Assignment 2 - Srujana Vanka, 2020102005

Code

Hide

```
library(readxl)
library(ggplot2)
library(corrplot)
library(MASS)
```

Hide

```
library(ppcor)
library(carData)
```

Hide

```
library(ltm)
```

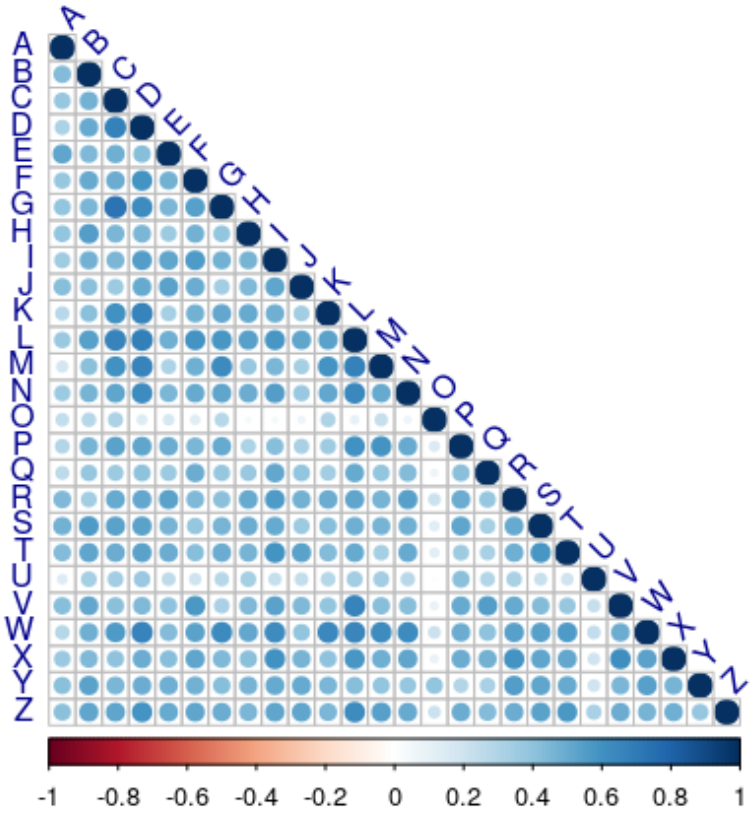
Hide

```
library(psych)
```

1 Advert Rating: Outlier Detection

Hide

```
data <- read_excel("BRSM_Assignment_2_datasets.xlsx", sheet = "Advert Rating")

cor_matrix <- cor(data)
corrplot(cor_matrix, type = "lower", tl.col = "darkblue", tl.srt = 45)
```



Observations and Inferences:

Upon examining the correlation heatmap, it becomes evident that participants ‘O’ and ‘U’ stand out as potential outliers. These individuals exhibit a striking characteristic - their correlation with almost every other participant is consistently zero.

This lack of correlation suggests a lack of linear relationship between the ratings provided by ‘O’ and ‘U’ and those of the other participants.

2 Reliable Job: Internal Consistency

Hide

```
data <- read_excel("BRSM_Assignment_2_datasets.xlsx", sheet = "Reliable Job")
```

1. Calculate Cronbach’s Alpha for Job Satisfaction (JS):

Hide

```
# Extract JS items
JS_items <- data[, c("JS1", "JS2", "JS3", "JS4")]

# Calculate Spearman correlations between JS items
cor_JS <- cor(JS_items, method = "spearman")

# Calculate mean correlation (lower triangle, excluding diagonal)
mean_cor_JS <- mean(cor_JS[lower.tri(cor_JS, diag = FALSE)])

# Calculate Cronbach's Alpha for JS
num_JS_items <- ncol(JS_items)
alpha_JS <- (num_JS_items / (num_JS_items - 1)) * (1 - (mean_cor_JS / (1 + mean_cor_J
S)))

cat("Cronbach's Alpha for Job Satisfaction (JS):", alpha_JS, "\n")
```

```
Cronbach's Alpha for Job Satisfaction (JS): 0.8319978
```

2. Calculate Cronbach’s Alpha for Job Performance (JP):

Hide

```
# Extract JP items
JP_items <- data[, c("JP1", "JP2", "JP3", "JP4")]

# Calculate Spearman correlations between JP items
cor_matrix_JP <- cor(JP_items, method = "spearman")

# Calculate mean correlation (lower triangle, excluding diagonal)
lower_tri_JP <- cor_matrix_JP[lower.tri(cor_matrix_JP)]

mean_corr_JP <- mean(lower_tri_JP)

n_JP <- ncol(JP_items)
alpha_JP <- n_JP * mean_corr_JP/(1 + (n_JP - 1)*mean_corr_JP)

cat("Cronbach's Alpha for Job Performance (JP):", alpha_JP, "\n")
```

```
Cronbach's Alpha for Job Performance (JP): 0.5242351
```

Internal Consistency Analysis

1. **Job Satisfaction (JS):**
    - **Cronbach’s Alpha:** 0.8319978
    - **Commentary:** The Cronbach’s alpha for Job Satisfaction is 0.832, indicating a high level of internal consistency. This suggests that the questions related to job satisfaction are closely related and measure a common underlying construct reliably.
2. **Job Performance (JP):**
    - **Cronbach’s Alpha:** 0.5242351
    - **Commentary:** The Cronbach’s alpha for Job Performance is 0.525, which is below the commonly accepted threshold of 0.7. This suggests potential issues with internal consistency in the Job

Performance questionnaire. It is advisable to review and possibly revise the items for improved reliability.

Summary: - Typically, Cronbach's Alpha values range from 0 to 1, where higher values indicate better internal consistency. - An Alpha value above 0.7 is often considered acceptable for reliability. - The Job Satisfaction questionnaire demonstrates good internal consistency. - The Job Performance questionnaire may benefit from further examination and refinement of its items to enhance internal consistency.

3 Yulu: Normality Testing

1. Conduct exploratory analysis

Hide

```
yulu_data <- read_excel("BRSM_Assignment_2_datasets.xlsx", sheet = "Yulu")
df <- yulu_data
```

Sampling is employed in statistical analyses, such as the Shapiro-Wilk test for normality, to manage issues related to sample size constraints.

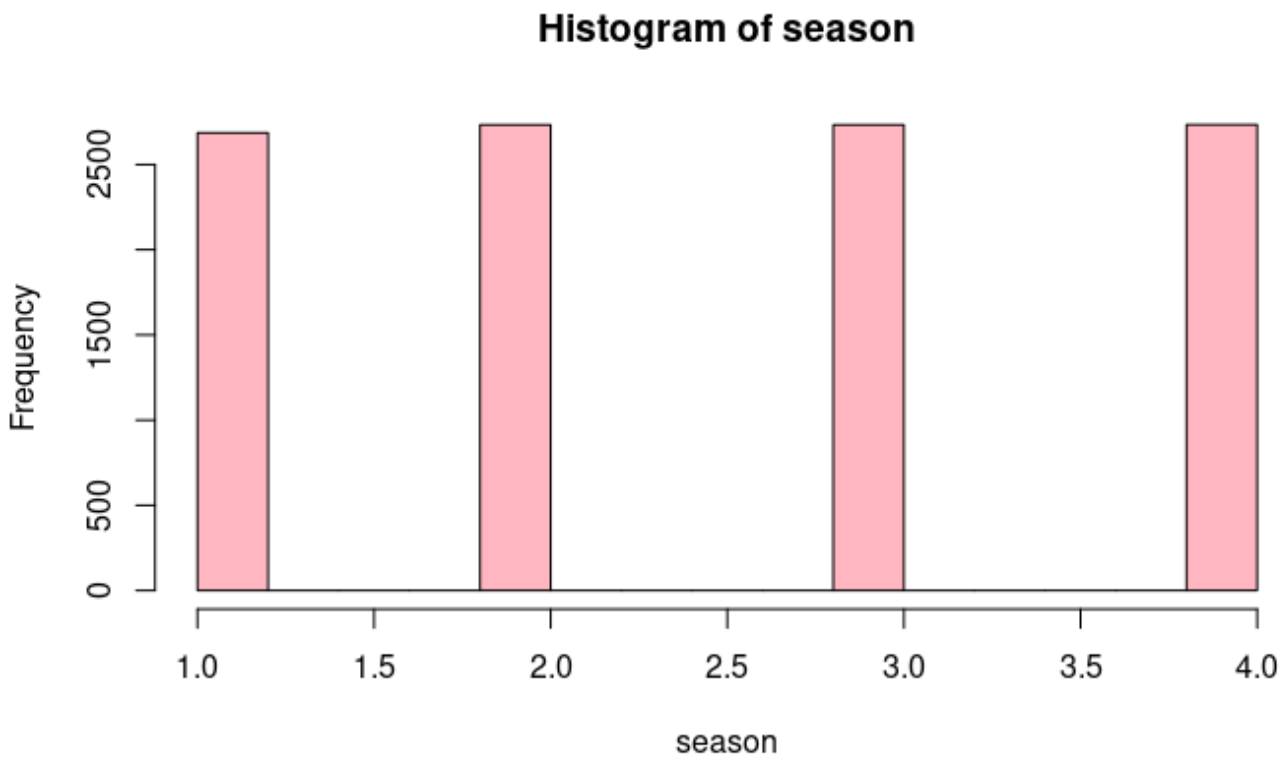
The Shapiro-Wilk test assumes that the sample size falls within a specific range, typically between 3 and 5000 observations. When dealing with larger datasets that surpass this threshold, sampling becomes a crucial step to comply with the test requirements.

The purpose of sampling is to create a representative subset of the original data, allowing statistical analyses to be performed on a manageable and statistically valid scale. By randomly selecting a subset of the data, we ensure that the assumptions and properties of the statistical tests hold, preventing biases or inaccuracies that might arise due to excessively large sample sizes.

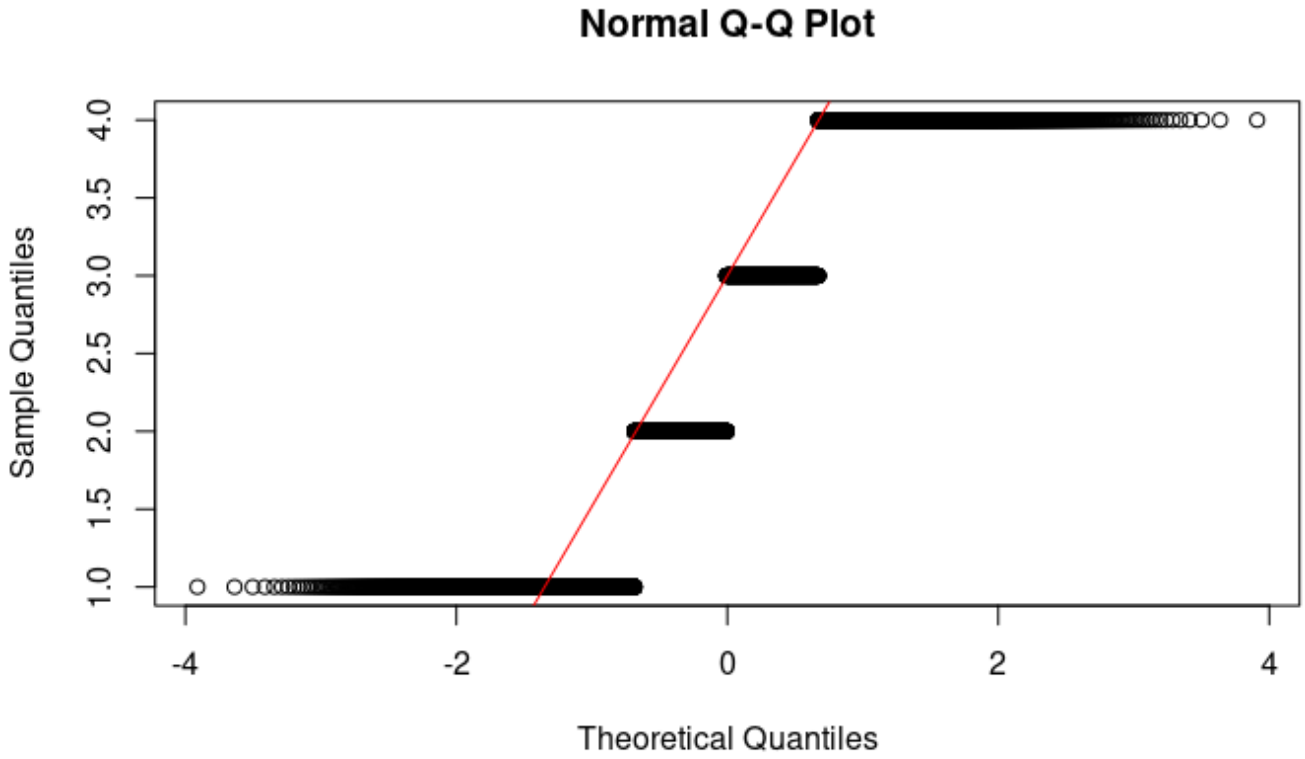
Hide

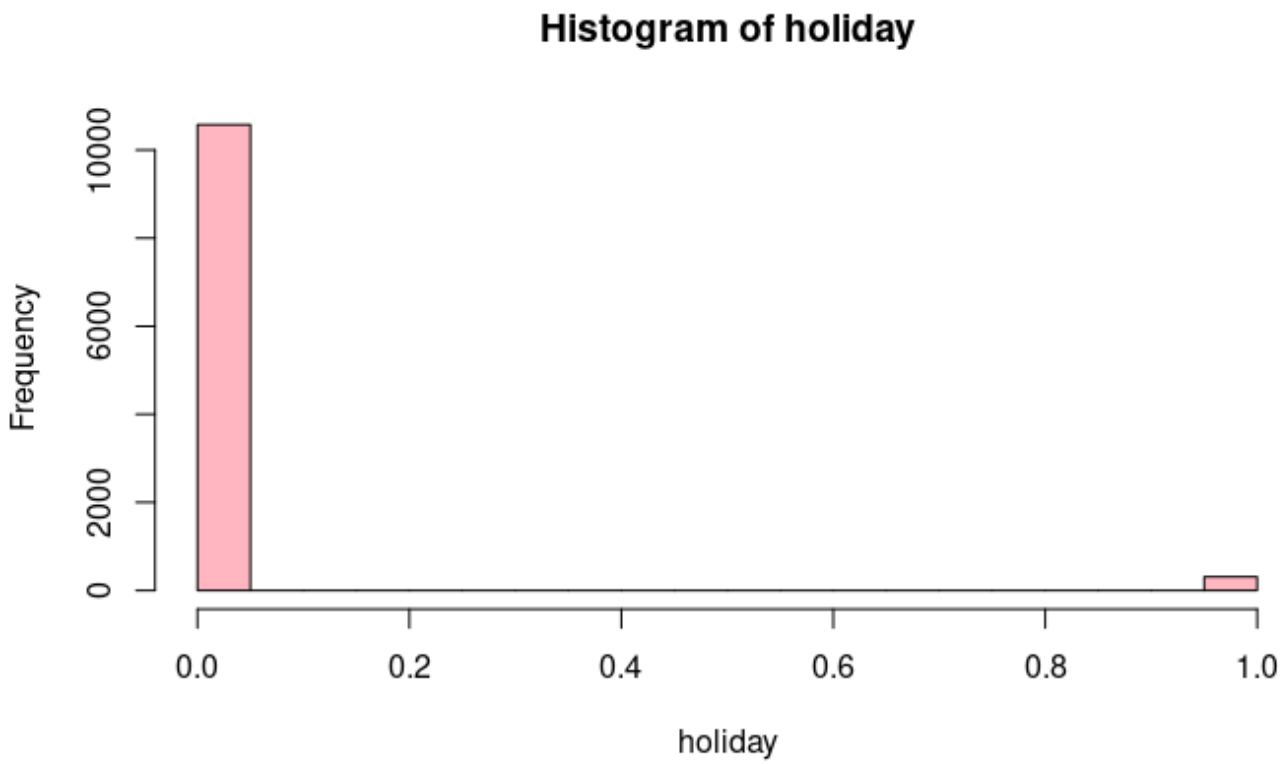
```
numeric_columns <- c('season', 'holiday', 'workingday', 'weather', 'temp', 'atemp',
'humidity', 'windspeed', 'casual', 'registered', 'count')

# Function for exploratory analysis and normality checks
explore_and_test_normality <- function(col) {
  # Check if the sample size is within the limit for Shapiro-Wilk test
  if (length(df[[col]]) > 5000) {
    # Set the desired sample size (e.g., 1000)
    desired_sample_size <- 1000

    # Take a random sample of size 'desired_sample_size'
    sampled_data <- sample(df[[col]], size = desired_sample_size)

    # Shapiro-Wilk test on the sampled data
    shapiro_test <- shapiro.test(sampled_data)
    cat("Shapiro-Wilk test for", col, "(sampled data): Statistic =", shapiro_test$sta
tistic, ", p-value =", shapiro_test$p.value, "\n")
  } else {
    # Shapiro-Wilk test directly
    shapiro_test <- shapiro.test(df[[col]])
    cat("Shapiro-Wilk test for", col, ": Statistic =", shapiro_test$statistic, ", p-v
alue =", shapiro_test$p.value, "\n")
  }

  # Histogram
  hist(df[[col]], main = paste("Histogram of", col), xlab = col, col = "lightpink", b
order = "black")

  # Q-Q plot
  qqnorm(df[[col]])
  qqline(df[[col]], col = 2)
}

# Apply the function for each numeric column
for (col in numeric_columns) {
  explore_and_test_normality(col)
}
```

```
Shapiro-Wilk test for season (sampled data): Statistic = 0.856705 , p-value = 3.43450
3e-29
```

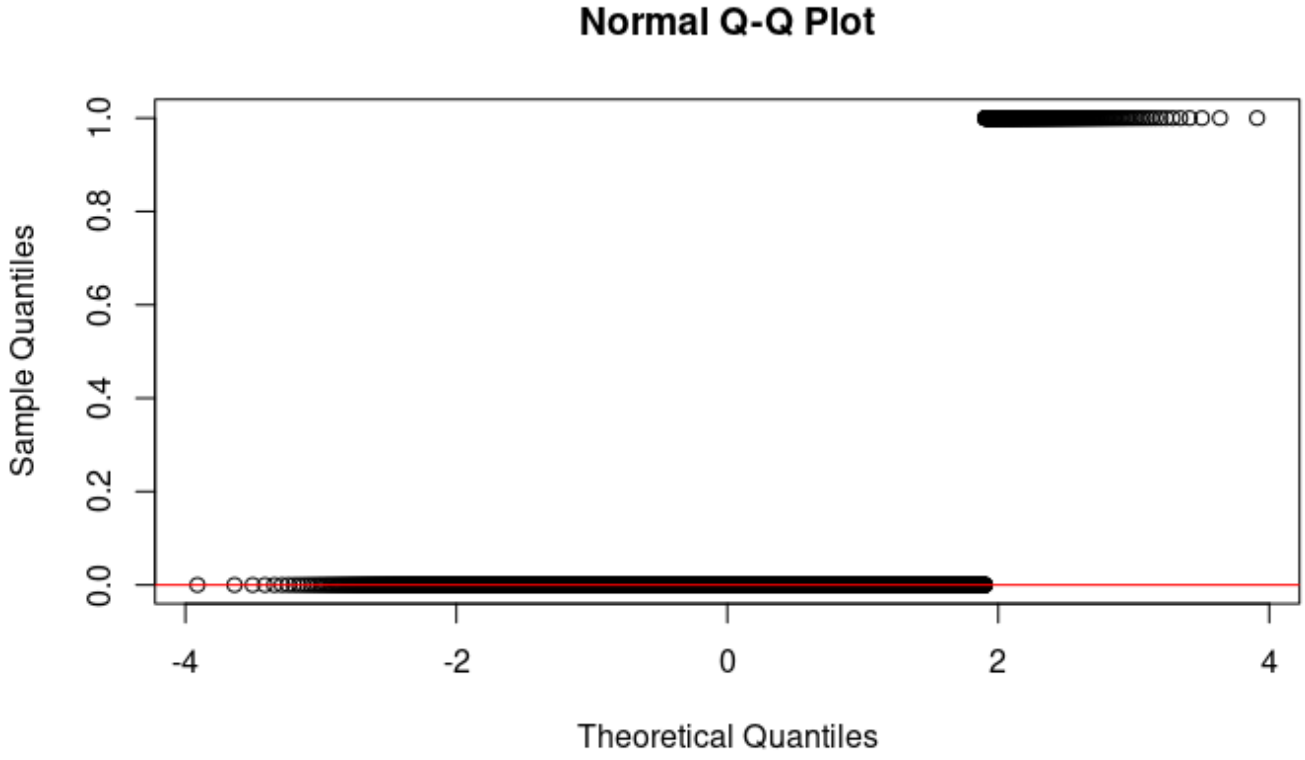


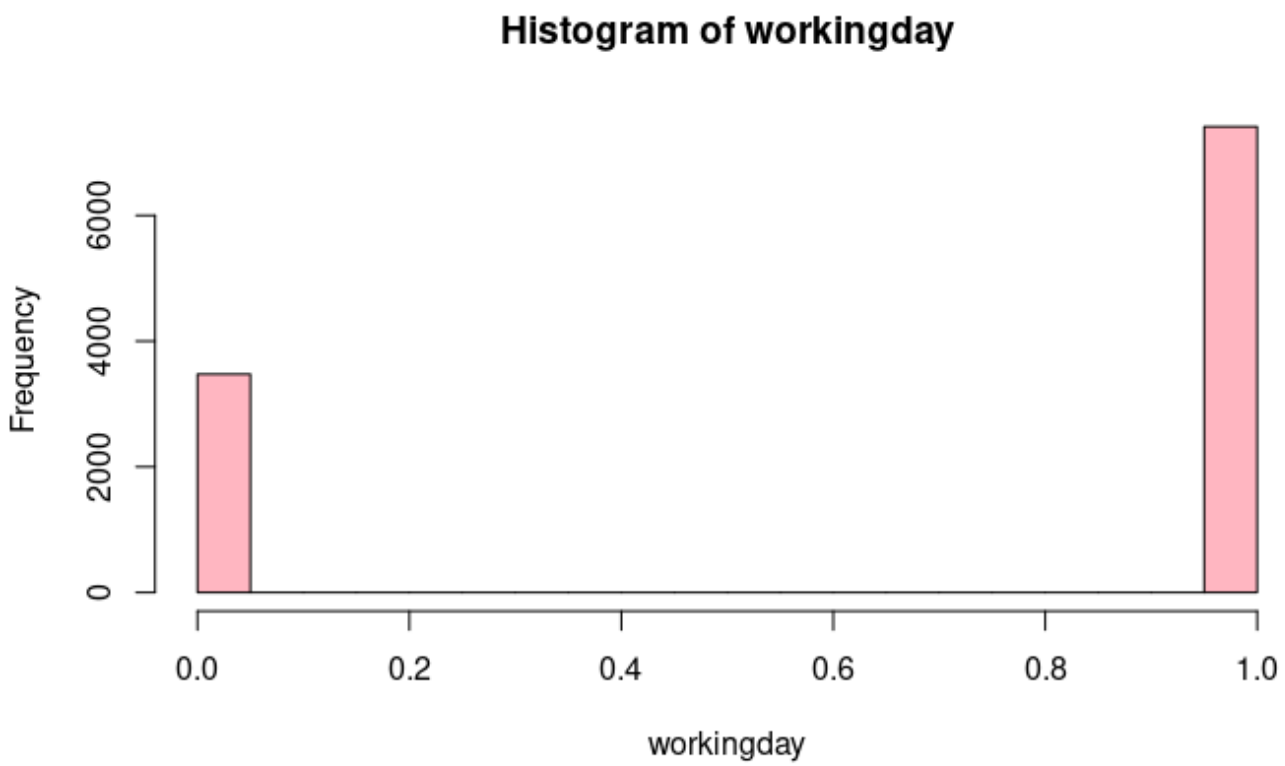

```
Shapiro-Wilk test for holiday (sampled data): Statistic = 0.1732696 , p-value = 1.910
05e-54
```



