0.47354161
##         115         116         117         118         119         120
##  0.12968791 -0.01241970  0.20744777 -0.20856132 -0.09474886  0.62983114
##         121         122         123         124         125         126
## -0.21639553  0.14164600 -0.08583123 -0.28663958  0.24948847  0.14254830
##         127         128         129         130         131         132
##  0.31012675  0.25087009  0.32597553 -0.21370440  0.24698421  0.18604913
##         133         134         135         136         137         138
## -0.29838171 -0.03768129 -0.53800666  0.23601041  0.28018122  0.48465049
##         139         140         141         142         143         144
##  0.04173258  0.22459785  0.27511832 -0.22863595 -0.27381623 -0.10786644
##         145         146         147         148         149         150
## -0.09041804 -0.22921536 -0.19082807  0.31557253  0.05054498  0.12168723
##         151         152         153         154         155         156
## -0.06293118 -0.17136547  0.24494161  0.14860499  0.26635124  0.35821335
##         157         158         159         160         161         162
## -0.21976594  0.31265782 -0.42198587  0.65835108 -0.14388993  0.23764511
##         163         164         165         166         167         168
## -0.22080183  0.04891601  0.42553691 -0.22281048  0.57632934  0.09611773
##         169         170         171         172         173         174
##  0.35298140  0.16265972  0.22234886 -0.49647702 -0.09066635  0.39797878
##         175         176         177         178         179         180
## -0.02135012  0.12254021 -0.56124019 -0.15266700  0.43731550 -0.36979761
##         181         182         183         184         185         186
##  0.29657264  0.13098841  0.06076842  0.23589889 -0.13509404 -0.02771750
##         187         188         189         190         191         192
##  0.24473379 -0.09007677  0.53851101  0.15272580 -0.12030003  0.36525948
##         193         194         195         196         197         198
##  0.25786647 -0.55915089 -0.30448031  0.06858735 -0.55106786  0.57981128
##         199         200
##  0.29551983  0.01007488
```

```r
print(hist(residuals,
     breaks = 30,
     main = "Histogram of Residuals",
     xlab = "Residuals"))
```




```
## $breaks
##  [1] -1.9 -1.8 -1.7 -1.6 -1.5 -1.4 -1.3 -1.2 -1.1 -1.0 -0.9 -0.8 -0.7 -0.6 -0.5
## [16] -0.4 -0.3 -0.2 -0.1  0.0  0.1  0.2  0.3  0.4  0.5  0.6  0.7
##
## $counts
##  [1]  1  0  0  0  0  0  0  1  1  0  0  3  4 10  6  7 20 12 29 16 23 29 18 11  7
## [26]  2
##
## $density
##  [1] 0.05 0.00 0.00 0.00 0.00 0.00 0.00 0.05 0.05 0.00 0.00 0.15 0.20 0.50 0.30
## [16] 0.35 1.00 0.60 1.45 0.80 1.15 1.45 0.90 0.55 0.35 0.10
##
## $mids
##  [1] -1.85 -1.75 -1.65 -1.55 -1.45 -1.35 -1.25 -1.15 -1.05 -0.95 -0.85 -0.75
## [13] -0.65 -0.55 -0.45 -0.35 -0.25 -0.15 -0.05  0.05  0.15  0.25  0.35  0.45
## [25]  0.55  0.65
##
## $xname
## [1] "residuals"
##
## $equidist
## [1] TRUE
##
## attr(,"class")
## [1] "histogram"
```

Despite the model's results being lackluster, the residual checks via a histogram, QQ plot, and a scatter plot to test for linearity all seem copacetic. The residuals appear roughly centered around zero with no strong curves in the plot. THis suggests that linearity is reasonably satisfied for the model. The histogram of residuals shows a nearly normal distribution, but with some mild skew, while the QQ plot demonstrates mostly linear behavior there is some deviation in the tails. Essentially mostly normal. Overall, the residual diagnostics do not show any major violations of model assumptions, suggesting the issue lies more in the weak predictor relationships than in poor model fit structure.

2. **Optimization:**

   - Use your model to **optimize inventory levels** for a **peak sales season**, balancing **minimizing stockouts** with **minimizing overstock**.

```r
## We want 95th Percentile for Peak Sales
peak_sales <- quantile(retail_data$Sales, 0.95)
## For the other two variables we can just use the mean value because were only dealing with peak sales
avg_lead  <- mean(retail_data$Lead_Time_Days)
avg_price <- mean(retail_data$Price)

## Mimicking the model, and transforming these variables before using to predict inventory needs.
log_sales <- log(peak_sales + 1)
log_lead <- log(avg_lead + 1)
log_price <- log(avg_price + 1)

peak_sales_data <- data.frame(log_Sales = log_sales,
                        log_LeadTime = log_lead,
                        log_price = log_price)

predicted_log_inventory <- predict(multi_model, newdata = peak_sales_data)
## Raw logged Predicted Inventoruy level need
print(predicted_log_inventory)
```

```
##      95%
## 6.148031
```

```r
#Scaling back to normal number
normal_scale_inventory_level <- exp(predicted_log_inventory) - 1
print(paste("Recommended Inventory Level for Peak Sales:", round(normal_scale_inventory_level, 2)))
```

```
## [1] "Recommended Inventory Level for Peak Sales: 466.8"
```

Based on the model in this assignment, which is not ideal with a super low r^2 value and high p-values, the predicted inventory level needs for the top 5% of sales is 467.