Untitled

Ritika Pandey

2024-12-07

```
options(repos = c(CRAN = "https://cran.rstudio.com"))
options(warn = -1)


library(lubridate)

##
## Attaching package: 'lubridate'

## The following objects are masked from 'package:base':
##
##     date, intersect, setdiff, union

library(dplyr)

##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union

library(ggplot2)
library('fastDummies')
library(gridExtra)

##
## Attaching package: 'gridExtra'

## The following object is masked from 'package:dplyr':
##
##     combine

library(corrplot)

## corrplot 0.95 loaded

library(ggcorrplot)
library(PerformanceAnalytics)

## Loading required package: xts
```

```
## Loading required package: zoo

##
## Attaching package: 'zoo'

## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric

##
## ############################ Warning from 'xts' package
############################
## #
#
## # The dplyr lag() function breaks how base R's lag() function is supposed
to  #
## # work, which breaks lag(my_xts). Calls to lag(my_xts) that you type or
#
## # source() into this session won't work correctly.
#
## #
#
## # Use stats::lag() to make sure you're not using dplyr::lag(), or you can
add #
## # conflictRules('dplyr', exclude = 'lag') to your .Rprofile to stop
#
## # dplyr from breaking base R's lag() function.
#
## #
#
## # Code in packages is not affected. It's protected by R's namespace
mechanism #
## # Set `options(xts.warn_dplyr_breaks_lag = FALSE)` to suppress this
warning.  #
## #
#
##
################################################################################
##

##
## Attaching package: 'xts'

## The following objects are masked from 'package:dplyr':
##
##     first, last

##
## Attaching package: 'PerformanceAnalytics'
```

```
## The following object is masked from 'package:graphics':
##
##     legend
```

```
library(tidyr)
library(GGally)
```

```
## Registered S3 method overwritten by 'GGally':
##   method from
##   +.gg   ggplot2
```

```
library(car)
```

```
## Loading required package: carData
```

```
##
## Attaching package: 'car'
```

```
## The following object is masked from 'package:dplyr':
##
##     recode
```

```
library(e1071)
```

```
##
## Attaching package: 'e1071'
```

```
## The following objects are masked from 'package:PerformanceAnalytics':
##
##     kurtosis, skewness
```

```
library(caret)
```

```
## Loading required package: lattice
```

```
library(glmnet)
```

```
## Loading required package: Matrix
```

```
##
## Attaching package: 'Matrix'
```

```
## The following objects are masked from 'package:tidyr':
##
##     expand, pack, unpack
```

```
## Loaded glmnet 4.1-8
```

```
library(class)
library(rpart)
library(randomForest)
```

```
## randomForest 4.7-1.2
```

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

##
## Attaching package: 'randomForest'

## The following object is masked from 'package:gridExtra':
##
##     combine

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

## The following object is masked from 'package:dplyr':
##
##     combine

library(FNN)

##
## Attaching package: 'FNN'

## The following objects are masked from 'package:class':
##
##     knn, knn.cv

library(xgboost)

##
## Attaching package: 'xgboost'

## The following object is masked from 'package:dplyr':
##
##     slice

library(stats)
library(lmtest)
library(MASS)

##
## Attaching package: 'MASS'

## The following object is masked from 'package:dplyr':
##
##     select

library(mgcv)

## Loading required package: nlme

##
## Attaching package: 'nlme'
```

```
## The following object is masked from 'package:dplyr':
##
##     collapse

## This is mgcv 1.9-1. For overview type 'help("mgcv-package")'.

library(Metrics)

##
## Attaching package: 'Metrics'

## The following objects are masked from 'package:caret':
##
##     precision, recall

library(randomForest)
library(WARN)



## 1. Data Understanding and Preprocessing

##a) Data Inspection:
##Loading the dataset
seoul_bike_data = read.csv("C:/Users/ritik/Downloads/SeoulBikeData.csv",
fileEncoding = "latin1")
head(seoul_bike_data)

##         Date Rented.Bike.Count Hour Temperature..C. Humidity...
## 1 01/12/2017               254    0            -5.2          37
## 2 01/12/2017               204    1            -5.5          38
## 3 01/12/2017               173    2            -6.0          39
## 4 01/12/2017               107    3            -6.2          40
## 5 01/12/2017                78    4            -6.0          36
## 6 01/12/2017               100    5            -6.4          37
##   Wind.speed..m.s. Visibility..10m. Dew.point.temperature..C.
## 1              2.2             2000                     -17.6
## 2              0.8             2000                     -17.6
## 3              1.0             2000                     -17.7
## 4              0.9             2000                     -17.6
## 5              2.3             2000                     -18.6
## 6              1.5             2000                     -18.7
##   Solar.Radiation..MJ.m2. Rainfall.mm. Snowfall..cm. Seasons    Holiday
## 1                       0           0             0  Winter No Holiday
## 2                       0           0             0  Winter No Holiday
## 3                       0           0             0  Winter No Holiday
## 4                       0           0             0  Winter No Holiday
## 5                       0           0             0  Winter No Holiday
## 6                       0           0             0  Winter No Holiday
##   Functioning.Day
## 1             Yes
## 2             Yes
```

```
## 3             Yes
## 4             Yes
## 5             Yes
## 6             Yes
```

```
##Manually assign new column names for better accessibility of variables
new_column_names = c("Date", "RBC", "Hour", "Temp", "Humid_Percent",
"Wind_Speed", "Visibility", "DPT", "Solar_Rad", "Rainfall", "Snowfall",
"Seasons", "Holiday", "Functioning_Day")
colnames(seoul_bike_data) = new_column_names
colnames(seoul_bike_data)
```

```
##  [1] "Date"            "RBC"             "Hour"            "Temp"
##  [5] "Humid_Percent"   "Wind_Speed"      "Visibility"      "DPT"
##  [9] "Solar_Rad"       "Rainfall"        "Snowfall"        "Seasons"
## [13] "Holiday"         "Functioning_Day"
```

```
##Exploring data types.
str(seoul_bike_data)
```

```
## 'data.frame':    8760 obs. of  14 variables:
##  $ Date           : chr  "01/12/2017" "01/12/2017" "01/12/2017"
"01/12/2017" ...
##  $ RBC            : int  254 204 173 107 78 100 181 460 930 490 ...
##  $ Hour           : int  0 1 2 3 4 5 6 7 8 9 ...
##  $ Temp           : num  -5.2 -5.5 -6 -6.2 -6 -6.4 -6.6 -7.4 -7.6 -6.5 ...
##  $ Humid_Percent  : int  37 38 39 40 36 37 35 38 37 27 ...
##  $ Wind_Speed     : num  2.2 0.8 1 0.9 2.3 1.5 1.3 0.9 1.1 0.5 ...
##  $ Visibility     : int  2000 2000 2000 2000 2000 2000 2000 2000 2000 1928
...
##  $ DPT            : num  -17.6 -17.6 -17.7 -17.6 -18.6 -18.7 -19.5 -19.3 -
19.8 -22.4 ...
##  $ Solar_Rad      : num  0 0 0 0 0 0 0 0 0.01 0.23 ...
##  $ Rainfall       : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ Snowfall       : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ Seasons        : chr  "Winter" "Winter" "Winter" "Winter" ...
##  $ Holiday        : chr  "No Holiday" "No Holiday" "No Holiday" "No
Holiday" ...
##  $ Functioning_Day: chr  "Yes" "Yes" "Yes" "Yes" ...
```

```
##Exploring Null Values.
colSums(is.na(seoul_bike_data))
```

```
##            Date             RBC            Hour            Temp
Humid_Percent
##               0               0               0               0
0
##      Wind_Speed      Visibility             DPT       Solar_Rad
Rainfall
##               0               0               0               0
0
```

```
##        Snowfall         Seasons         Holiday Functioning_Day
##               0               0               0               0
```

```
##Exploring Basic statistics.
summary(seoul_bike_data)
```

```
##      Date                 RBC              Hour            Temp
##  Length:8760        Min.   :   0.0   Min.   : 0.00   Min.   :-17.80
##  Class :character   1st Qu.: 191.0   1st Qu.: 5.75   1st Qu.:  3.50
##  Mode  :character   Median : 504.5   Median :11.50   Median : 13.70
##                     Mean   : 704.6   Mean   :11.50   Mean   : 12.88
##                     3rd Qu.:1065.2   3rd Qu.:17.25   3rd Qu.: 22.50
##                     Max.   :3556.0   Max.   :23.00   Max.   : 39.40
##  Humid_Percent     Wind_Speed      Visibility        DPT
##  Min.   : 0.00   Min.   :0.000   Min.   :  27   Min.   :-30.600
##  1st Qu.:42.00   1st Qu.:0.900   1st Qu.: 940   1st Qu.: -4.700
##  Median :57.00   Median :1.500   Median :1698   Median :  5.100
##  Mean   :58.23   Mean   :1.725   Mean   :1437   Mean   :  4.074
##  3rd Qu.:74.00   3rd Qu.:2.300   3rd Qu.:2000   3rd Qu.: 14.800
##  Max.   :98.00   Max.   :7.400   Max.   :2000   Max.   : 27.200
##    Solar_Rad         Rainfall          Snowfall         Seasons
##  Min.   :0.0000   Min.   : 0.0000   Min.   :0.00000   Length:8760
##  1st Qu.:0.0000   1st Qu.: 0.0000   1st Qu.:0.00000   Class :character
##  Median :0.0100   Median : 0.0000   Median :0.00000   Mode  :character
##  Mean   :0.5691   Mean   : 0.1487   Mean   :0.07507
##  3rd Qu.:0.9300   3rd Qu.: 0.0000   3rd Qu.:0.00000
##  Max.   :3.5200   Max.   :35.0000   Max.   :8.80000
##    Holiday          Functioning_Day
##  Length:8760        Length:8760
##  Class :character   Class :character
##  Mode  :character   Mode  :character
##
##
##
```

```
##Extraction of features like Day, Month, Year, and Weekday from Date.
seoul_bike_data$Date = as.Date(seoul_bike_data$Date, format = "%d/%m/%Y")
seoul_bike_data$Day = day(seoul_bike_data$Date)
seoul_bike_data$Month = month(seoul_bike_data$Date)
seoul_bike_data$Year = year(seoul_bike_data$Date)
seoul_bike_data$Weekday = weekdays(seoul_bike_data$Date)
head(seoul_bike_data)
```

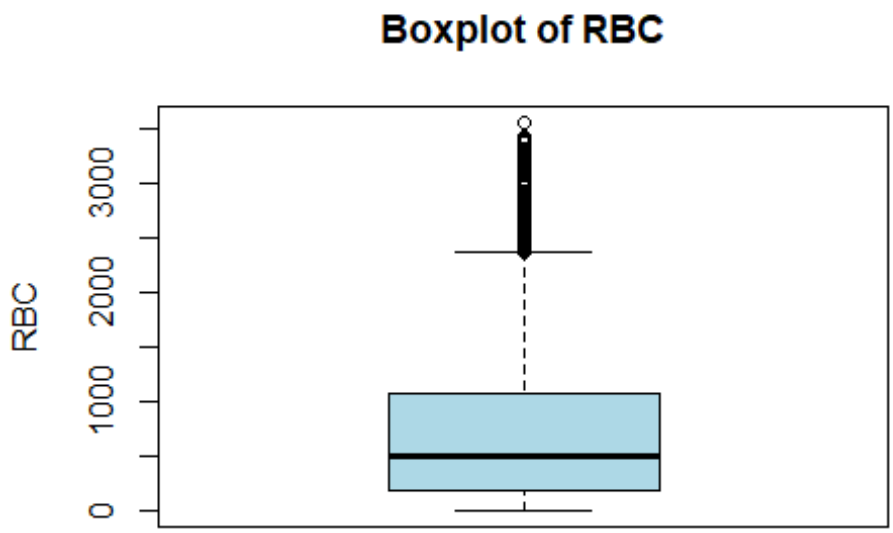
```
##         Date RBC Hour Temp Humid_Percent Wind_Speed Visibility   DPT
Solar_Rad
## 1 2017-12-01 254    0 -5.2            37        2.2       2000 -17.6
0
## 2 2017-12-01 204    1 -5.5            38        0.8       2000 -17.6
0
## 3 2017-12-01 173    2 -6.0            39        1.0       2000 -17.7
0
```

```
## 4 2017-12-01 107    3 -6.2            40        0.9       2000 -17.6
0
## 5 2017-12-01  78    4 -6.0            36        2.3       2000 -18.6
0
## 6 2017-12-01 100    5 -6.4            37        1.5       2000 -18.7
0
##   Rainfall Snowfall Seasons    Holiday Functioning_Day Day Month Year
Weekday
## 1        0        0  Winter No Holiday             Yes   1    12 2017
Friday
## 2        0        0  Winter No Holiday             Yes   1    12 2017
Friday
## 3        0        0  Winter No Holiday             Yes   1    12 2017
Friday
## 4        0        0  Winter No Holiday             Yes   1    12 2017
Friday
## 5        0        0  Winter No Holiday             Yes   1    12 2017
Friday
## 6        0        0  Winter No Holiday             Yes   1    12 2017
Friday
```

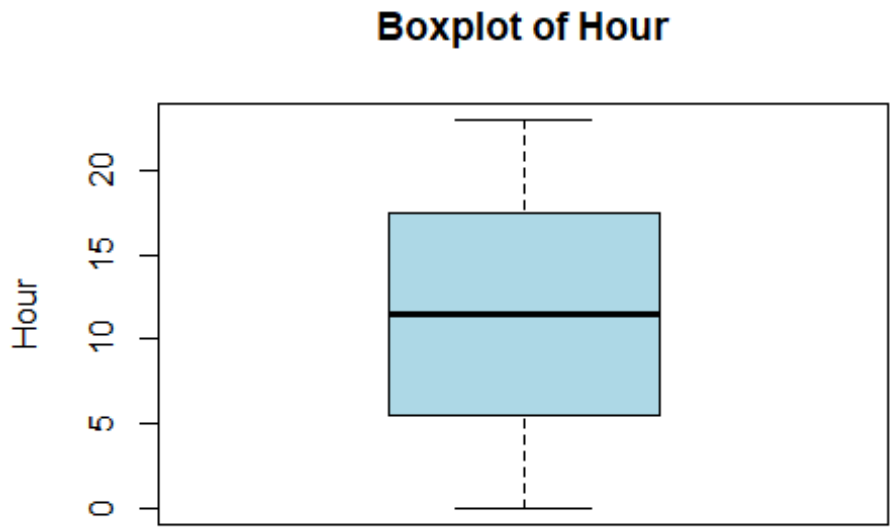
```
##b) Data Cleaning
##Since there are no missing values in the dataset we can move on the
analysis of the ouliers in the datset.
##Checking for outliers in numerical columns.
num_cols = seoul_bike_data %>% select_if(is.numeric)
colnames(num_cols)
```

```
##  [1] "RBC"           "Hour"          "Temp"          "Humid_Percent"
##  [5] "Wind_Speed"    "Visibility"    "DPT"           "Solar_Rad"
##  [9] "Rainfall"      "Snowfall"      "Day"           "Month"
## [13] "Year"
```

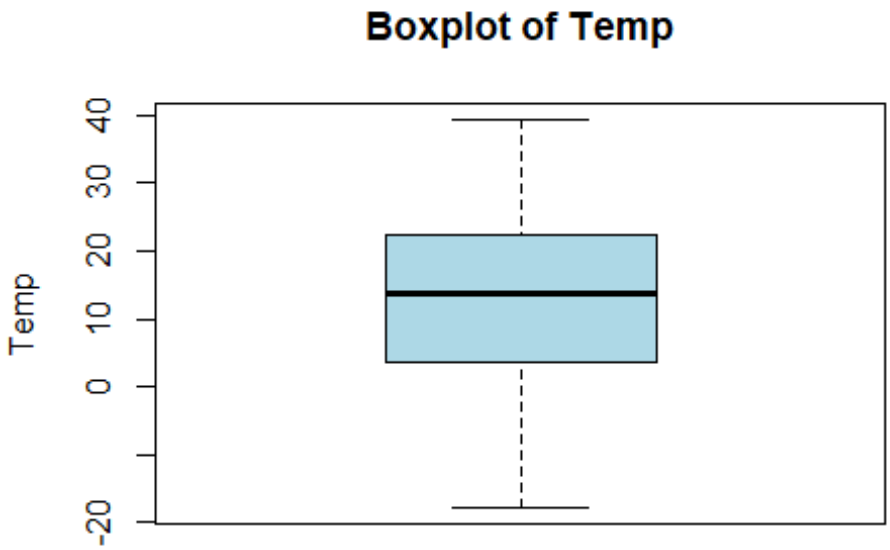
```
##Visualising outliers
for (col_name in colnames(num_cols)) {
  boxplot(num_cols[[col_name]],
          main = paste("Boxplot of", col_name),
          col = "lightblue",
          ylab = col_name)
  readline(prompt = "Press[Enter] to see the next plot...")
}
```



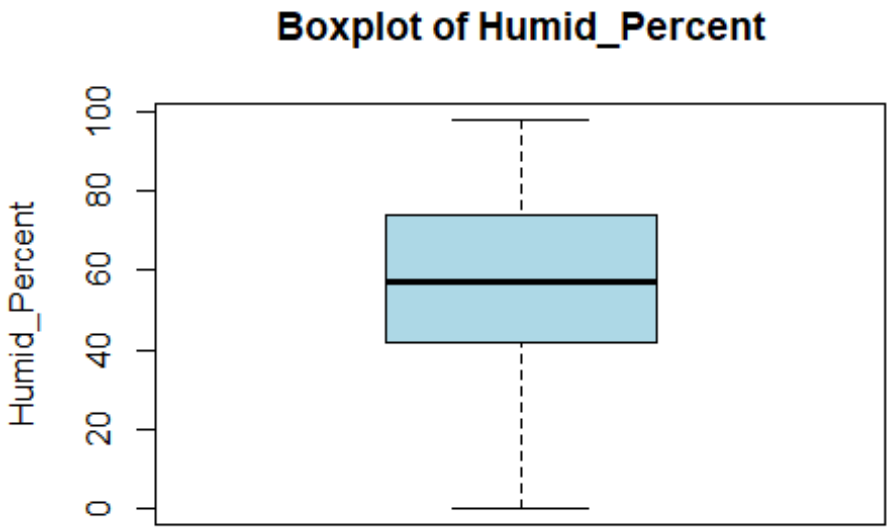

```
## Press[Enter] to see the next plot...
```



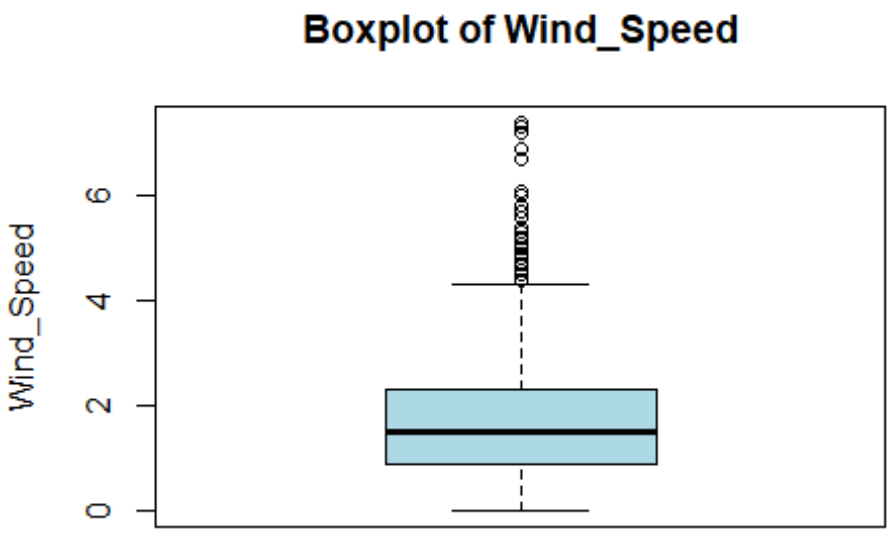

```
## Press[Enter] to see the next plot...
```



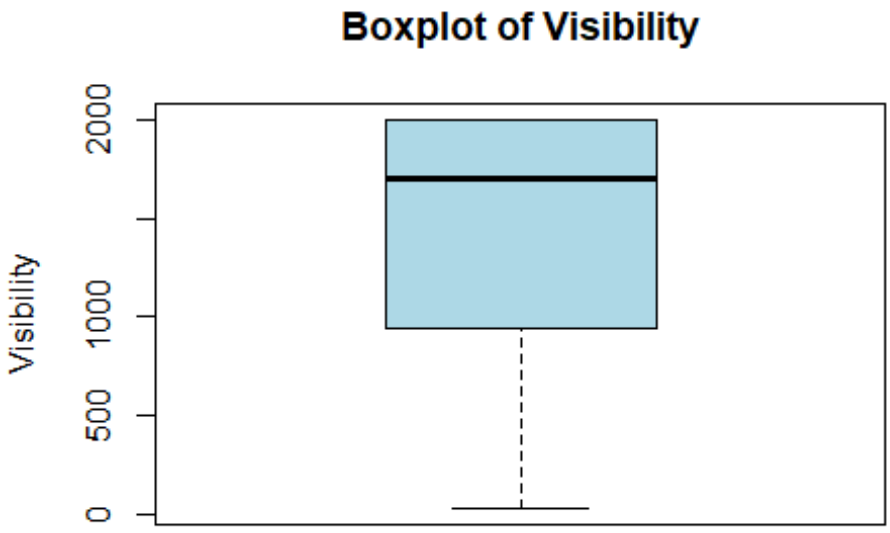

```
## Press[Enter] to see the next plot...
```



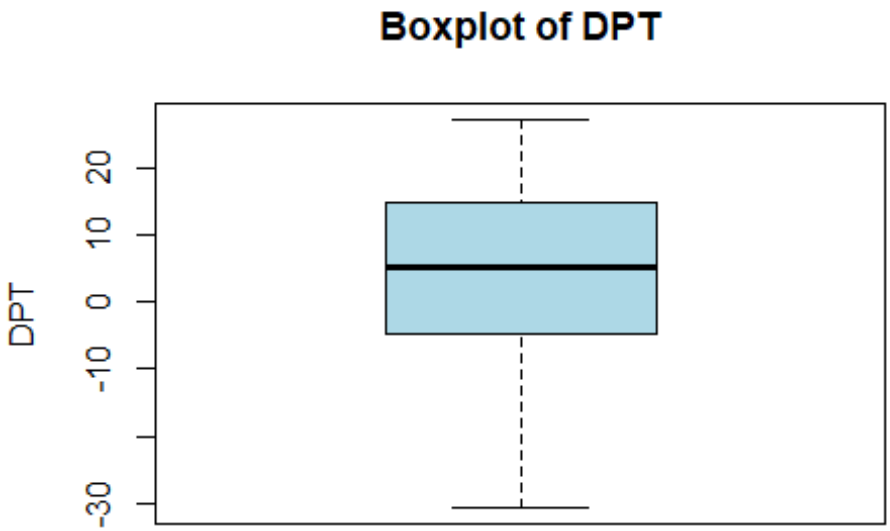

```
## Press[Enter] to see the next plot...
```



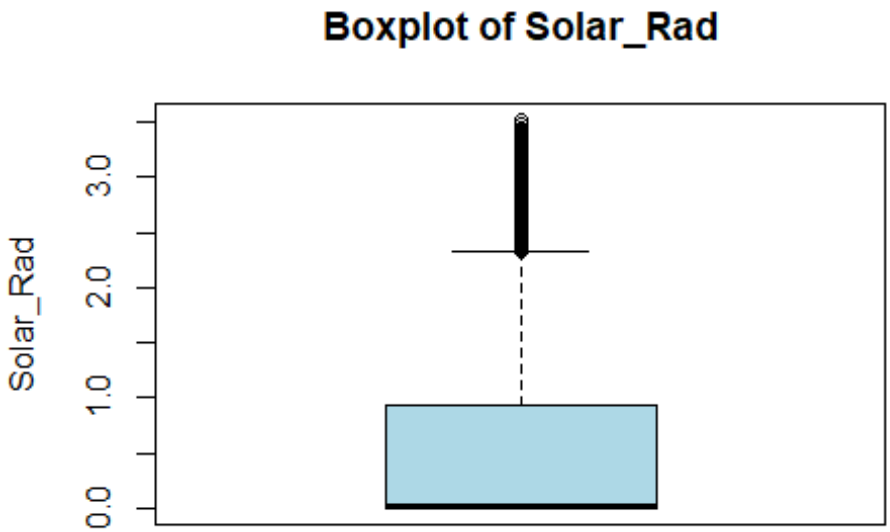

```
## Press[Enter] to see the next plot...
```



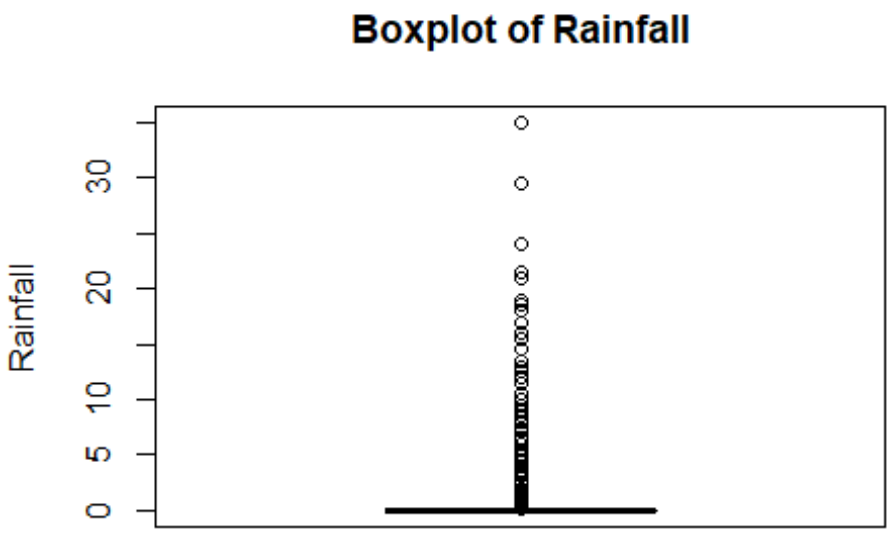