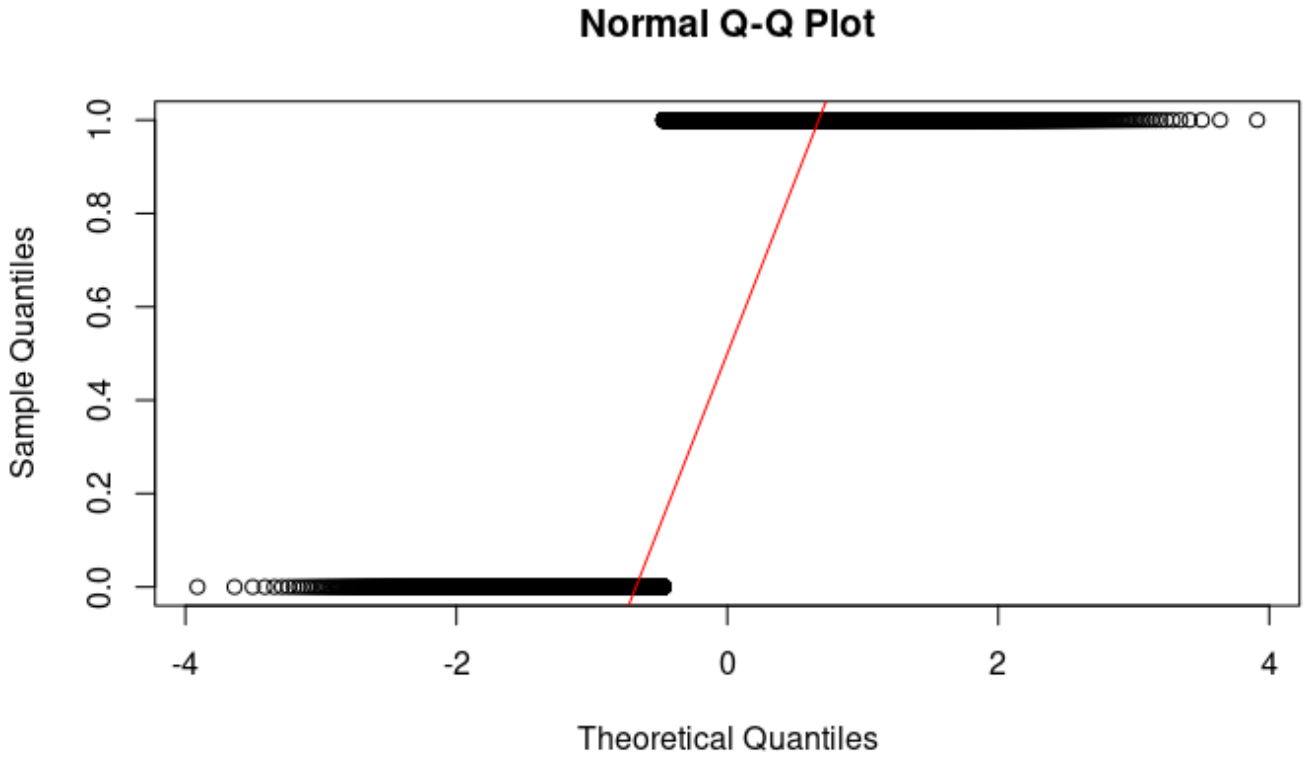


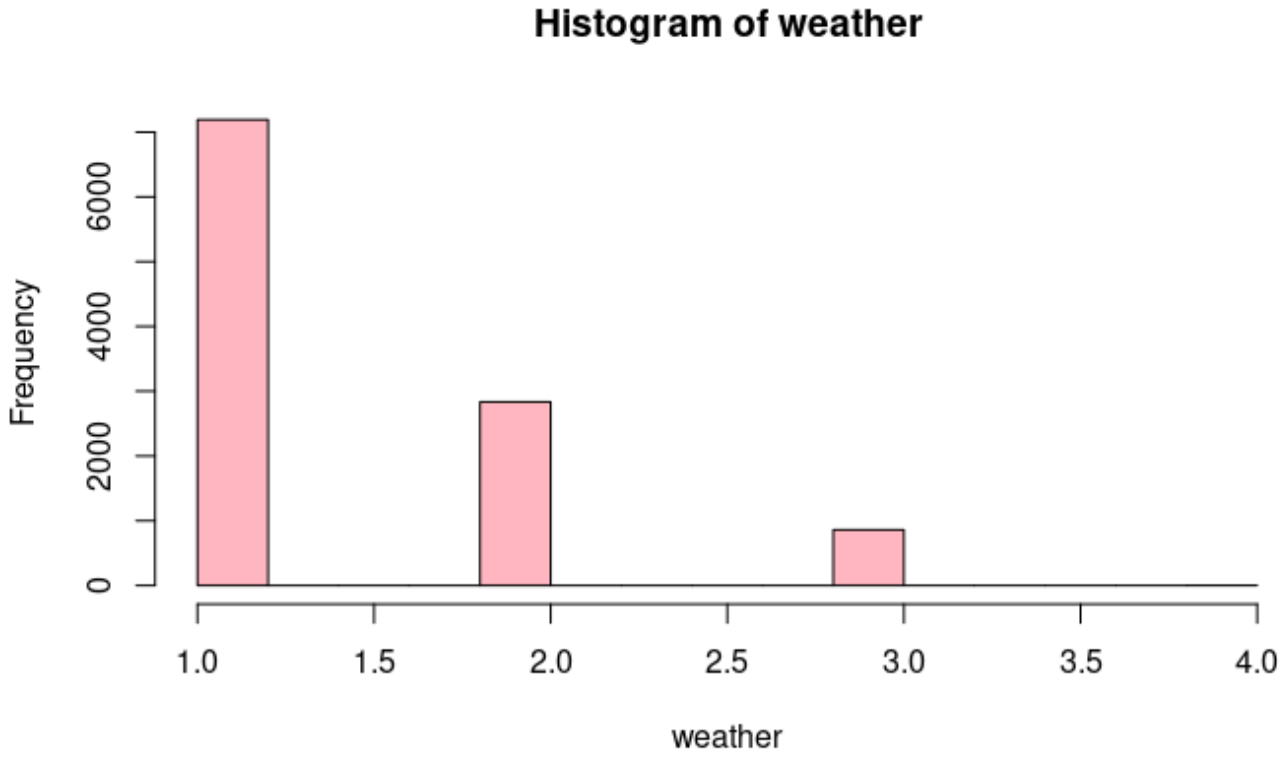

```
Shapiro-Wilk test for workingday (sampled data): Statistic = 0.573018 , p-value = 5.7
26442e-44
```



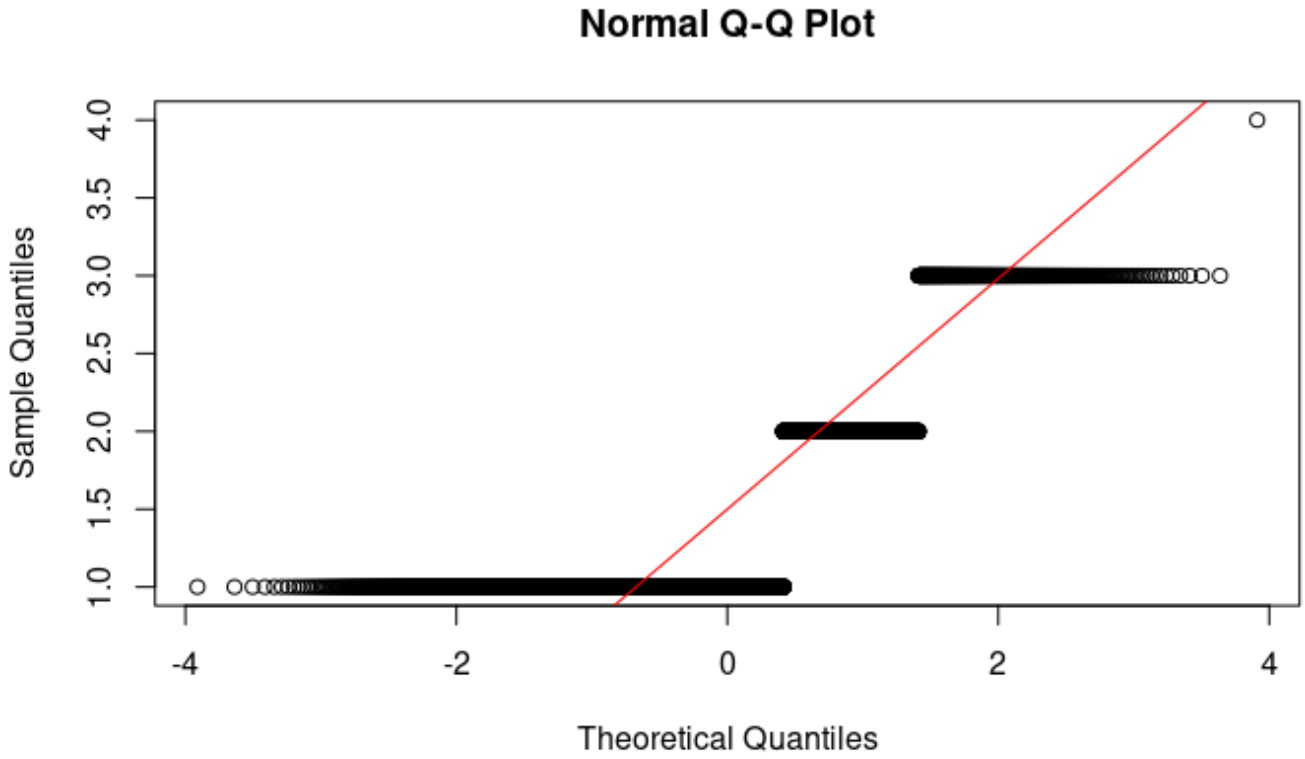


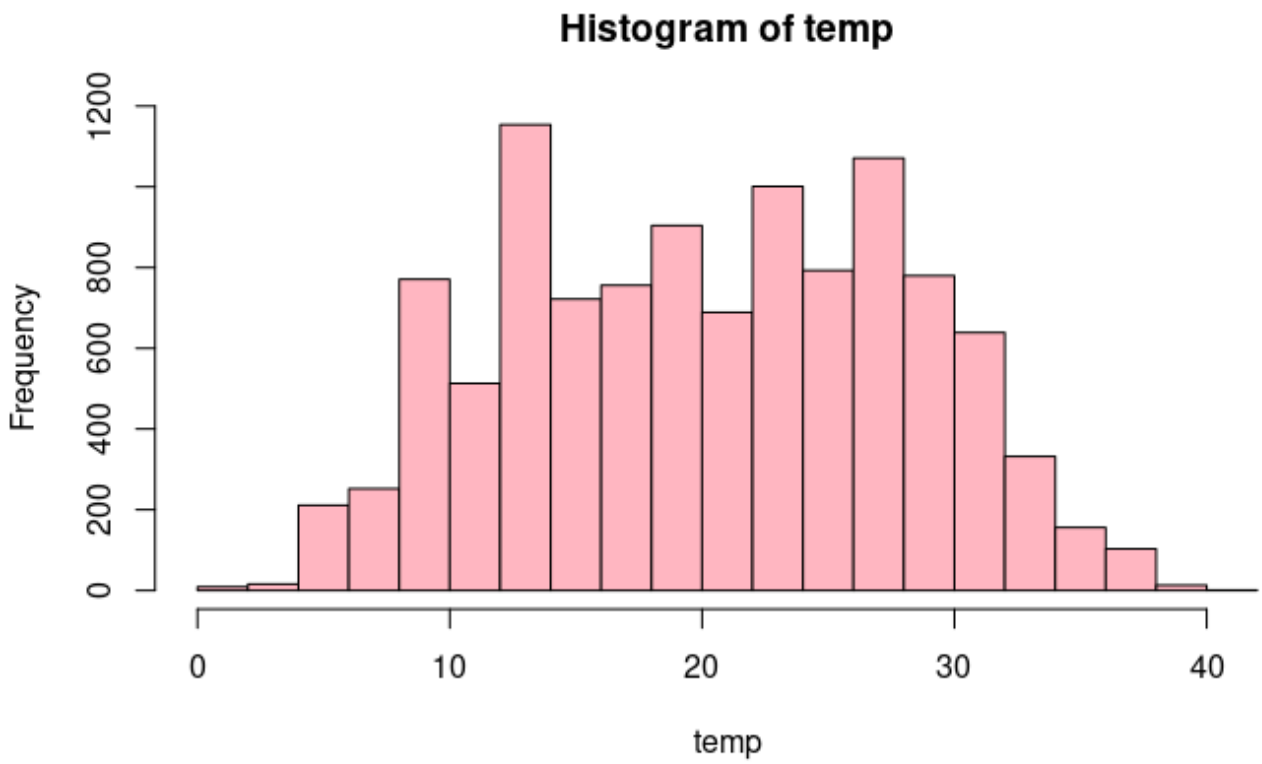

```
Shapiro-Wilk test for weather (sampled data): Statistic = 0.6414152 , p-value = 2.151
611e-41
```



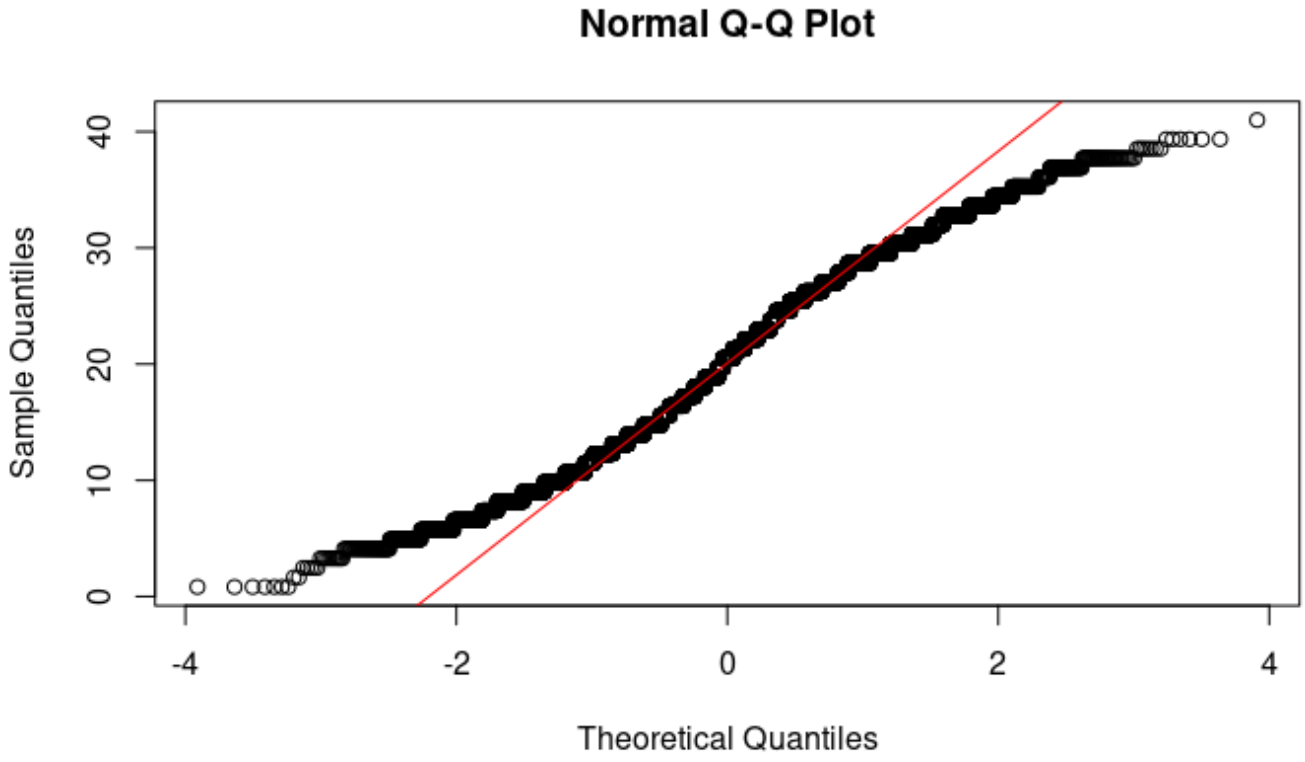


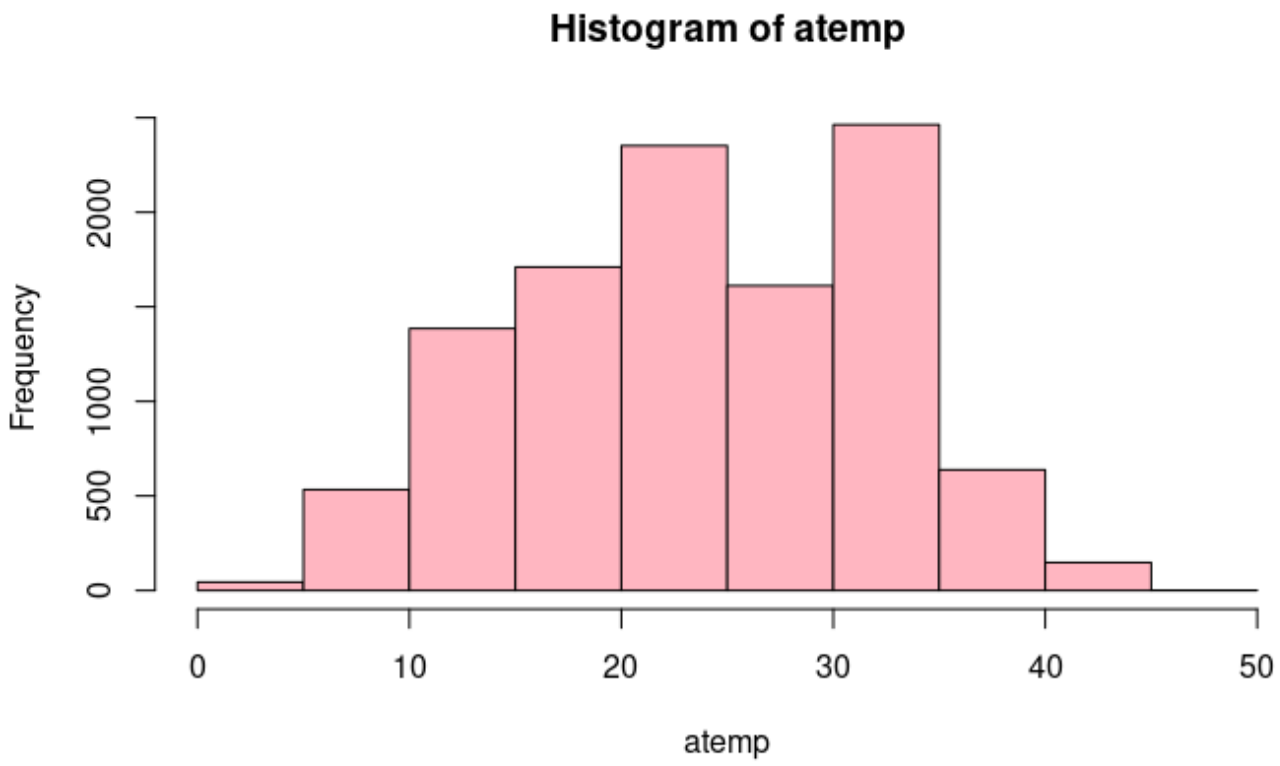

```
Shapiro-Wilk test for temp (sampled data): Statistic = 0.9810993 , p-value = 4.19952e
-10
```



