# Seoul Bike Sharing Demand Analysis and Prediction

2022-08-05

---

## Introduction

### Description of the data file

We select a data file which contains count of public bikes rented at each hour in Seoul Bike Sharing System with the corresponding weather data and holidays information. It is a substantially large data file with 8760 observations and 14 variables. We are interested in using Rented.Bike.Count (a numeric variable) as our response variable and explore how other factors (a mix of categorical variables and continuous numeric variables) affect the count of bikes rented at each hour. Among the other 13 variables which we plan to use as potential predictors, we know from intuition that some may have more importance than others, like temperature, humidity, wind speed, visibility, seasons, and holiday, etc.

### Our Interest

This data set is interesting to us both personally and business-wise. Recently we have seen a rise in the delivery, accessibility, and usage of regular and electric rental bikes. There are clear environmental, health, and economical benefits associated with the usage of bikes as a mode of transportation. We would like to find out what factors lead to an increase in number of bikes rented and what factors have inverse effect on using rental bikes. Learning about such factors can help a bike rental business manage its inventory and supply without any hindrance. It can also help cities plan accordingly due to an increase of bikers, e.g. opening up more bike lanes during certain days or seasons. Environmentally, we will have a better understanding of the feasibility of turning a city into a "bike city" or looking at alternative options if a city is not friendly to bikers due to harsh weather conditions.

### Background information on the data set

The original data comes from http://data.seoul.go.kr. The holiday information comes from SOUTH KOREA PUBLIC HOLIDAYS. A clean version can be found at UCI Machine Learning Repository.

Attribute Information:

- Date : month/day/year
- Rented Bike count - Count of bikes rented at each hour
- Hour - Hour of the day
- Temperature - Temperature in Celsius
- Humidity - %
- Windspeed - m/s
- Visibility - 10m
- Dew point temperature - Celsius
- Solar radiation - MJ/m2
- Rainfall - mm

- Snowfall - cm
- Seasons - Winter, Spring, Summer, Autumn
- Holiday - Holiday, No holiday
- Functional Day - Functional or Non-functional days of rental bike system

## Methods

### Data in R

First of all, we loaded the data file in R. Since the column names in the original `csv` file contain measurement units (like `Wind speed (m/s), Solar Radiation (MJ/m2)`) and characters such as ° and %, we loaded the data using cleaned up column names.

We printed out the structure and first few rows of the data file below. We also confirmed that the data file doesn't have any null values.

```
library(formattable)
columns = c("Date","Rented","Hour","Temp","Humidity",
            "Wind","Visibility","Dew",
            "Radiation","Rain","Snow","Season","Holiday",
            "Functioning")
bike = read.csv("../data/SeoulBikeData.csv", col.names = columns)
str(bike)
```

```
## 'data.frame':    8760 obs. of  14 variables:
##  $ Date       : chr  "01/12/2017" "01/12/2017" "01/12/2017" "01/12/2017" ...
##  $ Rented     : 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 ...
##  $ Humidity   : int  37 38 39 40 36 37 35 38 37 27 ...
##  $ Wind       : 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 ...
##  $ Dew        : num  -17.6 -17.6 -17.7 -17.6 -18.6 -18.7 -19.5 -19.3 -19.8 -22.4 ...
##  $ Radiation  : num  0 0 0 0 0 0 0 0 0.01 0.23 ...
##  $ Rain       : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ Snow       : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ Season     : chr  "Winter" "Winter" "Winter" "Winter" ...
##  $ Holiday    : chr  "No Holiday" "No Holiday" "No Holiday" "No Holiday" ...
##  $ Functioning: chr  "Yes" "Yes" "Yes" "Yes" ...
```

```
formattable(head(bike[,1:10]), digits = 2)
```