```
## Press[Enter] to see the next plot...
```



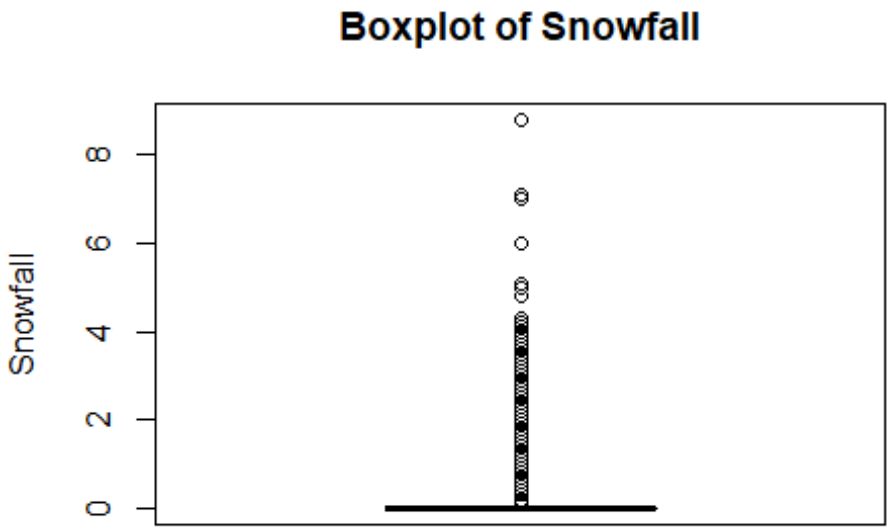

```
## Press[Enter] to see the next plot...
```



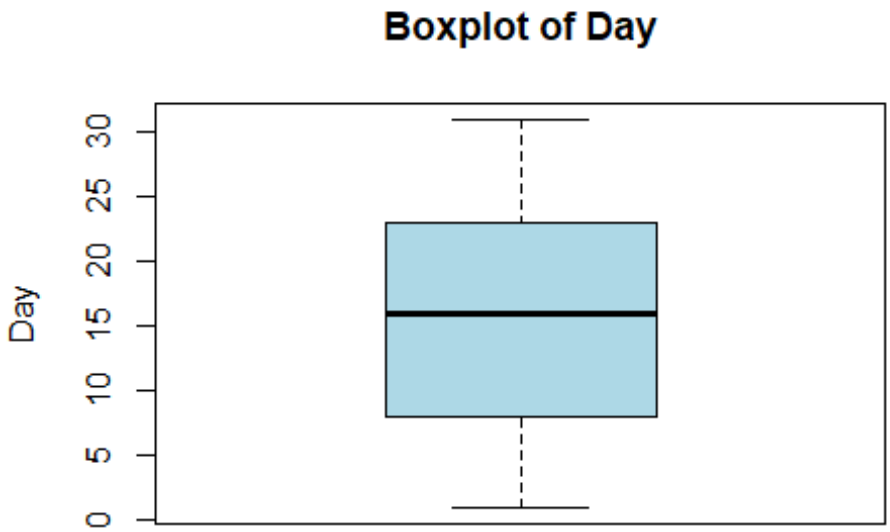

```
## Press[Enter] to see the next plot...
```



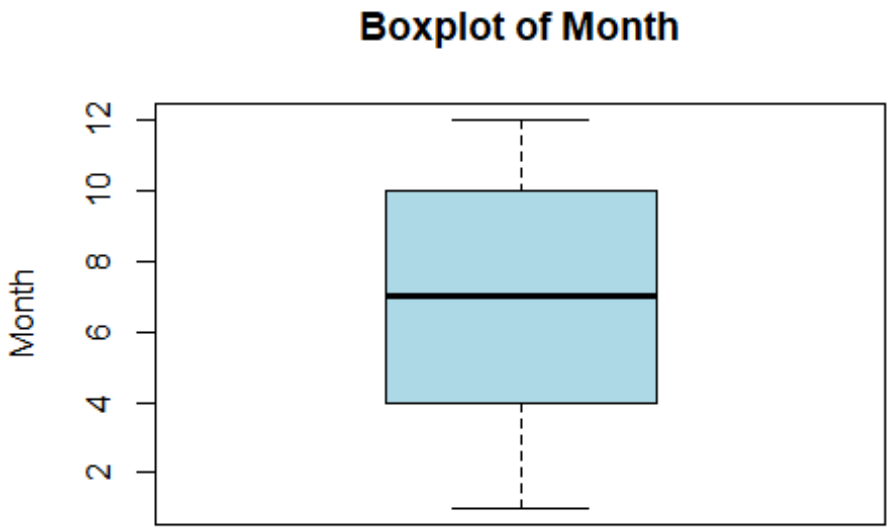

```
## Press[Enter] to see the next plot...
```



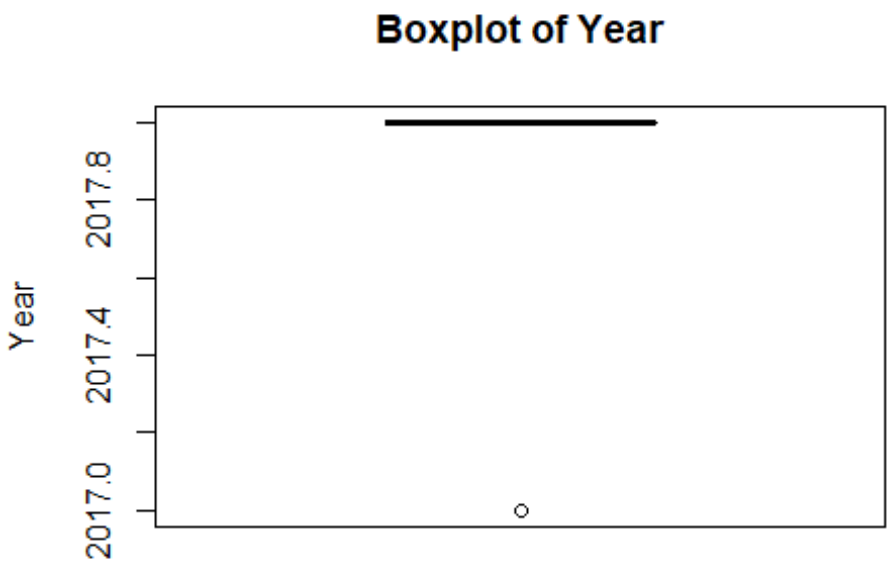

```
## Press[Enter] to see the next plot...
```



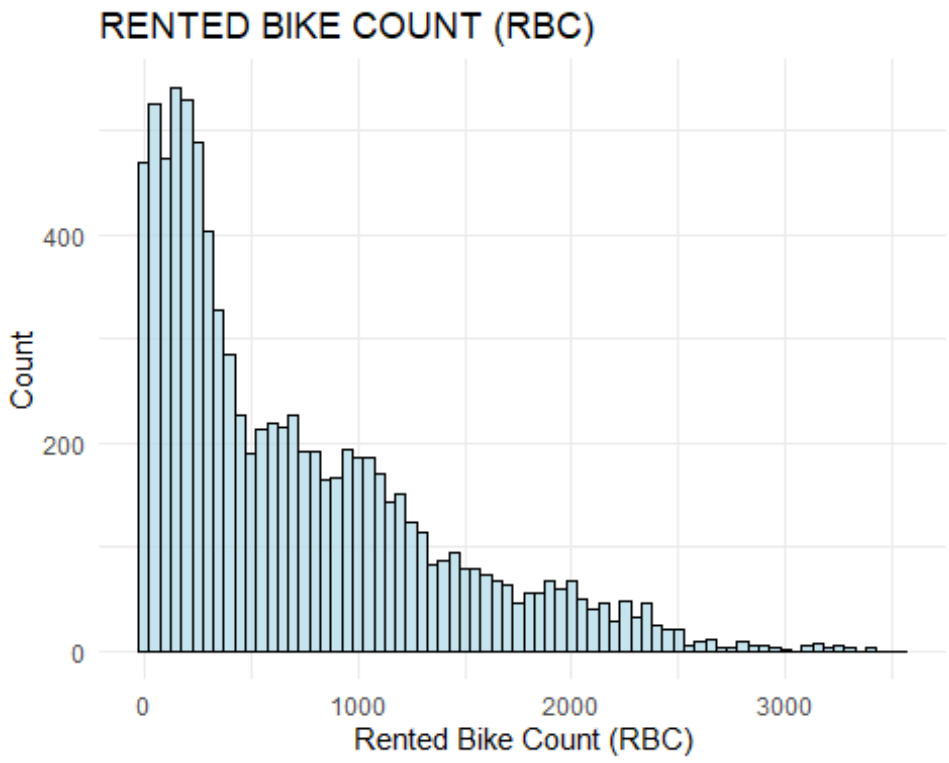

```
## Press[Enter] to see the next plot...
```




```
## Press[Enter] to see the next plot...
```




```
## Press[Enter] to see the next plot...

##2. Exploratory Data Analysis (EDA)
##a) Univariate Analysis
rbc_histogram = ggplot(seoul_bike_data, aes(x = RBC)) +
  geom_histogram(binwidth = 50, fill = "lightblue", color = "black", alpha =
0.7) +  # Adjust binwidth as needed
  labs(
    title = "RENTED BIKE COUNT (RBC)",
    x = "Rented Bike Count (RBC)",
    y = "Count"
  ) +
  theme_minimal() +
  theme(
    plot.title = element_text(hjust = 0.5),  # Center the title
    axis.title.x = element_text(size = 12),
    axis.title.y = element_text(size = 12)
  ) + theme_minimal()

print(rbc_histogram)
```




```r
skewness(seoul_bike_data$RBC)
```

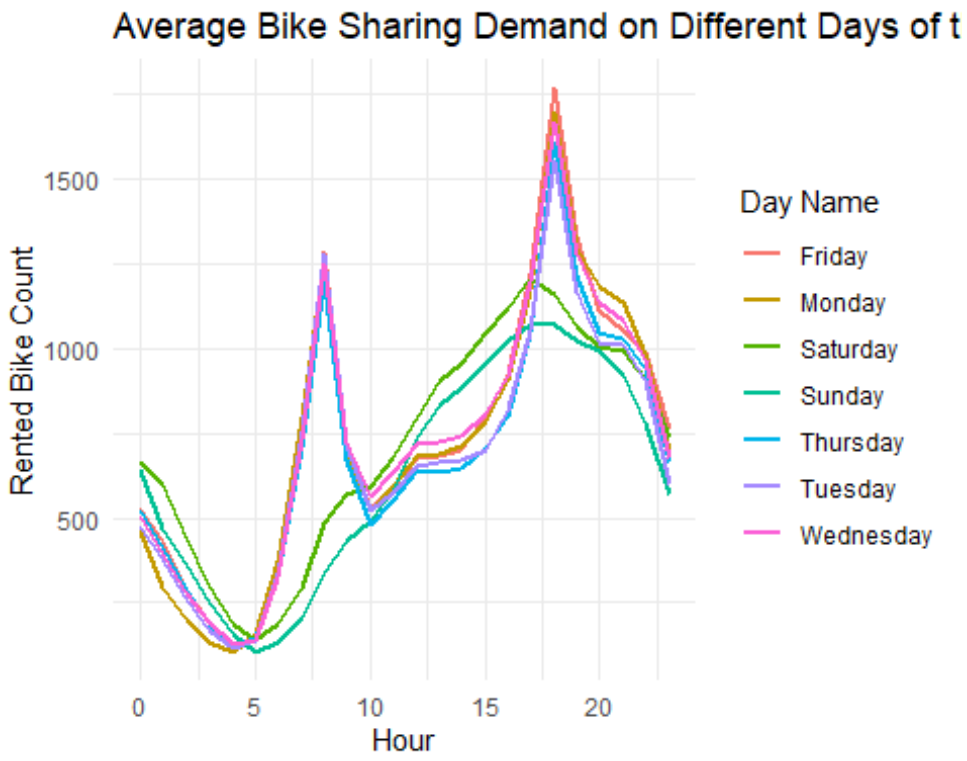
```
## [1] 1.153033
```

```r
##The skewness value of 1.153 indicates that the distribution of the RBC
(Rented Bike Count) variable is significantly positively skewed.
##We might encounter model assumption violation due to this, but lets tackle
that issue in the later stages if it gives rises to model assumption
violation.



##b)Bivariate Analysis
#Average Rented Bike Counts (RBC) for each weekday
plot_data = seoul_bike_data %>%
  group_by(Weekday, Hour) %>%
  summarise(Mean_RBC = mean(RBC, na.rm = TRUE))
```

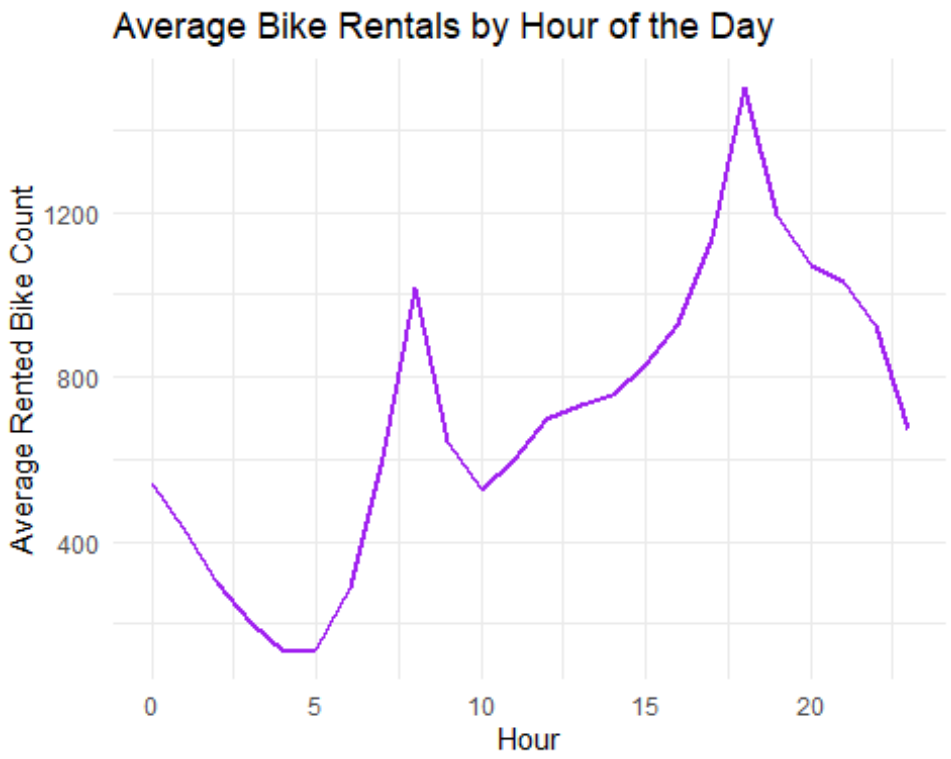
```
## `summarise()` has grouped output by 'Weekday'. You can override using the
## `.groups` argument.
```

```r
ggplot(plot_data, aes(x = Hour, y = Mean_RBC, color = Weekday)) +
  geom_line(size = 1) + # Add lines for each Weekday
  labs(
    title = "Average Bike Sharing Demand on Different Days of the Week",
    x = "Hour",
    y = "Rented Bike Count",
    color = "Day Name"
```

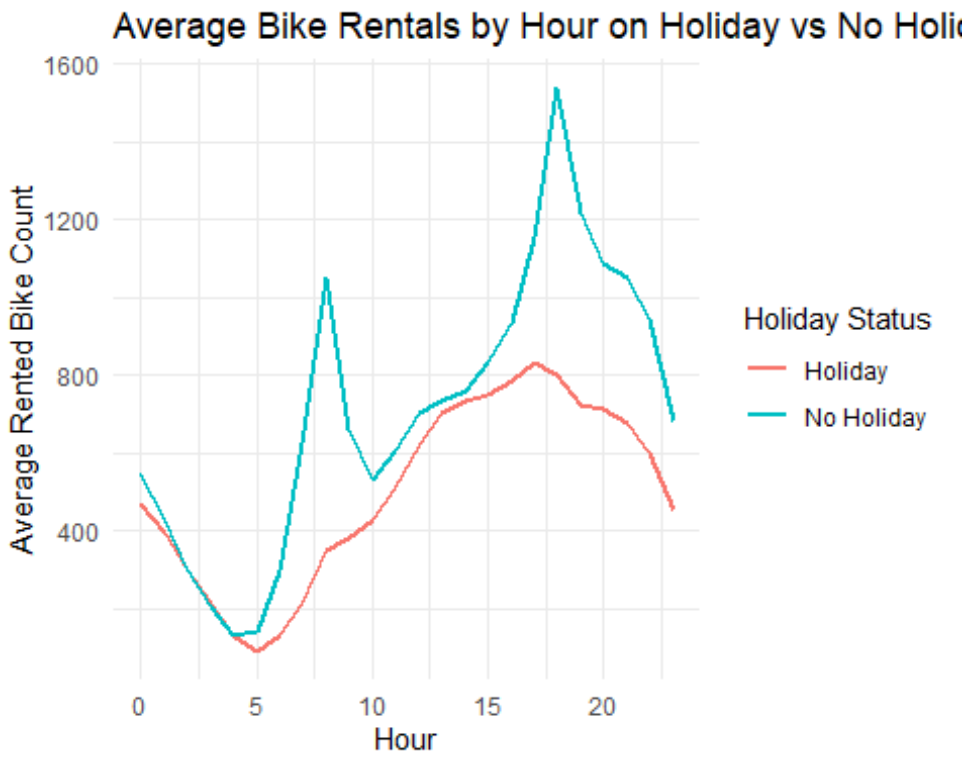
```
  ) +
  theme_minimal()
```



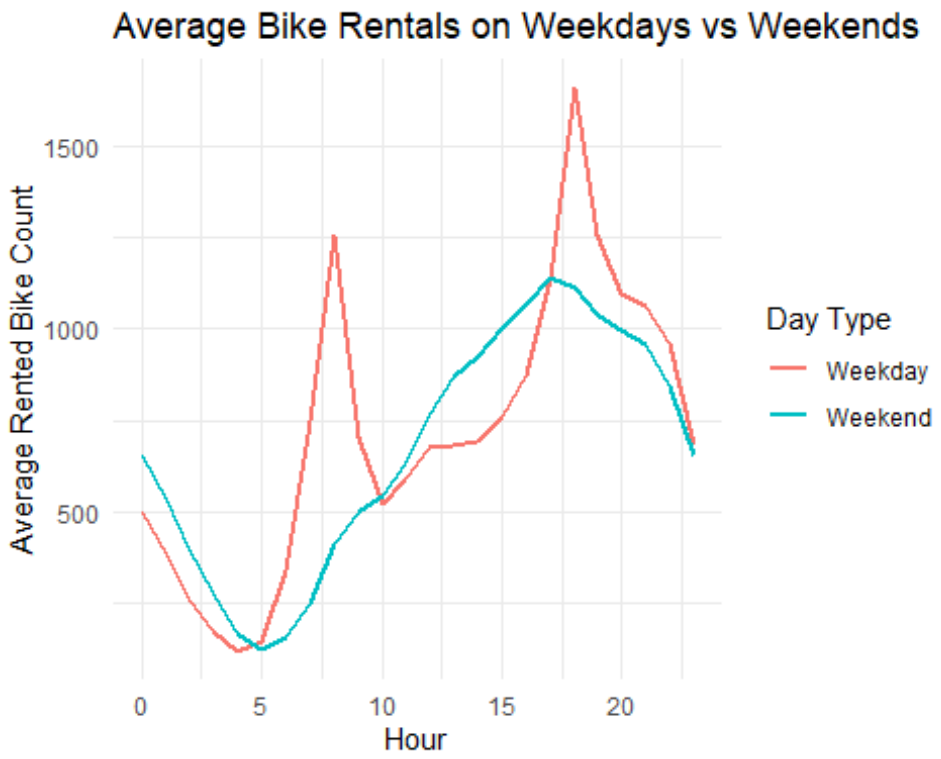

```
# RBC vs Hour (Line plot)
rbc_hour_plot = seoul_bike_data %>%
  group_by(Hour) %>%
  summarise(Mean_RBC = mean(RBC, na.rm = TRUE))

ggplot(rbc_hour_plot, aes(x = Hour, y = Mean_RBC)) +
  geom_line(color = "purple", size = 1) +  # Line for RBC against Hour
  labs(
    title = "Average Bike Rentals by Hour of the Day",
    x = "Hour",
    y = "Average Rented Bike Count"
  ) +
  theme_minimal()
```



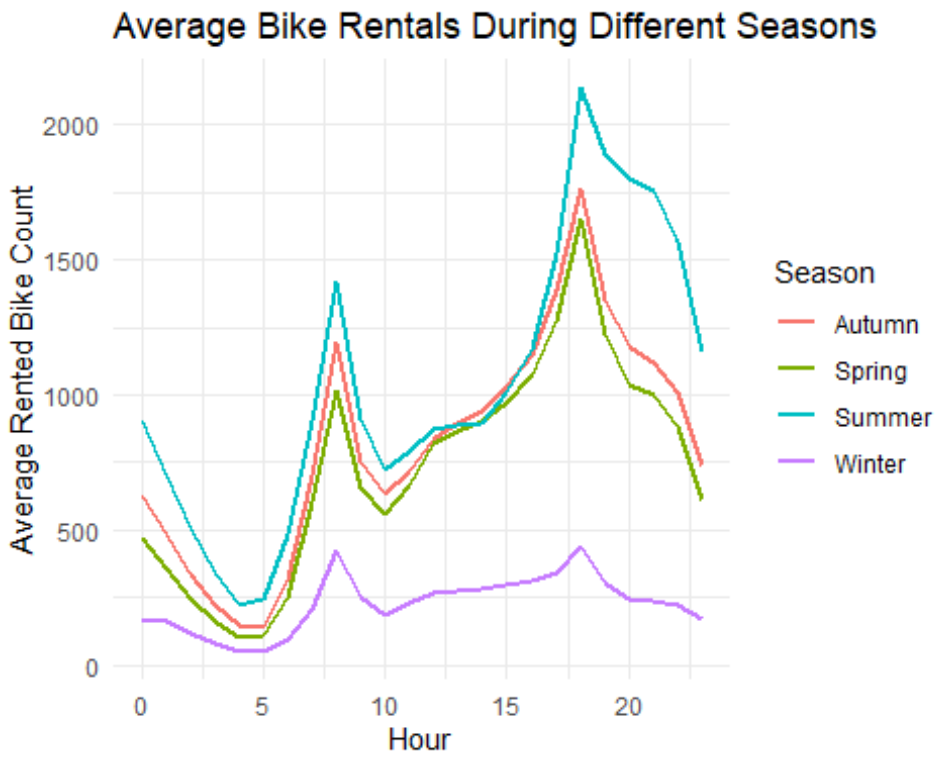

```r
#RBC vs Holiday/No Holiday (Line plot)
rbc_holiday_plot = seoul_bike_data %>%
  group_by(Holiday, Hour) %>%
  summarise(Mean_RBC = mean(RBC, na.rm = TRUE))

## `summarise()` has grouped output by 'Holiday'. You can override using the
## `.groups` argument.

ggplot(rbc_holiday_plot, aes(x = Hour, y = Mean_RBC, color = Holiday)) +
  geom_line(size = 1) +  # Line for RBC against Hour, grouped by Holiday
  labs(
    title = "Average Bike Rentals by Hour on Holiday vs No Holiday",
    x = "Hour",
    y = "Average Rented Bike Count",
    color = "Holiday Status"
  ) +
  theme_minimal()
```



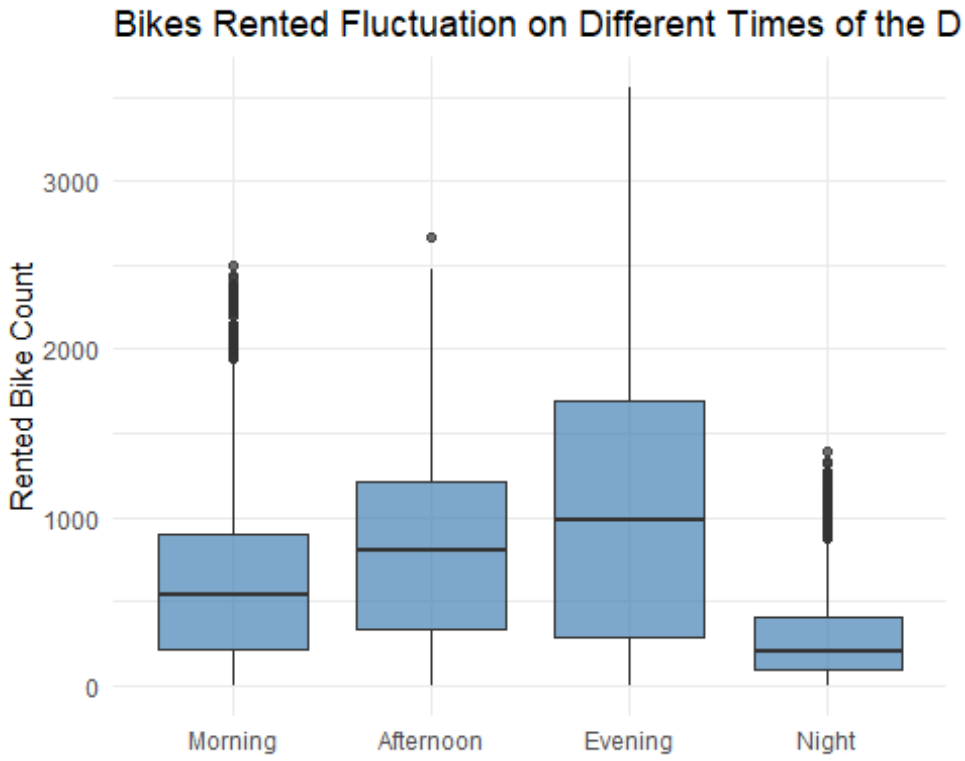

```
#Bike Demand on Weekdays vs Weekend
seoul_bike_data$Day_Type = ifelse(seoul_bike_data$Weekday %in% c("Saturday",
"Sunday"), "Weekend", "Weekday")

rbc_weekend_weekday_plot = seoul_bike_data %>%
  group_by(Day_Type, Hour) %>%
  summarise(Mean_RBC = mean(RBC, na.rm = TRUE))

## `summarise()` has grouped output by 'Day_Type'. You can override using the
## `.groups` argument.

ggplot(rbc_weekend_weekday_plot, aes(x = Hour, y = Mean_RBC, color =
Day_Type)) +
  geom_line(size = 1) +  # Line for RBC against Hour, grouped by Day Type
  labs(
    title = "Average Bike Rentals on Weekdays vs Weekends",
    x = "Hour",
    y = "Average Rented Bike Count",
    color = "Day Type"
  ) +
  theme_minimal()
```