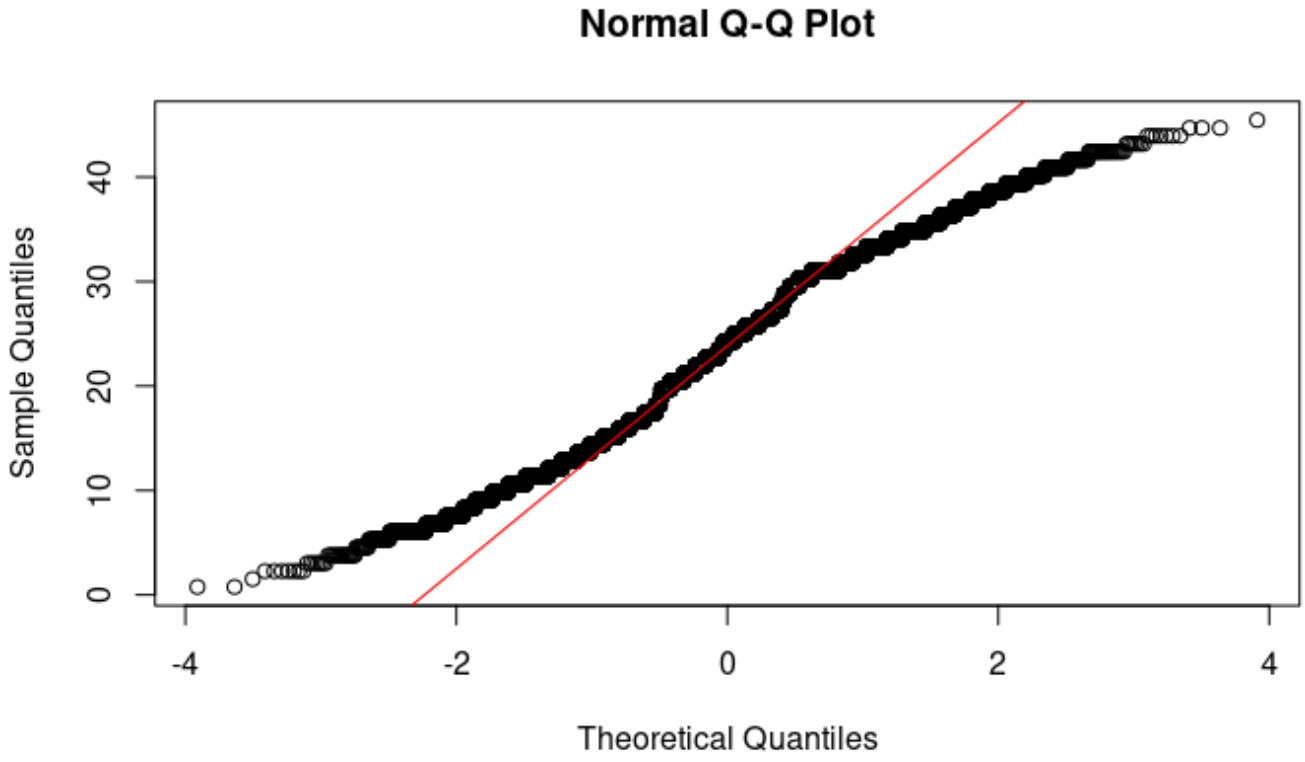


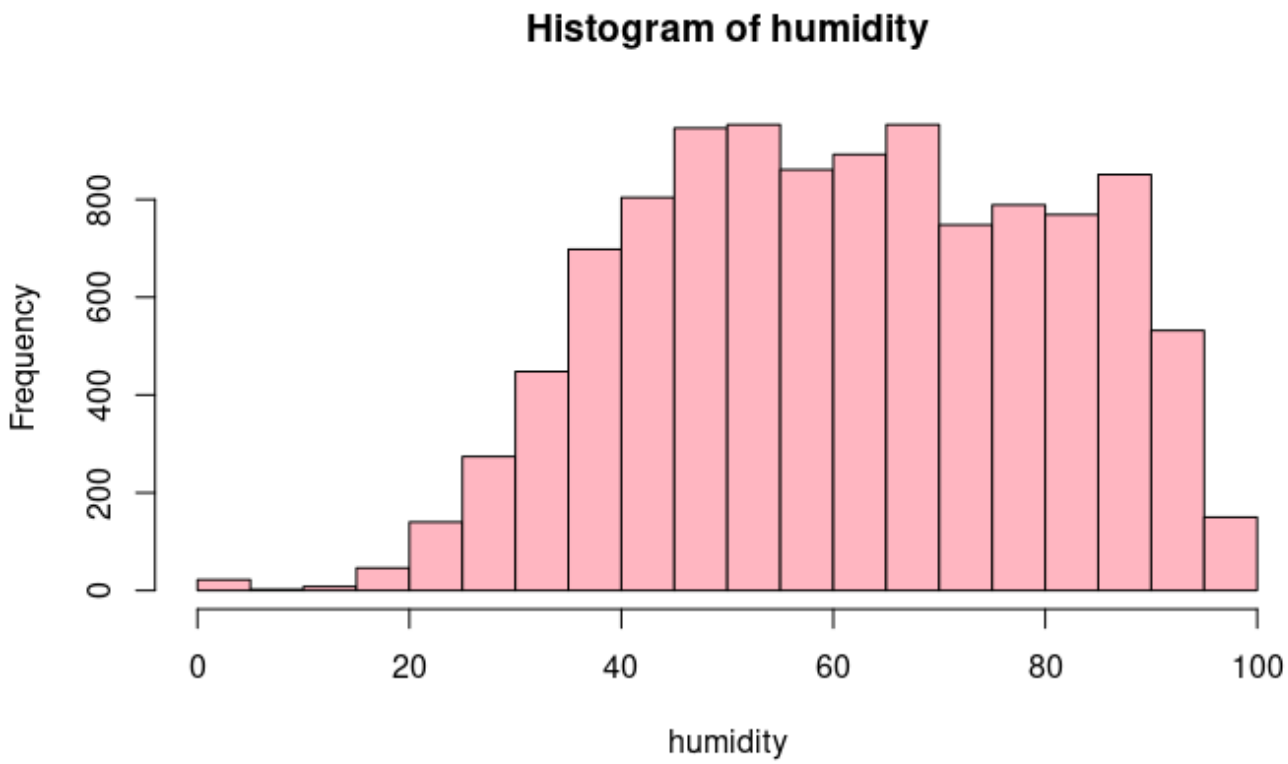

```
Shapiro-Wilk test for atemp (sampled data): Statistic = 0.9830694 , p-value = 2.25099
1e-09
```



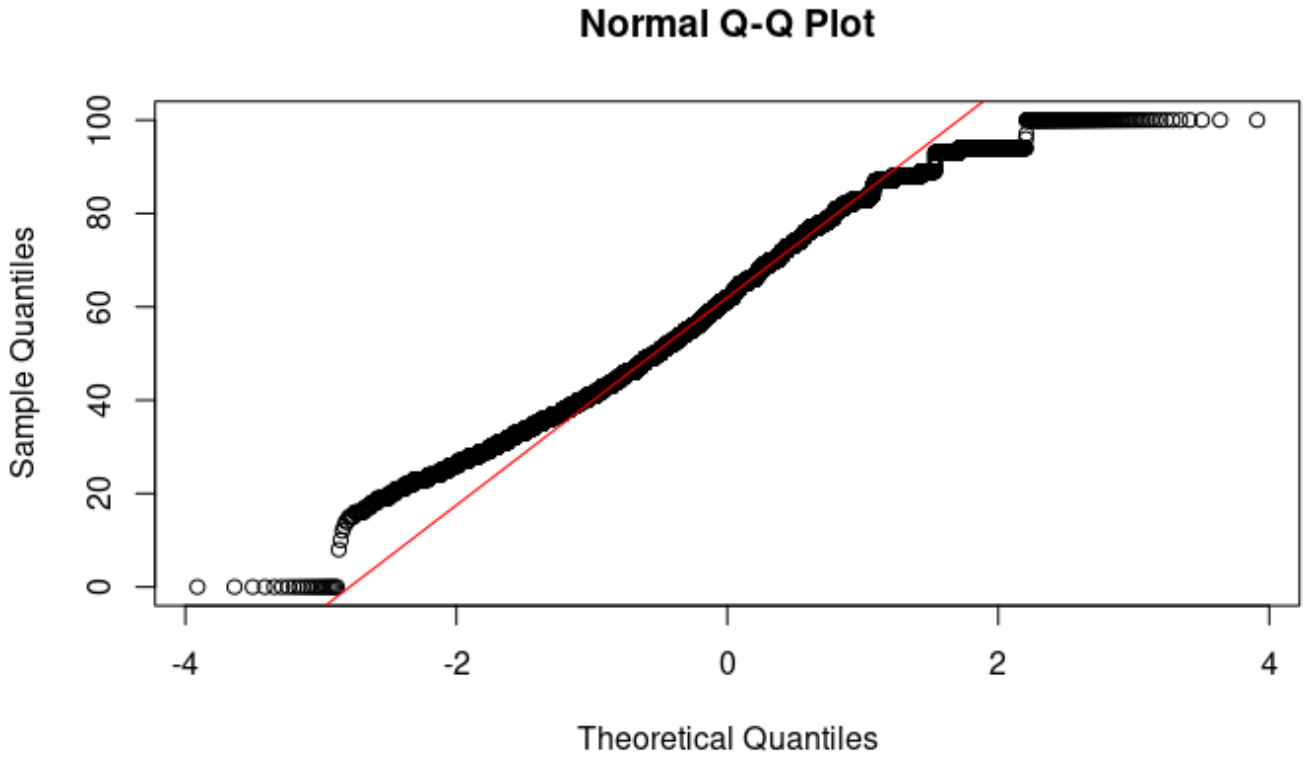


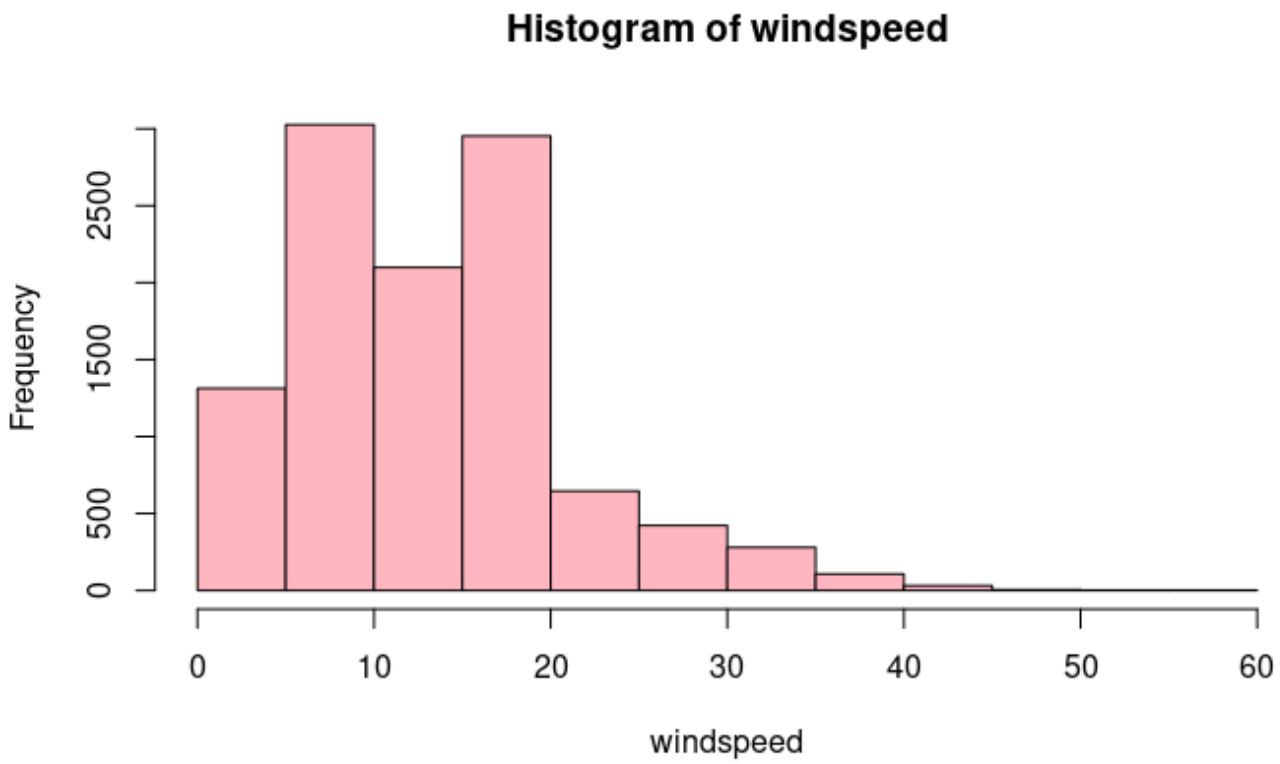

```
Shapiro-Wilk test for humidity (sampled data): Statistic = 0.9785663 , p-value = 5.64
2666e-11
```



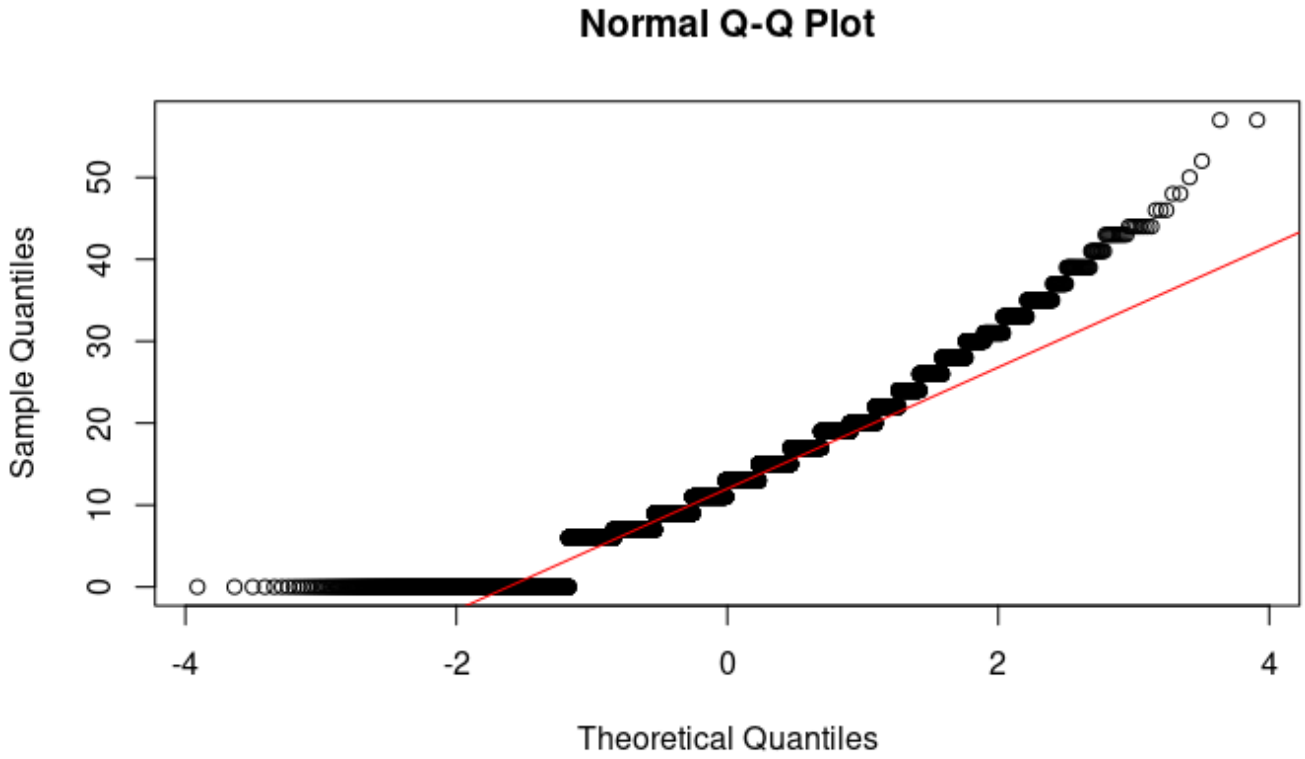


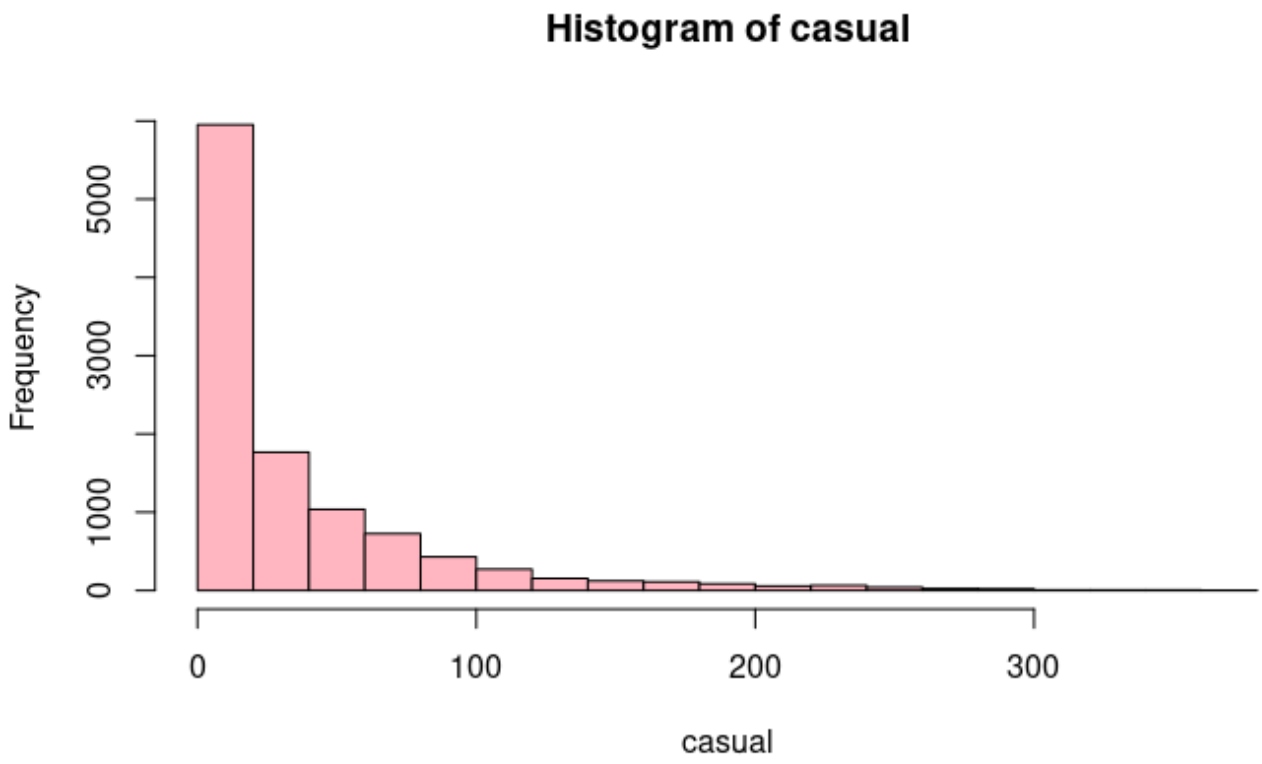

```
Shapiro-Wilk test for windspeed (sampled data): Statistic = 0.9536524 , p-value = 3.1
7664e-17
```



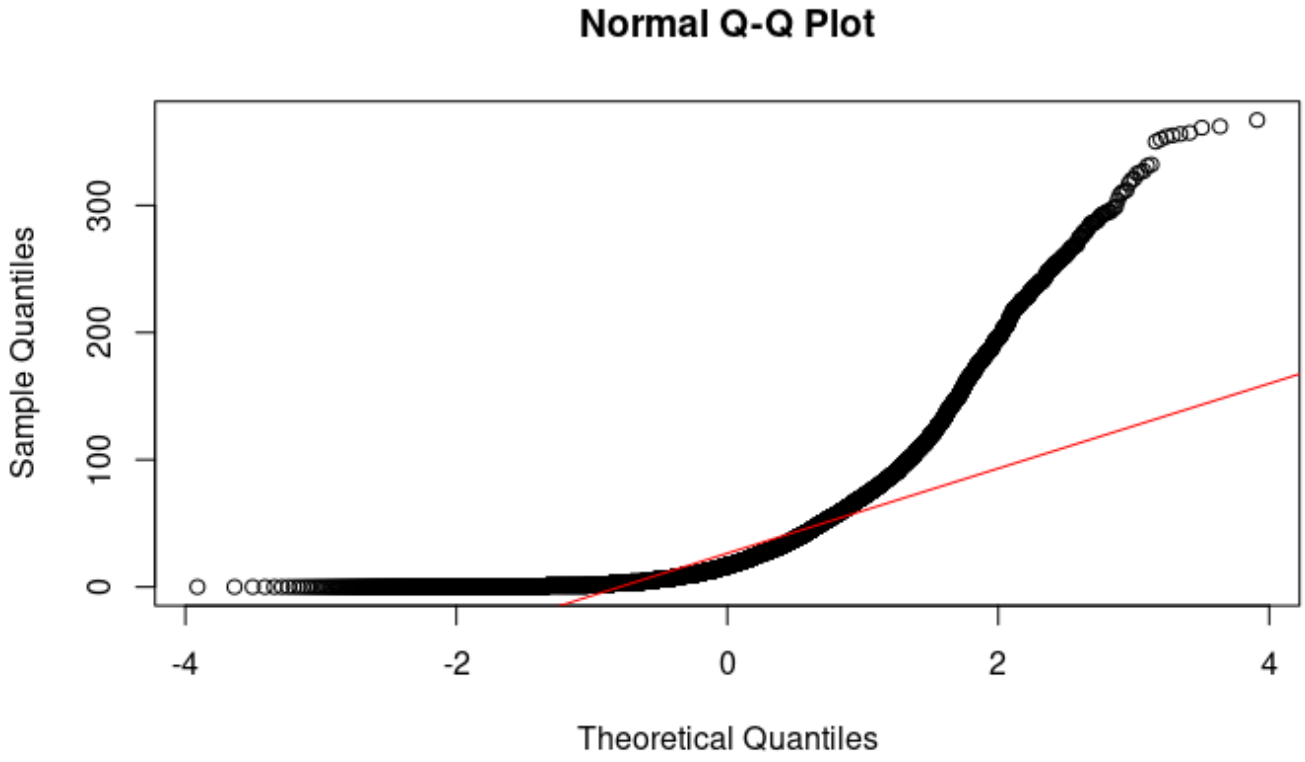


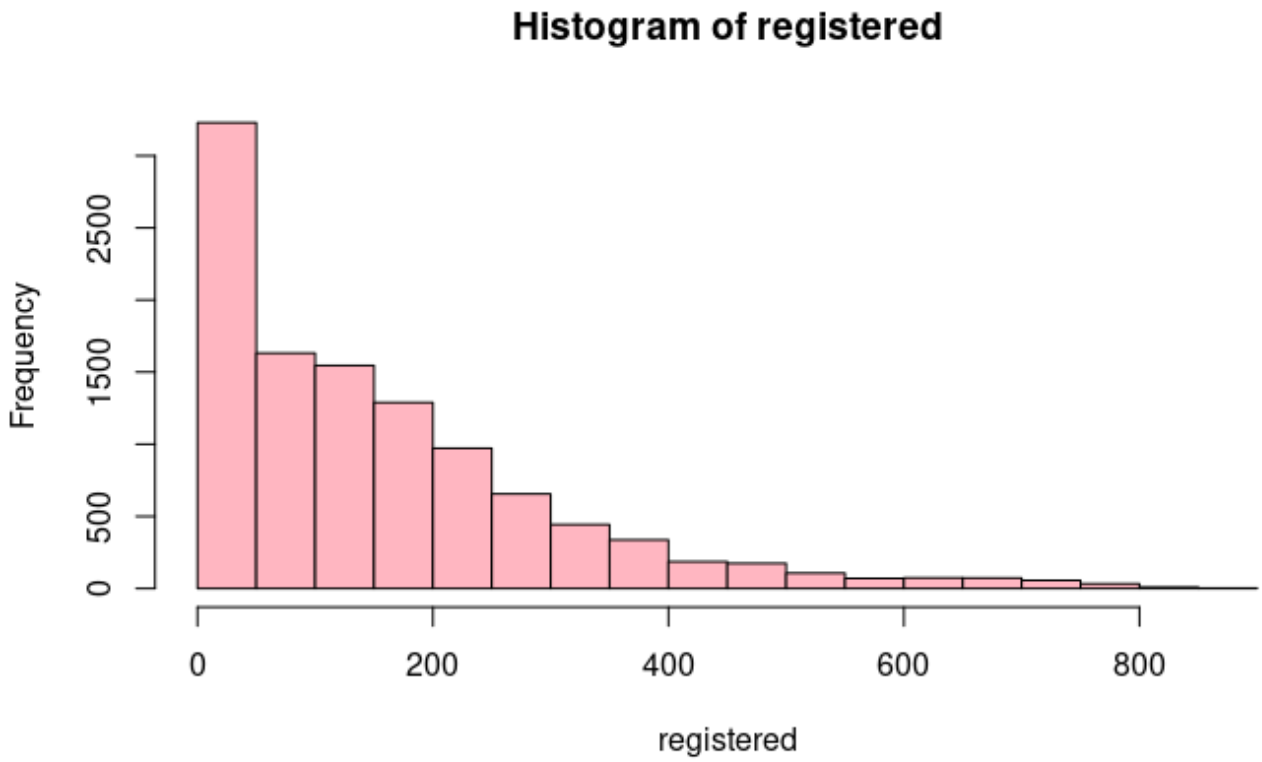

```
Shapiro-Wilk test for casual (sampled data): Statistic = 0.7200567 , p-value = 7.0133
9e-38
```