Date

Rented

Hour

Temp

Humidity

Wind

Visibility

Dew

Radiation

Rain

01/12/2017

254

0

-5.2

37

2.2

2000

-18

0

0

01/12/2017

204

1

-5.5

38

0.8

2000

-18

0

0

01/12/2017

173

2

-6.0

39

1.0

2000

-18

0

0

01/12/2017

107

3

-6.2

40

0.9

2000

-18

0

0

01/12/2017

78

4

-6.0

36

2.3

2000

-19

0

0

01/12/2017

100

5

-6.4

37

1.5

2000

-19

0

0

```
is.null(bike)
```

```
## [1] FALSE
```

```
bike$Date    = as.Date(bike$Date, '%d/%m/%Y')
bike$Month   = as.numeric(format(bike$Date,'%m'))
bike$Weekday = weekdays(bike$Date, abbreviate = TRUE)
bike$Weekend = ifelse(bike$Weekday == 'Sat' | bike$Weekday =='Sun', "Yes", "No")

range(bike$Date)
```

```
## [1] "2017-12-01" "2018-11-30"
```

We then converted the Date variable into proper date format for R to work with. Then we checked the range of the dates in our data set, which is one year's data from 2017-12-01 to 2018-11-30. So we probably don't need the year variable here. But we created several other variables like month, weekday and weekend and think these variables will help us better understand the seasonality and weekly fluctuations in bike demand.

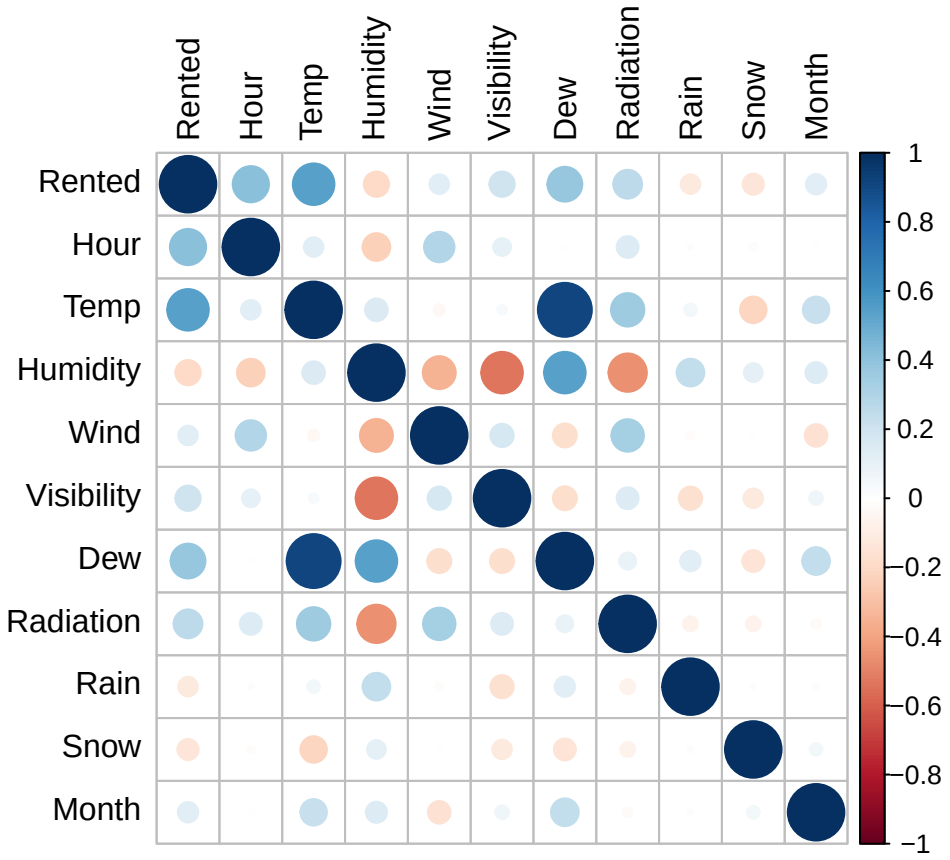
```
bike$Season      = as.factor(bike$Season)
bike$Holiday     = as.factor(bike$Holiday)
bike$Functioning = as.factor(bike$Functioning)
bike$Weekday     = as.factor(bike$Weekday)
bike$Weekend     = as.factor(bike$Weekend)

# Change the order of the Weekday factor variable
bike$Weekday = factor(bike$Weekday,
                      levels = c('Sun','Mon','Tue','Wed', 'Thu', 'Fri', 'Sat'))
```

We successfully coerced the categorical variables into factor variables to help with the data exploration and modeling process in the next steps.