```r
#Bike Demand During Different Seasons
rbc_season_plot = seoul_bike_data %>%
  group_by(Seasons, Hour) %>%
  summarise(Mean_RBC = mean(RBC, na.rm = TRUE))

## `summarise()` has grouped output by 'Seasons'. You can override using the
## `.groups` argument.

# Plot: RBC vs Hour (by Season)
ggplot(rbc_season_plot, aes(x = Hour, y = Mean_RBC, color = Seasons)) +
  geom_line(size = 1) +  # Line for RBC against Hour, grouped by Seasons
  labs(
    title = "Average Bike Rentals During Different Seasons",
    x = "Hour",
    y = "Average Rented Bike Count",
    color = "Season"
  ) +
  theme_minimal()
```

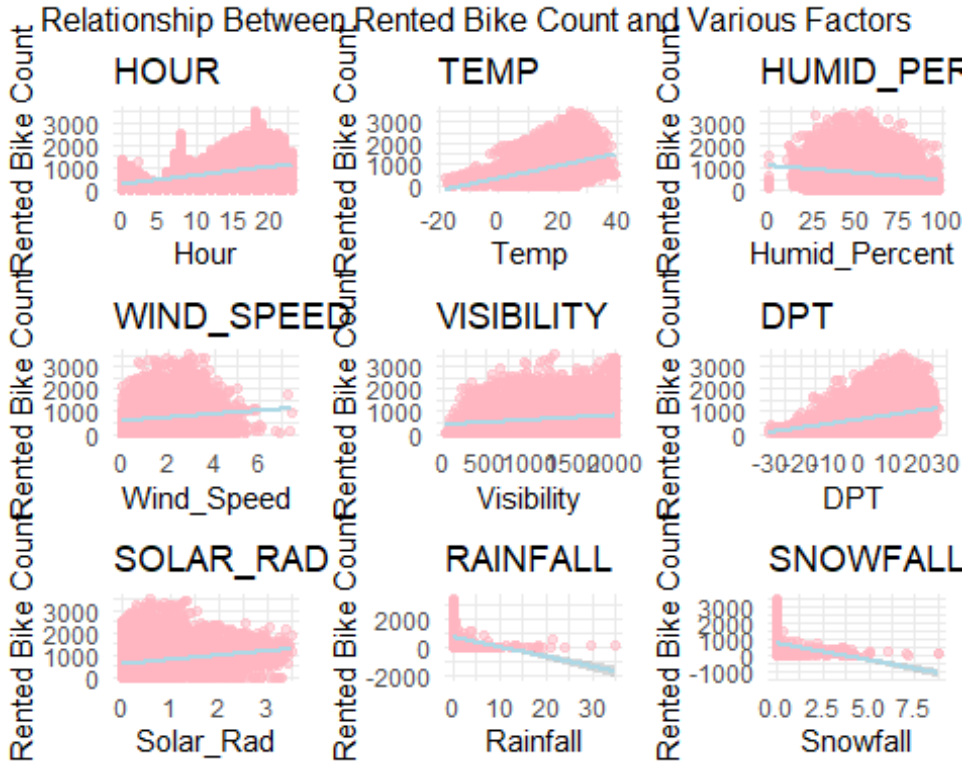



```r
#Bike Demand during Different Times  of the day
seoul_bike_data <- seoul_bike_data %>%
  mutate(Time_of_Day = case_when(
    Hour >= 0 & Hour < 6 ~ "Night",
    Hour >= 6 & Hour < 12 ~ "Morning",
    Hour >= 12 & Hour < 18 ~ "Afternoon",
    Hour >= 18 & Hour <= 23 ~ "Evening"
  )) %>%
  mutate(Time_of_Day = factor(Time_of_Day, levels = c("Morning", "Afternoon",
"Evening", "Night")))

# Summarize and plot data
ggplot(seoul_bike_data, aes(x = Time_of_Day, y = RBC)) +
  geom_boxplot(fill = "steelblue", alpha = 0.7) +
  labs(
    title = "Bikes Rented Fluctuation on Different Times of the Day",
    x = NULL,
    y = "Rented Bike Count"
  ) +
  theme_minimal()
```

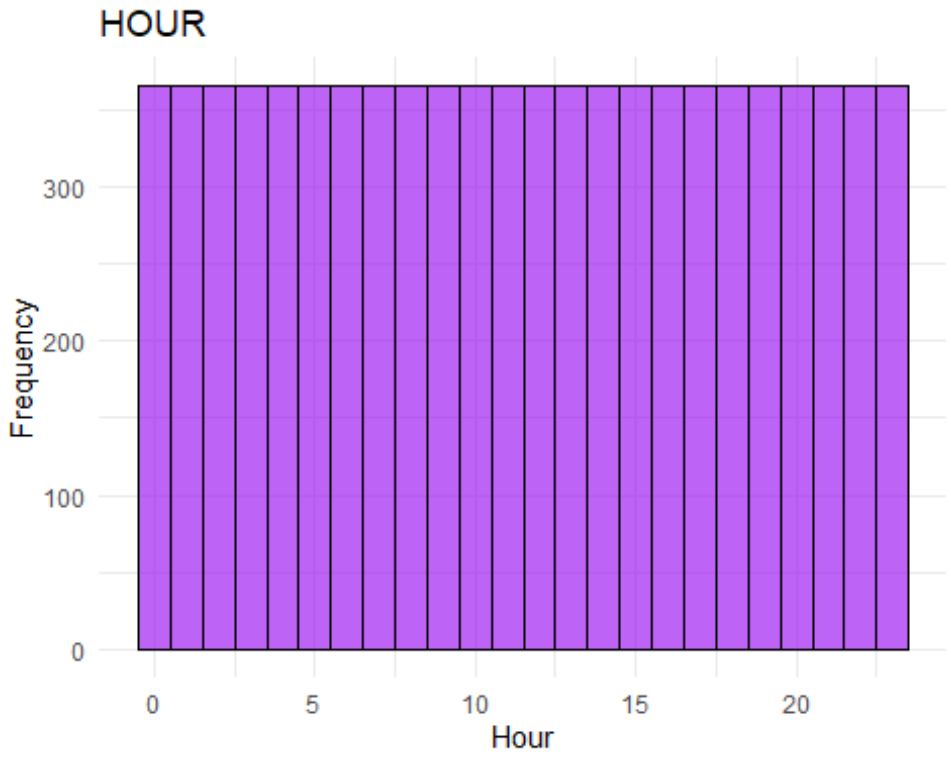


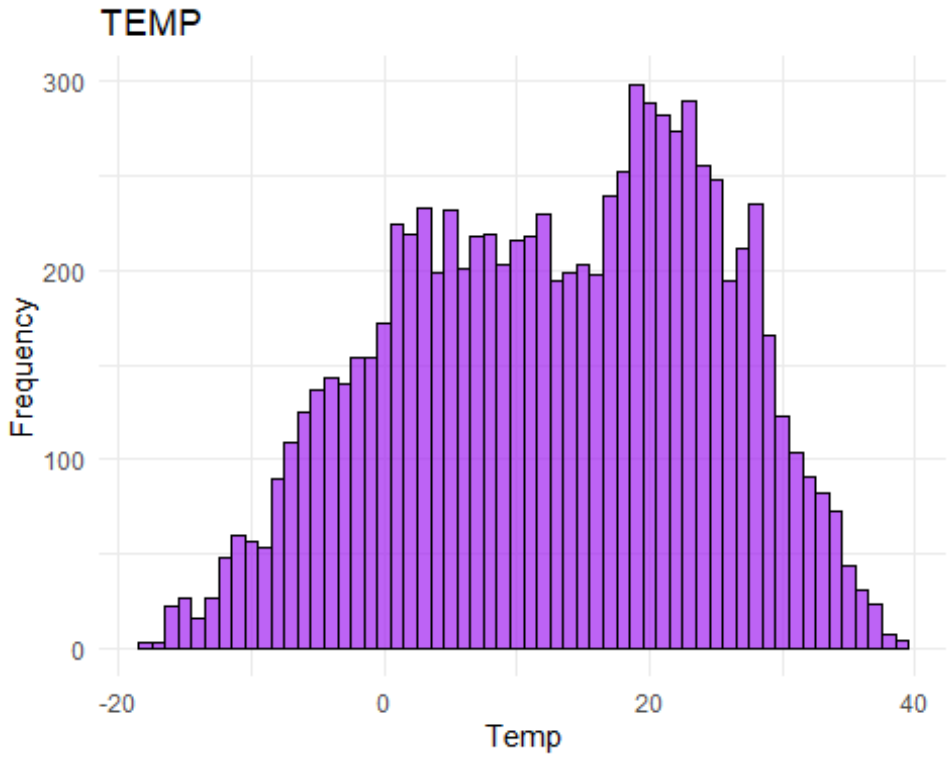

```
#Distribution plot of each numerical columns (excluding dummy variables)
Scatter Plot
variables_to_plot = c("Hour", "Temp", "Humid_Percent", "Wind_Speed",
                      "Visibility", "DPT", "Solar_Rad", "Rainfall",
"Snowfall")

scatter_plots = lapply(variables_to_plot, function(var) {
  ggplot(seoul_bike_data, aes_string(x = var, y = "RBC")) +
    geom_point(color = "lightpink", alpha = 0.5) +
    geom_smooth(method = "lm", color = "lightblue", se = TRUE) +
    labs(
      title = toupper(var),
      x = var,
      y = "Rented Bike Count"
    ) +
    theme_minimal()
})

grid.arrange(
  grobs = scatter_plots,
  ncol = 3,
  top = "Relationship Between Rented Bike Count and Various Factors"
)

## `geom_smooth()` using formula = 'y ~ x'
```

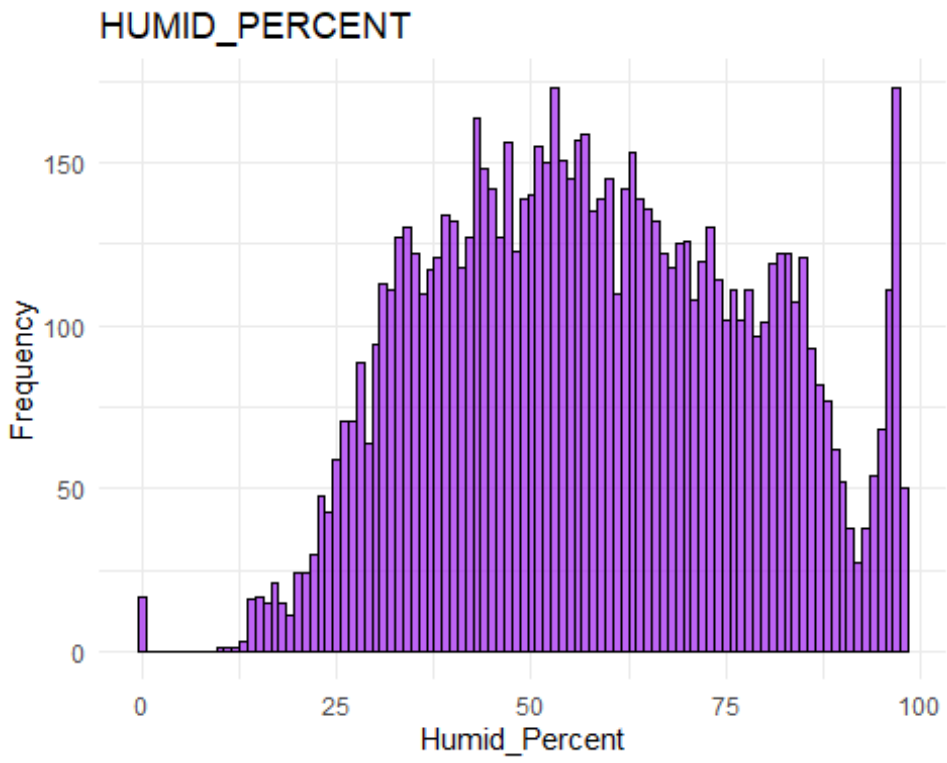
```
## `geom_smooth()` using formula = 'y ~ x'
## `geom_smooth()` using formula = 'y ~ x'
## `geom_smooth()` using formula = 'y ~ x'
## `geom_smooth()` using formula = 'y ~ x'
## `geom_smooth()` using formula = 'y ~ x'
## `geom_smooth()` using formula = 'y ~ x'
## `geom_smooth()` using formula = 'y ~ x'
## `geom_smooth()` using formula = 'y ~ x'
```

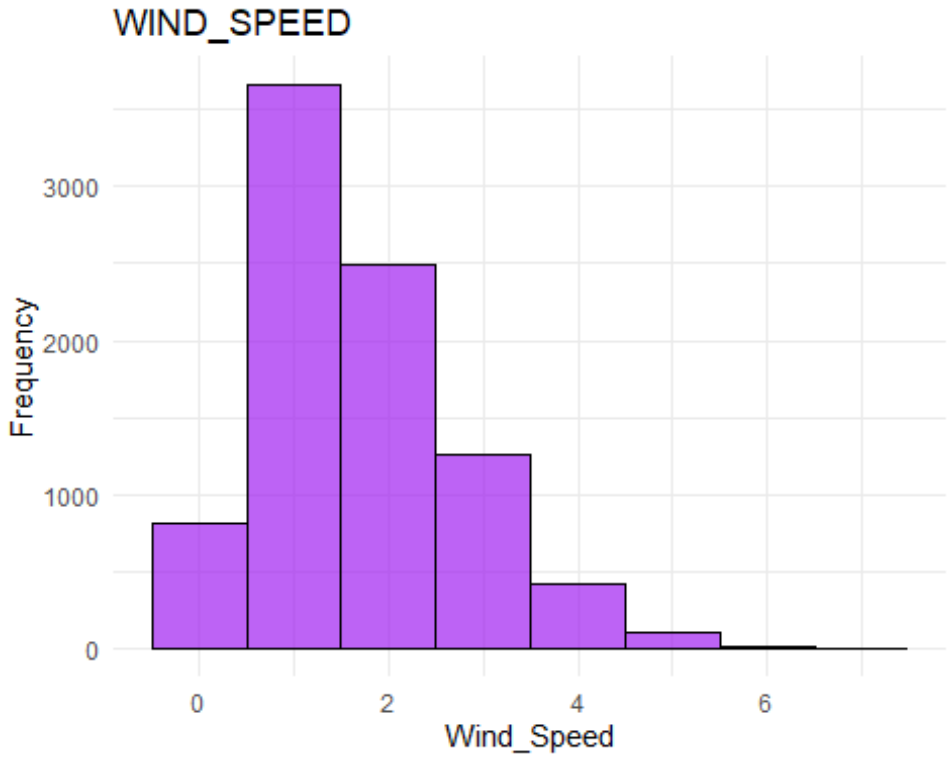


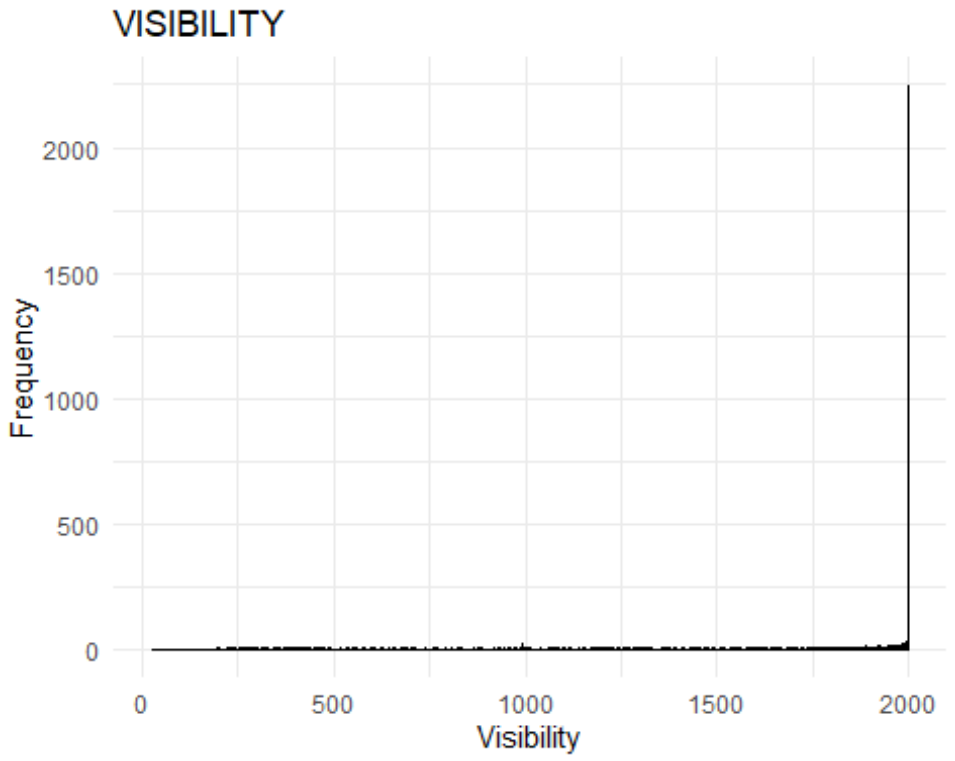

```
#Distribution plot of each numerical columns (excluding dummy variables)
Histograms
histograms = lapply(variables_to_plot, function(var) {
  ggplot(seoul_bike_data, aes_string(x = var)) +
    geom_histogram(binwidth = 1, fill = "purple", color = "black", alpha =
0.7) +
    labs(
      title = toupper(var),
      x = var,
      y = "Frequency"
    ) +
    theme_minimal()
})

for (plot in histograms) {
  print(plot)
}
```

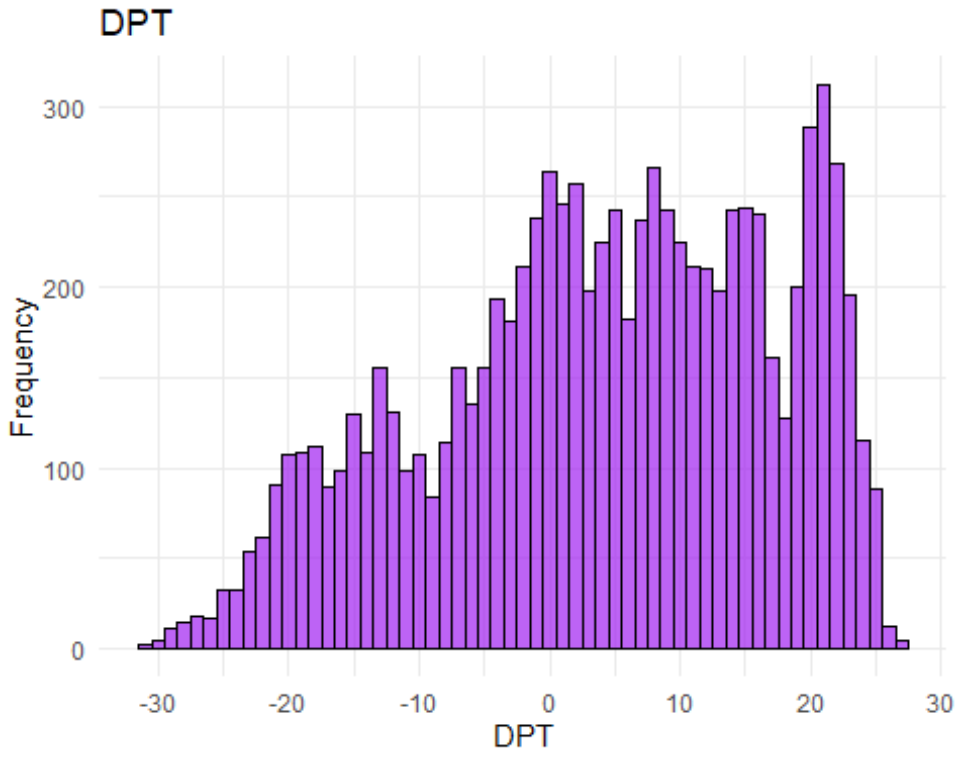

HOUR
Frequency
300
200
100
0
0
5
10
15
20
Hour

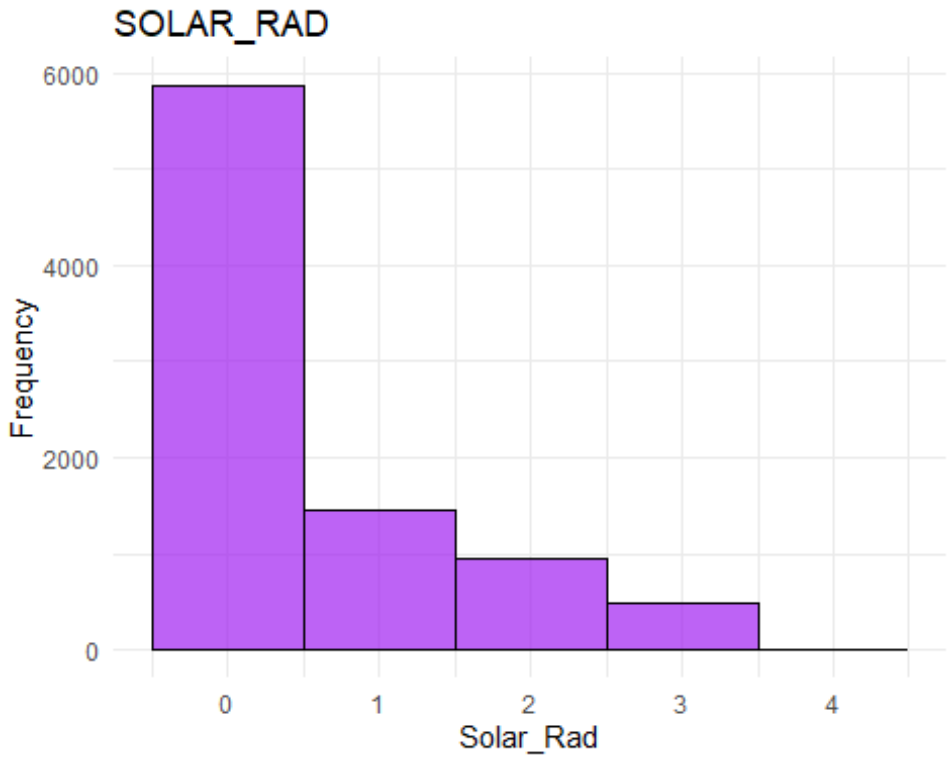

TEMP
Frequency
300
200
100
0
-20
0
20
40
Temp

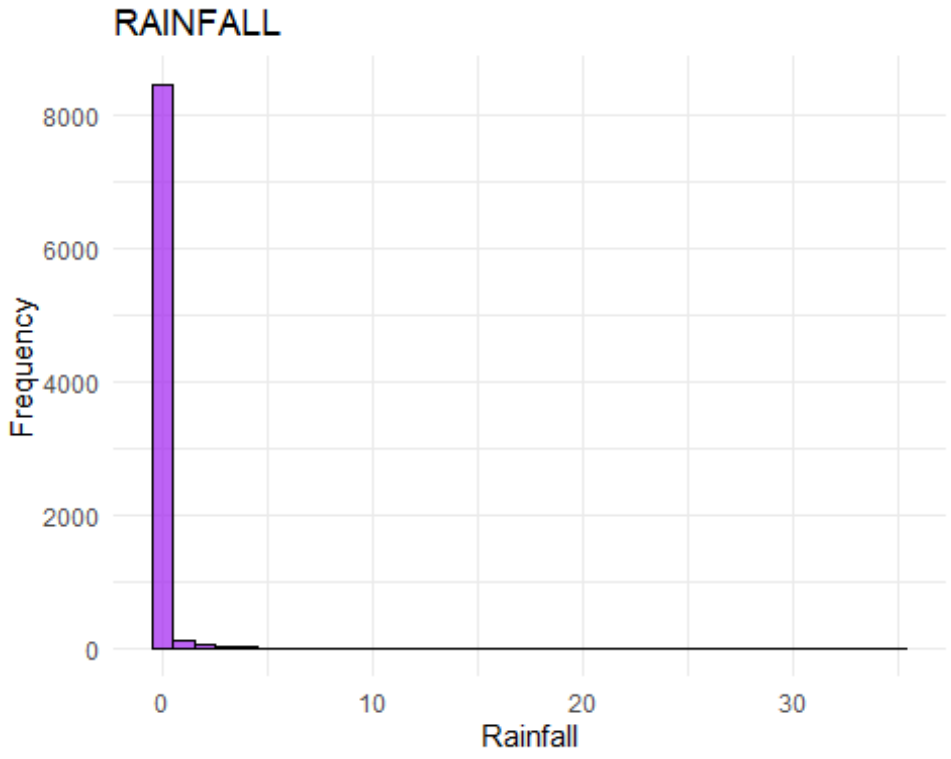

HUMID_PERCENT
Frequency
150
100
50
0
0
25
50
75
100
Humid_Percent

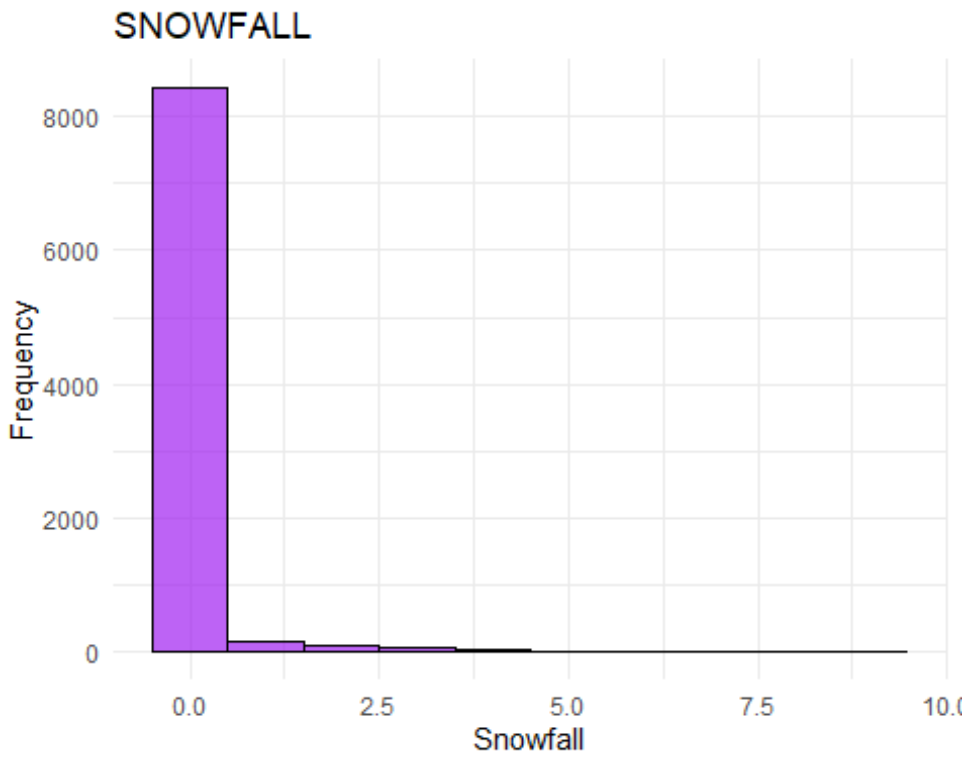

WIND_SPEED
Frequency
3000
2000
1000
0
0
2
4
6
Wind_Speed


VISIBILITY
2000
1500
1000
500
0
Frequency
0
500
1000
1500
2000
Visibility

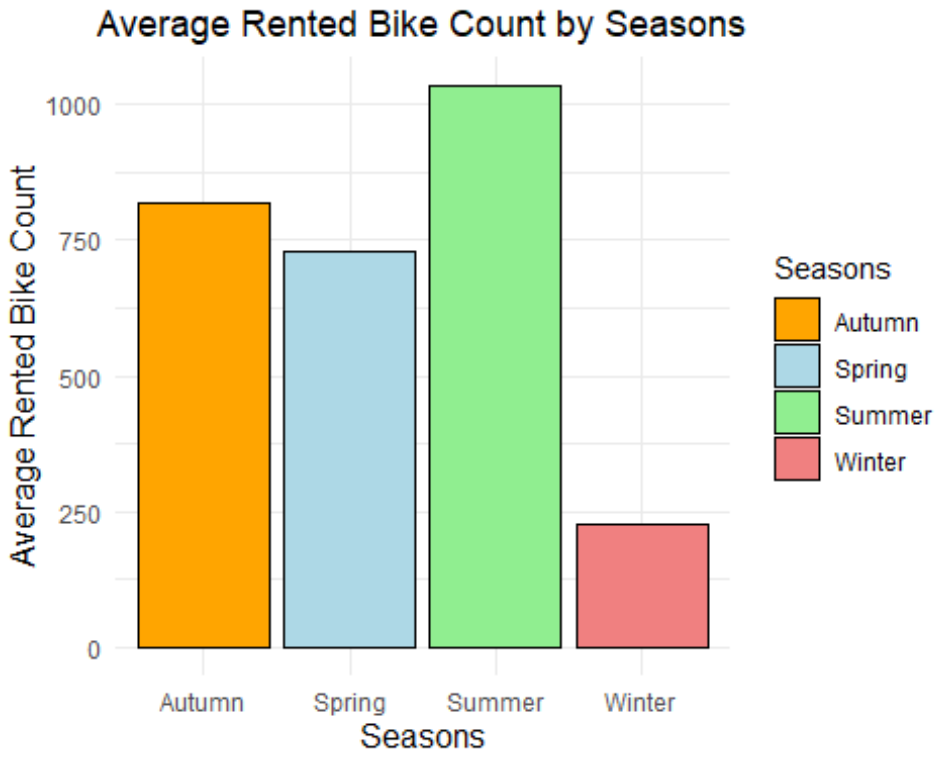

DPT
300
200
100
0
Frequency
-30
-20
-10
0
10
20
30
DPT


SOLAR_RAD
6000
4000
2000
0
Frequency
0
1
2
3
4
Solar_Rad

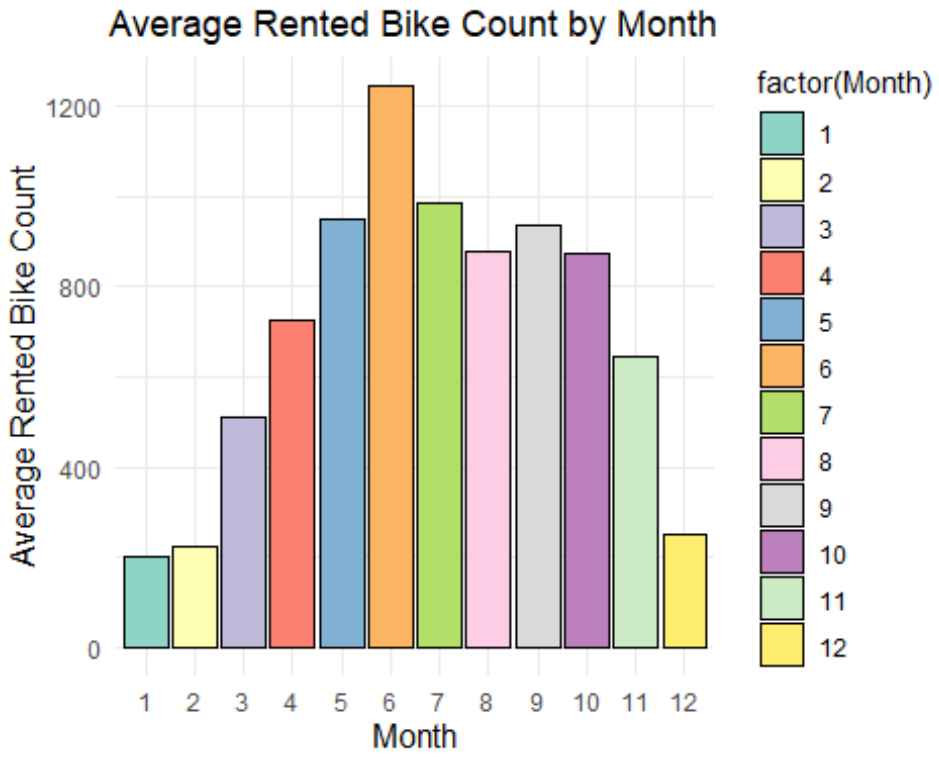

RAINFALL
8000
6000
4000
2000
0
Frequency
0
10
20
30
Rainfall



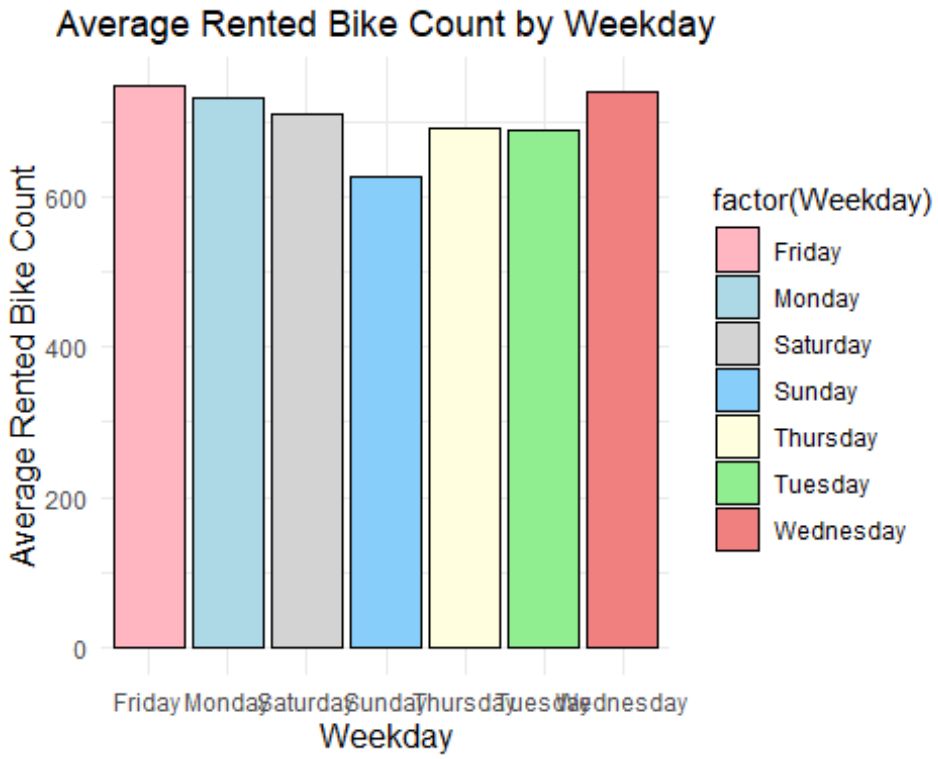