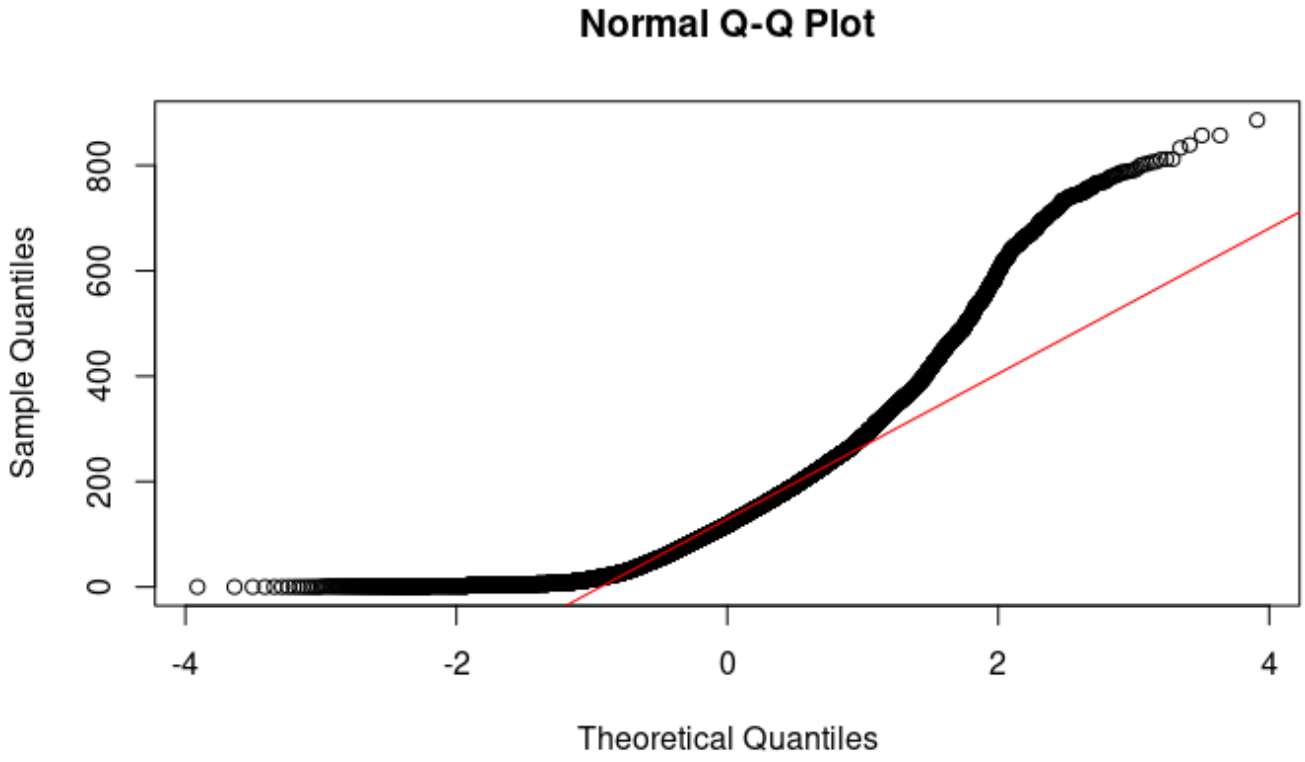


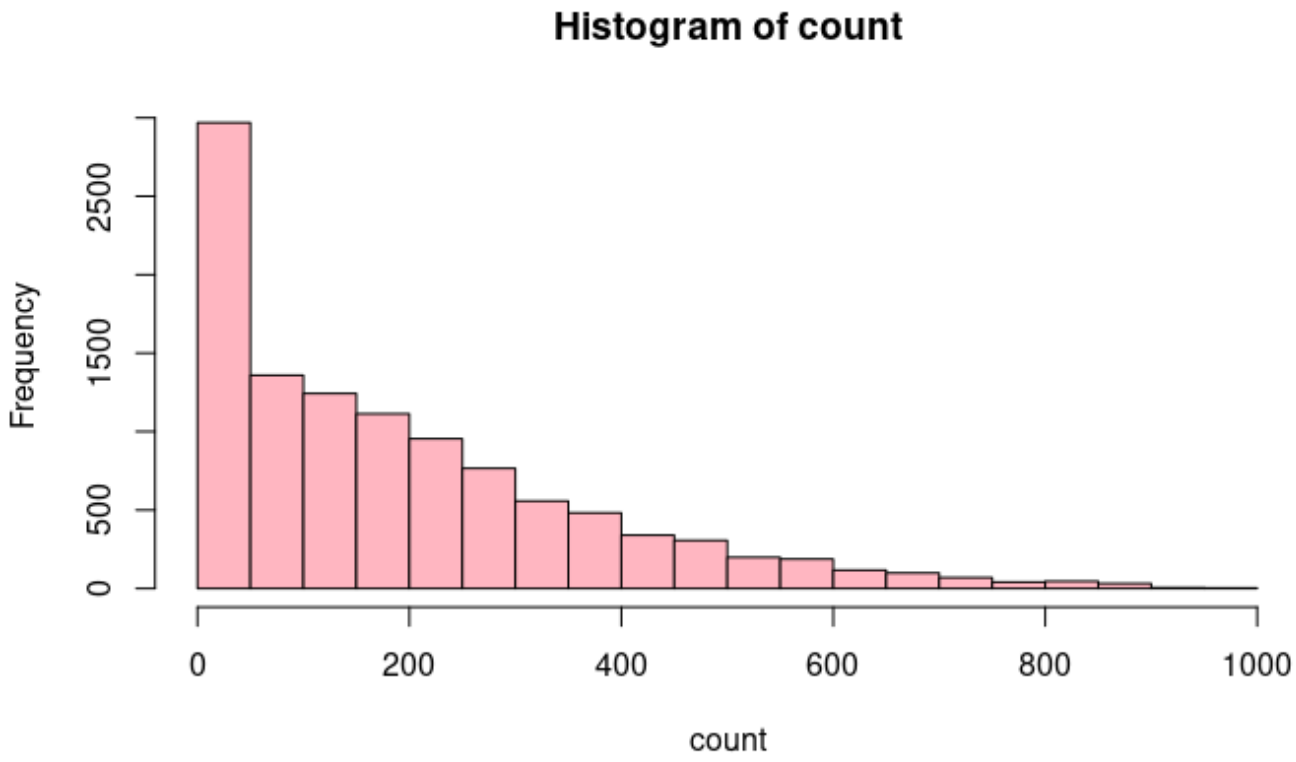

```
Shapiro-Wilk test for registered (sampled data): Statistic = 0.8465666 , p-value = 5.
048424e-30
```

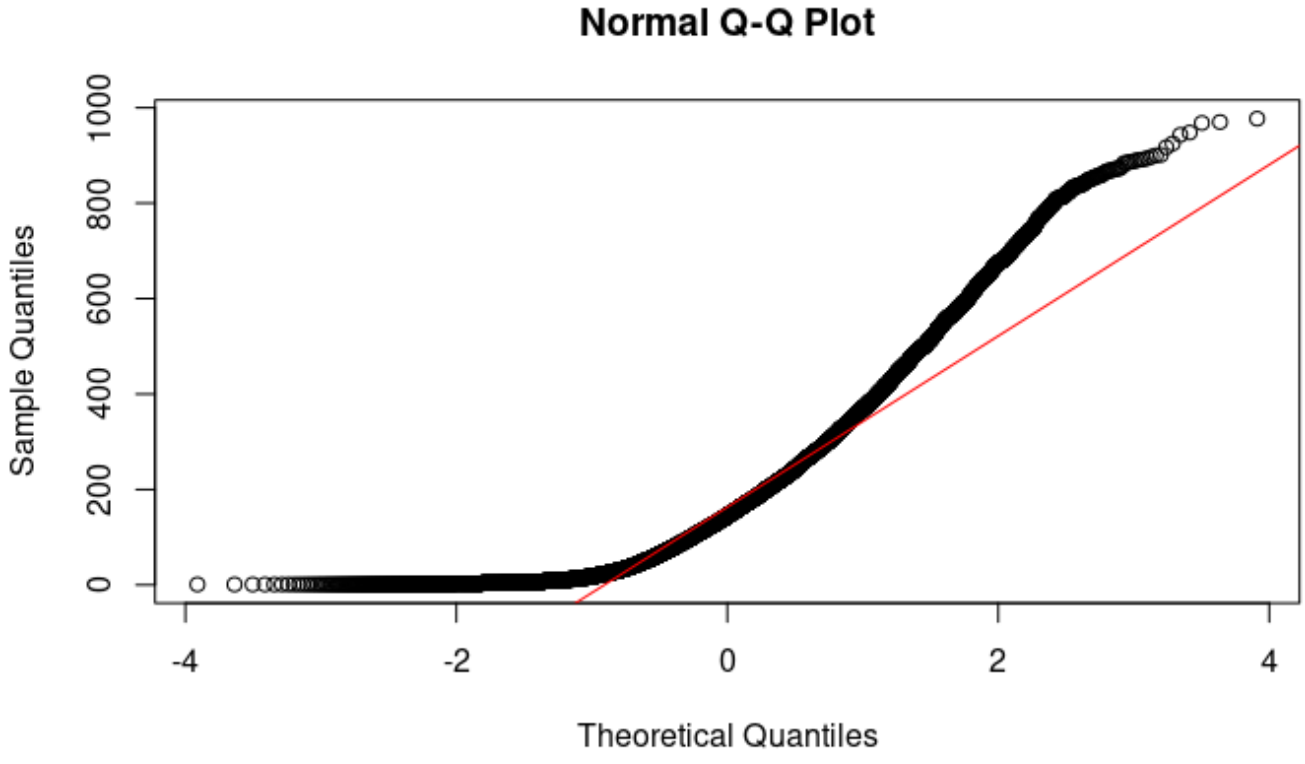





```
Shapiro-Wilk test for count (sampled data): Statistic = 0.886417 , p-value = 1.879587
e-26
```







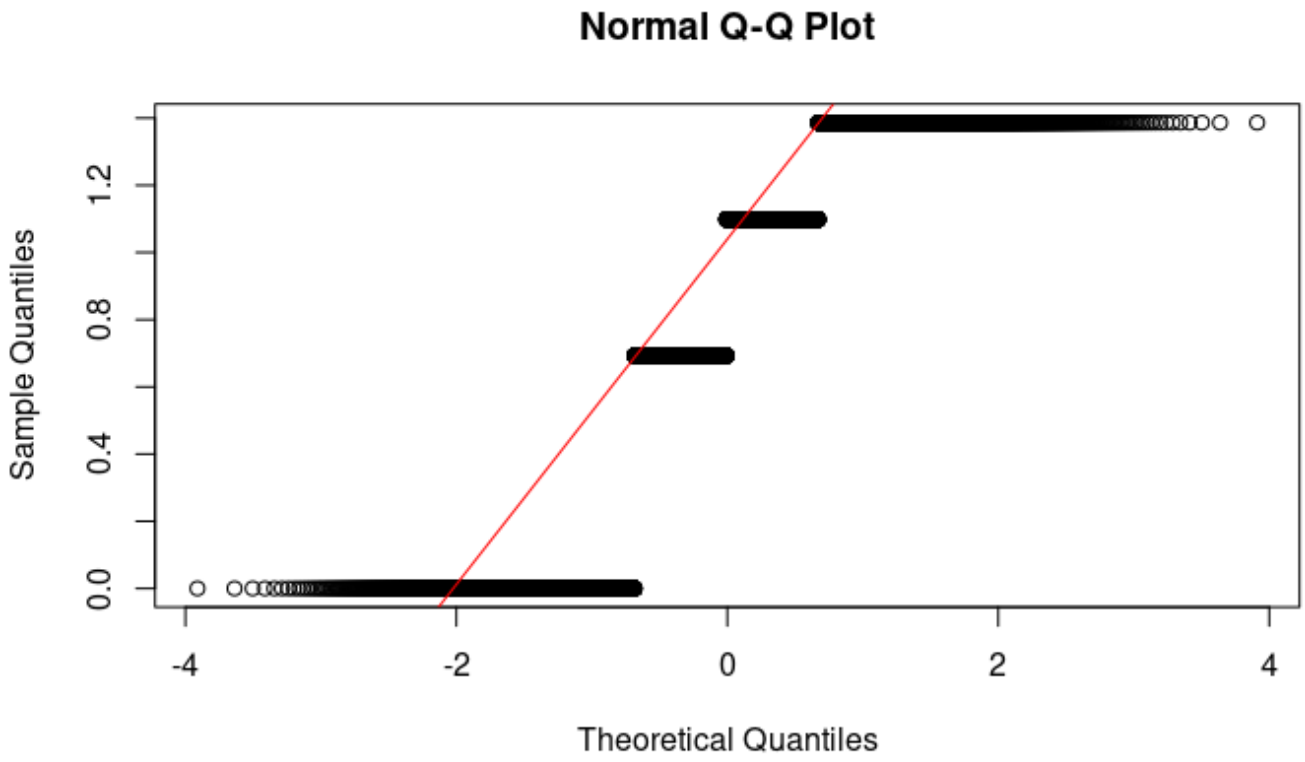
2. Apply Box-Cox transformation if not normally distributed

Hide

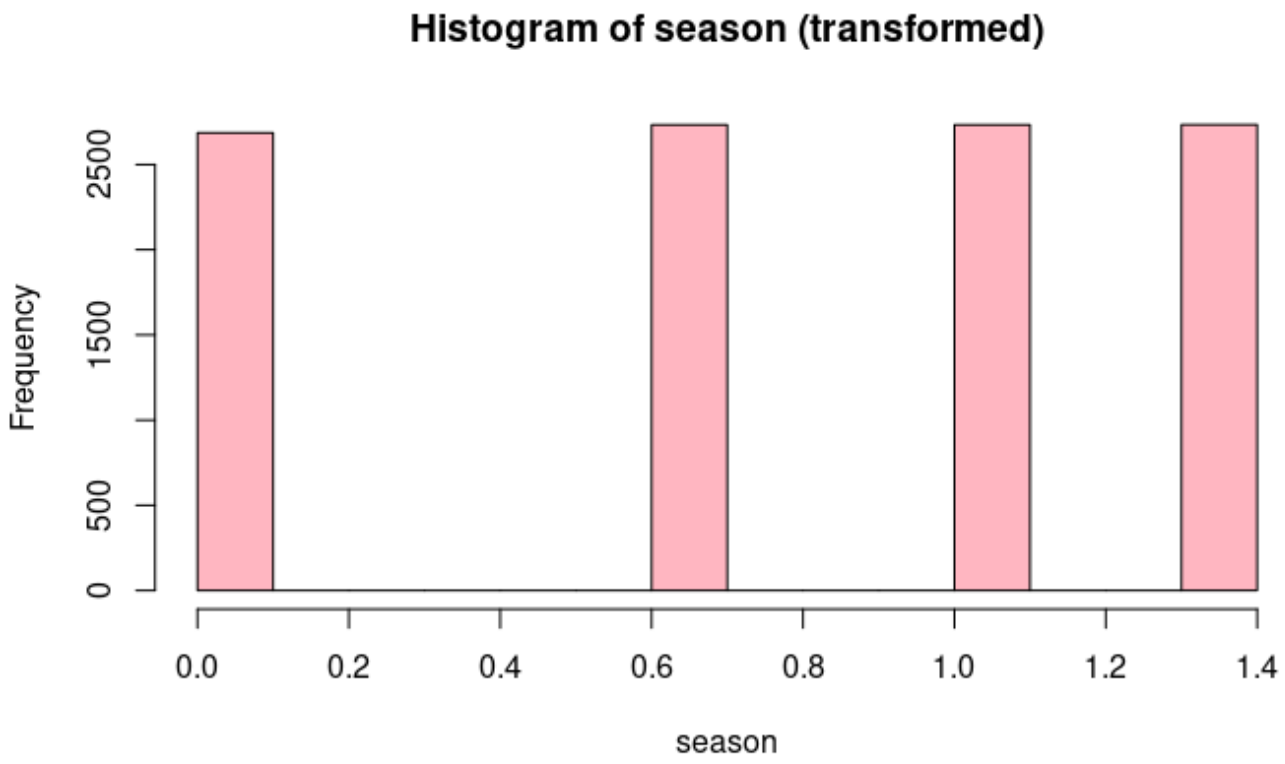
```
# Function to check normality and apply Box-Cox transformation
explore_and_transform <- function(col) {
  # Check if the sample size is within the limit for Shapiro-Wilk test
  if (length(df[[col]]) > 5000) {
    # Set the desired sample size (e.g., 1000)
    desired_sample_size <- 1000

    # Take a random sample of size 'desired_sample_size'
    sampled_data <- sample(df[[col]], size = desired_sample_size)

    # Shapiro-Wilk test on the sampled data
    shapiro_test <- shapiro.test(sampled_data)
    cat("Shapiro-Wilk test for", col, "(sampled data): Statistic =", shapiro_test$sta
tistic, ", p-value =", shapiro_test$p.value, "\n")

    # Apply Box-Cox transformation for non-normally distributed variables
    if (shapiro_test$p.value < 0.05) {
      # Handling zero values by adding a small constant
      df[[col]] <- ifelse(df[[col]] == 0, 0.01, df[[col]])
      transformed_col <- log(df[[col]])
      cat("Box-Cox transformation applied for", col, "\n")

      # Plot QQ plot
      qqnorm(transformed_col)
      qqline(transformed_col, col = 2)

      # Plot histogram
      hist(transformed_col, main = paste("Histogram of", col, "(transformed)"), xlab
= col, col = "lightpink", border = "black")
    }
  } else {
    # Shapiro-Wilk test directly
    shapiro_test <- shapiro.test(df[[col]])
    cat("Shapiro-Wilk test for", col, ": Statistic =", shapiro_test$statistic, ", p-v
alue =", shapiro_test$p.value, "\n")

    # Apply Box-Cox transformation for non-normally distributed variables
    if (shapiro_test$p.value < 0.05) {
      # Handling zero values by adding a small constant
      df[[col]] <- ifelse(df[[col]] == 0, 0.01, df[[col]])
      transformed_col <- log(df[[col]])
      cat("Box-Cox transformation applied for", col, "\n")

      # Plot QQ plot
      qqnorm(transformed_col)
      qqline(transformed_col, col = 2)

      # Plot histogram
      hist(transformed_col, main = paste("Histogram of", col, "(transformed)"), xlab
= col, col = "lightblue", border = "black")
    }
  }
}

# Apply the function for each numeric column
```

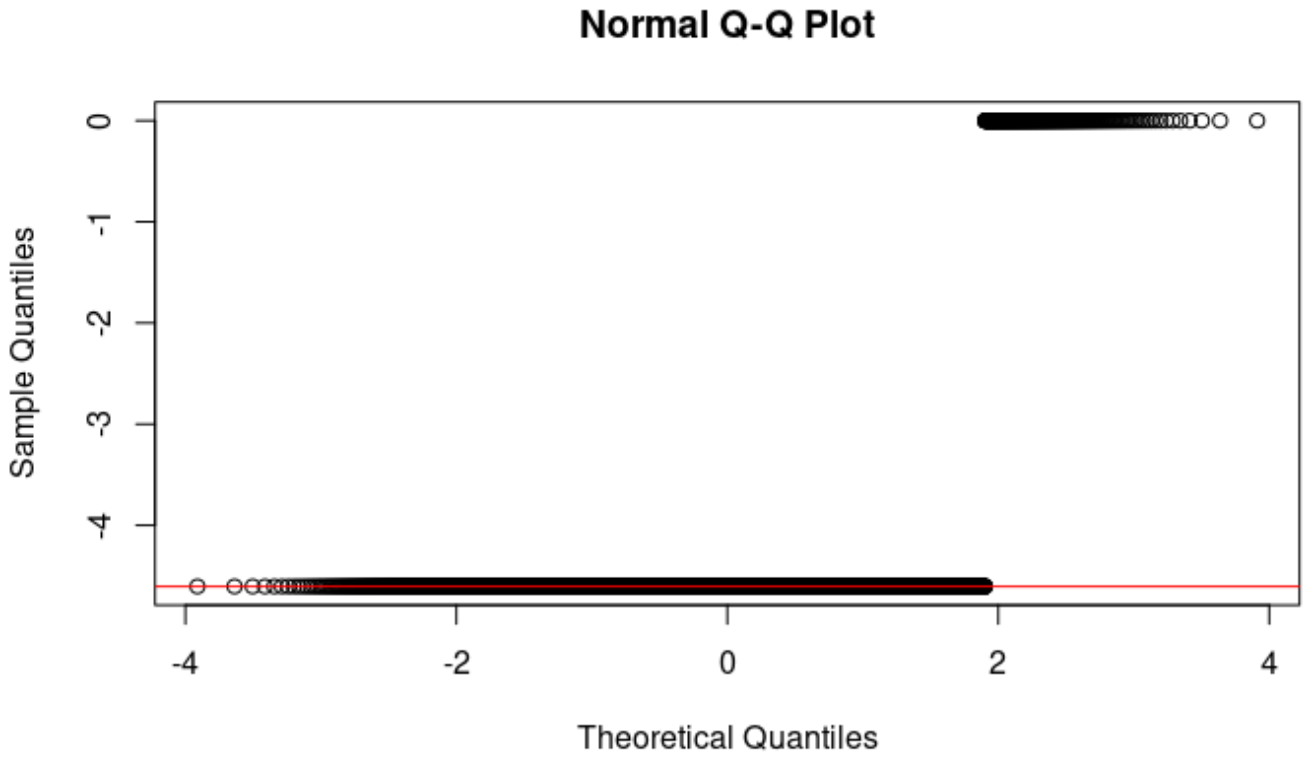
```
for (col in numeric_columns) {
  explore_and_transform(col)
}
```

```
Shapiro-Wilk test for season (sampled data): Statistic = 0.8633926 , p-value = 1.2903
82e-28
Box-Cox transformation applied for season
```