### Exploratory data analysis

Before we start any modeling process, let's try to understand our data better and gain a more intuitive understanding about our variables.

```
library(corrplot)
bike_num = subset(bike, select = -c(Date, Season, Holiday, Functioning, Weekday, Weekend))
M = round(cor(bike_num), 2)
corrplot(M, tl.col = "black")
```




Looking at the pairwise correlation among the variables, we can see Temperature has the highest correlation with rented bike counts. Hour and Dew also have high correlation with the response variable.

```
library(corrr)
library(dplyr)

correlations = corrr::correlate(bike_num)
top_5 = head(dplyr::arrange(corrr::stretch(correlations, remove.dups = TRUE),
                            desc(r)), 5)
formattable(top_5, align=c('l', 'l', 'l'), digits = 2)
```

x

y

r

Temp

Dew

0.91

Rented

Temp

0.54

Humidity

Dew

0.54

Rented

Hour

0.41

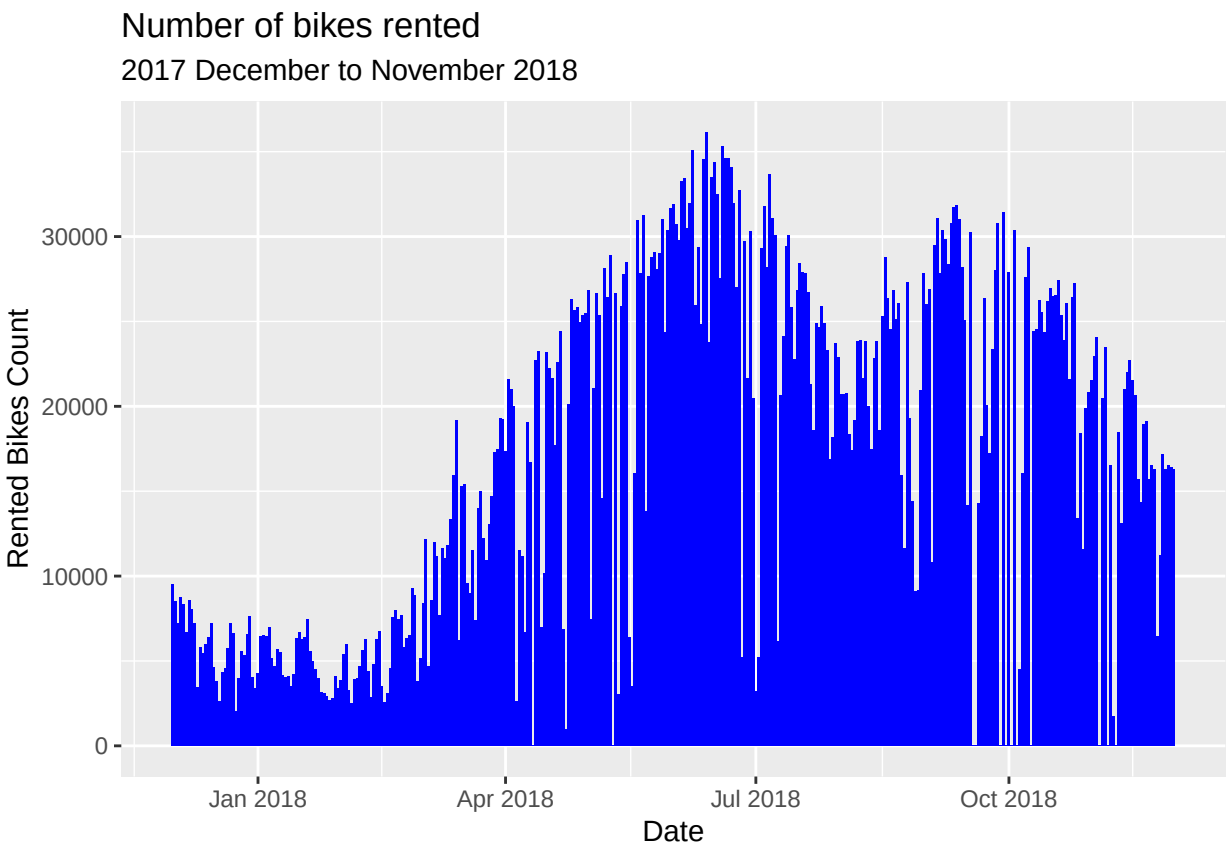
Rented

Dew

0.38

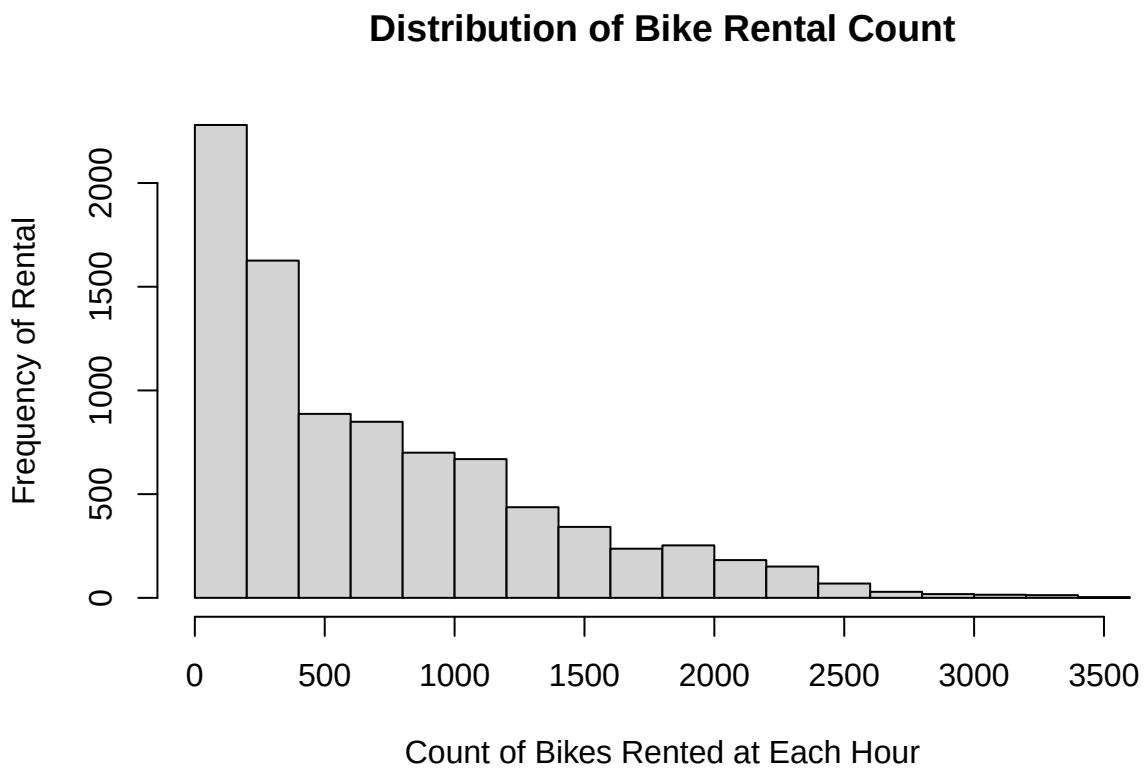
We printed out the top 5 highly correlated variables in the data set and can see we have some highly correlated variables in the data set, which could suggest multi-collinearity. We may want to address this later in the modeling process since we are interested in interpreting the coefficients.

```
library(ggplot2)
ggplot(data = bike, aes(x = Date, y = Rented)) +
       geom_bar(stat = "identity", fill = "blue") +
       labs(title = "Number of bikes rented ",
            subtitle = "2017 December to November 2018",
            x = "Date", y = "Rented Bikes Count")
```