```r
# Barplot for Seasons vs Rented Bike Count
ggplot(seoul_bike_data, aes(x = Seasons, y = RBC, fill = Seasons)) +
  geom_bar(stat = "summary", fun = "mean", color = "black") +
  labs(
    title = "Average Rented Bike Count by Seasons",
    x = "Seasons",
    y = "Average Rented Bike Count"
  ) +
  scale_fill_manual(values = c("Spring" = "lightblue", "Summer" =
"lightgreen", "Autumn" = "orange", "Winter" = "lightcoral")) +
  theme_minimal() +
  theme(
    plot.title = element_text(hjust = 0.5),
    axis.title.x = element_text(size = 12),
    axis.title.y = element_text(size = 12)
  )
```



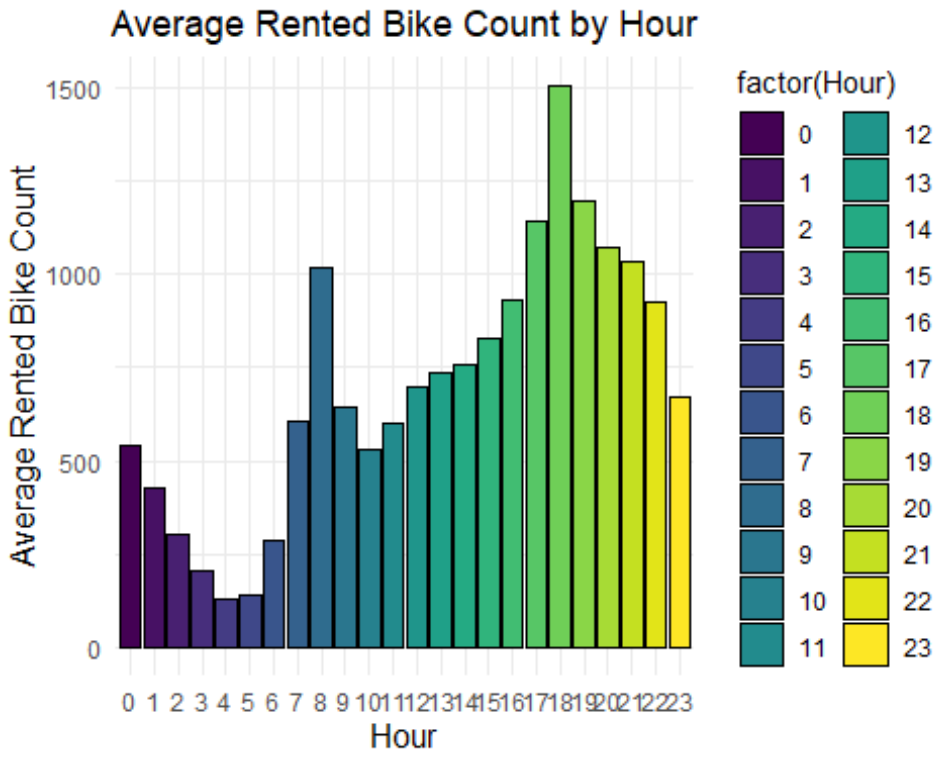

```r
# Barplot for Month vs Rented Bike Count
ggplot(seoul_bike_data, aes(x = factor(Month), y = RBC, fill =
factor(Month))) +
  geom_bar(stat = "summary", fun = "mean", color = "black") +
  labs(
    title = "Average Rented Bike Count by Month",
    x = "Month",
    y = "Average Rented Bike Count"
  ) +
  scale_fill_brewer(palette = "Set3") +  # Automatically use a color palette
  theme_minimal() +
  theme(
    plot.title = element_text(hjust = 0.5),
    axis.title.x = element_text(size = 12),
    axis.title.y = element_text(size = 12)
  )
```



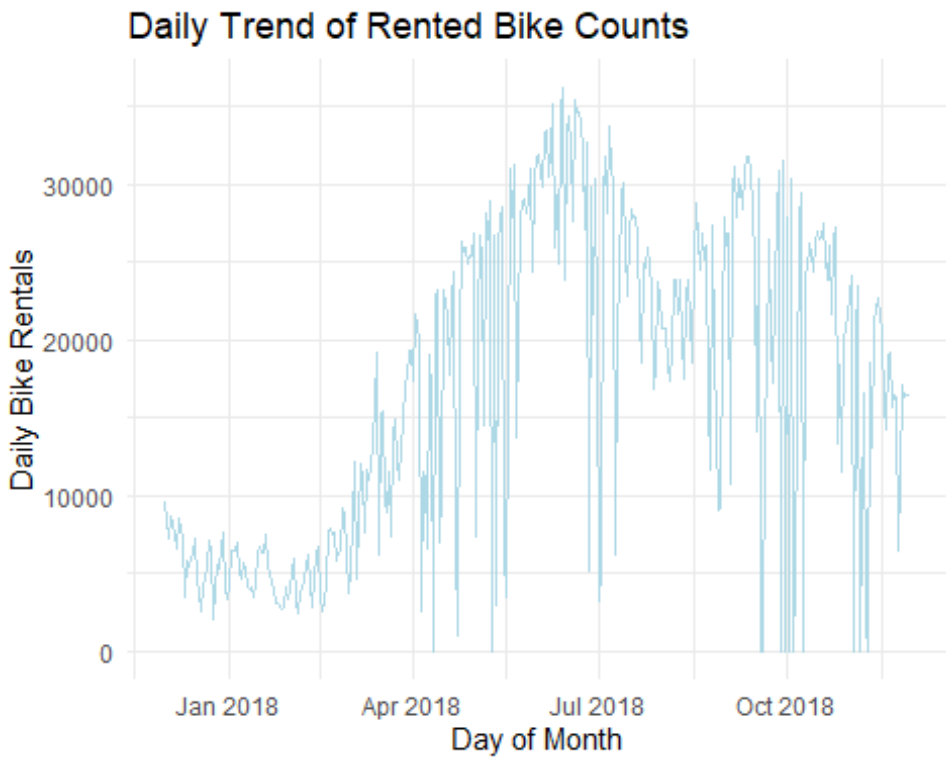

```r
# Barplot for Weekday vs Rented Bike Count
ggplot(seoul_bike_data, aes(x = factor(Weekday), y = RBC, fill =
factor(Weekday))) +
  geom_bar(stat = "summary", fun = "mean", color = "black") +
  labs(
    title = "Average Rented Bike Count by Weekday",
    x = "Weekday",
    y = "Average Rented Bike Count"
  ) +
  scale_fill_manual(values = c("Monday" = "lightblue", "Tuesday" =
"lightgreen", "Wednesday" = "lightcoral",
                               "Thursday" = "lightyellow", "Friday" =
"lightpink", "Saturday" = "lightgray", "Sunday" = "lightskyblue")) +
  theme_minimal() +
  theme(
    plot.title = element_text(hjust = 0.5),
    axis.title.x = element_text(size = 12),
    axis.title.y = element_text(size = 12)
  )
```



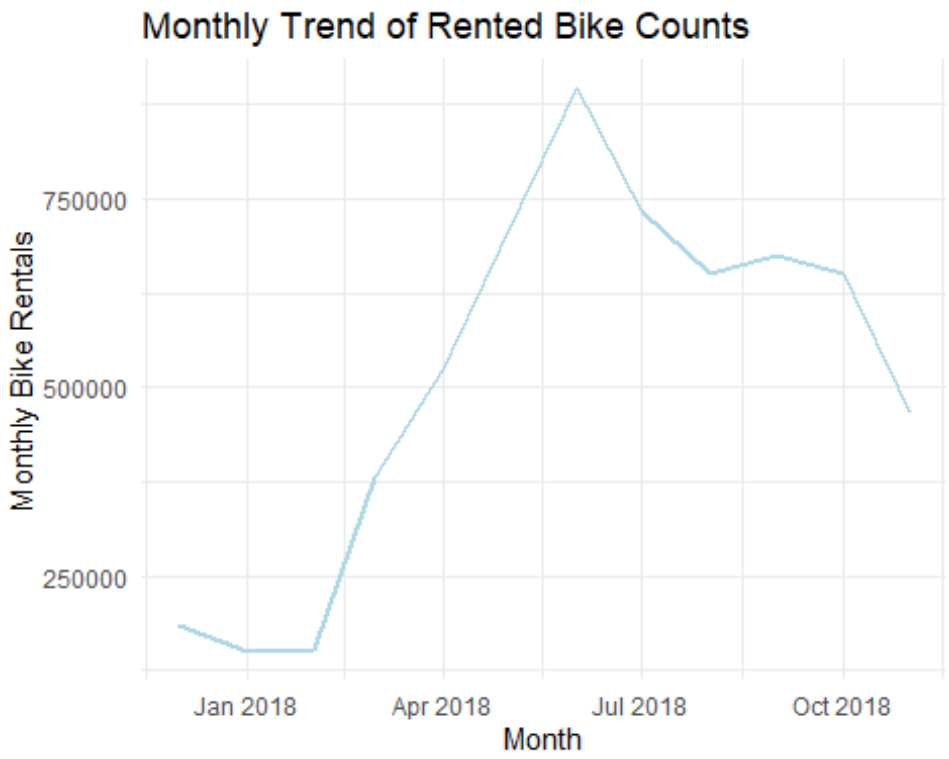

```r
# Barplot for Hour vs Rented Bike Count
ggplot(seoul_bike_data, aes(x = factor(Hour), y = RBC, fill = factor(Hour)))
+
  geom_bar(stat = "summary", fun = "mean", color = "black") +
  labs(
    title = "Average Rented Bike Count by Hour",
    x = "Hour",
    y = "Average Rented Bike Count"
  ) +
  scale_fill_viridis_d() +  # Use a color palette from the viridis package
  theme_minimal() +
  theme(
    plot.title = element_text(hjust = 0.5),
    axis.title.x = element_text(size = 12),
    axis.title.y = element_text(size = 12)
  )
```




```r
#RBC trend Analysis Over Time
#Daily Trends
daily_trends <- seoul_bike_data %>%
  group_by(Date) %>%
  summarise(Daily_RBC = sum(RBC, na.rm = TRUE))

ggplot(daily_trends, aes(x = Date, y = Daily_RBC)) +
  geom_line(color = "lightblue") +
  labs(title = "Daily Trend of Rented Bike Counts", x = "Day of Month", y =
"Daily Bike Rentals") +
  theme_minimal()
```




```r
#Monthly Trends
monthly_trends = seoul_bike_data %>%
  group_by(Year, Month) %>%
  summarise(Monthly_RBC = sum(RBC, na.rm = TRUE))

## `summarise()` has grouped output by 'Year'. You can override using the
## `.groups` argument.

monthly_trends = monthly_trends %>%
  mutate(YearMonth = as.Date(paste(Year, Month, "01", sep = "-")))

ggplot(monthly_trends, aes(x = YearMonth, y = Monthly_RBC)) +
  geom_line(color = "lightblue", size = 1) +
  labs(
    title = "Monthly Trend of Rented Bike Counts",
    x = "Month",
    y = "Monthly Bike Rentals"
  ) +
  theme_minimal()
```

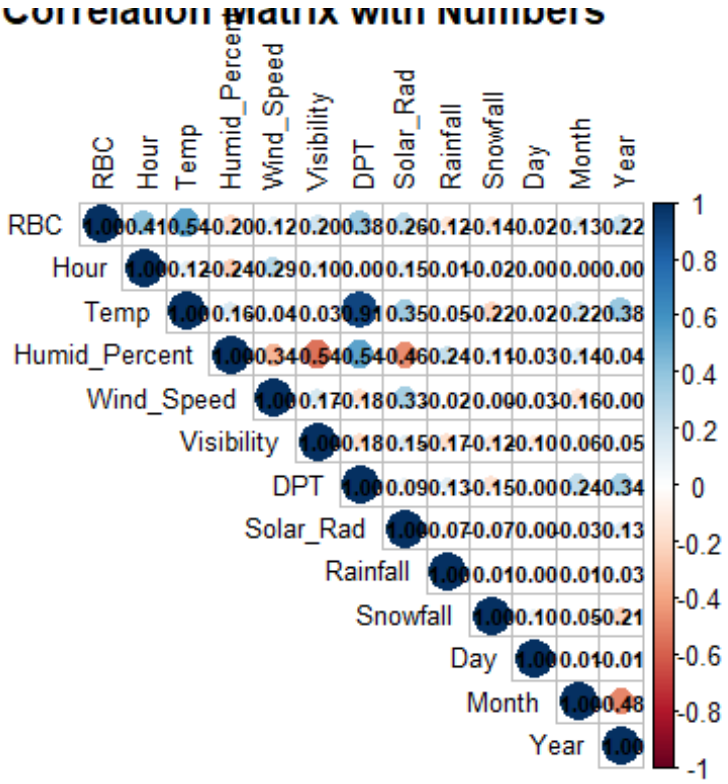



```
seoul_bike_data <- seoul_bike_data[, !(names(seoul_bike_data) %in%
c("Day_Type", "Time_of_Day"))]
head(seoul_bike_data)
```

```
##         Date RBC Hour Temp Humid_Percent Wind_Speed Visibility   DPT
Solar_Rad
## 1 2017-12-01 254    0 -5.2            37        2.2       2000 -17.6
0
## 2 2017-12-01 204    1 -5.5            38        0.8       2000 -17.6
0
## 3 2017-12-01 173    2 -6.0            39        1.0       2000 -17.7
0
## 4 2017-12-01 107    3 -6.2            40        0.9       2000 -17.6
0
## 5 2017-12-01  78    4 -6.0            36        2.3       2000 -18.6
0
## 6 2017-12-01 100    5 -6.4            37        1.5       2000 -18.7
0
##   Rainfall Snowfall Seasons    Holiday Functioning_Day Day Month Year
Weekday
## 1        0        0  Winter No Holiday             Yes   1    12 2017
Friday
## 2        0        0  Winter No Holiday             Yes   1    12 2017
Friday
## 3        0        0  Winter No Holiday             Yes   1    12 2017
Friday
## 4        0        0  Winter No Holiday             Yes   1    12 2017
```

```
Friday
## 5        0        0  Winter No Holiday             Yes    1    12 2017
Friday
## 6        0        0  Winter No Holiday             Yes    1    12 2017
Friday
```

```r
##c) Correlation Analysis for numerical Columns
seoul_bike_data$Hour = as.numeric(seoul_bike_data$Hour)
numeric_cols = seoul_bike_data[, sapply(seoul_bike_data, is.numeric)]
corr_matrix = cor(numeric_cols, use = "complete.obs")
corrplot(corr_matrix,
         method = "circle",        # Use circle method for visualization
         type = "upper",           # Show the upper triangle of the matrix
         tl.cex = 0.8,             # Adjust text label size
         addCoef.col = "black",    # Add correlation numbers in black color
         tl.col = "black",         # Add color to the axis labels
         number.cex = 0.7,         # Adjust the size of the correlation
numbers
         title = "Correlation Matrix with Numbers")
```



##The correlation between Temp and DPT is 0.91, which indicates a very high correlation. This suggests that both variables are likely capturing similar information, which could lead to multicollinearity if both are included in a model.
##There is also a high correlation (0.54) between Humid_Percent and DPT. These variables might be related because humidity and dew point temperature are often closely related in meteorological contexts.