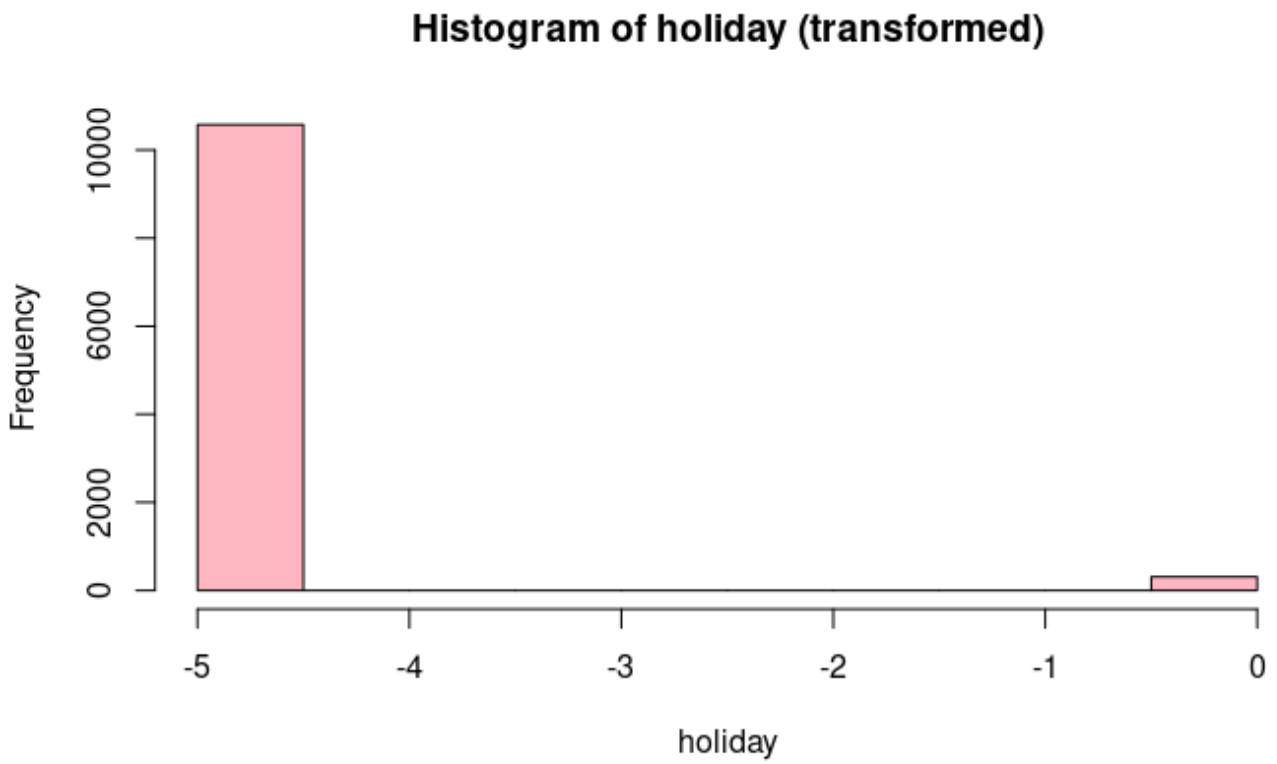


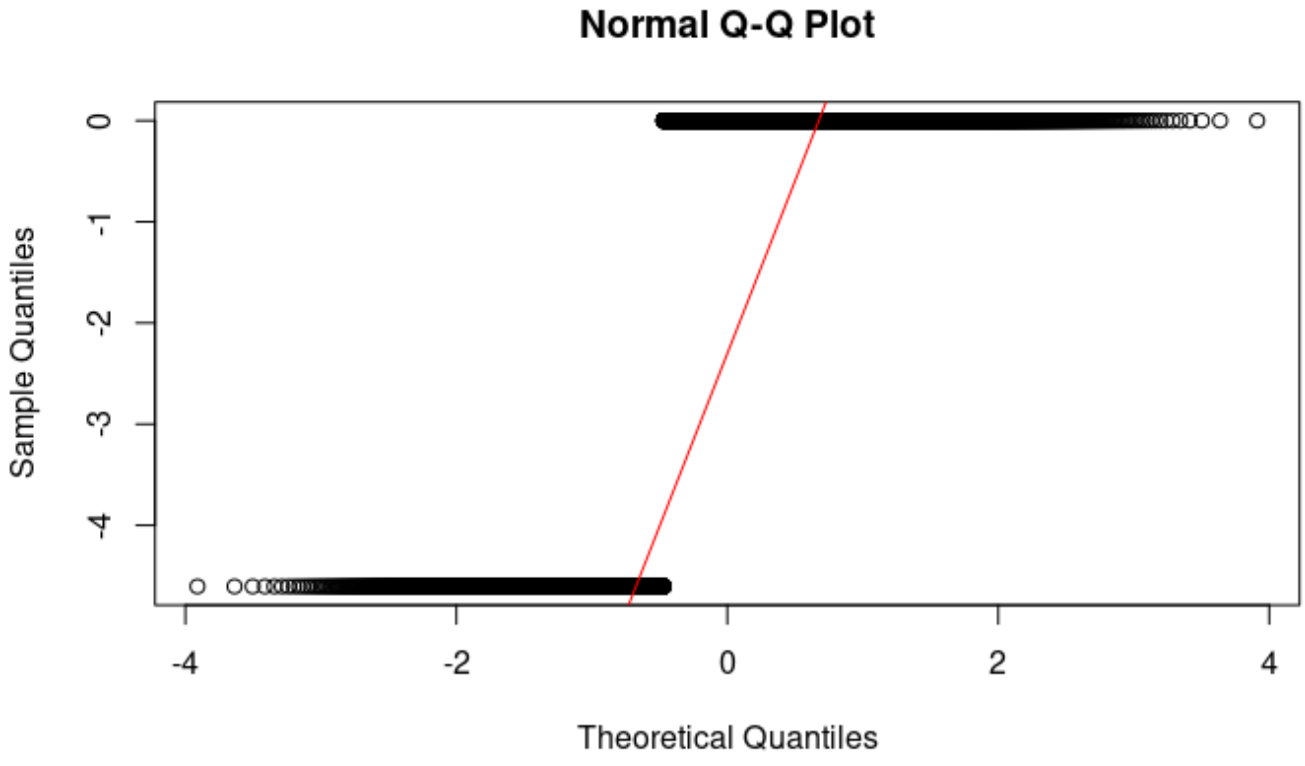

```
Shapiro-Wilk test for holiday (sampled data): Statistic = 0.1478553 , p-value = 5.932
161e-55
Box-Cox transformation applied for holiday
```



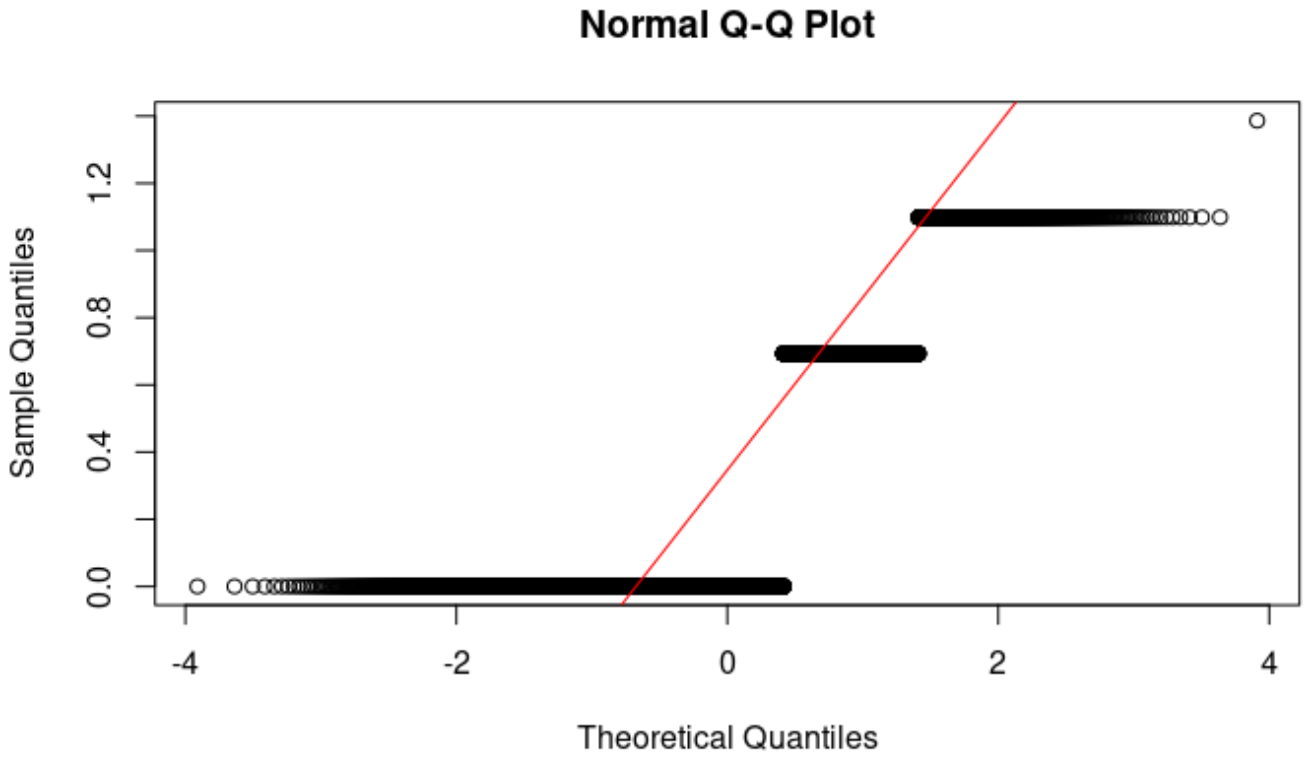



```
Shapiro-Wilk test for workingday (sampled data): Statistic = 0.5842002 , p-value = 1.
427267e-43
Box-Cox transformation applied for workingday
```

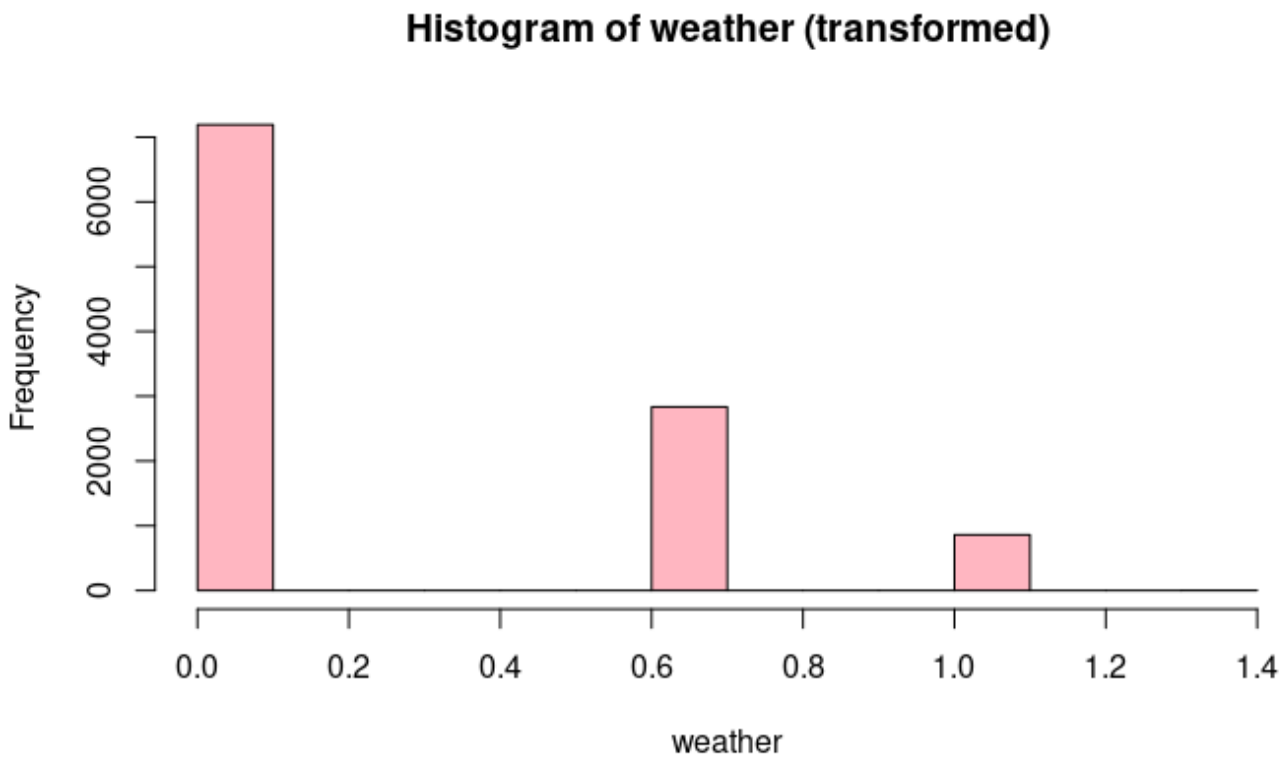


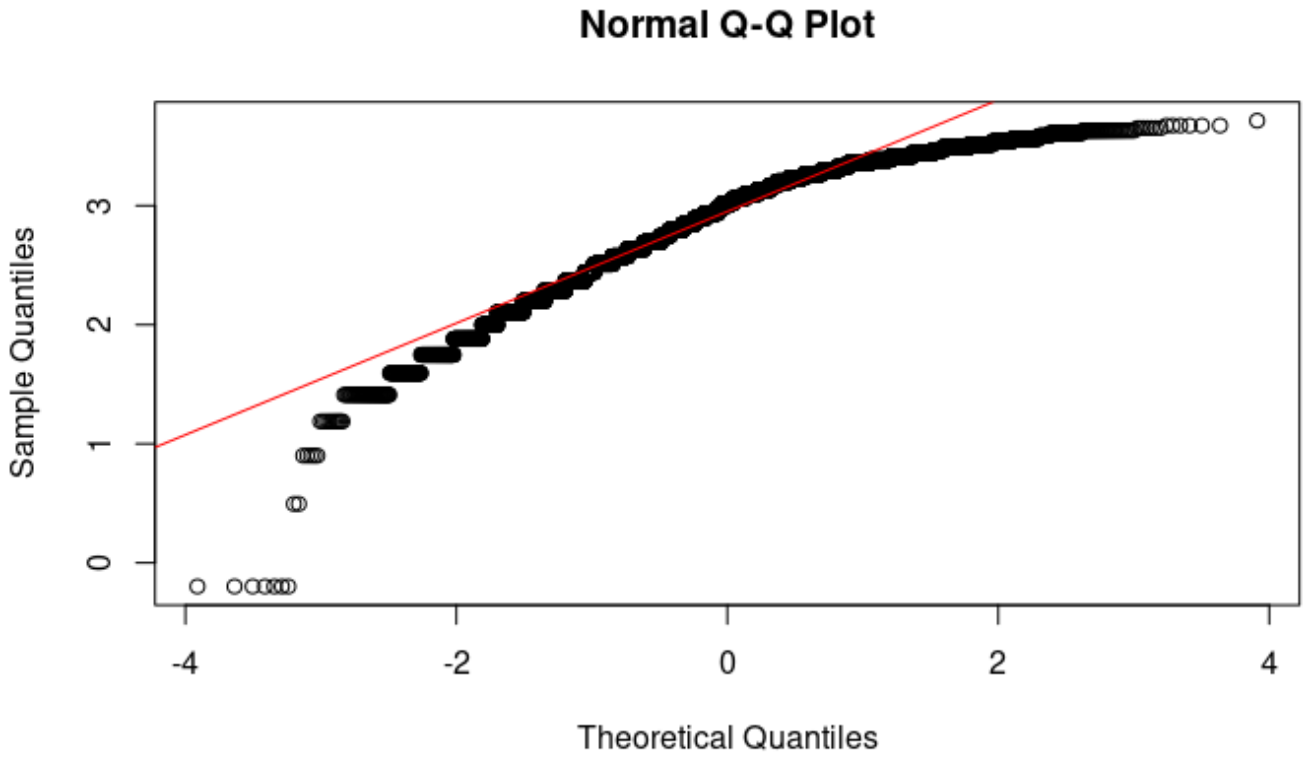



```
Shapiro-Wilk test for weather (sampled data): Statistic = 0.6546061 , p-value = 7.499
497e-41
Box-Cox transformation applied for weather
```

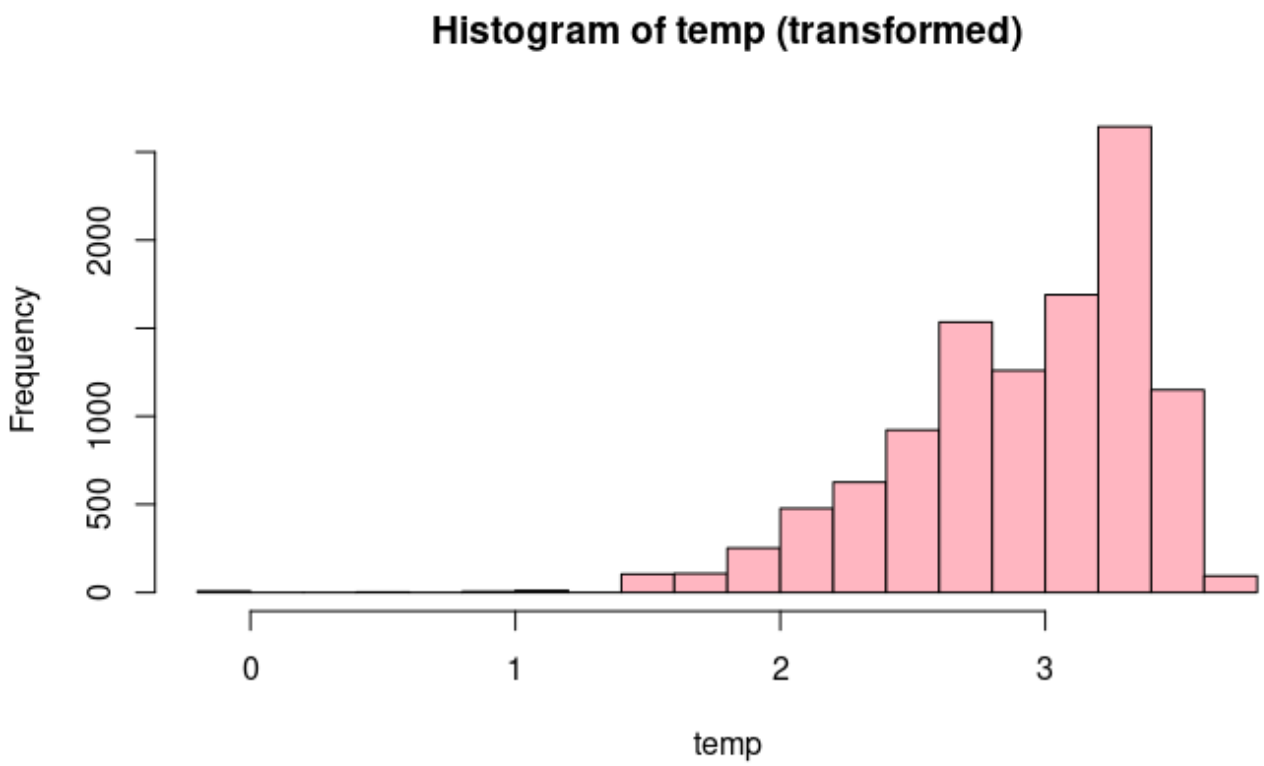


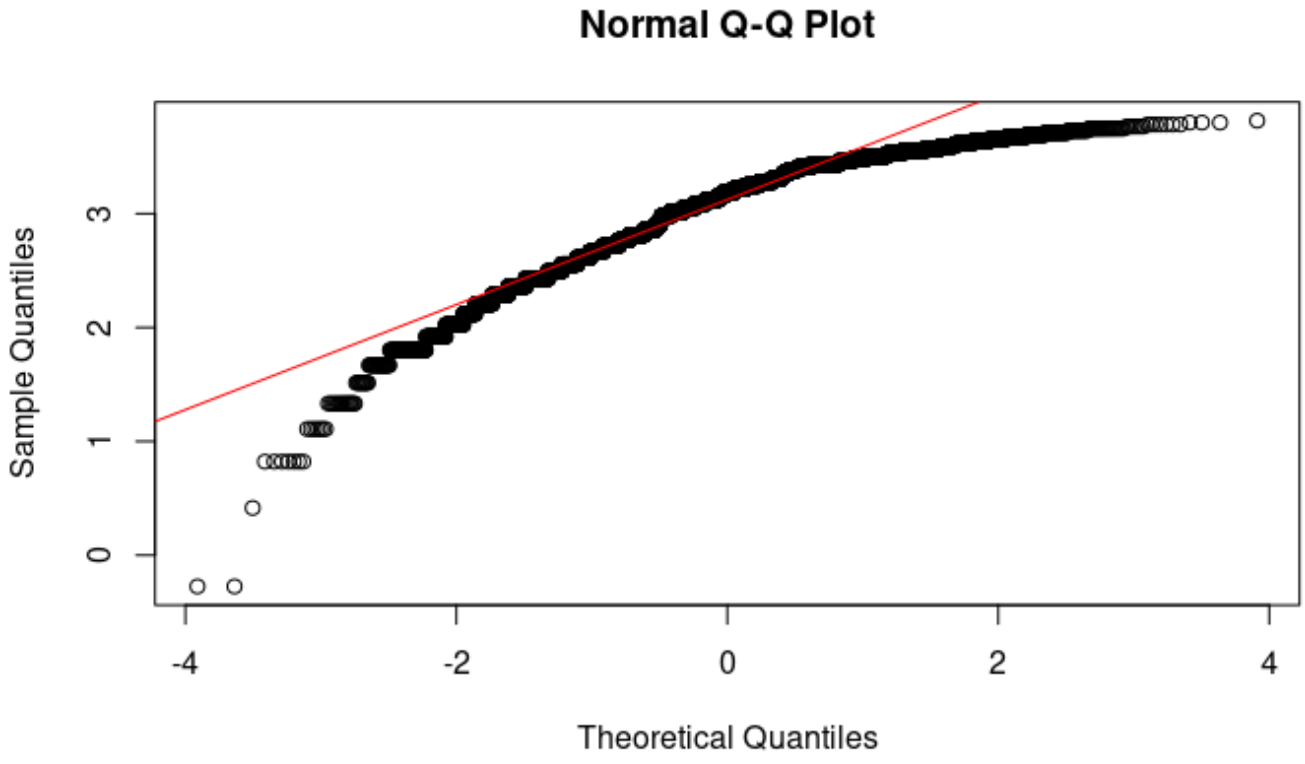

```
Shapiro-Wilk test for temp (sampled data): Statistic = 0.9782404 , p-value = 4.406632
e-11
Box-Cox transformation applied for temp
```