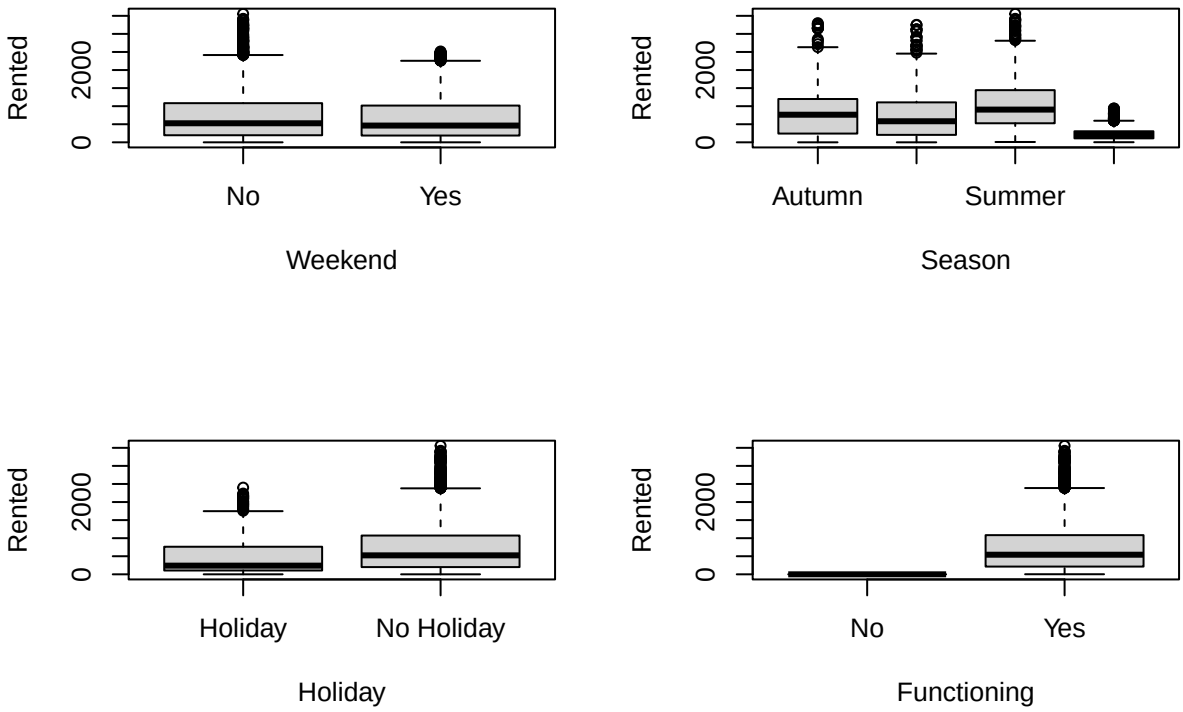

We can clearly see there is seasonality in the demand of rented bikes.

```
hist(bike$Rented,
     breaks = 25,
     ylab   = 'Frequency of Rental',
     xlab   = 'Count of Bikes Rented at Each Hour',
     main   = 'Distribution of Bike Rental Count')
```