```
##Solar_Rad and Temp have a correlation of 0.35, which is moderate, but still
might raise concerns about collinearity depending on the context.
##To Address multicollinearity we will examine VIF (Variance Inflation
Factor)for each variable to check the extent of multicollinearity and might
might consider removing one of the highly correlated variables by conducting
appropriate tests.
```

```
##3. Feature Engineering
##a) Variance Inflation Factor (VIF)
seoul_bike_data_numeric = seoul_bike_data %>% select_if(is.numeric)  # Select
only numeric columns
vif_result = vif(lm(RBC ~ ., data = seoul_bike_data_numeric))  # RBC as the
dependent variable
print(vif_result)
```

```
##          Hour          Temp Humid_Percent    Wind_Speed    Visibility
##      1.188483     87.735664     20.448556      1.303892      1.673881
##           DPT     Solar_Rad      Rainfall      Snowfall           Day
##    115.697293      2.034245      1.084795      1.130244      1.044487
##         Month          Year
##      1.880074      2.013793
```

```
##Just as the Correlation Matrix the VIF shows that Temp, Humid_Percent, DPT
have a really large VIF score, and thus are highly correlated.
##We might conider dropping one of this and testing how the model performs in
further stpes by checking the VIF scores by dropping one of these varaibles
after conducting appropriate tests.
##First lest check if dropping DPT improves the VIF scores of the other
predictors.
seoul_bike_data_numeric_1 = dplyr::select(seoul_bike_data_numeric, -DPT)
vif_result_1 = vif(lm(RBC ~ ., data = seoul_bike_data_numeric_1))  # RBC as
the dependent variable
print(vif_result_1)
```

```
##          Hour          Temp Humid_Percent    Wind_Speed    Visibility
##      1.186224      2.173417      2.627656      1.301828      1.663552
##     Solar_Rad      Rainfall      Snowfall           Day         Month
##      1.937487      1.070908      1.125286      1.044456      1.880066
##          Year
##      2.013747
```

```
##We see that by dropping DPT predictor our VIF scores for all the other
predictors presnt within the dataset improves and none of them show
significant signs of collinearity.
```

```r
##b) Encode categorical variables.
table(seoul_bike_data$Seasons)

##
## Autumn Spring Summer Winter
##   2184   2208   2208   2160

table(seoul_bike_data$Holiday)

##
##    Holiday No Holiday
##        432       8328

table(seoul_bike_data$Functioning_Day)

##
##   No  Yes
##  295 8465

table(seoul_bike_data$Weekday)

##
##    Friday    Monday  Saturday    Sunday  Thursday   Tuesday Wednesday
##      1272      1248      1248      1248      1248      1248      1248

seoul_bike_data$Holiday = ifelse(seoul_bike_data$Holiday == "No Holiday", 0,
1)
seoul_bike_data$`Functioning_Day`= ifelse(seoul_bike_data$`Functioning_Day`
== "No", 0, 1)

seoul_bike_data_season <- fastDummies::dummy_cols(seoul_bike_data,
select_columns = "Seasons", remove_first_dummy = TRUE)
seoul_bike_data_weekday <- fastDummies::dummy_cols(seoul_bike_data,
select_columns = "Weekday", remove_first_dummy = TRUE)
seoul_bike_data_season <- seoul_bike_data_season[, grep("Seasons",
colnames(seoul_bike_data_season))]
seoul_bike_data_weekday <- seoul_bike_data_weekday[, grep("Weekday",
colnames(seoul_bike_data_weekday))]
seoul_bike_data <- cbind(seoul_bike_data, seoul_bike_data_season,
seoul_bike_data_weekday)
seoul_bike_data <- seoul_bike_data[, !(names(seoul_bike_data) %in%
c("Seasons", "Weekday"))]
head(seoul_bike_data)

##         Date RBC Hour Temp Humid_Percent Wind_Speed Visibility   DPT
Solar_Rad
## 1 2017-12-01 254    0 -5.2            37        2.2       2000 -17.6
0
## 2 2017-12-01 204    1 -5.5            38        0.8       2000 -17.6
0
## 3 2017-12-01 173    2 -6.0            39        1.0       2000 -17.7
0
```

```
## 4 2017-12-01 107    3 -6.2            40        0.9       2000 -17.6
0
## 5 2017-12-01  78    4 -6.0            36        2.3       2000 -18.6
0
## 6 2017-12-01 100    5 -6.4            37        1.5       2000 -18.7
0
##   Rainfall Snowfall Holiday Functioning_Day Day Month Year Seasons_Spring
## 1        0        0       0               1   1    12 2017              0
## 2        0        0       0               1   1    12 2017              0
## 3        0        0       0               1   1    12 2017              0
## 4        0        0       0               1   1    12 2017              0
## 5        0        0       0               1   1    12 2017              0
## 6        0        0       0               1   1    12 2017              0
##   Seasons_Summer Seasons_Winter Weekday_Monday Weekday_Saturday
Weekday_Sunday
## 1              0              1              0                 0
0
## 2              0              1              0                 0
0
## 3              0              1              0                 0
0
## 4              0              1              0                 0
0
## 5              0              1              0                 0
0
## 6              0              1              0                 0
0
##   Weekday_Thursday Weekday_Tuesday Weekday_Wednesday
## 1                0               0                 0
## 2                0               0                 0
## 3                0               0                 0
## 4                0               0                 0
## 5                0               0                 0
## 6                0               0                 0

seoul_bike_data = dplyr::select(seoul_bike_data, -Date)
head(seoul_bike_data)

##   RBC Hour Temp Humid_Percent Wind_Speed Visibility   DPT Solar_Rad
Rainfall
## 1 254    0 -5.2            37        2.2       2000 -17.6         0
0
## 2 204    1 -5.5            38        0.8       2000 -17.6         0
0
## 3 173    2 -6.0            39        1.0       2000 -17.7         0
0
## 4 107    3 -6.2            40        0.9       2000 -17.6         0
0
## 5  78    4 -6.0            36        2.3       2000 -18.6         0
0
```

```
## 6 100    5 -6.4            37        1.5       2000 -18.7         0
0
##   Snowfall Holiday Functioning_Day Day Month Year Seasons_Spring
Seasons_Summer
## 1        0       0               1   1    12 2017              0
0
## 2        0       0               1   1    12 2017              0
0
## 3        0       0               1   1    12 2017              0
0
## 4        0       0               1   1    12 2017              0
0
## 5        0       0               1   1    12 2017              0
0
## 6        0       0               1   1    12 2017              0
0
##   Seasons_Winter Weekday_Monday Weekday_Saturday Weekday_Sunday
## 1              1              0                0              0
## 2              1              0                0              0
## 3              1              0                0              0
## 4              1              0                0              0
## 5              1              0                0              0
## 6              1              0                0              0
##   Weekday_Thursday Weekday_Tuesday Weekday_Wednesday
## 1                0               0                 0
## 2                0               0                 0
## 3                0               0                 0
## 4                0               0                 0
## 5                0               0                 0
## 6                0               0                 0
```

```r
ncol(seoul_bike_data)
```

```
## [1] 24
```

```r
model_with_dpt = lm(RBC ~ . , data = seoul_bike_data)
model_without_dpt = lm(RBC ~ . - DPT, data = seoul_bike_data)
summary(model_with_dpt)
```

```
##
## Call:
## lm(formula = RBC ~ ., data = seoul_bike_data)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1158.14  -275.28   -56.18   206.40  2224.00
##
## Coefficients:
##                      Estimate Std. Error t value Pr(>|t|)
## (Intercept)        1.130e+06  1.389e+05   8.136 4.65e-16 ***
## Hour               2.730e+01  7.291e-01  37.449  < 2e-16 ***
```

```
## Temp                1.833e+01  3.647e+00   5.026 5.11e-07 ***
## Humid_Percent      -1.081e+01  1.026e+00 -10.529  < 2e-16 ***
## Wind_Speed          1.734e+01  5.067e+00   3.423 0.000622 ***
## Visibility          2.967e-03  9.952e-03   0.298 0.765615
## DPT                 9.473e+00  3.817e+00   2.482 0.013093 *
## Solar_Rad          -8.310e+01  7.560e+00 -10.993  < 2e-16 ***
## Rainfall           -5.732e+01  4.238e+00 -13.524  < 2e-16 ***
## Snowfall            3.235e+01  1.125e+01   2.875 0.004053 **
## Holiday            -1.275e+02  2.154e+01  -5.916 3.41e-09 ***
## Functioning_Day     9.514e+02  2.670e+01  35.630  < 2e-16 ***
## Day                -1.260e+00  5.347e-01  -2.356 0.018510 *
## Month              -4.466e+01  6.267e+00  -7.126 1.12e-12 ***
## Year               -5.596e+02  6.878e+01  -8.135 4.67e-16 ***
## Seasons_Spring     -4.059e+02  3.991e+01 -10.170  < 2e-16 ***
## Seasons_Summer     -2.955e+02  2.596e+01 -11.385  < 2e-16 ***
## Seasons_Winter     -7.733e+02  5.640e+01 -13.711  < 2e-16 ***
## Weekday_Monday     -5.446e+01  1.717e+01  -3.171 0.001526 **
## Weekday_Saturday   -6.781e+01  1.713e+01  -3.958 7.61e-05 ***
## Weekday_Sunday     -1.395e+02  1.714e+01  -8.141 4.45e-16 ***
## Weekday_Thursday   -3.037e+01  1.714e+01  -1.772 0.076491 .
## Weekday_Tuesday    -2.716e+01  1.721e+01  -1.579 0.114479
## Weekday_Wednesday  -2.250e+00  1.719e+01  -0.131 0.895883
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 429 on 8736 degrees of freedom
## Multiple R-squared:  0.5587, Adjusted R-squared:  0.5576
## F-statistic:   481 on 23 and 8736 DF,  p-value: < 2.2e-16

summary(model_without_dpt)

##
## Call:
## lm(formula = RBC ~ . - DPT, data = seoul_bike_data)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1154.32  -278.27   -55.26   208.61  2227.60
##
## Coefficients:
##                     Estimate Std. Error t value Pr(>|t|)
## (Intercept)        1.134e+06  1.389e+05   8.168 3.58e-16 ***
## Hour               2.724e+01  7.289e-01  37.376  < 2e-16 ***
## Temp               2.712e+01  8.684e-01  31.226  < 2e-16 ***
## Humid_Percent     -8.431e+00  3.718e-01 -22.677  < 2e-16 ***
## Wind_Speed         1.685e+01  5.065e+00   3.327 0.000882 ***
## Visibility         4.283e-03  9.941e-03   0.431 0.666620
## Solar_Rad         -8.700e+01  7.398e+00 -11.760  < 2e-16 ***
## Rainfall          -5.855e+01  4.210e+00 -13.907  < 2e-16 ***
## Snowfall           3.027e+01  1.123e+01   2.696 0.007026 **
```

```
## Holiday           -1.273e+02  2.155e+01  -5.905 3.65e-09 ***
## Functioning_Day    9.497e+02  2.670e+01  35.569  < 2e-16 ***
## Day               -1.267e+00  5.348e-01  -2.369 0.017875 *
## Month             -4.490e+01  6.268e+00  -7.164 8.48e-13 ***
## Year              -5.620e+02  6.880e+01  -8.169 3.55e-16 ***
## Seasons_Spring    -4.085e+02  3.991e+01 -10.237  < 2e-16 ***
## Seasons_Summer    -2.926e+02  2.594e+01 -11.281  < 2e-16 ***
## Seasons_Winter    -7.761e+02  5.641e+01 -13.760  < 2e-16 ***
## Weekday_Monday    -5.275e+01  1.717e+01  -3.073 0.002128 **
## Weekday_Saturday  -6.865e+01  1.713e+01  -4.007 6.20e-05 ***
## Weekday_Sunday    -1.403e+02  1.714e+01  -8.184 3.12e-16 ***
## Weekday_Thursday  -3.014e+01  1.715e+01  -1.758 0.078780 .
## Weekday_Tuesday   -2.514e+01  1.719e+01  -1.462 0.143659
## Weekday_Wednesday -1.770e+00  1.720e+01  -0.103 0.918017
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 429.1 on 8737 degrees of freedom
## Multiple R-squared:  0.5584, Adjusted R-squared:  0.5573
## F-statistic: 502.3 on 22 and 8737 DF,  p-value: < 2.2e-16

pred_with_dpt <- predict(model_with_dpt, newdata = seoul_bike_data)
pred_without_dpt <- predict(model_without_dpt, newdata = seoul_bike_data)
mse_with_dpt <- mse(seoul_bike_data$RBC, pred_with_dpt)
mse_without_dpt <- mse(seoul_bike_data$RBC, pred_without_dpt)
cat("MSE with DPT:", mse_with_dpt, "\n")

## MSE with DPT: 183548.8

cat("MSE without DPT:", mse_without_dpt, "\n")

## MSE without DPT: 183678.2

anova(model_without_dpt, model_with_dpt)

## Analysis of Variance Table
##
## Model 1: RBC ~ (Hour + Temp + Humid_Percent + Wind_Speed + Visibility +
##     DPT + Solar_Rad + Rainfall + Snowfall + Holiday + Functioning_Day +
##     Day + Month + Year + Seasons_Spring + Seasons_Summer + Seasons_Winter
+
##     Weekday_Monday + Weekday_Saturday + Weekday_Sunday + Weekday_Thursday
+
##     Weekday_Tuesday + Weekday_Wednesday) - DPT
## Model 2: RBC ~ Hour + Temp + Humid_Percent + Wind_Speed + Visibility +
##     DPT + Solar_Rad + Rainfall + Snowfall + Holiday + Functioning_Day +
##     Day + Month + Year + Seasons_Spring + Seasons_Summer + Seasons_Winter
+
##     Weekday_Monday + Weekday_Saturday + Weekday_Sunday + Weekday_Thursday
+
##     Weekday_Tuesday + Weekday_Wednesday
```

```
##   Res.Df        RSS Df Sum of Sq     F  Pr(>F)
## 1   8737 1609021440
## 2   8736 1607887849  1   1133590 6.159 0.01309 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
seoul_bike_data = seoul_bike_data[, !names(seoul_bike_data) %in% "DPT"]
##Drop DPT
head(seoul_bike_data)
```

```
##   RBC Hour Temp Humid_Percent Wind_Speed Visibility Solar_Rad Rainfall
Snowfall
## 1 254    0 -5.2            37        2.2       2000         0        0
0
## 2 204    1 -5.5            38        0.8       2000         0        0
0
## 3 173    2 -6.0            39        1.0       2000         0        0
0
## 4 107    3 -6.2            40        0.9       2000         0        0
0
## 5  78    4 -6.0            36        2.3       2000         0        0
0
## 6 100    5 -6.4            37        1.5       2000         0        0
0
##   Holiday Functioning_Day Day Month Year Seasons_Spring Seasons_Summer
## 1       0               1   1    12 2017              0              0
## 2       0               1   1    12 2017              0              0
## 3       0               1   1    12 2017              0              0
## 4       0               1   1    12 2017              0              0
## 5       0               1   1    12 2017              0              0
## 6       0               1   1    12 2017              0              0
##   Seasons_Winter Weekday_Monday Weekday_Saturday Weekday_Sunday
## 1              1              0                0              0
## 2              1              0                0              0
## 3              1              0                0              0
## 4              1              0                0              0
## 5              1              0                0              0
## 6              1              0                0              0
##   Weekday_Thursday Weekday_Tuesday Weekday_Wednesday
## 1                0               0                 0
## 2                0               0                 0
## 3                0               0                 0
## 4                0               0                 0
## 5                0               0                 0
## 6                0               0                 0
```

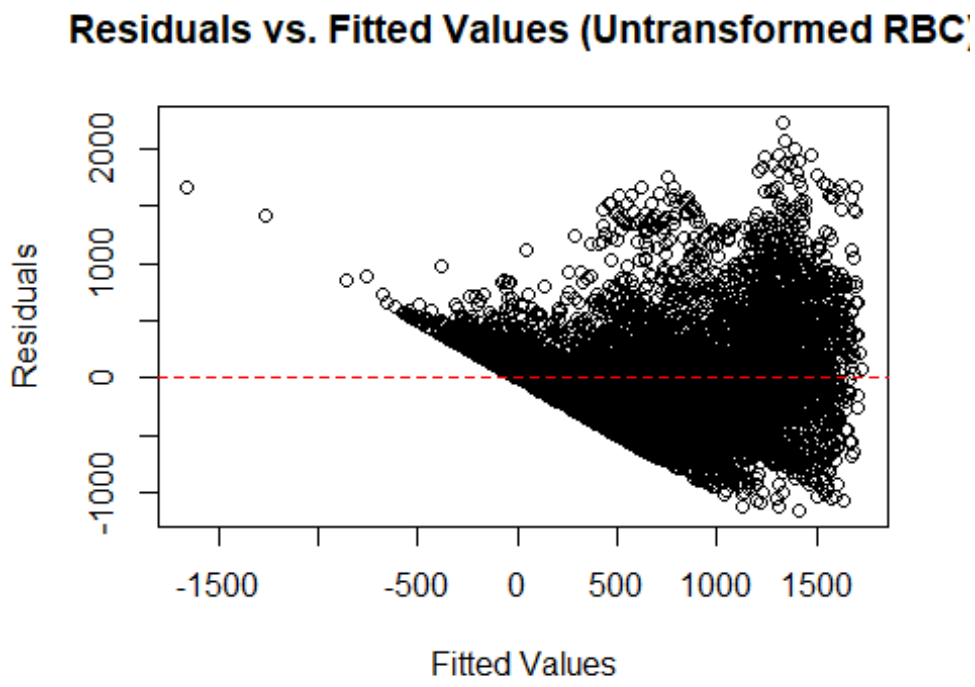
```
ncol(seoul_bike_data)
```

```
## [1] 23
```

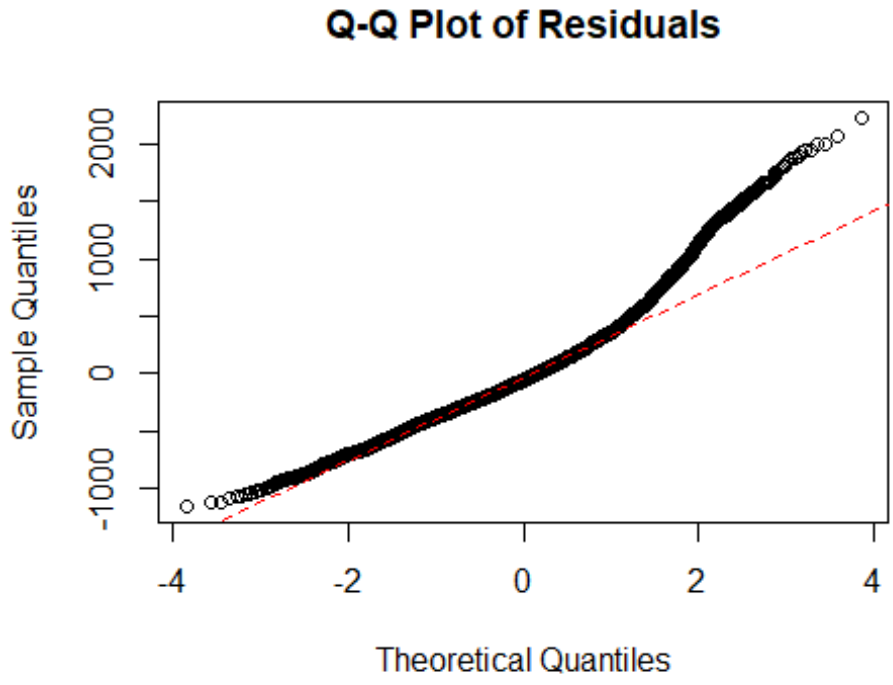
```r
##Fitting a preliminary linear model using the untransformed RBC
model_raw = lm(RBC ~ ., data = seoul_bike_data)
summary(model_raw)
```

```
##
## Call:
## lm(formula = RBC ~ ., data = seoul_bike_data)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1154.32  -278.27   -55.26   208.61  2227.60
##
## Coefficients:
##                      Estimate Std. Error t value Pr(>|t|)
## (Intercept)         1.134e+06  1.389e+05   8.168 3.58e-16 ***
## Hour                2.724e+01  7.289e-01  37.376  < 2e-16 ***
## Temp                2.712e+01  8.684e-01  31.226  < 2e-16 ***
## Humid_Percent      -8.431e+00  3.718e-01 -22.677  < 2e-16 ***
## Wind_Speed          1.685e+01  5.065e+00   3.327 0.000882 ***
## Visibility          4.283e-03  9.941e-03   0.431 0.666620
## Solar_Rad          -8.700e+01  7.398e+00 -11.760  < 2e-16 ***
## Rainfall           -5.855e+01  4.210e+00 -13.907  < 2e-16 ***
## Snowfall            3.027e+01  1.123e+01   2.696 0.007026 **
## Holiday            -1.273e+02  2.155e+01  -5.905 3.65e-09 ***
## Functioning_Day     9.497e+02  2.670e+01  35.569  < 2e-16 ***
## Day                -1.267e+00  5.348e-01  -2.369 0.017875 *
## Month              -4.490e+01  6.268e+00  -7.164 8.48e-13 ***
## Year               -5.620e+02  6.880e+01  -8.169 3.55e-16 ***
## Seasons_Spring     -4.085e+02  3.991e+01 -10.237  < 2e-16 ***
## Seasons_Summer     -2.926e+02  2.594e+01 -11.281  < 2e-16 ***
## Seasons_Winter     -7.761e+02  5.641e+01 -13.760  < 2e-16 ***
## Weekday_Monday     -5.275e+01  1.717e+01  -3.073 0.002128 **
## Weekday_Saturday   -6.865e+01  1.713e+01  -4.007 6.20e-05 ***
## Weekday_Sunday     -1.403e+02  1.714e+01  -8.184 3.12e-16 ***
## Weekday_Thursday   -3.014e+01  1.715e+01  -1.758 0.078780 .
## Weekday_Tuesday    -2.514e+01  1.719e+01  -1.462 0.143659
## Weekday_Wednesday  -1.770e+00  1.720e+01  -0.103 0.918017
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 429.1 on 8737 degrees of freedom
## Multiple R-squared:  0.5584, Adjusted R-squared:  0.5573
## F-statistic: 502.3 on 22 and 8737 DF,  p-value: < 2.2e-16
```

```r
##Plotting residuals vs. fitted values to check for model assumptions
plot(model_raw$fitted.values, residuals(model_raw),
     xlab = "Fitted Values",
     ylab = "Residuals",
     main = "Residuals vs. Fitted Values (Untransformed RBC)")
abline(h = 0, col = "red", lty = 2)
```




```r
##Q-Q plot
qqnorm(residuals(model_raw), main = "Q-Q Plot of Residuals")
qqline(residuals(model_raw), col = "red", lty = 2)  # Add a reference line
```



```
##The Q-Q plot of Residuals shows significant deviations from the diagonal line, especially at the tails.This suggests that the residuals are not normally distributed, particularly with heavy tails or skewness.
##The Residuals vs. Fitted Values plot shows a funnel-like shape. This indicates non equal variance of residuals across the range of fitted values.
##Similary it also violates Linearity assumption.


##Lets check for normality and EV using the tests
bp_test_raw = bptest(model_raw)
print(bp_test_raw)
```

```
##
##  studentized Breusch-Pagan test
##
## data:  model_raw
## BP = 856.04, df = 22, p-value < 2.2e-16
```

```
##Since Shapiro test can only be done on sample size between 3 and 5000 we will use Kolmogorov-Smirnov Test for normality assumption.
residuals = resid(model_raw)
ks_test = ks.test(residuals, "pnorm", mean(residuals), sd(residuals))
print(ks_test)
```

```
##
##  Asymptotic one-sample Kolmogorov-Smirnov test
##
## data:  residuals
## D = 0.069661, p-value < 2.2e-16
## alternative hypothesis: two-sided
```

```
##From the BP test and the KS test we get the same results that the model assumptions are not satisfied by the model.


# Calculate residuals and influence metrics
cooks_distances = cooks.distance(model_raw) # Cook's distance
standardized_residuals = rstandard(model_raw) # Standardized residuals
cooks_threshold = 4 / nrow(seoul_bike_data) # Common threshold for Cook's distance
residuals_threshold = 2 # Common threshold for standardized residuals (|Residual| > 2 is considered an outlier)
influential_points = which(cooks_distances > cooks_threshold)
outliers = which(abs(standardized_residuals) > residuals_threshold)
overlap_points = intersect(influential_points, outliers)
cat("Number of influential points:", length(influential_points), "\n")
```

```
## Number of influential points: 393
```

```
cat("Number of outliers:", length(outliers), "\n")
```

```
## Number of outliers: 450

cat("Number of overlapping points (outliers and influential):",
length(overlap_points), "\n")

## Number of overlapping points (outliers and influential): 321

cat("Indices of influential points:", influential_points, "\n")

## Indices of influential points: 81 129 153 562 565 2020 2025 2208 2506 2947
2971 2995 3033 3105 3115 3141 3142 3143 3211 3273 3283 3297 3321 3465 3475
3489 3499 3513 3523 3537 3547 3609 3619 3638 3639 3640 3641 3642 3643 3644
3645 3646 3681 3691 3705 3715 3762 3763 3786 3801 3825 3835 3836 3873 3883
3925 3926 3927 3928 3929 3930 3931 3932 3945 3955 3956 3969 3979 3997 3998
4013 4017 4028 4051 4070 4071 4072 4073 4074 4075 4098 4099 4107 4108 4109
4110 4123 4124 4125 4126 4132 4133 4134 4135 4140 4141 4142 4144 4149 4150
4151 4161 4171 4172 4185 4195 4196 4209 4219 4254 4256 4265 4266 4267 4268
4277 4278 4281 4291 4292 4293 4305 4315 4329 4339 4340 4341 4353 4363 4364
4365 4377 4387 4436 4437 4449 4459 4460 4461 4462 4473 4483 4484 4505 4506
4507 4508 4509 4510 4521 4531 4532 4545 4555 4556 4557 4558 4559 4581 4582
4601 4602 4603 4605 4617 4641 4651 4652 4653 4654 4655 4671 4673 4674 4675
4676 4699 4713 4723 4724 4749 4750 4769 4770 4771 4772 4773 4785 4809 4819
4820 4821 4822 4833 4843 4844 4845 4846 4857 4867 4868 4869 4870 4881 4891
4892 4953 4963 4964 4965 4985 4986 4987 4988 4989 4990 5001 5011 5012 5013
5025 5035 5049 5059 5060 5061 5100 5133 5145 5155 5156 5157 5169 5179 5180
5193 5203 5217 5227 5228 5313 5347 5371 5372 5385 5395 5491 5515 5846 5847
5848 5849 5861 5871 5872 5873 5895 6181 6182 6183 6184 6185 6235 6307 6316
6317 6331 6381 6382 6475 6489 6499 6502 6503 6521 6525 6527 6528 6571 6572
6629 6640 6641 6642 6644 6645 6646 6657 6667 6681 6691 6705 6715 6720 6729
6739 6801 6811 6825 6835 6849 6859 6873 6883 6897 6907 6911 6953 6969 6979
6980 7001 7002 7024 7025 7026 7032 7075 7209 7219 7305 7315 7358 7359 7360
7361 7362 7363 7413 7414 7415 7416 7422 7423 7473 7483 7521 7531 7545 7555
7569 7579 7641 7651 7665 7675 7689 7713 7723 7737 7747 7809 7819 7836 7843
7857 7867 7881 7891 7977 8001 8011 8025 8049 8059 8073 8083 8145 8155 8229
8231 8232 8233 8313 8337 8361 8385 8409 8419 8481 8505 8520 8529 8553 8577
8603 8604 8605 8606 8649 8673 8721 8745

cat("Indices of outliers:", outliers, "\n")

## Indices of outliers: 2327 2673 2697 2947 2961 2971 2995 3033 3105 3115
3129 3141 3142 3143 3177 3187 3211 3273 3283 3297 3321 3369 3425 3428 3465
3475 3489 3499 3513 3523 3537 3547 3592 3609 3619 3620 3642 3643 3681 3691
3705 3715 3762 3763 3784 3785 3786 3801 3811 3812 3825 3835 3836 3873 3883
3925 3926 3927 3928 3929 3930 3931 3932 3933 3944 3945 3955 3956 3969 3979
3998 4028 4051 4070 4071 4072 4073 4074 4075 4097 4098 4099 4109 4113 4122
4123 4124 4125 4126 4141 4142 4161 4171 4172 4173 4185 4195 4196 4209 4219
4254 4264 4265 4266 4267 4268 4277 4278 4281 4291 4292 4293 4305 4314 4315
4329 4338 4339 4340 4341 4342 4352 4353 4363 4364 4365 4377 4387 4436 4437
4449 4459 4460 4461 4462 4473 4483 4484 4486 4507 4509 4520 4521 4531 4532
4533 4534 4545 4554 4555 4556 4557 4558 4559 4581 4582 4599 4600 4601 4602
4603 4604 4605 4617 4640 4641 4650 4651 4652 4653 4654 4655 4671 4672 4673
```