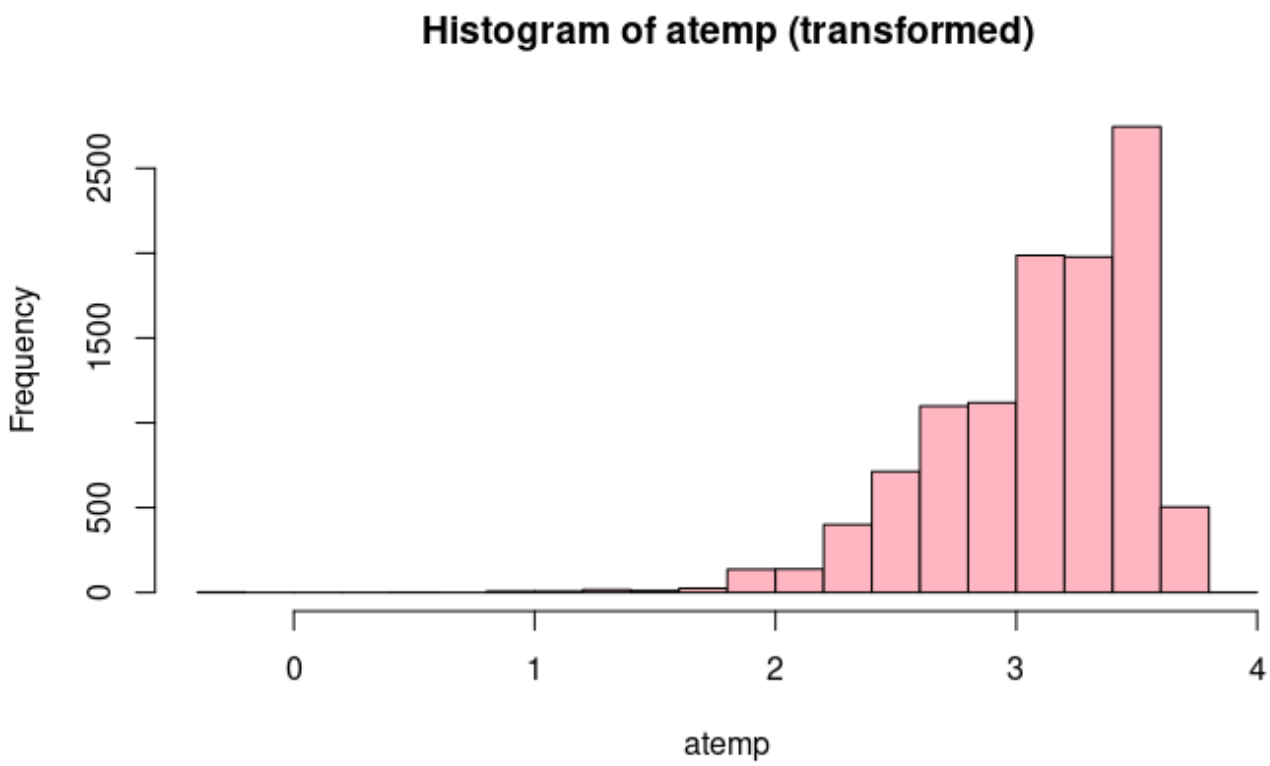


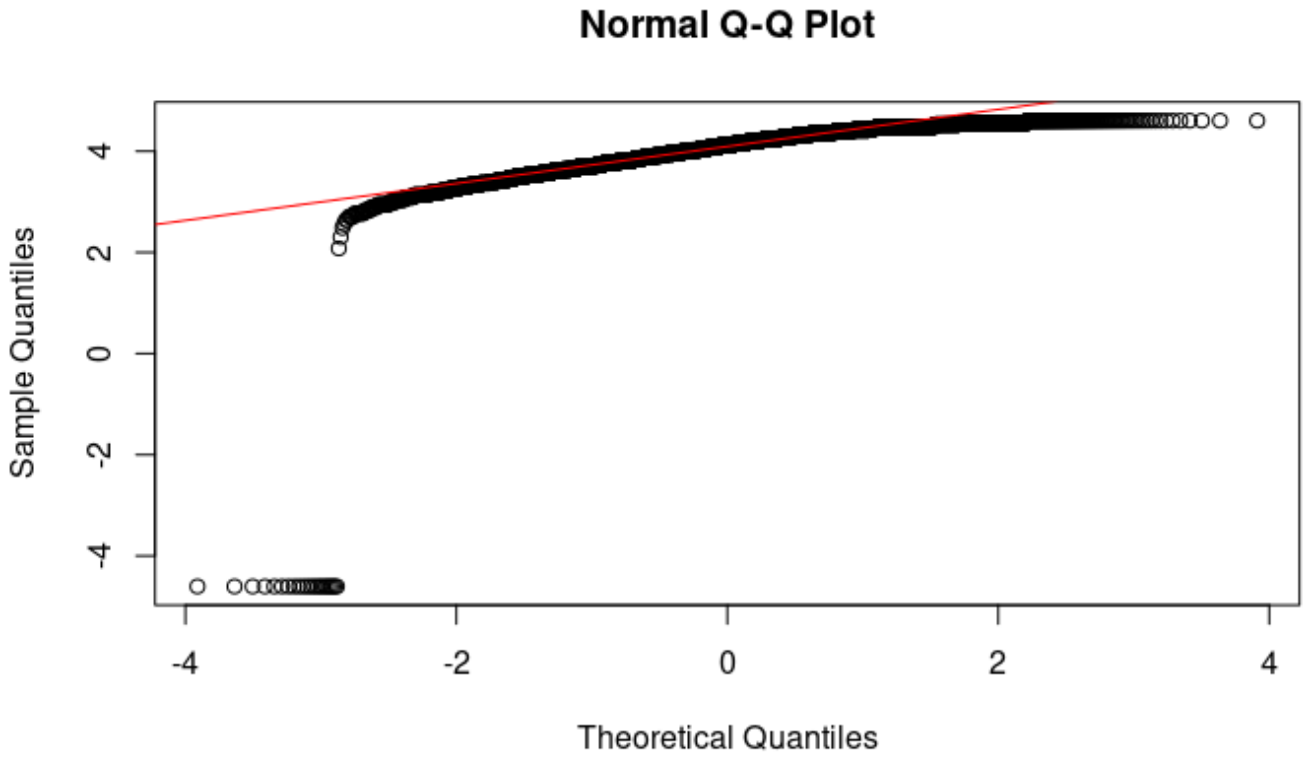

```
Shapiro-Wilk test for atemp (sampled data): Statistic = 0.9807313 , p-value = 3.10610
2e-10
Box-Cox transformation applied for atemp
```



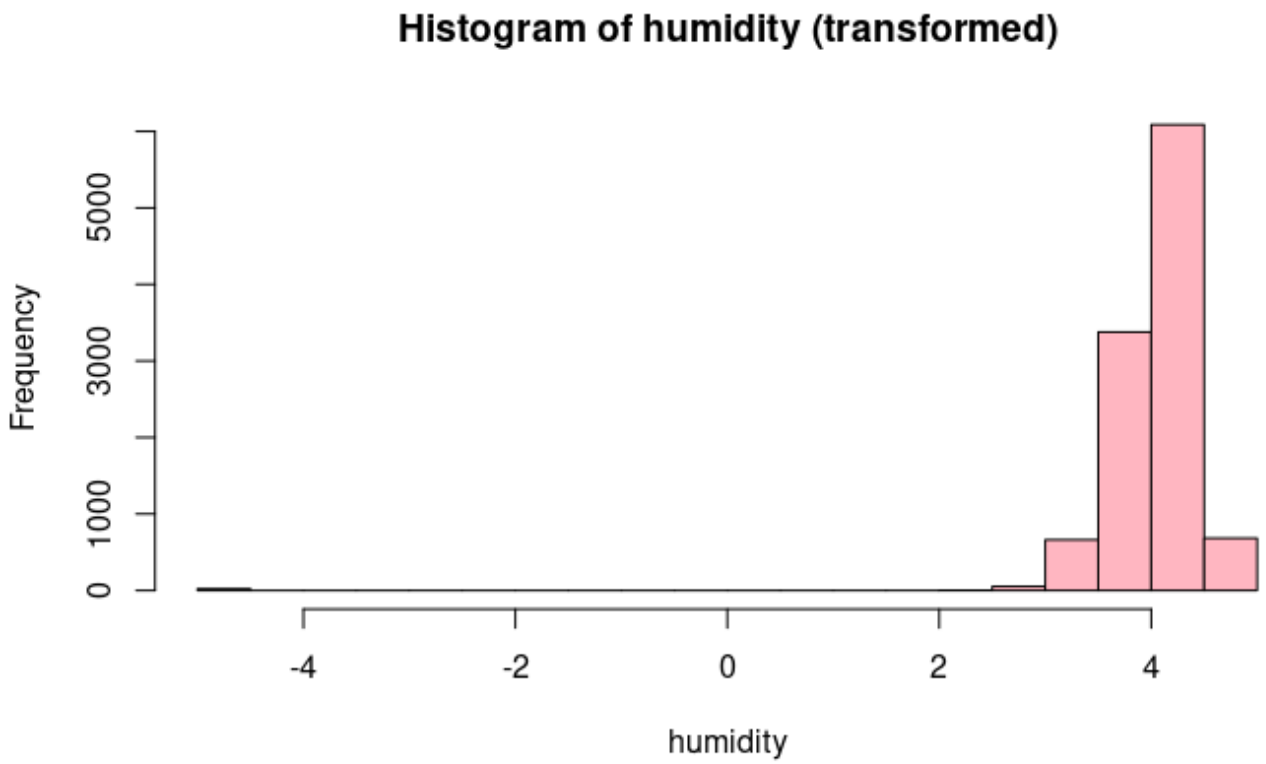


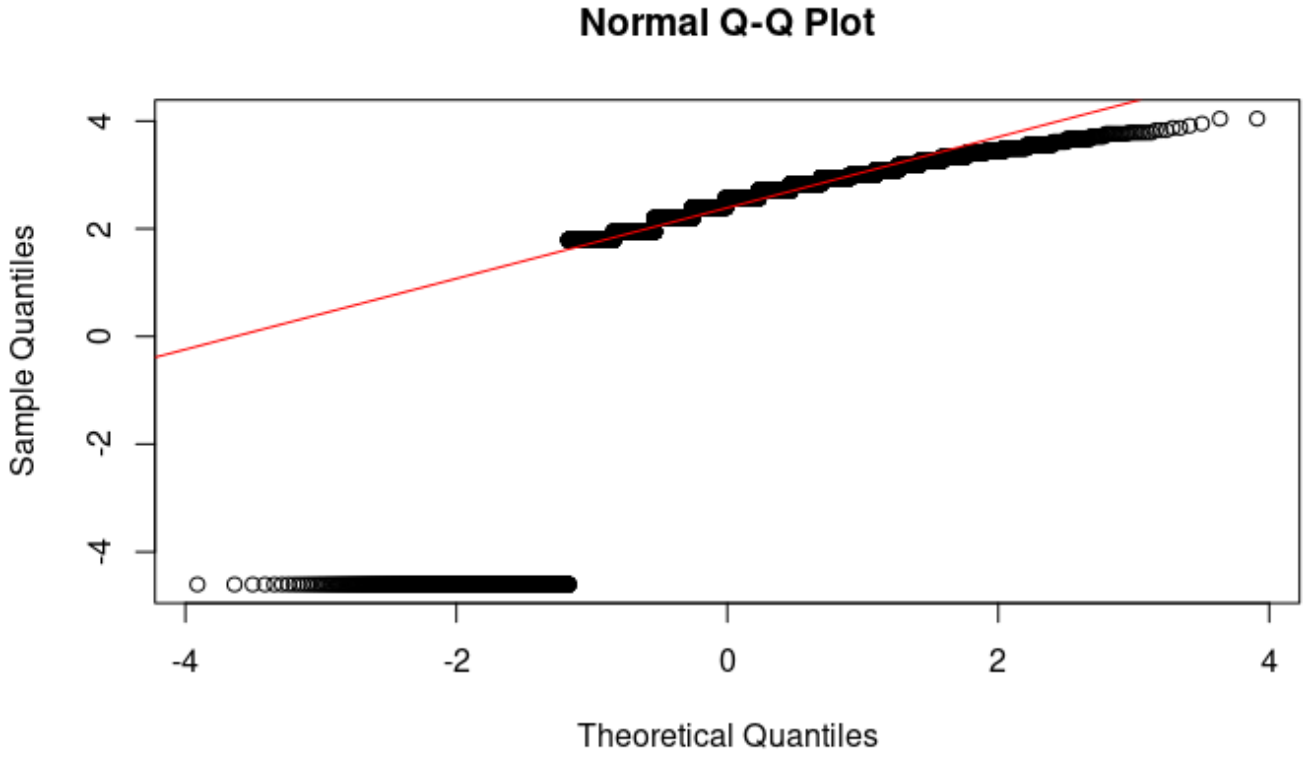

```
Shapiro-Wilk test for humidity (sampled data): Statistic = 0.9792964 , p-value = 9.90
4162e-11
Box-Cox transformation applied for humidity
```



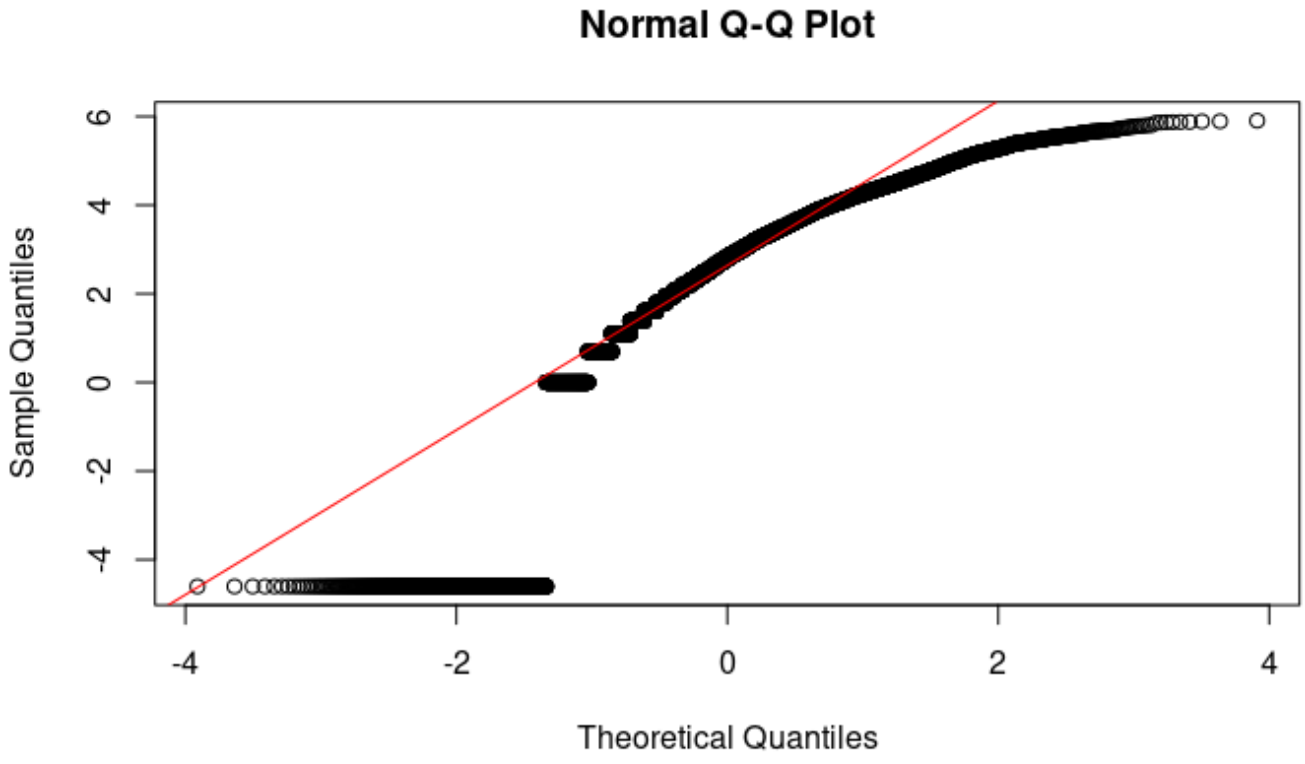



```
Shapiro-Wilk test for windspeed (sampled data): Statistic = 0.9543312 , p-value = 4.3
26822e-17
Box-Cox transformation applied for windspeed
```

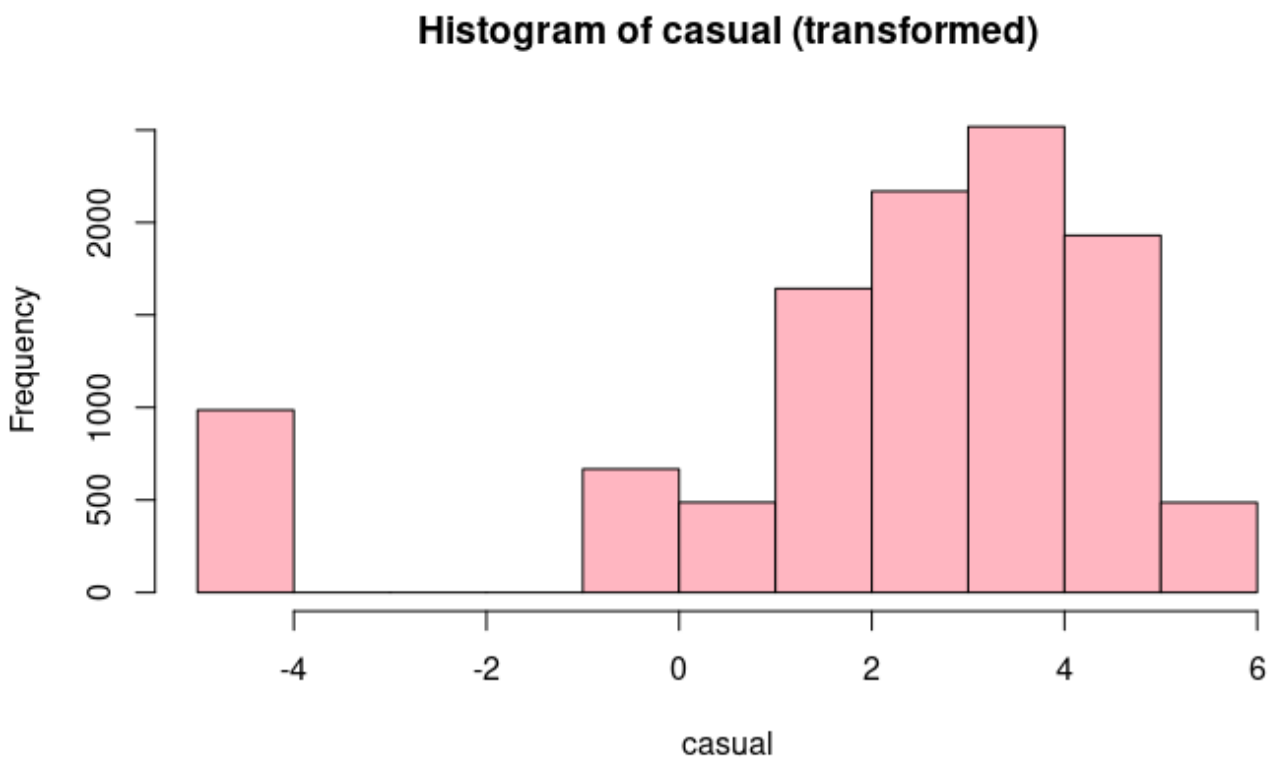


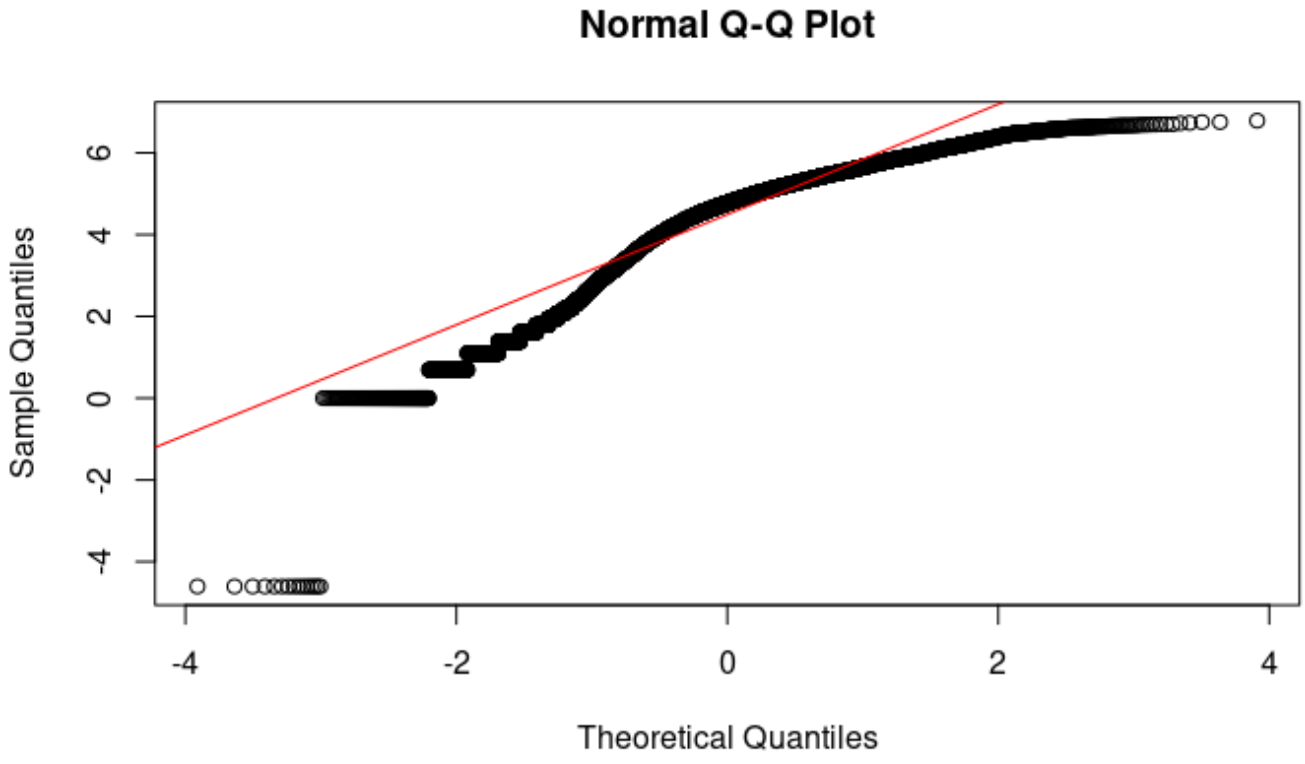



```
Shapiro-Wilk test for casual (sampled data): Statistic = 0.7028145 , p-value = 1.0295
68e-38
Box-Cox transformation applied for casual
```