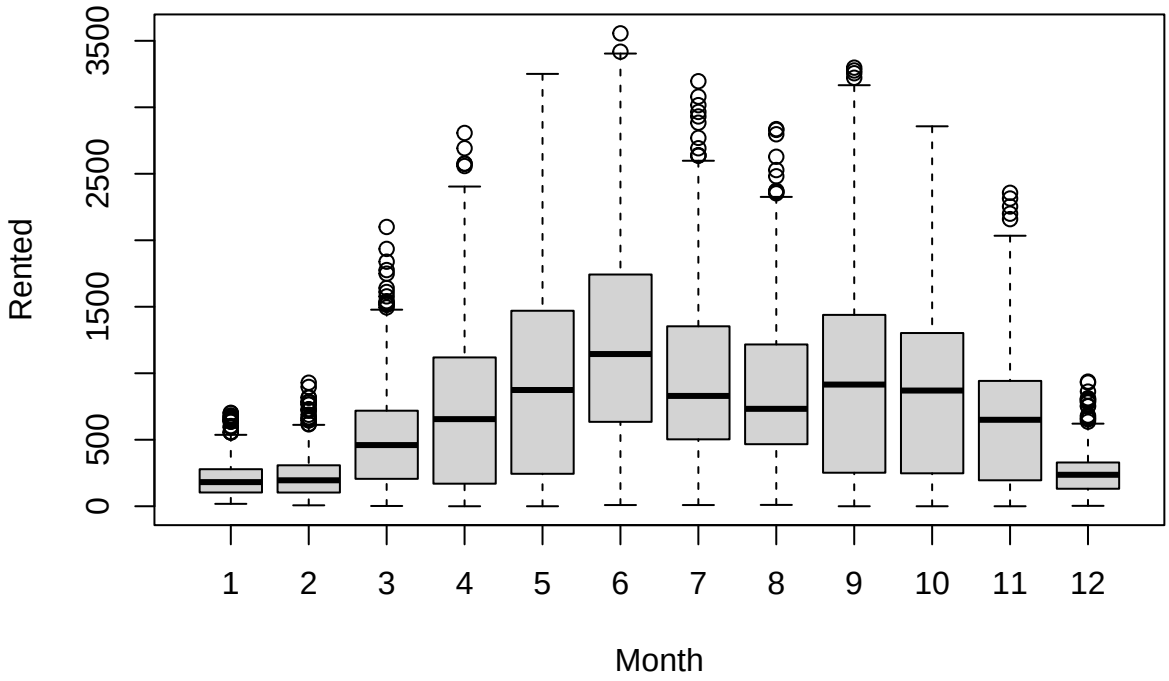

From the histogram of the response variable above, we can see the distribution is highly skewed, which means transformation may help our modeling process later.

```
par(mfrow=c(2, 2))

plot(Rented ~ Weekend,     data = bike)
plot(Rented ~ Season,      data = bike)
plot(Rented ~ Holiday,     data = bike)
plot(Rented ~ Functioning, data = bike)
```