```
4674 4675 4676 4677 4678 4699 4713 4722 4723 4724 4725 4726 4746 4747 4748
4749 4750 4753 4768 4769 4770 4771 4772 4773 4784 4785 4797 4809 4818 4819
4820 4821 4822 4832 4833 4842 4843 4844 4845 4846 4857 4867 4868 4869 4870
4881 4891 4892 4915 4916 4917 4918 4941 4942 4953 4963 4964 4965 4966 4973
4974 4989 5001 5010 5011 5012 5013 5014 5025 5035 5049 5059 5060 5061 5062
5132 5134 5145 5155 5156 5157 5158 5169 5179 5180 5181 5182 5193 5203 5204
5217 5226 5227 5228 5251 5252 5274 5275 5276 5277 5303 5304 5313 5347 5348
5361 5371 5372 5373 5374 5385 5395 5396 5457 5467 5469 5481 5491 5492 5494
5505 5515 5529 5539 5553 5621 5635 5683 5707 5727 5751 5823 5824 5846 5847
5848 5849 5861 5870 5871 5872 5873 5894 5895 5896 6139 6211 6225 6235 6297
6307 6331 6332 6345 6365 6381 6382 6383 6384 6475 6489 6502 6503 6521 6561
6571 6572 6640 6641 6644 6645 6646 6657 6667 6681 6691 6705 6715 6720 6729
6739 6801 6811 6812 6825 6835 6849 6859 6873 6883 6897 6907 6911 6953 6969
6979 6980 7075 7209 7219 7220 7305 7315 7316 7360 7362 7412 7413 7414 7415
7416 7423 7456 7457 7473 7483 7521 7531 7545 7555 7569 7579 7641 7651 7665
7675 7689 7699 7713 7723 7737 7747 7809 7819 7843 7857 7867 7881 7891 7977
7987 8001 8011 8025 8035 8049 8059 8073 8083 8145 8155 8229 8231 8232 8233
8313 8323 8337 8347 8361 8371 8385 8409 8419 8481 8491 8505 8520 8529 8553
8577 8649 8673 8721 8745
```

```
cat("Indices of overlapping points:", overlap_points, "\n")
```

```
## Indices of overlapping points: 2947 2971 2995 3033 3105 3115 3141 3142
3143 3211 3273 3283 3297 3321 3465 3475 3489 3499 3513 3523 3537 3547 3609
3619 3642 3643 3681 3691 3705 3715 3762 3763 3786 3801 3825 3835 3836 3873
3883 3925 3926 3927 3928 3929 3930 3931 3932 3945 3955 3956 3969 3979 3998
4028 4051 4070 4071 4072 4073 4074 4075 4098 4099 4109 4123 4124 4125 4126
4141 4142 4161 4171 4172 4185 4195 4196 4209 4219 4254 4265 4266 4267 4268
4277 4278 4281 4291 4292 4293 4305 4315 4329 4339 4340 4341 4353 4363 4364
4365 4377 4387 4436 4437 4449 4459 4460 4461 4462 4473 4483 4484 4507 4509
4521 4531 4532 4545 4555 4556 4557 4558 4559 4581 4582 4601 4602 4603 4605
4617 4641 4651 4652 4653 4654 4655 4671 4673 4674 4675 4676 4699 4713 4723
4724 4749 4750 4769 4770 4771 4772 4773 4785 4809 4819 4820 4821 4822 4833
4843 4844 4845 4846 4857 4867 4868 4869 4870 4881 4891 4892 4953 4963 4964
4965 4989 5001 5011 5012 5013 5025 5035 5049 5059 5060 5061 5145 5155 5156
5157 5169 5179 5180 5193 5203 5217 5227 5228 5313 5347 5371 5372 5385 5395
5491 5515 5846 5847 5848 5849 5861 5871 5872 5873 5895 6235 6307 6331 6381
6382 6475 6489 6502 6503 6521 6571 6572 6640 6641 6644 6645 6646 6657 6667
6681 6691 6705 6715 6720 6729 6739 6801 6811 6825 6835 6849 6859 6873 6883
6897 6907 6911 6953 6969 6979 6980 7075 7209 7219 7305 7315 7360 7362 7413
7414 7415 7416 7423 7473 7483 7521 7531 7545 7555 7569 7579 7641 7651 7665
7675 7689 7713 7723 7737 7747 7809 7819 7843 7857 7867 7881 7891 7977 8001
8011 8025 8049 8059 8073 8083 8145 8155 8229 8231 8232 8233 8313 8337 8361
8385 8409 8419 8481 8505 8520 8529 8553 8577 8649 8673 8721 8745
```

```
##Checking if removing these points improves the model
cleaned_data <- seoul_bike_data[-outliers, ]
model_cleaned <- lm(RBC ~ ., data = cleaned_data)
summary(model_cleaned)
```

```
##
## Call:
## lm(formula = RBC ~ ., data = cleaned_data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -789.14 -233.34  -30.98  198.71 1028.60
##
## Coefficients:
##                     Estimate Std. Error t value Pr(>|t|)
## (Intercept)        8.294e+05  1.118e+05   7.419 1.30e-13 ***
## Hour               2.429e+01  5.791e-01  41.934  < 2e-16 ***
## Temp               2.601e+01  6.962e-01  37.352  < 2e-16 ***
## Humid_Percent     -7.421e+00  2.998e-01 -24.757  < 2e-16 ***
## Wind_Speed         1.277e+01  4.044e+00   3.159  0.00159 **
## Visibility         7.934e-03  7.961e-03   0.997  0.31897
## Solar_Rad         -6.330e+01  5.922e+00 -10.690  < 2e-16 ***
## Rainfall          -6.510e+01  3.846e+00 -16.927  < 2e-16 ***
## Snowfall           2.714e+01  8.766e+00   3.096  0.00197 **
## Holiday           -1.039e+02  1.696e+01  -6.123 9.60e-10 ***
## Functioning_Day    9.018e+02  2.095e+01  43.048  < 2e-16 ***
## Day               -1.380e+00  4.278e-01  -3.225  0.00126 **
## Month             -3.133e+01  5.058e+00  -6.195 6.11e-10 ***
## Year              -4.109e+02  5.537e+01  -7.421 1.28e-13 ***
## Seasons_Spring    -3.373e+02  3.243e+01 -10.401  < 2e-16 ***
## Seasons_Summer    -2.899e+02  2.071e+01 -13.997  < 2e-16 ***
## Seasons_Winter    -6.062e+02  4.561e+01 -13.291  < 2e-16 ***
## Weekday_Monday    -5.583e+01  1.382e+01  -4.040 5.38e-05 ***
## Weekday_Saturday  -1.924e+01  1.361e+01  -1.413  0.15767
## Weekday_Sunday    -1.167e+02  1.369e+01  -8.523  < 2e-16 ***
## Weekday_Thursday  -2.247e+01  1.377e+01  -1.631  0.10289
## Weekday_Tuesday   -2.614e+01  1.381e+01  -1.893  0.05833 .
## Weekday_Wednesday -1.209e+01  1.382e+01  -0.875  0.38175
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 334.3 on 8287 degrees of freedom
## Multiple R-squared:  0.632,  Adjusted R-squared:  0.631
## F-statistic: 646.9 on 22 and 8287 DF,  p-value: < 2.2e-16
```

```
summary(model_raw)
```

```
##
## Call:
## lm(formula = RBC ~ ., data = seoul_bike_data)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1154.32  -278.27   -55.26   208.61  2227.60
##
```

```
## Coefficients:
##                      Estimate Std. Error t value Pr(>|t|)
## (Intercept)         1.134e+06  1.389e+05   8.168 3.58e-16 ***
## Hour                2.724e+01  7.289e-01  37.376  < 2e-16 ***
## Temp                2.712e+01  8.684e-01  31.226  < 2e-16 ***
## Humid_Percent      -8.431e+00  3.718e-01 -22.677  < 2e-16 ***
## Wind_Speed          1.685e+01  5.065e+00   3.327 0.000882 ***
## Visibility          4.283e-03  9.941e-03   0.431 0.666620
## Solar_Rad          -8.700e+01  7.398e+00 -11.760  < 2e-16 ***
## Rainfall           -5.855e+01  4.210e+00 -13.907  < 2e-16 ***
## Snowfall            3.027e+01  1.123e+01   2.696 0.007026 **
## Holiday            -1.273e+02  2.155e+01  -5.905 3.65e-09 ***
## Functioning_Day     9.497e+02  2.670e+01  35.569  < 2e-16 ***
## Day                -1.267e+00  5.348e-01  -2.369 0.017875 *
## Month              -4.490e+01  6.268e+00  -7.164 8.48e-13 ***
## Year               -5.620e+02  6.880e+01  -8.169 3.55e-16 ***
## Seasons_Spring     -4.085e+02  3.991e+01 -10.237  < 2e-16 ***
## Seasons_Summer     -2.926e+02  2.594e+01 -11.281  < 2e-16 ***
## Seasons_Winter     -7.761e+02  5.641e+01 -13.760  < 2e-16 ***
## Weekday_Monday     -5.275e+01  1.717e+01  -3.073 0.002128 **
## Weekday_Saturday   -6.865e+01  1.713e+01  -4.007 6.20e-05 ***
## Weekday_Sunday     -1.403e+02  1.714e+01  -8.184 3.12e-16 ***
## Weekday_Thursday   -3.014e+01  1.715e+01  -1.758 0.078780 .
## Weekday_Tuesday    -2.514e+01  1.719e+01  -1.462 0.143659
## Weekday_Wednesday  -1.770e+00  1.720e+01  -0.103 0.918017
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 429.1 on 8737 degrees of freedom
## Multiple R-squared:  0.5584, Adjusted R-squared:  0.5573
## F-statistic: 502.3 on 22 and 8737 DF,  p-value: < 2.2e-16

seoul_bike_data_clean = seoul_bike_data[-overlap_points, ]
head(seoul_bike_data_clean)

##   RBC Hour Temp Humid_Percent Wind_Speed Visibility Solar_Rad Rainfall
Snowfall
## 1 254    0 -5.2            37        2.2       2000         0        0
0
## 2 204    1 -5.5            38        0.8       2000         0        0
0
## 3 173    2 -6.0            39        1.0       2000         0        0
0
## 4 107    3 -6.2            40        0.9       2000         0        0
0
## 5  78    4 -6.0            36        2.3       2000         0        0
0
## 6 100    5 -6.4            37        1.5       2000         0        0
0
##   Holiday Functioning_Day Day Month Year Seasons_Spring Seasons_Summer
```

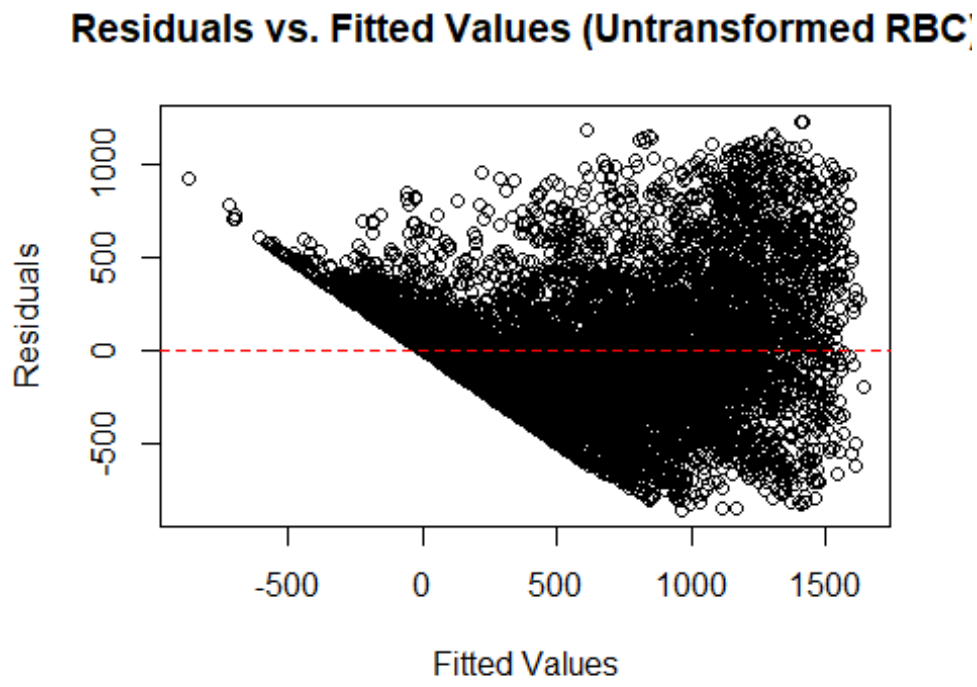
```
## 1       0               1   1    12 2017              0              0
## 2       0               1   1    12 2017              0              0
## 3       0               1   1    12 2017              0              0
## 4       0               1   1    12 2017              0              0
## 5       0               1   1    12 2017              0              0
## 6       0               1   1    12 2017              0              0
##   Seasons_Winter Weekday_Monday Weekday_Saturday Weekday_Sunday
## 1              1              0                0              0
## 2              1              0                0              0
## 3              1              0                0              0
## 4              1              0                0              0
## 5              1              0                0              0
## 6              1              0                0              0
##   Weekday_Thursday Weekday_Tuesday Weekday_Wednesday
## 1                0               0                 0
## 2                0               0                 0
## 3                0               0                 0
## 4                0               0                 0
## 5                0               0                 0
## 6                0               0                 0
```

```
##By removing the outliers, we have improved the fit and accuracy of your model, as reflected by the better residual standard error and higher R-squared values.
model_clean = lm(RBC ~ ., data = seoul_bike_data_clean)
summary(model_clean)
```

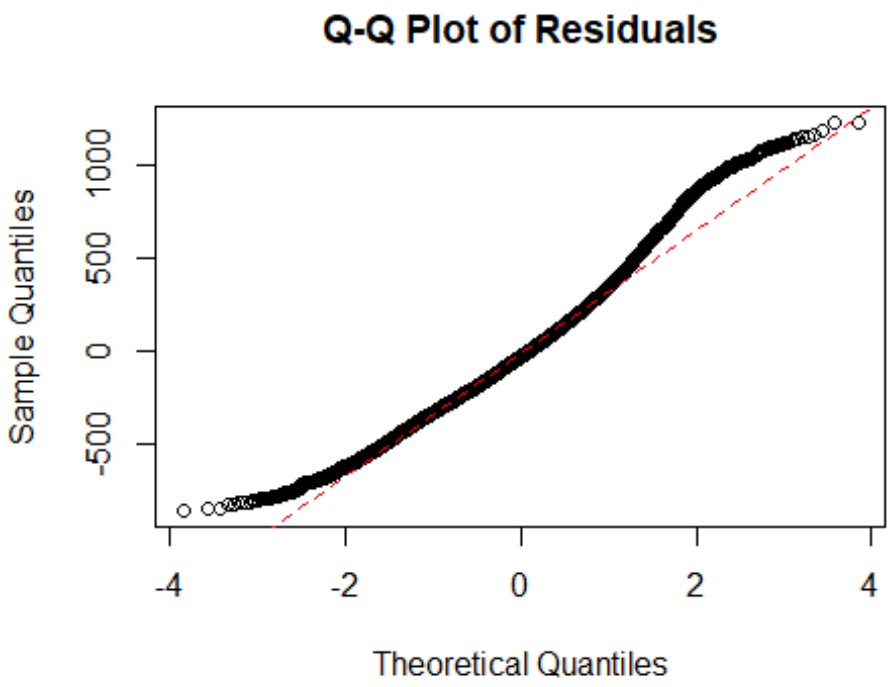
```
##
## Call:
## lm(formula = RBC ~ ., data = seoul_bike_data_clean)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -859.84 -243.57  -33.89  201.27 1224.50
##
## Coefficients:
##                      Estimate Std. Error t value Pr(>|t|)
## (Intercept)        9.404e+05  1.175e+05   8.002 1.38e-15 ***
## Hour               2.533e+01  6.091e-01  41.593  < 2e-16 ***
## Temp               2.620e+01  7.340e-01  35.697  < 2e-16 ***
## Humid_Percent     -7.806e+00  3.157e-01 -24.729  < 2e-16 ***
## Wind_Speed         1.376e+01  4.266e+00   3.225  0.00127 **
## Visibility         4.496e-03  8.385e-03   0.536  0.59179
## Solar_Rad         -7.236e+01  6.238e+00 -11.600  < 2e-16 ***
## Rainfall          -6.775e+01  4.073e+00 -16.635  < 2e-16 ***
## Snowfall           2.920e+01  9.284e+00   3.146  0.00166 **
## Holiday           -1.122e+02  1.797e+01  -6.244 4.48e-10 ***
## Functioning_Day    9.081e+02  2.218e+01  40.933  < 2e-16 ***
## Day               -1.442e+00  4.503e-01  -3.202  0.00137 **
## Month             -3.633e+01  5.314e+00  -6.837 8.66e-12 ***
```

```
## Year               -4.659e+02  5.821e+01  -8.004 1.36e-15 ***
## Seasons_Spring     -3.630e+02  3.404e+01 -10.664  < 2e-16 ***
## Seasons_Summer     -2.778e+02  2.185e+01 -12.713  < 2e-16 ***
## Seasons_Winter     -6.596e+02  4.787e+01 -13.778  < 2e-16 ***
## Weekday_Monday     -4.911e+01  1.452e+01  -3.383  0.00072 ***
## Weekday_Saturday   -2.063e+01  1.435e+01  -1.438  0.15059
## Weekday_Sunday     -1.111e+02  1.440e+01  -7.716 1.34e-14 ***
## Weekday_Thursday   -1.742e+01  1.449e+01  -1.202  0.22921
## Weekday_Tuesday    -2.234e+01  1.452e+01  -1.538  0.12410
## Weekday_Wednesday -2.107e+00  1.453e+01  -0.145  0.88470
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 354.3 on 8416 degrees of freedom
## Multiple R-squared:  0.6186, Adjusted R-squared:  0.6176
## F-statistic: 620.6 on 22 and 8416 DF,  p-value: < 2.2e-16
```

```r
##Plotting residuals vs. fitted values to check for model assumptions
plot(model_clean$fitted.values, residuals(model_clean),
     xlab = "Fitted Values",
     ylab = "Residuals",
     main = "Residuals vs. Fitted Values (Untransformed RBC)")
abline(h = 0, col = "red", lty = 2)
```




```r
##Q-Q plot
qqnorm(residuals(model_clean), main = "Q-Q Plot of Residuals")
qqline(residuals(model_clean), col = "red", lty = 2)  # Add a reference line
```




```
##Removing the outliers doesn't correct model assumptions.

##Lets check for normality and EV using the tests
bp_test_raw_1 = bptest(model_clean)
print(bp_test_raw_1)

##
##  studentized Breusch-Pagan test
##
## data:  model_clean
## BP = 1317.4, df = 22, p-value < 2.2e-16

##Since Shapiro test can only be done on sample size between 3 and 5000 we
will use Kolmogorov-Smirnov Test for normality assumption.
residuals_1 = resid(model_clean)
ks_test_1 = ks.test(residuals, "pnorm", mean(residuals_1), sd(residuals_1))
print(ks_test_1)

##
##  Asymptotic one-sample Kolmogorov-Smirnov test
##
## data:  residuals
## D = 0.065323, p-value < 2.2e-16
## alternative hypothesis: two-sided

##From the BP test and the KS test we get the same results that the model
assumptions are not satisfied by the model.
```

```r
##However, comparing the R^2 of the two models we see that the model without
outliers performs better.


full_model = lm(RBC ~ ., data = seoul_bike_data_clean)
stepwise_model = step(full_model,
                      scope = list(lower = ~1, upper = full_model),
                      direction = "both")
```

```
## Start:  AIC=99097.91
## RBC ~ Hour + Temp + Humid_Percent + Wind_Speed + Visibility +
##     Solar_Rad + Rainfall + Snowfall + Holiday + Functioning_Day +
##     Day + Month + Year + Seasons_Spring + Seasons_Summer + Seasons_Winter
+
##     Weekday_Monday + Weekday_Saturday + Weekday_Sunday + Weekday_Thursday
+
##     Weekday_Tuesday + Weekday_Wednesday
##
##                      Df Sum of Sq        RSS    AIC
## - Weekday_Wednesday  1      2640 1056281750  99096
## - Visibility         1     36094 1056315204  99096
## - Weekday_Thursday   1    181484 1056460594  99097
## <none>                           1056279110  99098
## - Weekday_Saturday   1    259377 1056538487  99098
## - Weekday_Tuesday    1    296867 1056575977  99098
## - Snowfall           1   1241908 1057521019  99106
## - Day                1   1286431 1057565541  99106
## - Wind_Speed         1   1305208 1057584318  99106
## - Weekday_Monday     1   1436491 1057715601  99107
## - Holiday            1   4892686 1061171796  99135
## - Month              1   5866693 1062145804  99143
## - Weekday_Sunday     1   7472011 1063751121  99155
## - Year               1   8040624 1064319734  99160
## - Seasons_Spring     1  14274077 1070553187  99209
## - Solar_Rad          1  16889565 1073168675  99230
## - Seasons_Summer     1  20284481 1076563591  99256
## - Seasons_Winter     1  23825855 1080104965  99284
## - Rainfall           1  34729054 1091008165  99369
## - Humid_Percent      1  76751379 1133030489  99688
## - Temp               1 159933533 1216212643 100286
## - Functioning_Day    1 210285963 1266565073 100628
## - Hour               1 217122949 1273402059 100673
##
## Step:  AIC=99095.93
## RBC ~ Hour + Temp + Humid_Percent + Wind_Speed + Visibility +
##     Solar_Rad + Rainfall + Snowfall + Holiday + Functioning_Day +
##     Day + Month + Year + Seasons_Spring + Seasons_Summer + Seasons_Winter
+
##     Weekday_Monday + Weekday_Saturday + Weekday_Sunday + Weekday_Thursday
+
```

```
##     Weekday_Tuesday
##
##                      Df Sum of Sq        RSS    AIC
## - Visibility          1     34988 1056316738  99094
## - Weekday_Thursday    1    213150 1056494900  99096
## <none>                             1056281750  99096
## - Weekday_Saturday    1    311328 1056593078  99096
## - Weekday_Tuesday     1    358499 1056640249  99097
## + Weekday_Wednesday   1      2640 1056279110  99098
## - Snowfall            1   1239271 1057521021  99104
## - Day                 1   1290047 1057571797  99104
## - Wind_Speed          1   1302963 1057584713  99104
## - Weekday_Monday      1   1822602 1058104352  99108
## - Holiday             1   4899415 1061181165  99133
## - Month               1   5868998 1062150748  99141
## - Year                1   8047789 1064329539  99158
## - Weekday_Sunday      1   9742194 1066023943  99171
## - Seasons_Spring      1  14280929 1070562679  99207
## - Solar_Rad           1  16890758 1073172508  99228
## - Seasons_Summer      1  20283288 1076565038  99254
## - Seasons_Winter      1  23850562 1080132312  99282
## - Rainfall            1  34728982 1091010732  99367
## - Humid_Percent       1  77073736 1133355485  99688
## - Temp                1 160155008 1216436758 100285
## - Functioning_Day     1 210283580 1266565330 100626
## - Hour                1 217130821 1273412571 100672
##
## Step:  AIC=99094.21
## RBC ~ Hour + Temp + Humid_Percent + Wind_Speed + Solar_Rad +
##     Rainfall + Snowfall + Holiday + Functioning_Day + Day + Month +
##     Year + Seasons_Spring + Seasons_Summer + Seasons_Winter +
##     Weekday_Monday + Weekday_Saturday + Weekday_Sunday + Weekday_Thursday
+
##     Weekday_Tuesday
##
##                      Df Sum of Sq        RSS    AIC
## - Weekday_Thursday    1    212633 1056529371  99094
## <none>                             1056316738  99094
## - Weekday_Saturday    1    320109 1056636847  99095
## - Weekday_Tuesday     1    356744 1056673482  99095
## + Visibility          1     34988 1056281750  99096
## + Weekday_Wednesday   1      1534 1056315204  99096
## - Snowfall            1   1236674 1057553412  99102
## - Wind_Speed          1   1351702 1057668440  99103
## - Day                 1   1394644 1057711382  99103
## - Weekday_Monday      1   1840416 1058157154  99107
## - Holiday             1   4884972 1061201710  99131
## - Month               1   5878839 1062195577  99139
## - Year                1   8062643 1064379381  99156
## - Weekday_Sunday      1   9793874 1066110612  99170
```

```
## - Seasons_Spring     1  14520841 1070837579  99207
## - Solar_Rad          1  18213945 1074530683  99236
## - Seasons_Summer     1  20248977 1076565715  99252
## - Seasons_Winter     1  24132615 1080449353  99283
## - Rainfall           1  34903335 1091220073  99367
## - Humid_Percent      1 116847770 1173164508  99978
## - Temp               1 160162681 1216479419 100284
## - Functioning_Day    1 210285597 1266602335 100624
## - Hour               1 217938315 1274255053 100675
##
## Step:  AIC=99093.91
## RBC ~ Hour + Temp + Humid_Percent + Wind_Speed + Solar_Rad +
##     Rainfall + Snowfall + Holiday + Functioning_Day + Day + Month +
##     Year + Seasons_Spring + Seasons_Summer + Seasons_Winter +
##     Weekday_Monday + Weekday_Saturday + Weekday_Sunday + Weekday_Tuesday
##
##                      Df Sum of Sq        RSS    AIC
## - Weekday_Saturday   1    190973 1056720344  99093
## - Weekday_Tuesday    1    222799 1056752170  99094
## <none>                           1056529371  99094
## + Weekday_Thursday   1    212633 1056316738  99094
## + Weekday_Wednesday  1     38062 1056491310  99096
## + Visibility         1     34471 1056494900  99096
## - Snowfall           1   1255621 1057784992  99102
## - Wind_Speed         1   1338795 1057868166  99103
## - Day                1   1381446 1057910817  99103
## - Weekday_Monday     1   1627992 1058157363  99105
## - Holiday            1   4869278 1061398649  99131
## - Month              1   5898772 1062428144  99139
## - Year               1   8089011 1064618382  99156
## - Weekday_Sunday     1   9956638 1066486009  99171
## - Seasons_Spring     1  14556214 1071085585  99207
## - Solar_Rad          1  18255729 1074785100  99236
## - Seasons_Summer     1  20286997 1076816369  99252
## - Seasons_Winter     1  24156143 1080685514  99283
## - Rainfall           1  34991220 1091520591  99367
## - Humid_Percent      1 117434992 1173964363  99981
## - Temp               1 160437247 1216966619 100285
## - Functioning_Day    1 210283980 1266813351 100624
## - Hour               1 217813022 1274342393 100674
##
## Step:  AIC=99093.44
## RBC ~ Hour + Temp + Humid_Percent + Wind_Speed + Solar_Rad +
##     Rainfall + Snowfall + Holiday + Functioning_Day + Day + Month +
##     Year + Seasons_Spring + Seasons_Summer + Seasons_Winter +
##     Weekday_Monday + Weekday_Sunday + Weekday_Tuesday
##
##                      Df Sum of Sq        RSS    AIC
## - Weekday_Tuesday    1    141001 1056861345  99093
## <none>                           1056720344  99093
```