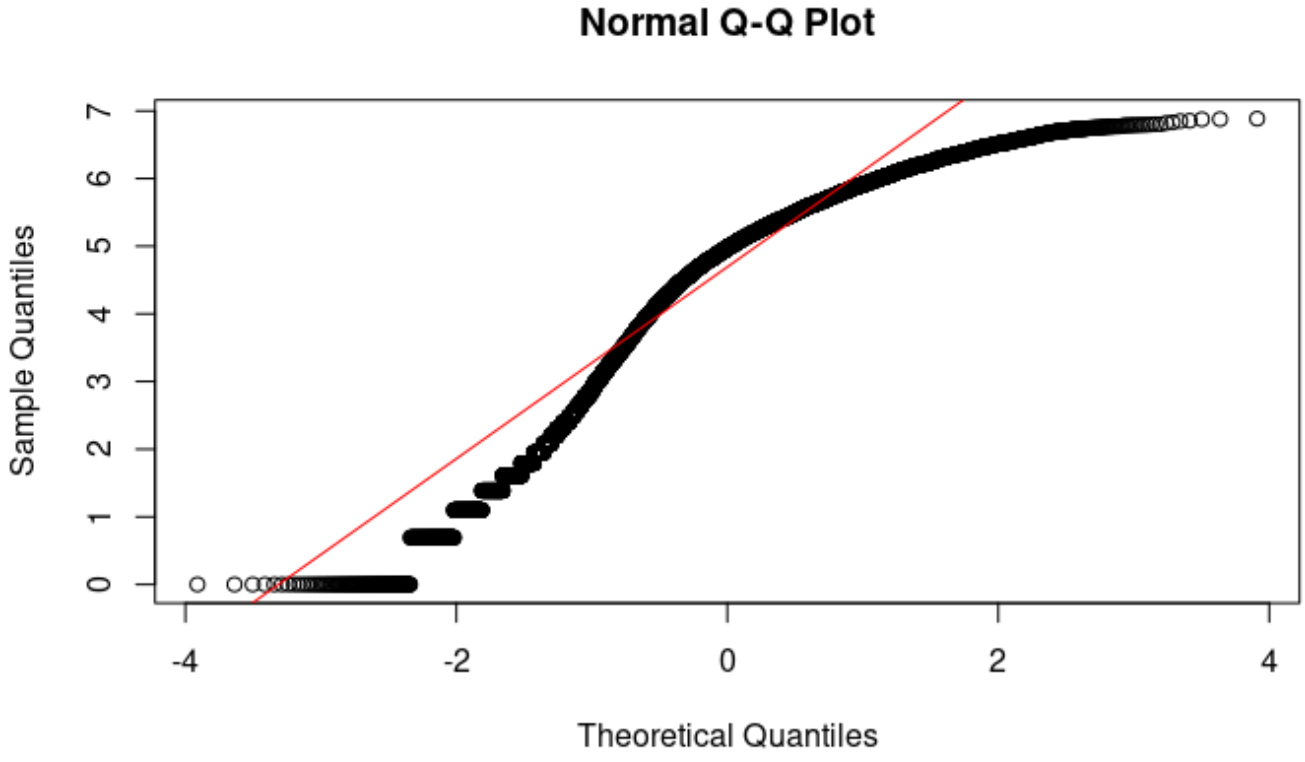

```
Shapiro-Wilk test for registered (sampled data): Statistic = 0.8406175 , p-value = 1.
715686e-30
Box-Cox transformation applied for registered
```

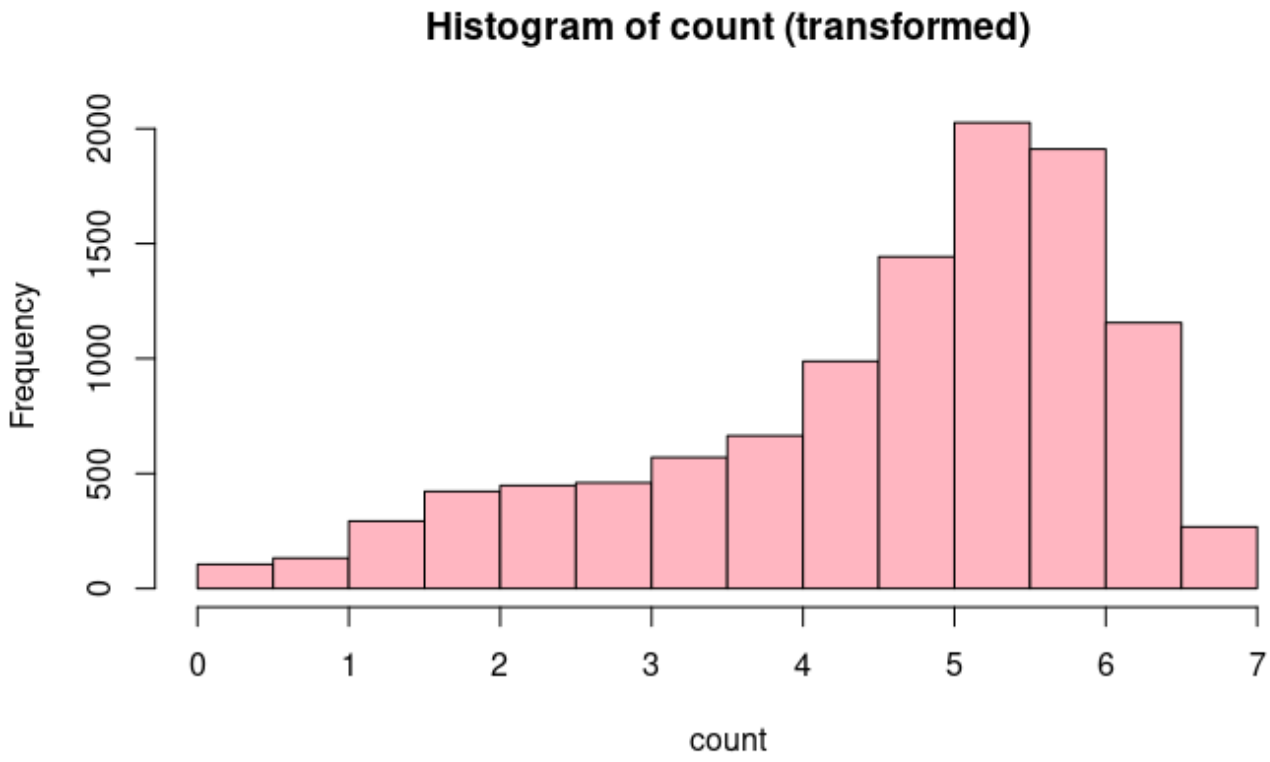





```
Shapiro-Wilk test for count (sampled data): Statistic = 0.8735969 , p-value = 1.07549
8e-27
Box-Cox transformation applied for count
```


Histogram of registered (transformed)
Frequency
0
1000
2000
3000
-4
-2
0
2
4
6
registered

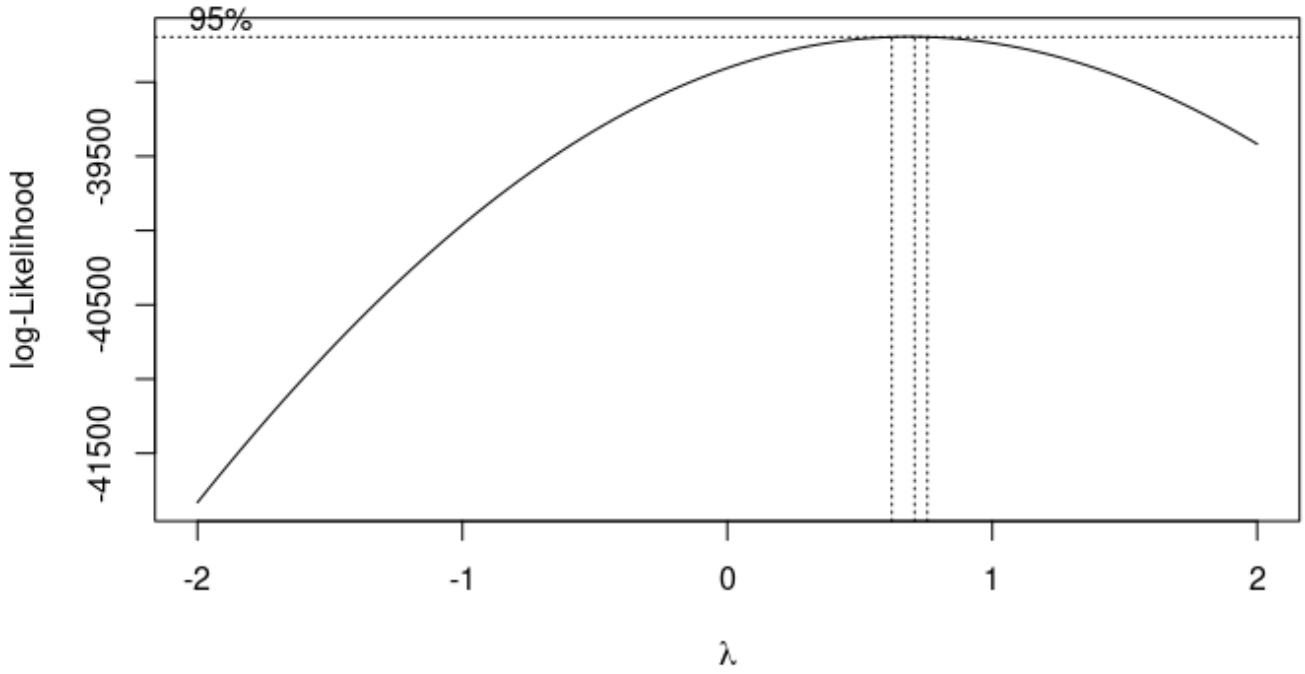

Normal Q-Q Plot
Sample Quantiles
0
1
2
3
4
5
6
7
-4
-2
0
2
4
Theoretical Quantiles



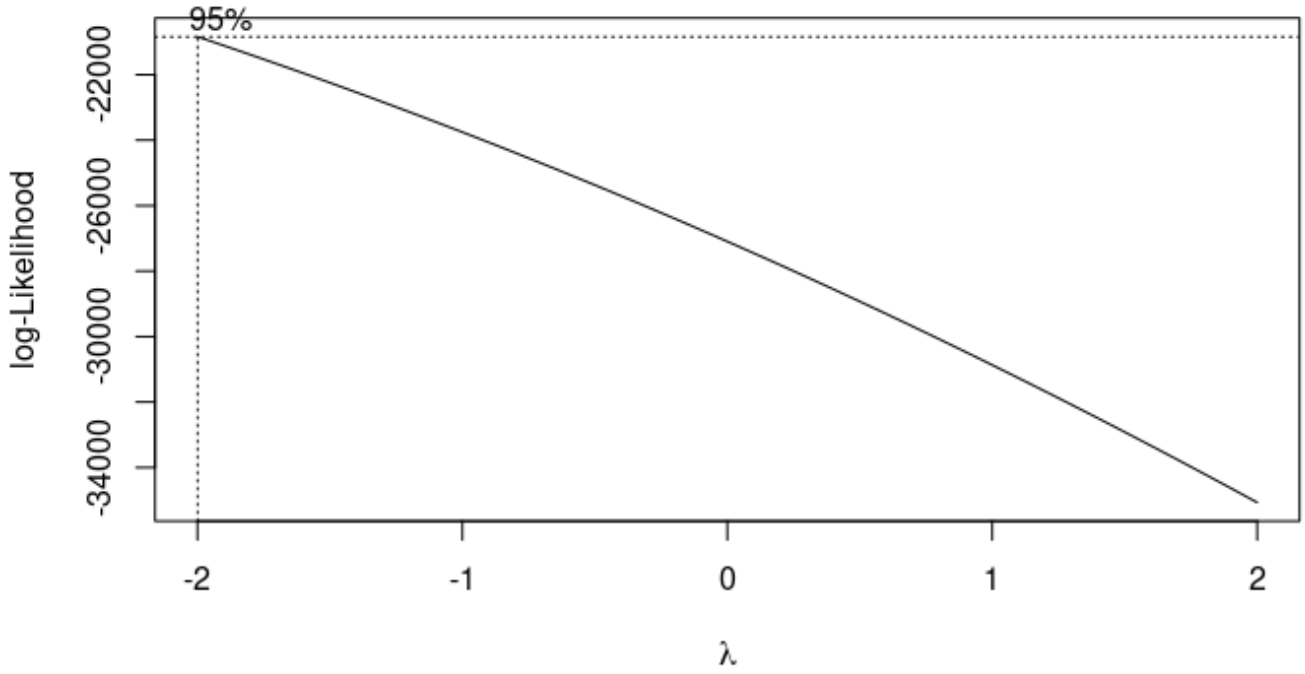
Hide

```
library(MASS)
num_cols <- names(df)[sapply(df, is.numeric)]
# Apply Box-Cox to Non-Normal Variables
for (col in num_cols) {
  # Perform Shapiro-Wilk test for normality
  if (length(df[[col]]) > 5000) {
    sample_data <- sample(df[[col]], 5000)
  } else {
    sample_data <- df[[col]]
  }
  test_result <- shapiro.test(sample_data)
  if (test_result$p.value <= 0.05) {
    bc_transform <- boxcox(df[[col]] + 1 ~ 1, lambda = seq(-2, 2, 0.1))
    optimal_lambda <- bc_transform$x[which.max(bc_transform$y)]
    df[[col]] <- (df[[col]] + 1)^optimal_lambda - 1
    cat(sprintf("Box-Cox transformed '%s' (optimal lambda: %.2f)\n", col, optimal_lam
bda))
  }
}
```

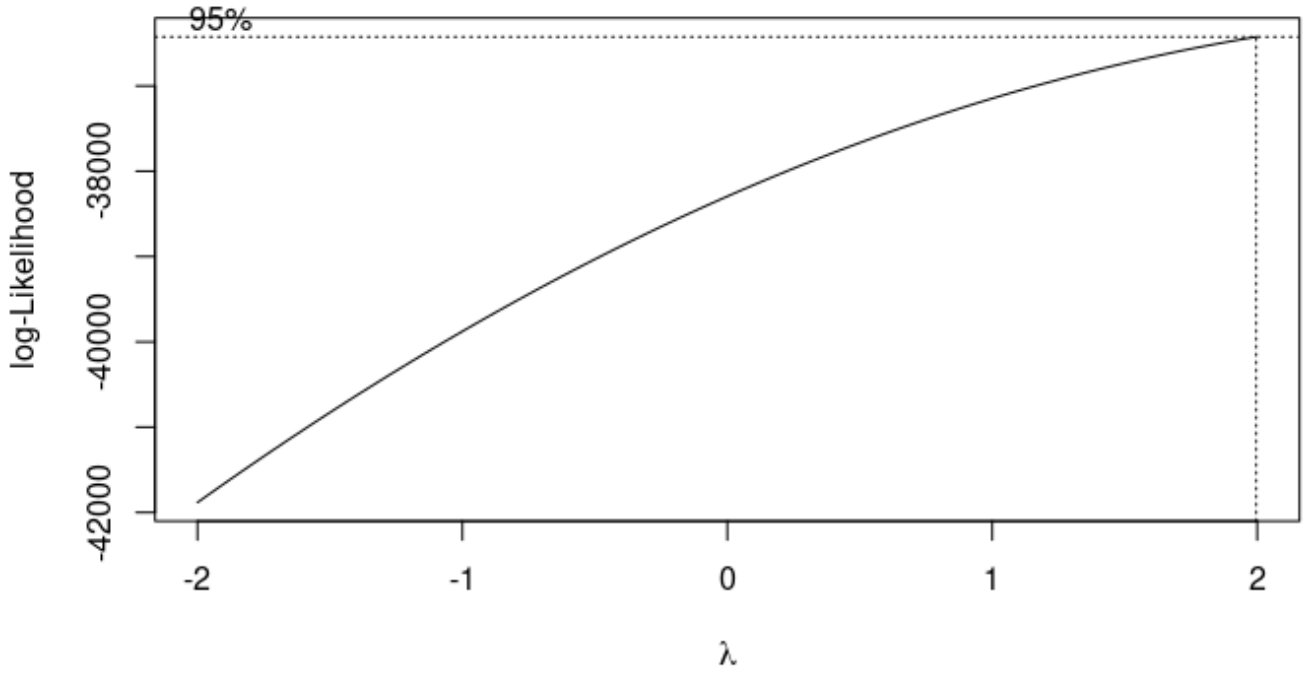
```
Box-Cox transformed 'season' (optimal lambda: 0.71)
```



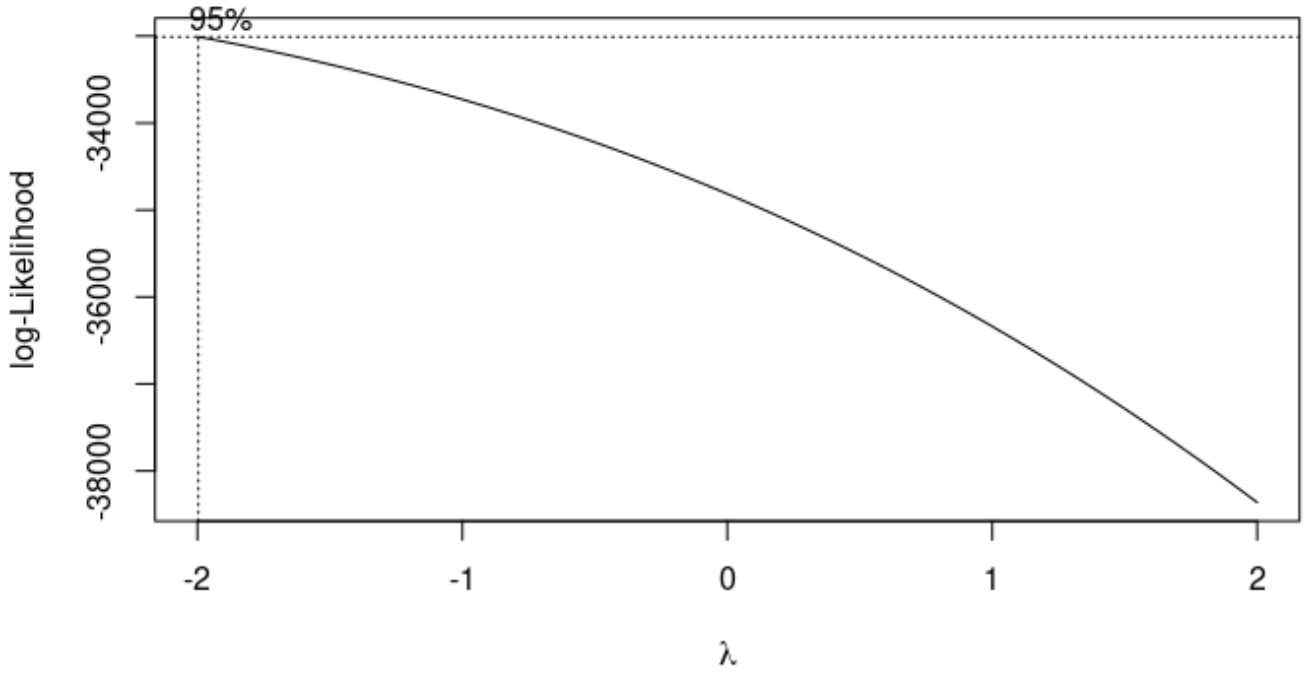

```
Box-Cox transformed 'holiday' (optimal lambda: -2.00)
```



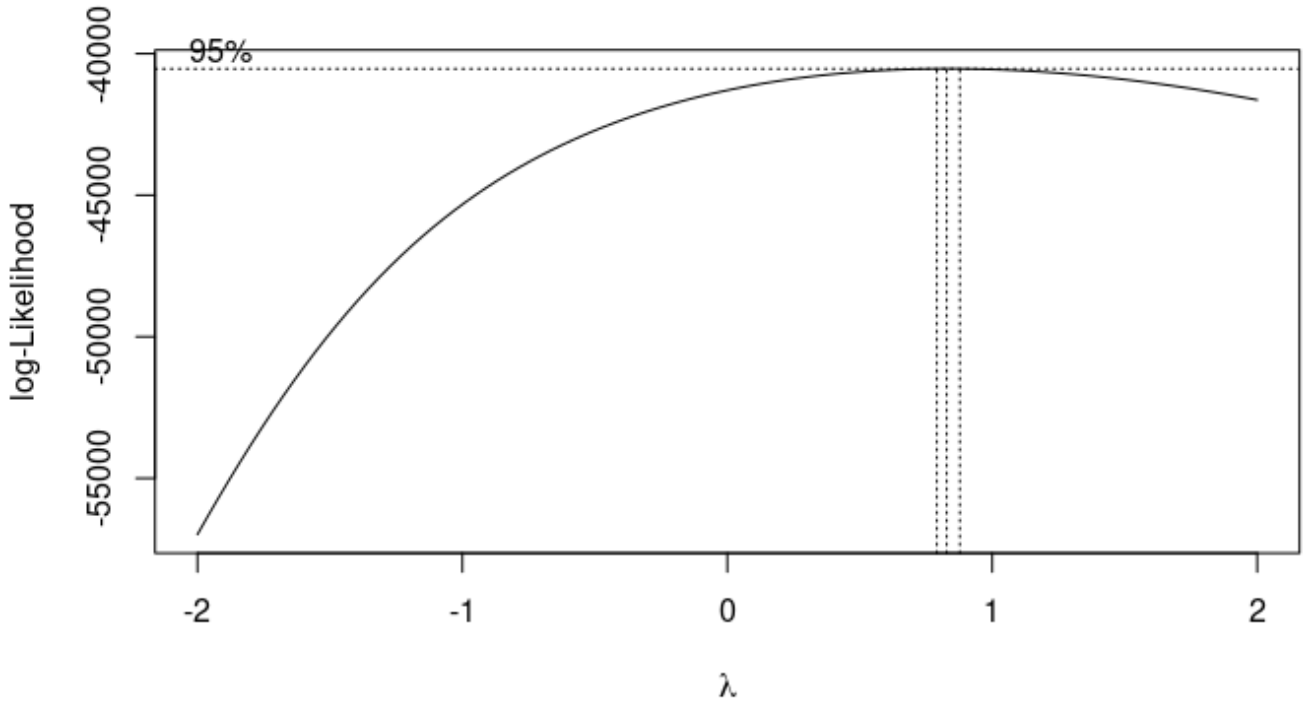

```
Box-Cox transformed 'workingday' (optimal lambda: 2.00)
```



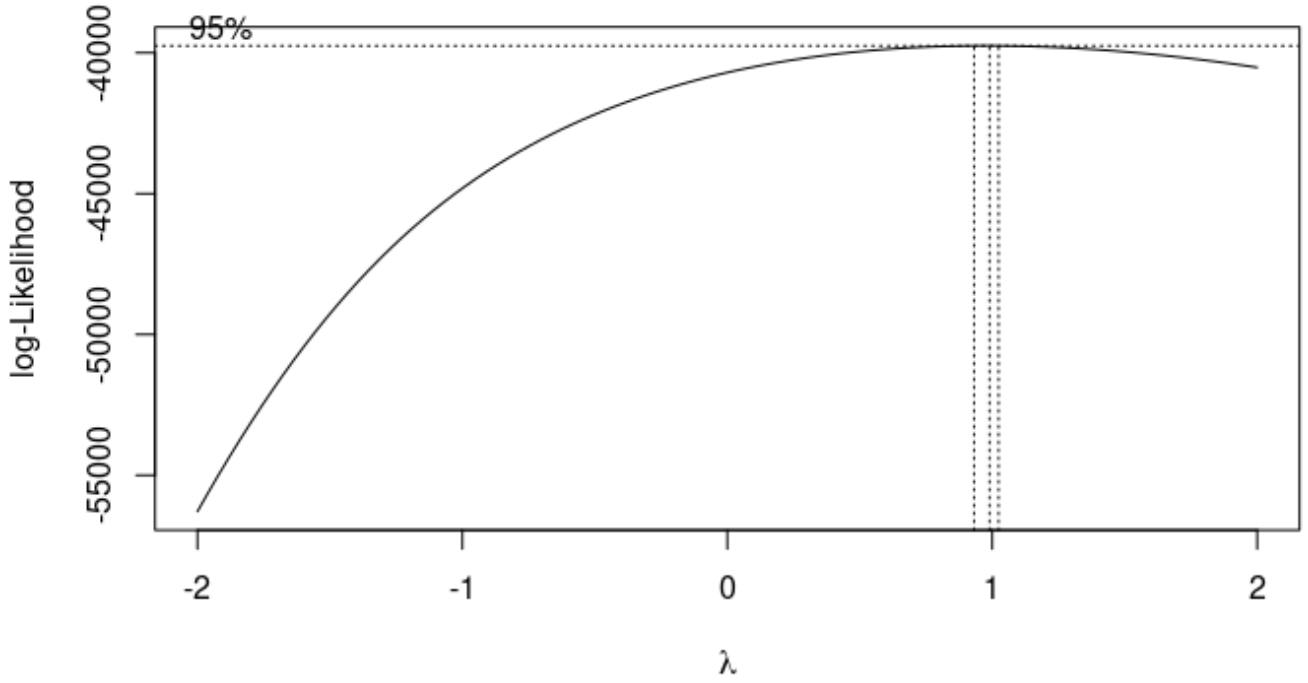

```
Box-Cox transformed 'weather' (optimal lambda: -2.00)
```