```
## + Weekday_Saturday   1    190973 1056529371  99094
## + Weekday_Wednesday  1    109006 1056611338  99095
## + Weekday_Thursday   1     83498 1056636847  99095
## + Visibility         1     41709 1056678635  99095
## - Snowfall           1   1258595 1057978940  99101
## - Wind_Speed         1   1333105 1058053449  99102
## - Day                1   1374354 1058094698  99102
## - Weekday_Monday     1   1450896 1058171241  99103
## - Holiday            1   4792853 1061513197  99130
## - Month              1   5821751 1062542096  99138
## - Year               1   7993228 1064713573  99155
## - Weekday_Sunday     1   9910637 1066630982  99170
## - Seasons_Spring     1  14440142 1071160487  99206
## - Solar_Rad          1  18222072 1074942417  99236
## - Seasons_Summer     1  20146404 1076866749  99251
## - Seasons_Winter     1  24031059 1080751403  99281
## - Rainfall           1  34991434 1091711778  99366
## - Humid_Percent      1 117260307 1173980651  99979
## - Temp               1 160246395 1216966739 100283
## - Functioning_Day    1 210093798 1266814142 100622
## - Hour               1 217998217 1274718562 100674
##
## Step:  AIC=99092.56
## RBC ~ Hour + Temp + Humid_Percent + Wind_Speed + Solar_Rad +
##     Rainfall + Snowfall + Holiday + Functioning_Day + Day + Month +
##     Year + Seasons_Spring + Seasons_Summer + Seasons_Winter +
##     Weekday_Monday + Weekday_Sunday
##
##                     Df Sum of Sq        RSS    AIC
## <none>                           1056861345  99093
## + Weekday_Wednesday  1    170951 1056690394  99093
## + Weekday_Tuesday    1    141001 1056720344  99093
## + Weekday_Saturday   1    109175 1056752170  99094
## + Visibility         1     38899 1056822446  99094
## + Weekday_Thursday   1     35340 1056826005  99094
## - Snowfall           1   1241300 1058102645  99100
## - Weekday_Monday     1   1331135 1058192480  99101
## - Wind_Speed         1   1347830 1058209174  99101
## - Day                1   1378322 1058239667  99102
## - Holiday            1   4901371 1061762716  99130
## - Month              1   5825406 1062686751  99137
## - Year               1   8009950 1064871295  99154
## - Weekday_Sunday     1   9833954 1066695299  99169
## - Seasons_Spring     1  14465875 1071327220  99205
## - Solar_Rad          1  18142674 1075004019  99234
## - Seasons_Summer     1  20168735 1077030080  99250
## - Seasons_Winter     1  24075850 1080937195  99281
## - Rainfall           1  35100204 1091961549  99366
## - Humid_Percent      1 117122330 1173983675  99977
## - Temp               1 160115150 1216976494 100281
```

```
## - Functioning_Day    1 212495085 1269356430 100637
## - Hour               1 218110622 1274971967 100674
```

```
summary(stepwise_model)
```

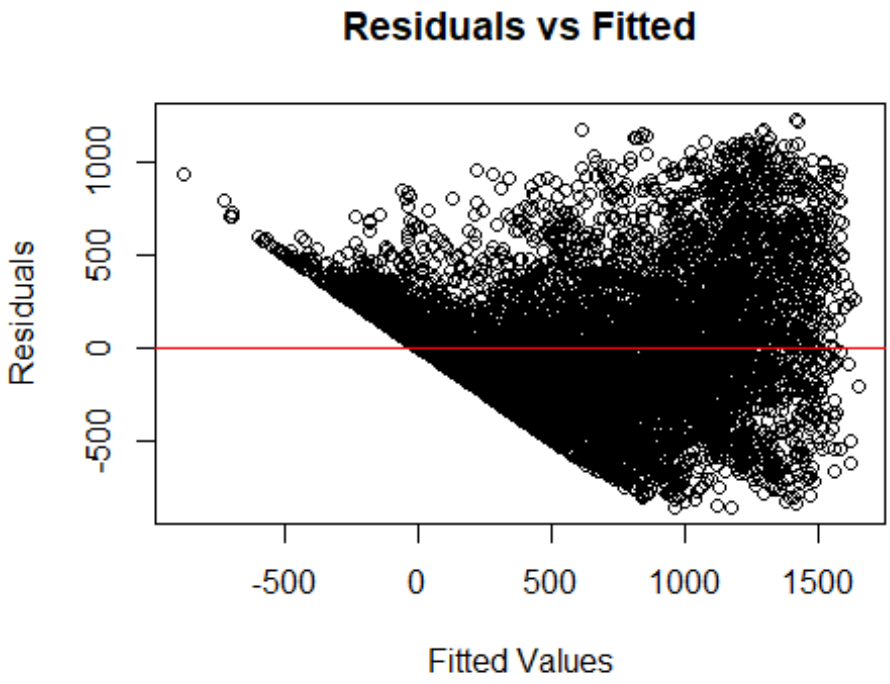
```
##
## Call:
## lm(formula = RBC ~ Hour + Temp + Humid_Percent + Wind_Speed +
##     Solar_Rad + Rainfall + Snowfall + Holiday + Functioning_Day +
##     Day + Month + Year + Seasons_Spring + Seasons_Summer + Seasons_Winter
+
##     Weekday_Monday + Weekday_Sunday, data = seoul_bike_data_clean)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -861.31 -243.34  -34.65  202.71 1225.32
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)      9.373e+05  1.174e+05   7.987 1.56e-15 ***
## Hour             2.531e+01  6.072e-01  41.688  < 2e-16 ***
## Temp             2.617e+01  7.326e-01  35.718  < 2e-16 ***
## Humid_Percent   -7.897e+00  2.585e-01 -30.549  < 2e-16 ***
## Wind_Speed       1.392e+01  4.246e+00   3.277 0.001053 **
## Solar_Rad       -7.294e+01  6.066e+00 -12.023  < 2e-16 ***
## Rainfall        -6.802e+01  4.067e+00 -16.724  < 2e-16 ***
## Snowfall         2.916e+01  9.271e+00   3.145 0.001667 **
## Holiday         -1.119e+02  1.791e+01  -6.249 4.32e-10 ***
## Functioning_Day  9.092e+02  2.209e+01  41.148  < 2e-16 ***
## Day             -1.472e+00  4.443e-01  -3.314 0.000924 ***
## Month           -3.617e+01  5.308e+00  -6.813 1.02e-11 ***
## Year            -4.644e+02  5.813e+01  -7.989 1.54e-15 ***
## Seasons_Spring  -3.636e+02  3.387e+01 -10.736  < 2e-16 ***
## Seasons_Summer  -2.765e+02  2.181e+01 -12.677  < 2e-16 ***
## Seasons_Winter  -6.604e+02  4.768e+01 -13.850  < 2e-16 ***
## Weekday_Monday  -3.679e+01  1.130e+01  -3.257 0.001131 **
## Weekday_Sunday  -9.883e+01  1.116e+01  -8.852  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 354.3 on 8421 degrees of freedom
## Multiple R-squared:  0.6184, Adjusted R-squared:  0.6177
## F-statistic: 802.8 on 17 and 8421 DF,  p-value: < 2.2e-16
```

```
##In stepwise selection, the significant predictors to include are: Hour,
Temp, Humid_Percent, Visibility, DPT, Rainfall, Holiday, Functioning_Day,
Month, Year, Seasons_Spring, Seasons_Summer, Seasons_Winter, Weekday_Monday,
Weekday_Saturday, Weekday_Sunday.
head(seoul_bike_data_clean)
```

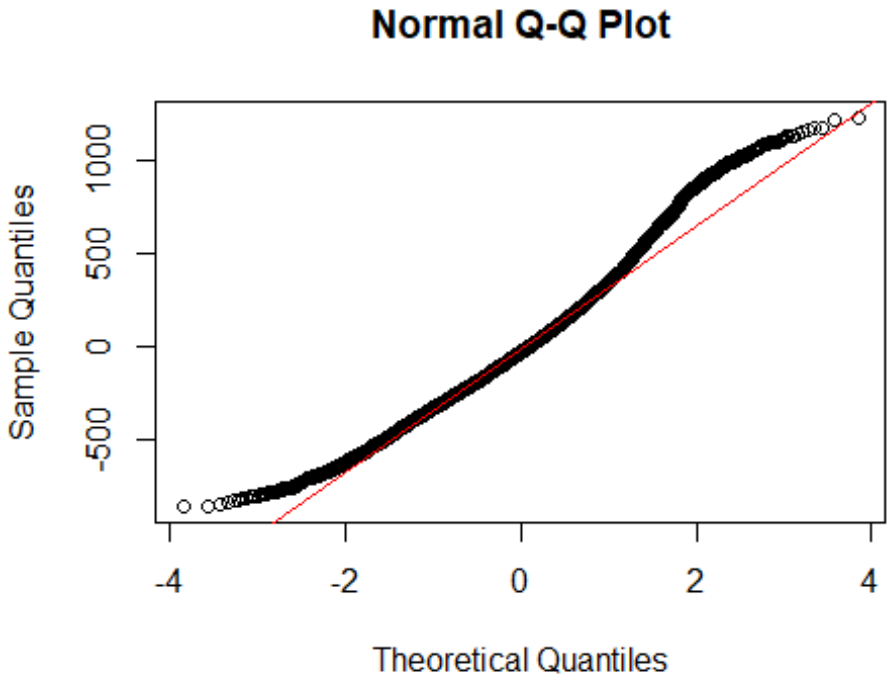
```
##   RBC Hour Temp Humid_Percent Wind_Speed Visibility Solar_Rad Rainfall
Snowfall
## 1 254    0 -5.2            37        2.2       2000         0        0
0
## 2 204    1 -5.5            38        0.8       2000         0        0
0
## 3 173    2 -6.0            39        1.0       2000         0        0
0
## 4 107    3 -6.2            40        0.9       2000         0        0
0
## 5  78    4 -6.0            36        2.3       2000         0        0
0
## 6 100    5 -6.4            37        1.5       2000         0        0
0
##   Holiday Functioning_Day Day Month Year Seasons_Spring Seasons_Summer
## 1       0               1   1    12 2017              0              0
## 2       0               1   1    12 2017              0              0
## 3       0               1   1    12 2017              0              0
## 4       0               1   1    12 2017              0              0
## 5       0               1   1    12 2017              0              0
## 6       0               1   1    12 2017              0              0
##   Seasons_Winter Weekday_Monday Weekday_Saturday Weekday_Sunday
## 1              1              0                 0              0
## 2              1              0                 0              0
## 3              1              0                 0              0
## 4              1              0                 0              0
## 5              1              0                 0              0
## 6              1              0                 0              0
##   Weekday_Thursday Weekday_Tuesday Weekday_Wednesday
## 1                0               0                 0
## 2                0               0                 0
## 3                0               0                 0
## 4                0               0                 0
## 5                0               0                 0
## 6                0               0                 0
```

```r
# Specify the columns we want to keep
columns_to_keep = c("RBC", "Hour", "Temp", "Humid_Percent", "Wind_Speed",
"Solar_Rad", "Rainfall", "Snowfall",
                    "Holiday", "Functioning_Day", "Day", "Month", "Year",
"Seasons_Spring", "Seasons_Summer",
                    "Seasons_Winter", "Weekday_Monday", "Weekday_Sunday")

seoul_bike_data_clean <- seoul_bike_data_clean[, columns_to_keep]


full_model = lm(RBC ~ ., data = seoul_bike_data_clean)
summary(full_model)
```

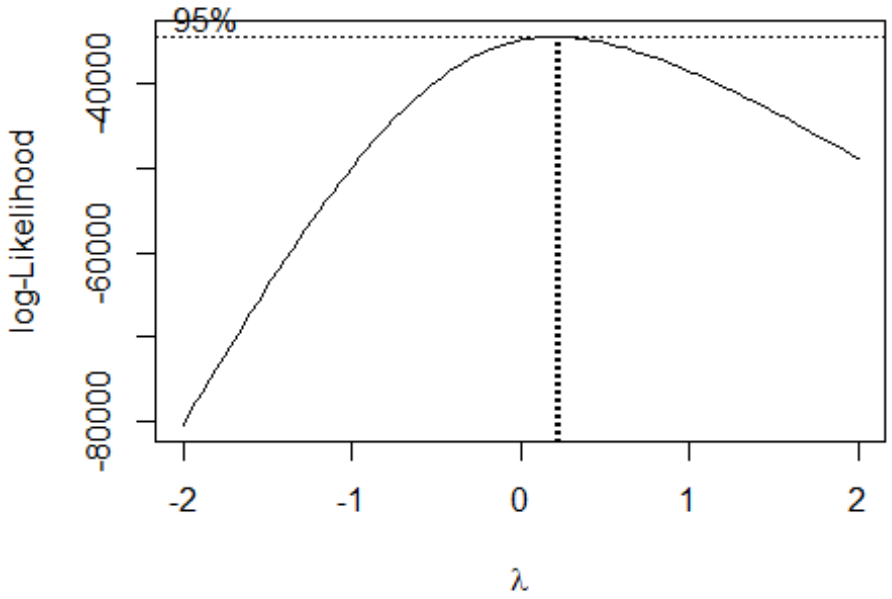
```
##
## Call:
## lm(formula = RBC ~ ., data = seoul_bike_data_clean)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -861.31 -243.34  -34.65  202.71 1225.32
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)      9.373e+05  1.174e+05   7.987 1.56e-15 ***
## Hour             2.531e+01  6.072e-01  41.688  < 2e-16 ***
## Temp             2.617e+01  7.326e-01  35.718  < 2e-16 ***
## Humid_Percent   -7.897e+00  2.585e-01 -30.549  < 2e-16 ***
## Wind_Speed       1.392e+01  4.246e+00   3.277 0.001053 **
## Solar_Rad       -7.294e+01  6.066e+00 -12.023  < 2e-16 ***
## Rainfall        -6.802e+01  4.067e+00 -16.724  < 2e-16 ***
## Snowfall         2.916e+01  9.271e+00   3.145 0.001667 **
## Holiday         -1.119e+02  1.791e+01  -6.249 4.32e-10 ***
## Functioning_Day  9.092e+02  2.209e+01  41.148  < 2e-16 ***
## Day             -1.472e+00  4.443e-01  -3.314 0.000924 ***
## Month           -3.617e+01  5.308e+00  -6.813 1.02e-11 ***
## Year            -4.644e+02  5.813e+01  -7.989 1.54e-15 ***
## Seasons_Spring  -3.636e+02  3.387e+01 -10.736  < 2e-16 ***
## Seasons_Summer  -2.765e+02  2.181e+01 -12.677  < 2e-16 ***
## Seasons_Winter  -6.604e+02  4.768e+01 -13.850  < 2e-16 ***
## Weekday_Monday  -3.679e+01  1.130e+01  -3.257 0.001131 **
## Weekday_Sunday  -9.883e+01  1.116e+01  -8.852  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 354.3 on 8421 degrees of freedom
## Multiple R-squared:  0.6184, Adjusted R-squared:  0.6177
## F-statistic: 802.8 on 17 and 8421 DF,  p-value: < 2.2e-16

residuals <- residuals(full_model)
fitted_values <- fitted(full_model)

plot(fitted_values, residuals, main = "Residuals vs Fitted",
     xlab = "Fitted Values", ylab = "Residuals")
abline(h = 0, col = "red")
```




```r
# QQ Plot
qqnorm(residuals)
qqline(residuals, col = "red")
```



```
##Box-Cox Transformation
seoul_bike_data_clean$RBC_shifted = seoul_bike_data_clean$RBC +
abs(min(seoul_bike_data_clean$RBC)) + 1
full_model_shifted = lm(RBC_shifted ~ Hour + Temp + Humid_Percent +
Wind_Speed + Solar_Rad + Rainfall +
                          Snowfall + Holiday + Functioning_Day + Day + Month
+ Year +
                          Seasons_Spring + Seasons_Summer + Seasons_Winter +
Weekday_Monday + Weekday_Sunday,
                        data = seoul_bike_data_clean)
boxcox_transformation_shifted = boxcox(full_model_shifted, plotit = TRUE)
```



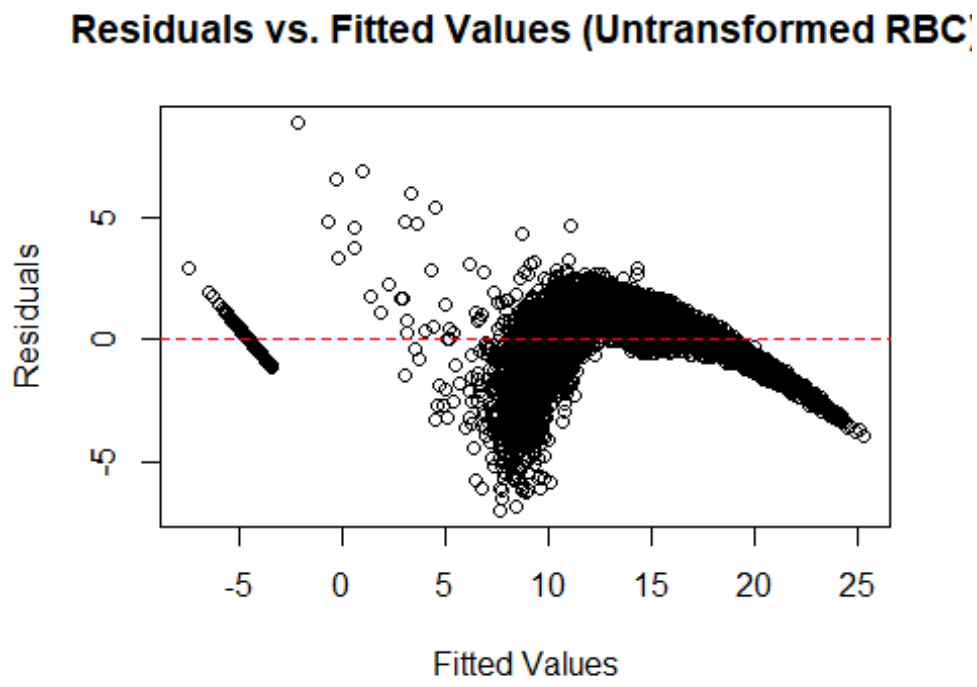

```
optimal_lambda_shifted =
boxcox_transformation_shifted$x[which.max(boxcox_transformation_shifted$y)]
print(paste("The optimal lambda for the shifted response is:",
optimal_lambda_shifted))

## [1] "The optimal lambda for the shifted response is: 0.222222222222222"

seoul_bike_data_clean$RBC_transformed = (seoul_bike_data_clean$RBC^0.222 - 1)
/ 0.222
head(seoul_bike_data_clean)

##   RBC Hour Temp Humid_Percent Wind_Speed Solar_Rad Rainfall Snowfall
Holiday
## 1 254    0 -5.2            37        2.2         0        0        0
0
```

```
## 2 204    1 -5.5            38        0.8         0        0        0
0
## 3 173    2 -6.0            39        1.0         0        0        0
0
## 4 107    3 -6.2            40        0.9         0        0        0
0
## 5  78    4 -6.0            36        2.3         0        0        0
0
## 6 100    5 -6.4            37        1.5         0        0        0
0
##   Functioning_Day Day Month Year Seasons_Spring Seasons_Summer
Seasons_Winter
## 1               1   1    12 2017              0              0
1
## 2               1   1    12 2017              0              0
1
## 3               1   1    12 2017              0              0
1
## 4               1   1    12 2017              0              0
1
## 5               1   1    12 2017              0              0
1
## 6               1   1    12 2017              0              0
1
##   Weekday_Monday Weekday_Sunday RBC_shifted RBC_transformed
## 1              0              0         255       10.895474
## 2              0              0         205       10.163969
## 3              0              0         174        9.636923
## 4              0              0         108        8.206218
## 5              0              0          79        7.344777
## 6              0              0         101        8.016726
```

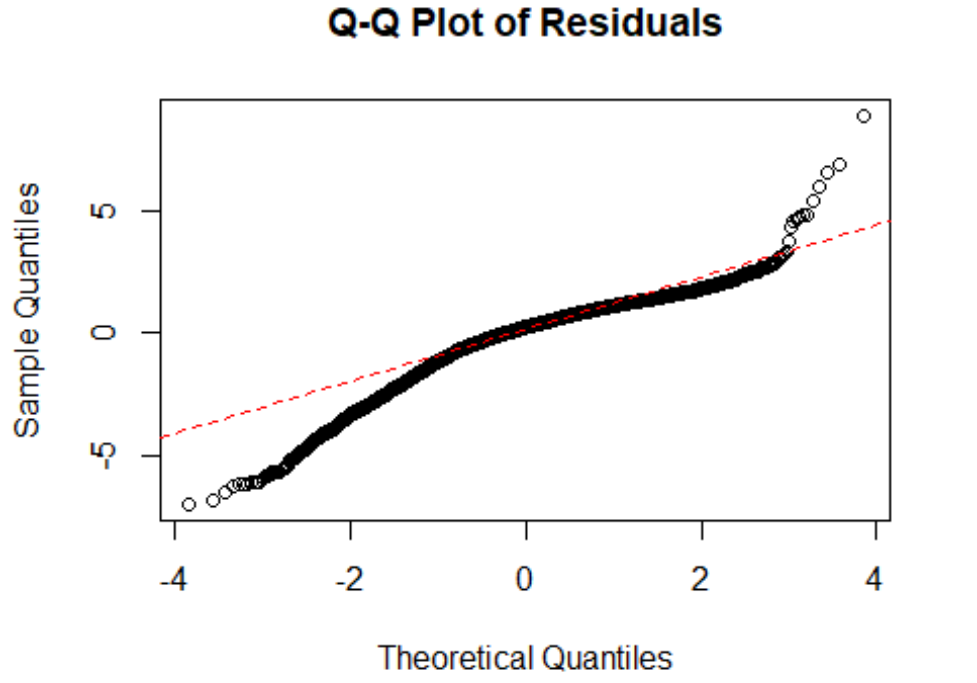
```
trans_model = lm(RBC_transformed ~ . -RBC, data = seoul_bike_data_clean)
summary(trans_model)
```

```
##
## Call:
## lm(formula = RBC_transformed ~ . - RBC, data = seoul_bike_data_clean)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -6.9500 -0.5451  0.2537  0.8739  8.8557
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)     -3.557e+01  4.333e+02  -0.082  0.93459
## Hour             2.220e-02  2.454e-03   9.047  < 2e-16 ***
## Temp             2.143e-02  2.892e-03   7.411 1.37e-13 ***
## Humid_Percent   -1.532e-02  1.002e-03 -15.288  < 2e-16 ***
## Wind_Speed      -8.301e-02  1.563e-02  -5.311 1.12e-07 ***
```

```
## Solar_Rad         2.147e-01  2.251e-02   9.537  < 2e-16 ***
## Rainfall         -5.012e-01  1.521e-02 -32.954  < 2e-16 ***
## Snowfall         -2.383e-01  3.412e-02  -6.985 3.07e-12 ***
## Holiday          -5.713e-01  6.605e-02  -8.651  < 2e-16 ***
## Functioning_Day   1.451e+01  8.908e-02 162.930  < 2e-16 ***
## Day               2.824e-03  1.636e-03   1.726  0.08435 .
## Month             5.288e-02  1.958e-02   2.701  0.00694 **
## Year              1.536e-02  2.146e-01   0.072  0.94295
## Seasons_Spring   -1.153e-01  1.254e-01  -0.919  0.35795
## Seasons_Summer   -5.566e-02  8.099e-02  -0.687  0.49195
## Seasons_Winter   -7.806e-01  1.774e-01  -4.400 1.09e-05 ***
## Weekday_Monday   -1.828e-01  4.158e-02  -4.396 1.12e-05 ***
## Weekday_Sunday   -3.669e-01  4.126e-02  -8.892  < 2e-16 ***
## RBC_shifted       5.514e-03  4.009e-05 137.551  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.303 on 8420 degrees of freedom
## Multiple R-squared:  0.9362, Adjusted R-squared:  0.936
## F-statistic:  6862 on 18 and 8420 DF,  p-value: < 2.2e-16

##The R^2 of the model is significantly improved, lets check if the model
assumptions are satisfied.
##Plotting residuals vs. fitted values to check for model assumptions
plot(trans_model$fitted.values, residuals(trans_model),
     xlab = "Fitted Values",
     ylab = "Residuals",
     main = "Residuals vs. Fitted Values (Untransformed RBC)")
abline(h = 0, col = "red", lty = 2)
```




```r
##Q-Q plot
qqnorm(residuals(trans_model), main = "Q-Q Plot of Residuals")
qqline(residuals(trans_model), col = "red", lty = 2)  # Add a reference line
```



```r
##The transformation does not improve the model assumptions.

##Lets check for normality and EV using the tests for the transformed model
bp_test_raw_2 = bptest(trans_model)
print(bp_test_raw_2)

##
##  studentized Breusch-Pagan test
##
## data:  trans_model
## BP = 1598.4, df = 18, p-value < 2.2e-16

residuals_2 = resid(trans_model)
ks_test_2 = ks.test(residuals_2, "pnorm", mean(residuals_2), sd(residuals_2))
print(ks_test_1)

##
##  Asymptotic one-sample Kolmogorov-Smirnov test
##
## data:  residuals
## D = 0.065323, p-value < 2.2e-16
## alternative hypothesis: two-sided

##The p-value (< 2.2e-16) from the KS test suggests that the residuals are
not normally distributed. This indicates that the normality assumption is
still violated, even after the transformation.
##The p-value from the BP test (< 2.2e-16) suggests that there is still
significant unequal variance in the residuals, meaning the assumption of
constant variance is still violated after transformation.
##The linear model appears to be a better fit for the data, providing clearer
results and better diagnostic metrics (e.g., residual standard error,
adjusted R-squared). The Poisson regression model is likely not suitable,
considering the infinite AIC and issues with convergence. Therefore, we
prefer the linear model for further analysis.