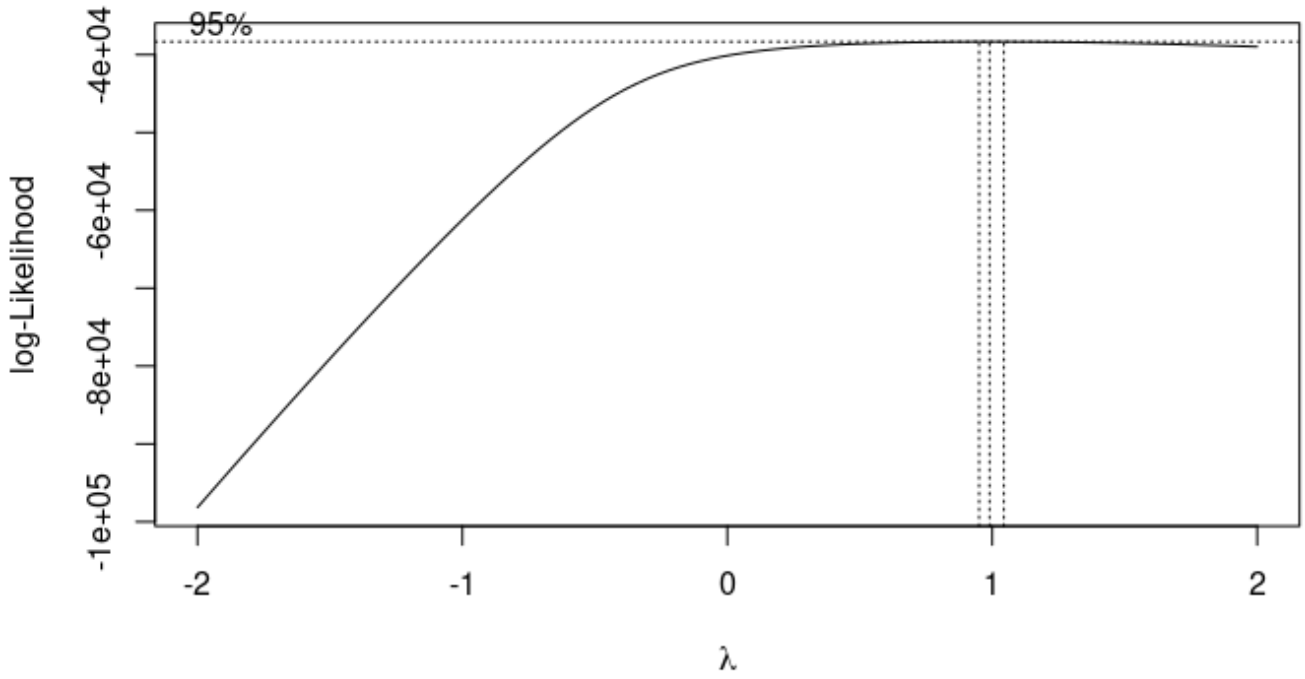

```
Box-Cox transformed 'temp' (optimal lambda: 0.83)
```



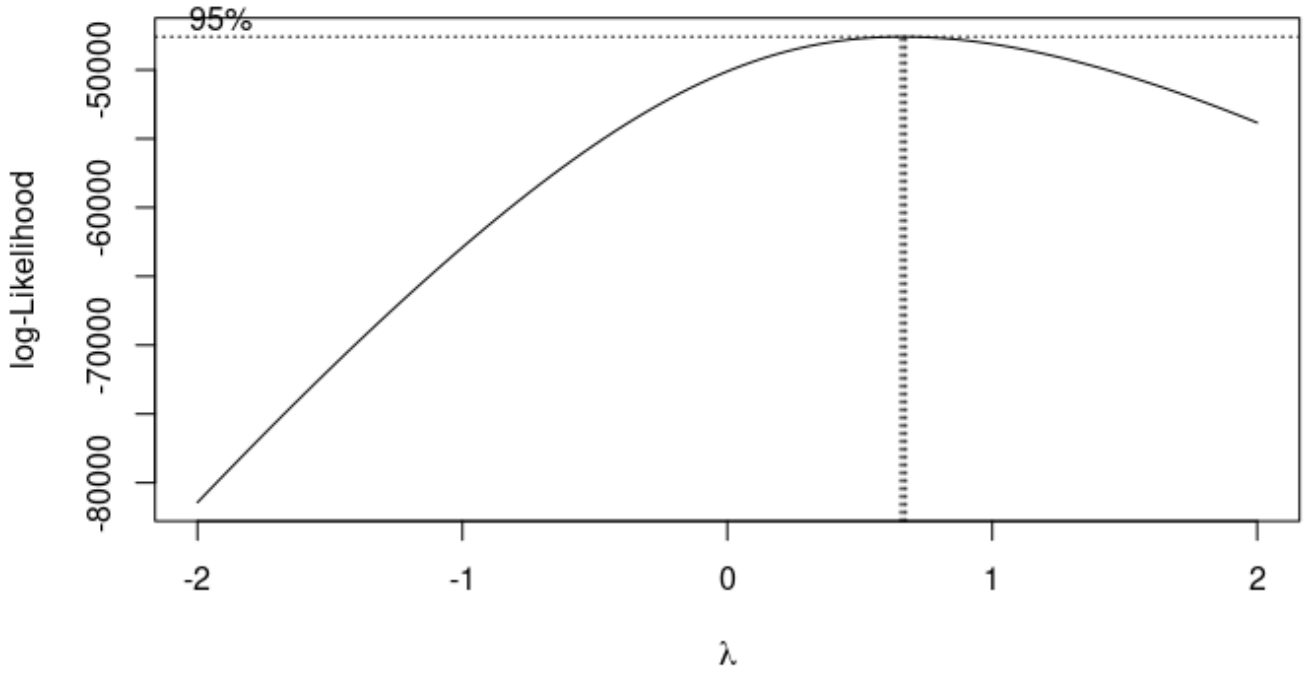

```
Box-Cox transformed 'atemp' (optimal lambda: 0.99)
```



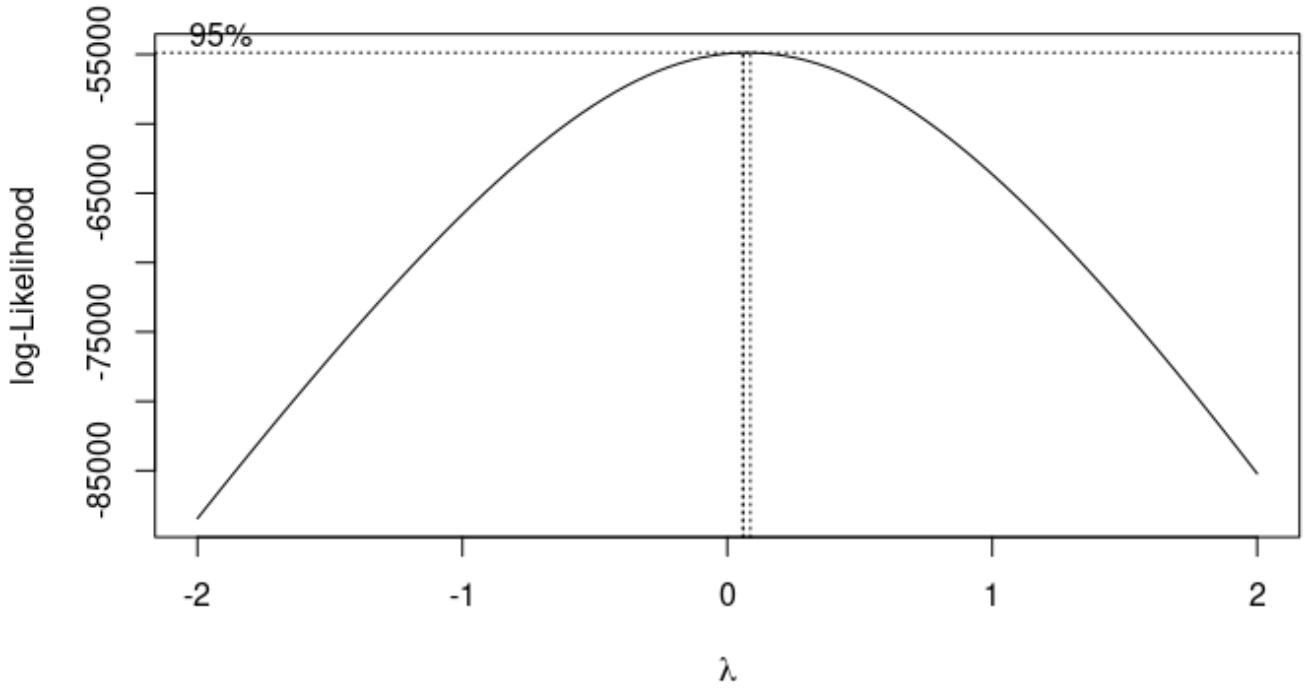

```
Box-Cox transformed 'humidity' (optimal lambda: 0.99)
```



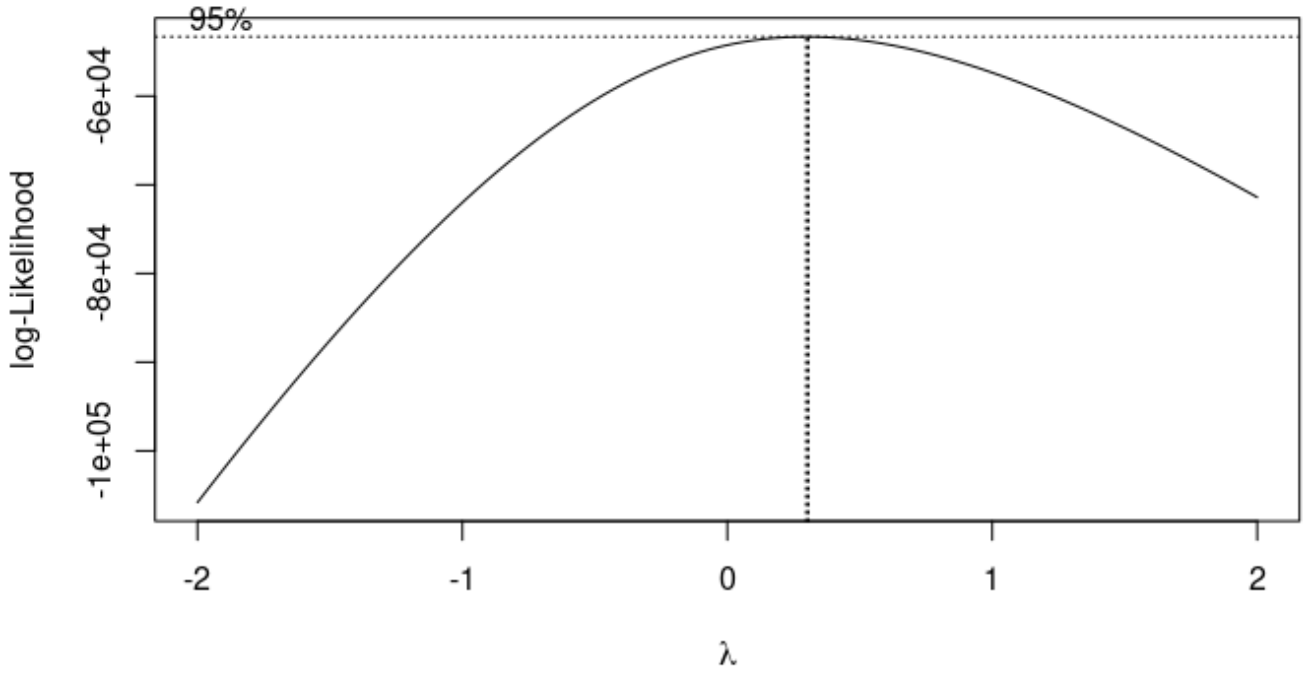

```
Box-Cox transformed 'windspeed' (optimal lambda: 0.67)
```



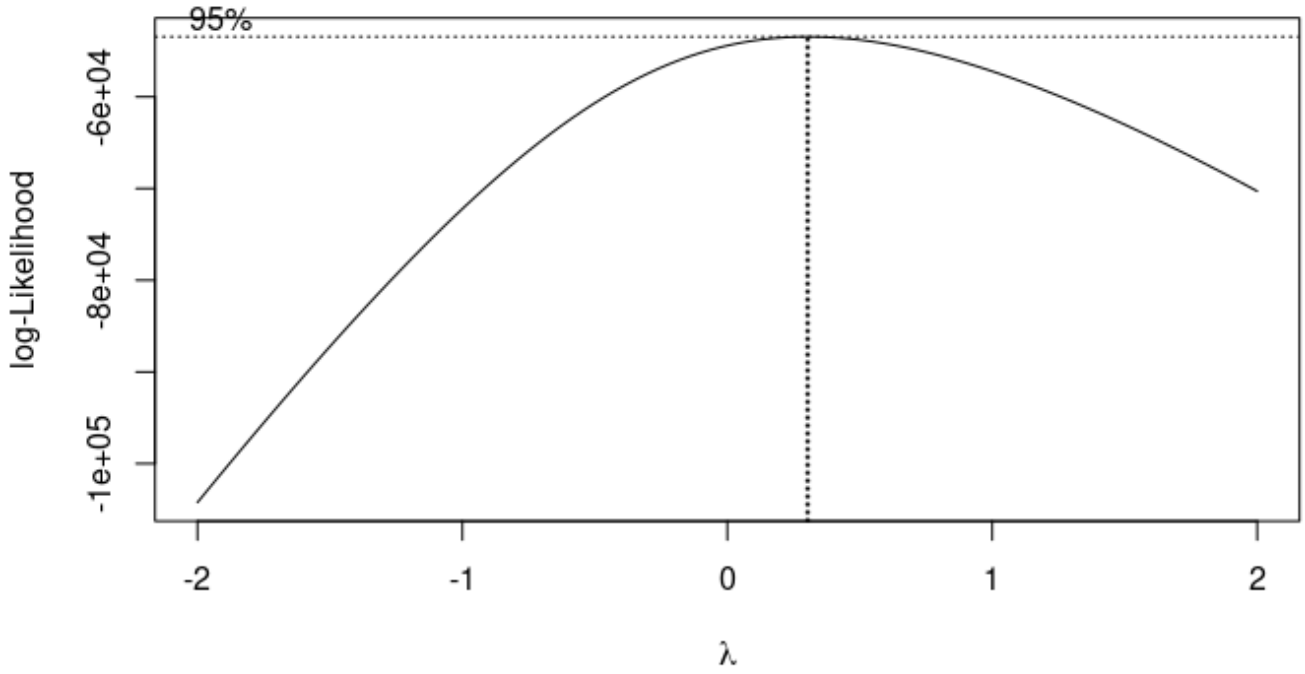

```
Box-Cox transformed 'casual' (optimal lambda: 0.06)
```




```
Box-Cox transformed 'registered' (optimal lambda: 0.30)
```




```
Box-Cox transformed 'count' (optimal lambda: 0.30)
```



Purpose of Box-Cox Transformation:

- The Box-Cox transformation is employed to reshape distributions that deviate from normality, making them more Gaussian in nature.
- The primary goal is to enhance the robustness of various statistical methods by ensuring that the data adheres more closely to normal distribution assumptions.

Role of Lambda (λ) Parameter:

- The λ parameter plays a crucial role in defining the nature of the transformation.
- A λ value around 1 suggests that minimal alteration is required for normalization.
- A λ close to 0 indicates that a logarithmic transformation provides an optimal approximation.
- λ values less than 0 signify the need for inverse transformations to achieve normality.

Application to Specific Variables:

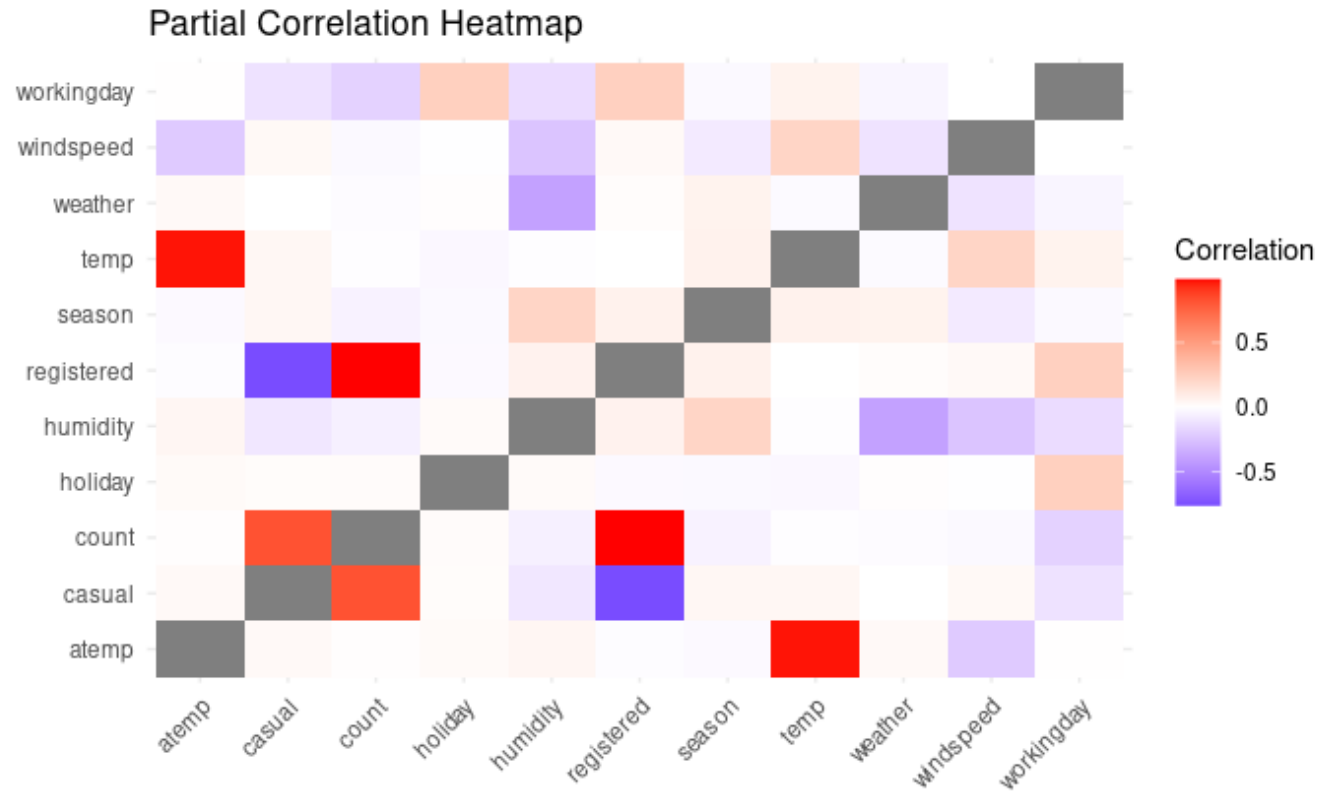
- Variables such as ‘holiday’ and ‘weather’ exhibited significantly negative lambda values, indicating highly skewed distributions.
- Negative lambda values suggest that applying the Box-Cox transformation in an inverse form brought these variables closer to a normal distribution.
- On the other hand, variables like ‘season’, ‘windspeed’, ‘temp’, ‘registered’, and ‘count’ showed positive lambda values closer to 1.
- Positive lambda values imply moderate skewness, and the Box-Cox transformation effectively improved the distributional shape of these variables.
- Variables like ‘atemp’ and ‘humidity’ with a lambda of 1 indicate that the Box-Cox transformation had a minimal effect on their distributions, suggesting that they were already reasonably normal in their raw form.

In summary, the Box-Cox transformation serves as a versatile tool for normalizing distributions, and the interpretation of lambda values provides insights into the degree of skewness and the effectiveness of the transformation for specific variables.

3. Explore partial or semi-partial correlations

Hide

```
library(ppcor)
library(reshape)
numeric_columns <- names(df)[sapply(df, is.numeric)]
partial_corrs <- matrix(NA, nrow = length(numeric_columns), ncol = length(numeric_col
umns))
rownames(partial_corrs) <- numeric_columns
colnames(partial_corrs) <- numeric_columns
# Loop over all pairs of variables
for (var1 in numeric_columns) {
  for (var2 in numeric_columns) {
    if (var1 != var2) {
      # Get the names of the control variables
      covars <- setdiff(numeric_columns, c(var1, var2))
      # Calculate the partial correlation
      partial_corr <- pcor.test(df[[var1]], df[[var2]], df[covars])
      # Store the partial correlation in the matrix
      partial_corrs[var1, var2] <- partial_corr$estimate
    }
  }
}
# Convert the matrix to a dataframe for plotting
partial_corrs_df <- melt(partial_corrs)
names(partial_corrs_df) <- c("Var1", "Var2", "Correlation")
# Plot the heatmap
ggplot(data = partial_corrs_df, aes(x = Var1, y = Var2, fill = Correlation))+geom_til
e() +scale_fill_gradient2(low = "blue", high ="red", mid = "white", midpoint =0) +the
me_minimal() +theme(axis.text.x = element_text(angle = 45, hjust = 1)) + labs(title =
"Partial Correlation Heatmap", x = "", y = "")
```



High Positive Correlation:

- **atemp vs temp:**
  - The variables `atemp` (feels-like temperature) and `temp` (actual temperature) exhibit a high positive correlation. This suggests that as the actual temperature increases, the perceived or feels-like temperature also tends to increase.
- **count vs registered:**
  - There is a strong positive correlation between the total count of users (`count`) and the count of registered users (`registered`). This indicates that as the overall usage increases, the number of registered users also tends to increase.
- **count vs casual:**
  - The total count of users (`count`) shows a significant positive correlation with the count of casual users (`casual`). This implies that as the total usage increases, the number of casual users also tends to increase.

High Negative Correlation:

- **casual vs registered:**
  - There is a substantial negative correlation between the count of casual users (`casual`) and the count of registered users (`registered`). This suggests an inverse relationship, indicating that when the count of casual users increases, the count of registered users tends to decrease and vice versa.
- **humidity vs weather:**
  - The variables `humidity` and `weather` exhibit a notable negative correlation. This implies that as humidity increases, the weather condition tends to be less severe or extreme.

It's important to note that these observations are made while controlling for all other variables (covariates) in the dataset. Partial correlation allows us to assess the relationship between two variables while taking into account the potential influence of other variables, providing a more refined understanding of their association.