##Prefered model so far
seoul_bike_data_clean <- subset(seoul_bike_data_clean, select = -
c(RBC_transformed, RBC_shifted))
full_model = lm(RBC ~ ., data = seoul_bike_data_clean)
summary(full_model)

##
## Call:
## lm(formula = RBC ~ ., data = seoul_bike_data_clean)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -861.31 -243.34  -34.65  202.71 1225.32
##
## Coefficients:
```

```
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)      9.373e+05  1.174e+05   7.987 1.56e-15 ***
## Hour             2.531e+01  6.072e-01  41.688  < 2e-16 ***
## Temp             2.617e+01  7.326e-01  35.718  < 2e-16 ***
## Humid_Percent   -7.897e+00  2.585e-01 -30.549  < 2e-16 ***
## Wind_Speed       1.392e+01  4.246e+00   3.277 0.001053 **
## Solar_Rad       -7.294e+01  6.066e+00 -12.023  < 2e-16 ***
## Rainfall        -6.802e+01  4.067e+00 -16.724  < 2e-16 ***
## Snowfall         2.916e+01  9.271e+00   3.145 0.001667 **
## Holiday         -1.119e+02  1.791e+01  -6.249 4.32e-10 ***
## Functioning_Day  9.092e+02  2.209e+01  41.148  < 2e-16 ***
## Day             -1.472e+00  4.443e-01  -3.314 0.000924 ***
## Month           -3.617e+01  5.308e+00  -6.813 1.02e-11 ***
## Year            -4.644e+02  5.813e+01  -7.989 1.54e-15 ***
## Seasons_Spring  -3.636e+02  3.387e+01 -10.736  < 2e-16 ***
## Seasons_Summer  -2.765e+02  2.181e+01 -12.677  < 2e-16 ***
## Seasons_Winter  -6.604e+02  4.768e+01 -13.850  < 2e-16 ***
## Weekday_Monday  -3.679e+01  1.130e+01  -3.257 0.001131 **
## Weekday_Sunday  -9.883e+01  1.116e+01  -8.852  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 354.3 on 8421 degrees of freedom
## Multiple R-squared:  0.6184, Adjusted R-squared:  0.6177
## F-statistic: 802.8 on 17 and 8421 DF,  p-value: < 2.2e-16
```

```
lin_r2 = summary(full_model)$r.squared
print(lin_r2)
```

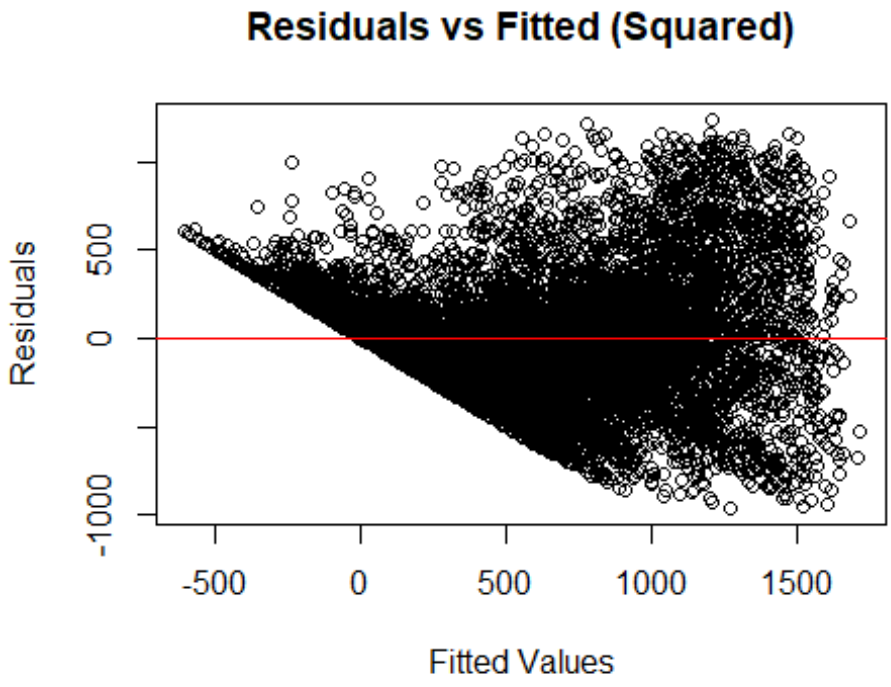
```
## [1] 0.6184307
```

```
# Make predictions
full_model_pred <- predict(full_model, newdata = seoul_bike_data_clean)

# Calculate residuals
full_model_residuals <- seoul_bike_data_clean$RBC - full_model_pred

# Calculate MSE
mse_full_model <- mean(full_model_residuals^2)

# Print the MSE
cat("MSE for the full model:", mse_full_model, "\n")
```

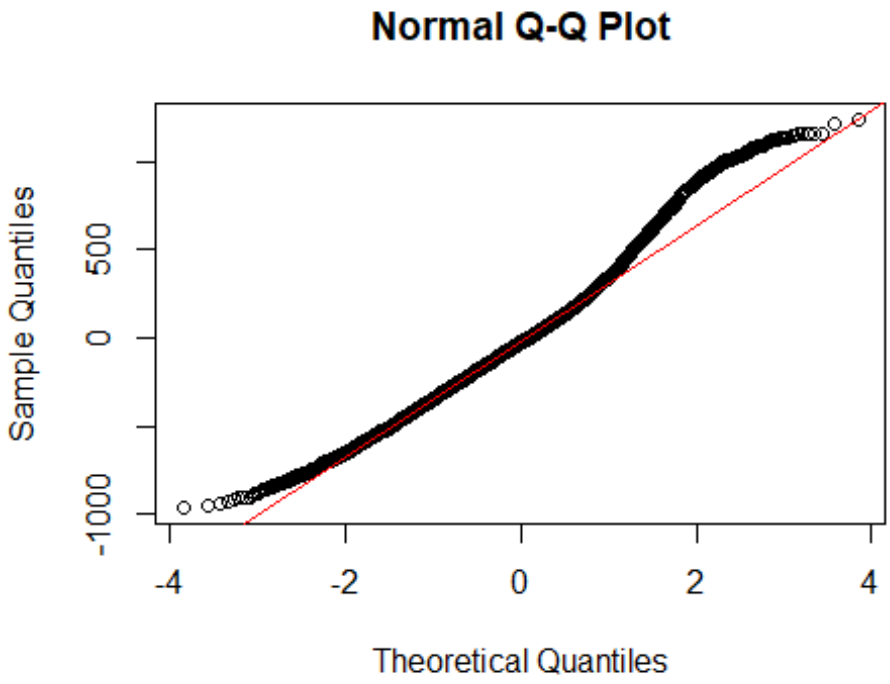
```
## MSE for the full model: 125235.4
```

```
# Apply square transformation to all numeric variables except the target
variable
seoul_bike_data_clean_sq <- seoul_bike_data_clean
seoul_bike_data_clean_sq[, -which(names(seoul_bike_data_clean) == "RBC")] <-
  seoul_bike_data_clean_sq[, -which(names(seoul_bike_data_clean) == "RBC")]^2
full_model_sq <- lm(RBC ~ ., data = seoul_bike_data_clean_sq)
```

```r
residuals_sq <- residuals(full_model_sq)
fitted_values_sq <- fitted(full_model_sq)
plot(fitted_values_sq, residuals_sq, main = "Residuals vs Fitted (Squared)",
     xlab = "Fitted Values", ylab = "Residuals")
abline(h = 0, col = "red")
```




```r
qqnorm(residuals_sq)
qqline(residuals_sq, col = "red")
```




```
summary(full_model_sq)

##
## Call:
## lm(formula = RBC ~ ., data = seoul_bike_data_clean_sq)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -962.86 -243.55  -28.24  199.11 1237.44
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)      7.153e+05  5.760e+04  12.418  < 2e-16 ***
## Hour             1.135e+00  2.526e-02  44.923  < 2e-16 ***
## Temp             5.995e-01  2.417e-02  24.804  < 2e-16 ***
## Humid_Percent   -6.679e-02  2.023e-03 -33.013  < 2e-16 ***
## Wind_Speed      -2.192e-01  9.084e-01  -0.241   0.8093
## Solar_Rad       -1.029e+01  2.238e+00  -4.597 4.34e-06 ***
## Rainfall        -2.215e+00  2.924e-01  -7.575 3.97e-14 ***
## Snowfall         2.973e+00  2.082e+00   1.428   0.1534
## Holiday         -8.004e+01  1.841e+01  -4.348 1.39e-05 ***
## Functioning_Day  8.781e+02  2.267e+01  38.733  < 2e-16 ***
## Day             -3.305e-02  1.399e-02  -2.363   0.0182 *
## Month           -4.154e+00  3.850e-01 -10.789  < 2e-16 ***
## Year            -1.756e-01  1.414e-02 -12.419  < 2e-16 ***
## Seasons_Spring  -4.950e+02  3.451e+01 -14.345  < 2e-16 ***
## Seasons_Summer  -3.510e+02  2.489e+01 -14.100  < 2e-16 ***
```

```
## Seasons_Winter  -1.044e+03  4.089e+01 -25.544  < 2e-16 ***
## Weekday_Monday  -2.695e+01  1.162e+01  -2.319   0.0204 *
## Weekday_Sunday  -8.538e+01  1.149e+01  -7.429 1.20e-13 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 364.6 on 8421 degrees of freedom
## Multiple R-squared:  0.5958, Adjusted R-squared:  0.5949
## F-statistic:   730 on 17 and 8421 DF,  p-value: < 2.2e-16
```

##Model performance is not good compared to the original model.

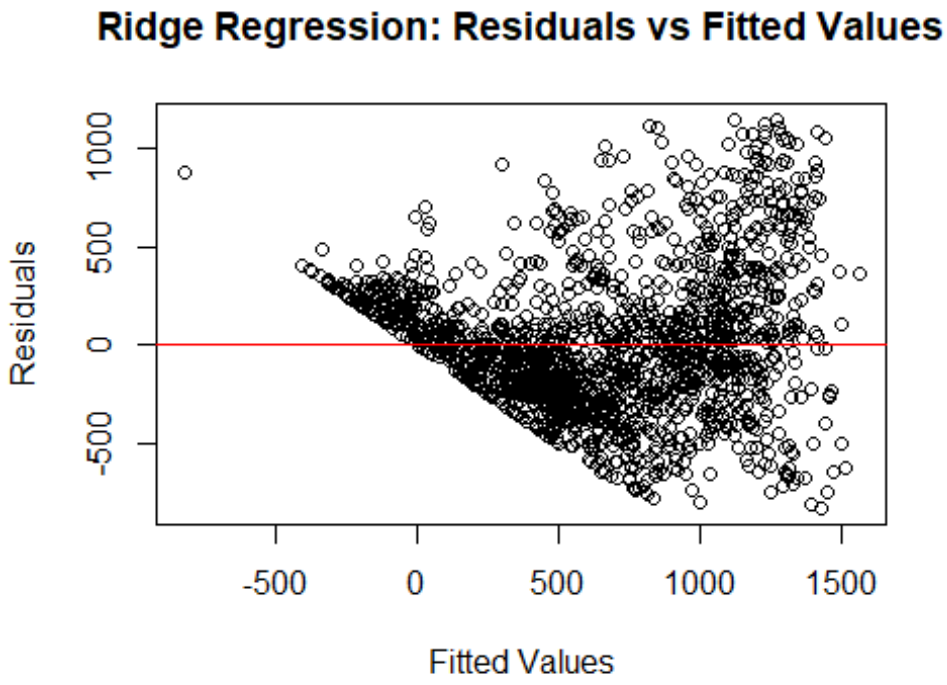
##Since the linear model hasn't satisfied assumptions even after transformations, let's explore alternative models.

```
# Set seed for reproducibility
set.seed(42)
trainIndex <- createDataPartition(seoul_bike_data_clean$RBC, p = 0.8,
                                  list = FALSE,
                                  times = 1)

train_data <- seoul_bike_data_clean[trainIndex, ]
test_data <- seoul_bike_data_clean[-trainIndex, ]

# Ridge Regression (alpha = 0)
X_train <- model.matrix(RBC ~ . -1, data = train_data)
y_train <- train_data$RBC
ridge_cv <- cv.glmnet(X_train, y_train, alpha = 0, standardize = TRUE)
ridge_best_lambda <- ridge_cv$lambda.min
ridge_model <- glmnet(X_train, y_train, alpha = 0, lambda =
ridge_best_lambda, standardize = TRUE)

# Predictions on test set
X_test <- model.matrix(RBC ~ . -1, data = test_data)
ridge_pred <- predict(ridge_model, s = ridge_best_lambda, newx = X_test)

# Evaluate Ridge
ridge_mse <- mean((test_data$RBC - ridge_pred)^2)
ridge_r2 <- 1 - sum((test_data$RBC - ridge_pred)^2) / sum((test_data$RBC -
mean(test_data$RBC))^2)

cat("Ridge Regression MSE on test data:", ridge_mse, "\n")

## Ridge Regression MSE on test data: 130393

cat("Ridge Regression R-squared on test data:", ridge_r2, "\n")
```

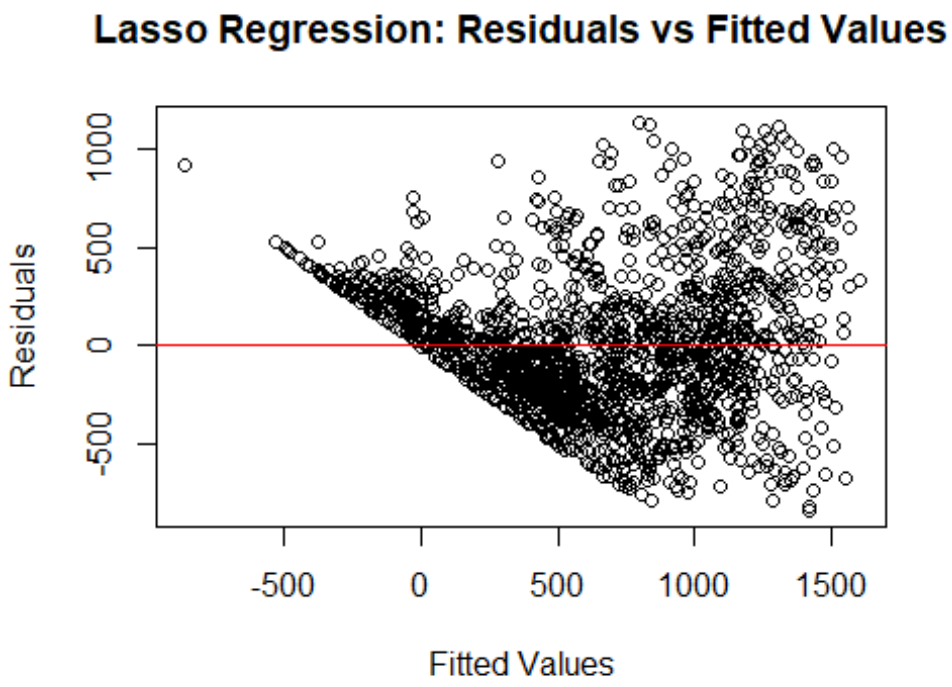
```
## Ridge Regression R-squared on test data: 0.6074424

# Lasso Regression (alpha = 1)
lasso_cv <- cv.glmnet(X_train, y_train, alpha = 1, standardize = TRUE)
lasso_best_lambda <- lasso_cv$lambda.min
lasso_model <- glmnet(X_train, y_train, alpha = 1, lambda =
lasso_best_lambda, standardize = TRUE)

# Predictions on test set
lasso_pred <- predict(lasso_model, s = lasso_best_lambda, newx = X_test)

# Evaluate Lasso
lasso_mse <- mean((test_data$RBC - lasso_pred)^2)
lasso_r2 <- 1 - sum((test_data$RBC - lasso_pred)^2) / sum((test_data$RBC -
mean(test_data$RBC))^2)

cat("Lasso Regression MSE on test data:", lasso_mse, "\n")

## Lasso Regression MSE on test data: 127368

cat("Lasso Regression R-squared on test data:", lasso_r2, "\n")

## Lasso Regression R-squared on test data: 0.6165493

# Random Forest
rf_model <- randomForest(RBC ~ ., data = train_data, ntree = 500, mtry = 3,
importance = TRUE)

# Predictions on test set
rf_predictions <- predict(rf_model, newdata = test_data)

# Evaluate Random Forest
rf_mse <- mean((test_data$RBC - rf_predictions)^2)
rf_r2 <- 1 - sum((test_data$RBC - rf_predictions)^2) / sum((test_data$RBC -
mean(test_data$RBC))^2)

cat("Random Forest MSE on test data:", rf_mse, "\n")

## Random Forest MSE on test data: 40938.36

cat("Random Forest R-squared on test data:", rf_r2, "\n")

## Random Forest R-squared on test data: 0.8767521

# XGBoost
library(xgboost)
dtrain <- xgb.DMatrix(data = as.matrix(train_data[, -which(names(train_data)
== "RBC")]),
                      label = train_data$RBC)
dtest <- xgb.DMatrix(data = as.matrix(test_data[, -which(names(test_data) ==
"RBC")]),
```

```
                      label = test_data$RBC)

params <- list(
  objective = "reg:squarederror", # For regression problems
  max_depth = 6,
  eta = 0.1,
  subsample = 0.8,
  colsample_bytree = 0.8,
  eval_metric = "rmse"
)

# Train the XGBoost model
xgb_model <- xgb.train(params = params, data = dtrain, nrounds = 500)

# Predictions on test set
xgb_predictions <- predict(xgb_model, newdata = dtest)

# Evaluate XGBoost
xgb_mse <- mean((test_data$RBC - xgb_predictions)^2)
xgb_r2 <- 1 - sum((test_data$RBC - xgb_predictions)^2) / sum((test_data$RBC -
mean(test_data$RBC))^2)

cat("XGBoost MSE on test data:", xgb_mse, "\n")

## XGBoost MSE on test data: 24433.74

cat("XGBoost R-squared on test data:", xgb_r2, "\n")

## XGBoost R-squared on test data: 0.9264404

# Comparison of all models
cat("\nModel Comparison:\n")

##
## Model Comparison:

cat("Linear Regression MSE:", mse_full_model, "R-squared:", lin_r2, "\n")

## Linear Regression MSE: 125235.4 R-squared: 0.6184307

cat("Ridge Regression MSE:", ridge_mse, ", R-squared:", ridge_r2, "\n")

## Ridge Regression MSE: 130393 , R-squared: 0.6074424

cat("Lasso Regression MSE:", lasso_mse, ", R-squared:", lasso_r2, "\n")

## Lasso Regression MSE: 127368 , R-squared: 0.6165493

cat("Random Forest MSE:", rf_mse, ", R-squared:", rf_r2, "\n")

## Random Forest MSE: 40938.36 , R-squared: 0.8767521
```

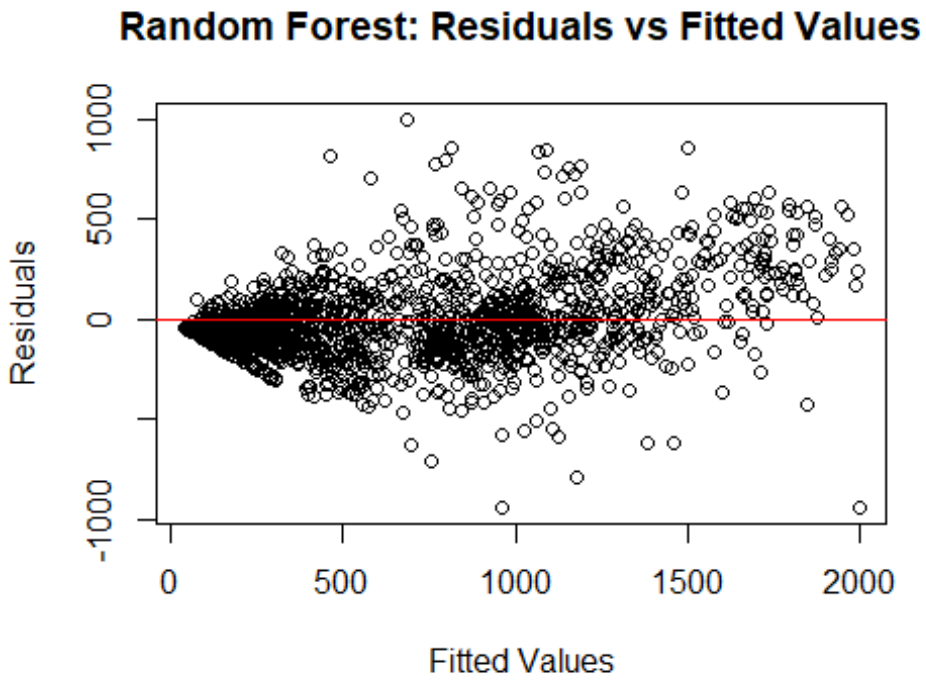
```r
cat("XGBoost MSE:", xgb_mse, ", R-squared:", xgb_r2, "\n")

## XGBoost MSE: 24433.74 , R-squared: 0.9264404

# Calculate Residuals for each model
ridge_residuals <- test_data$RBC - ridge_pred
lasso_residuals <- test_data$RBC - lasso_pred
rf_residuals <- test_data$RBC - rf_predictions
xgb_residuals <- test_data$RBC - xgb_predictions

# For Ridge Regression
ridge_fitted <- predict(ridge_model, s = ridge_best_lambda, newx = X_test)
ridge_residuals <- test_data$RBC - ridge_fitted
plot(ridge_fitted, ridge_residuals, main = "Ridge Regression: Residuals vs
Fitted Values",
     xlab = "Fitted Values", ylab = "Residuals", col = "black")
abline(h = 0, col = "red")
```



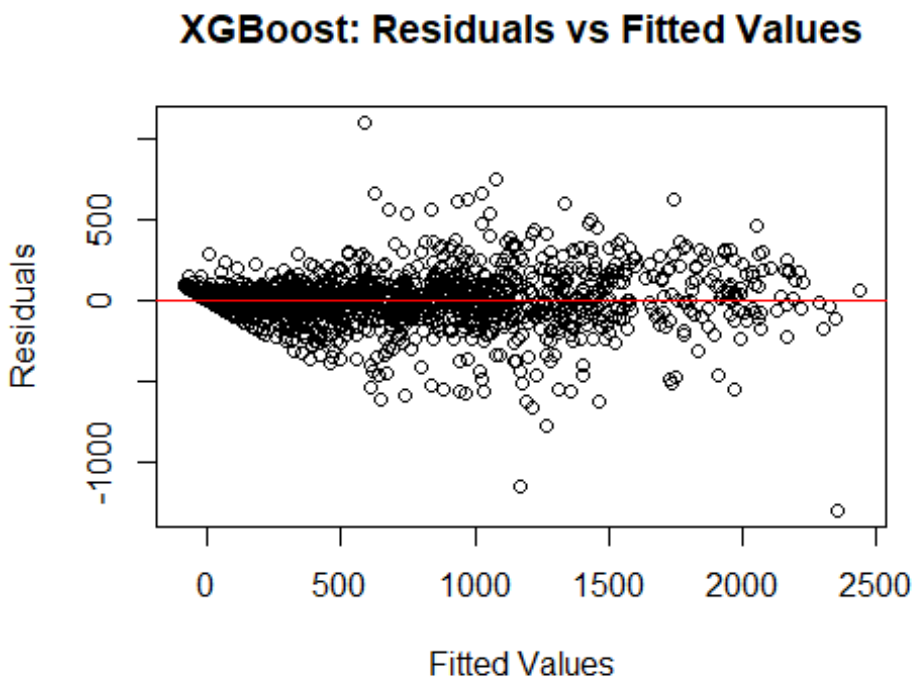

```r
# For Lasso Regression
lasso_fitted <- predict(lasso_model, s = lasso_best_lambda, newx = X_test)
lasso_residuals <- test_data$RBC - lasso_fitted
plot(lasso_fitted, lasso_residuals, main = "Lasso Regression: Residuals vs
Fitted Values",
     xlab = "Fitted Values", ylab = "Residuals", col = "black")
abline(h = 0, col = "red")
```




```r
# For Random Forest
rf_fitted <- rf_predictions
rf_residuals <- test_data$RBC - rf_fitted
plot(rf_fitted, rf_residuals, main = "Random Forest: Residuals vs Fitted
Values",
     xlab = "Fitted Values", ylab = "Residuals", col = "black")
abline(h = 0, col = "red")
```




```r
# For XGBoost
xgb_fitted <- xgb_predictions
xgb_residuals <- test_data$RBC - xgb_fitted
plot(xgb_fitted, xgb_residuals, main = "XGBoost: Residuals vs Fitted Values",
     xlab = "Fitted Values", ylab = "Residuals", col = "black")
abline(h = 0, col = "red")
```




```
##In this analysis, several models were evaluated to predict the target
variable, with performance measured using Mean Squared Error (MSE) and R-
squared. Below is a summary of the results for each model:

##Linear Regression:
##MSE: 125,235.4
##R-squared: 0.618
##The linear regression model demonstrates a moderate level of performance,
with a fairly high MSE and an R-squared value that indicates it explains
about 61.8% of the variance in the data. Although it provides a baseline
model, its predictive power is relatively limited compared to more complex
models.

##Ridge Regression:
#3MSE: 130,393
##R-squared: 0.607
##Ridge regression, which includes L2 regularization, results in slightly
worse performance compared to linear regression. With a higher MSE and a
slightly lower R-squared (60.7%), it appears that the regularization does not
significantly improve model accuracy in this case.

##Lasso Regression:
#3MSE: 127,368
##R-squared: 0.617
##Lasso regression, which incorporates L1 regularization, shows performance
similar to ridge regression. Its MSE is slightly lower than that of ridge
```

regression, but still relatively high compared to more complex models. The R-squared value (61.7%) is comparable to linear regression.

##Random Forest:
##MSE: 40,938.36
##R-squared: 0.877
##Random Forest significantly outperforms the simpler regression models, with a much lower MSE and an R-squared value of 0.877. This indicates that the Random Forest model is able to capture more of the data's variance and make more accurate predictions.

##XGBoost:
##MSE: 24,433.74
##R-squared: 0.926
##The XGBoost model achieves the best results among all evaluated models. With the lowest MSE and the highest R-squared (92.6%), XGBoost demonstrates the highest predictive accuracy, making it the most effective model for this task.

##Conclusion
##Based on the performance metrics, it is evident that XGBoost is the most effective model in terms of minimizing error and explaining the variance in the data. It consistently outperforms all other models, including Random Forest, which also shows strong performance. In comparison, the Linear Regression, Ridge Regression, and Lasso Regression models fall short in both MSE and R-squared, indicating that they are less capable of accurately modeling the data. Moving forward, XGBoost would be the preferred model for achieving the best predictive performance.