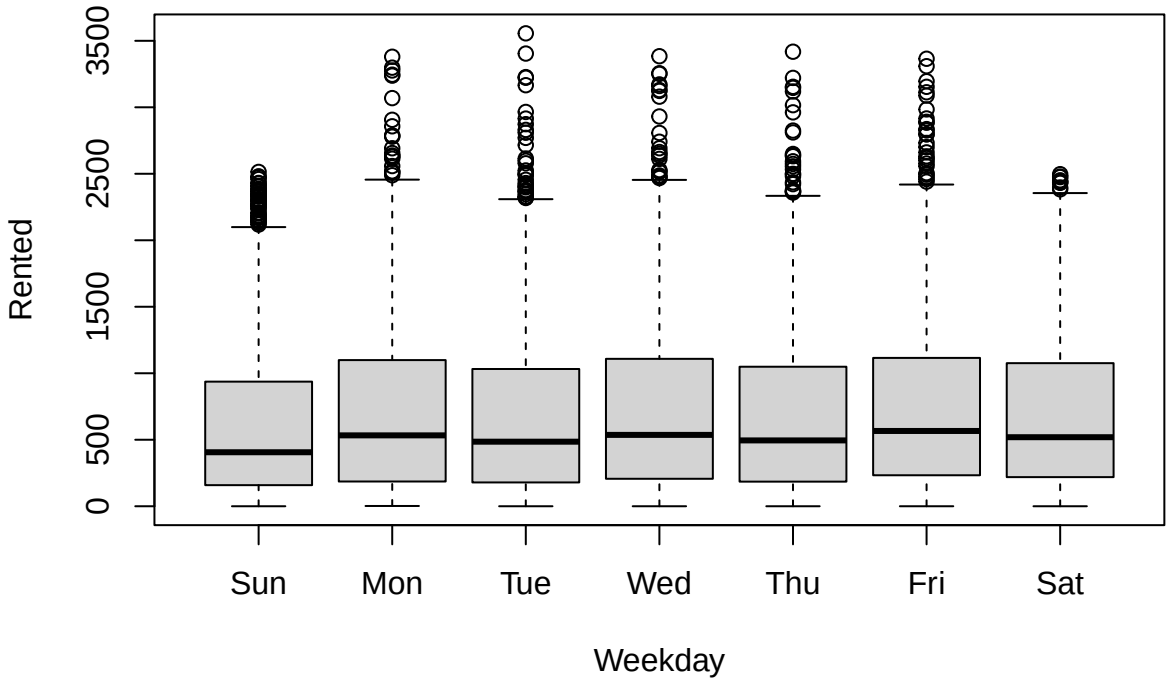

We can see we usually have higher rented bike counts on weekdays and non-holidays - perhaps more people use rental bikes as a commute method instead of using it for leisure purpose. We have highest rented bike counts during summer and lowest counts during winter, which makes sense. We actually don't have any rented bikes on non-functioning days of the rental bike system.

```
plot(Rented ~ as.factor(Month), xlab = "Month", data = bike)
```



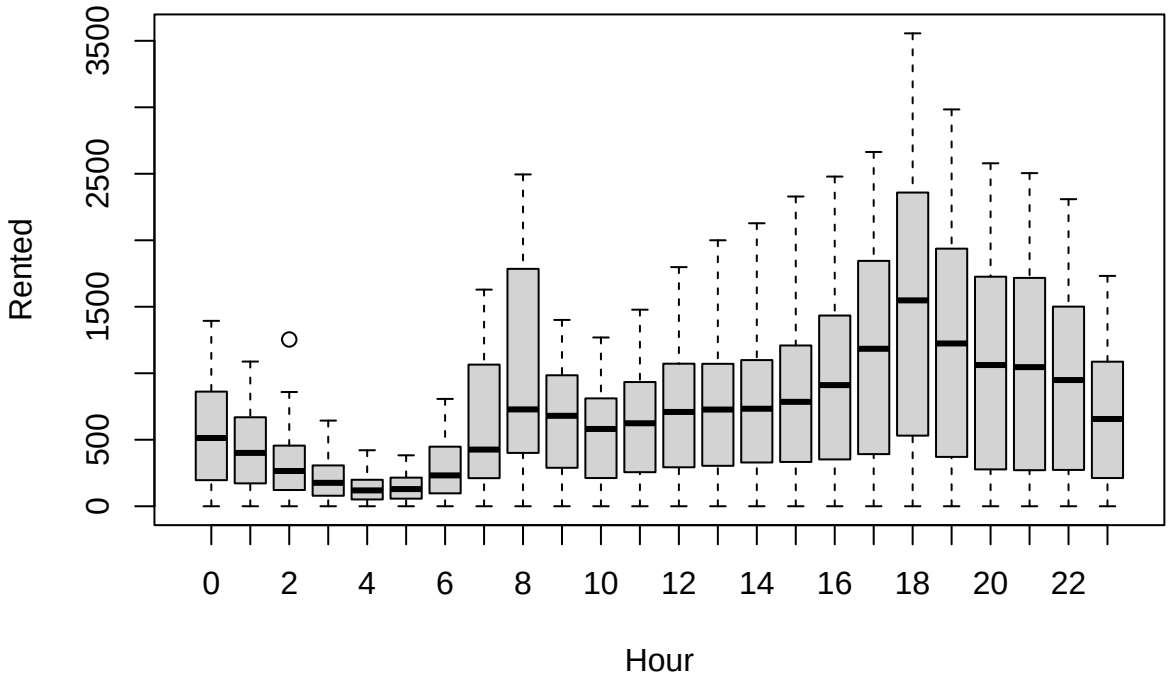

Further drilling seasons down to month, we can see the rental bike count reaches the peak in Jun and the lowest point in Jan.

```
plot(Rented ~ Weekday, data = bike)
```



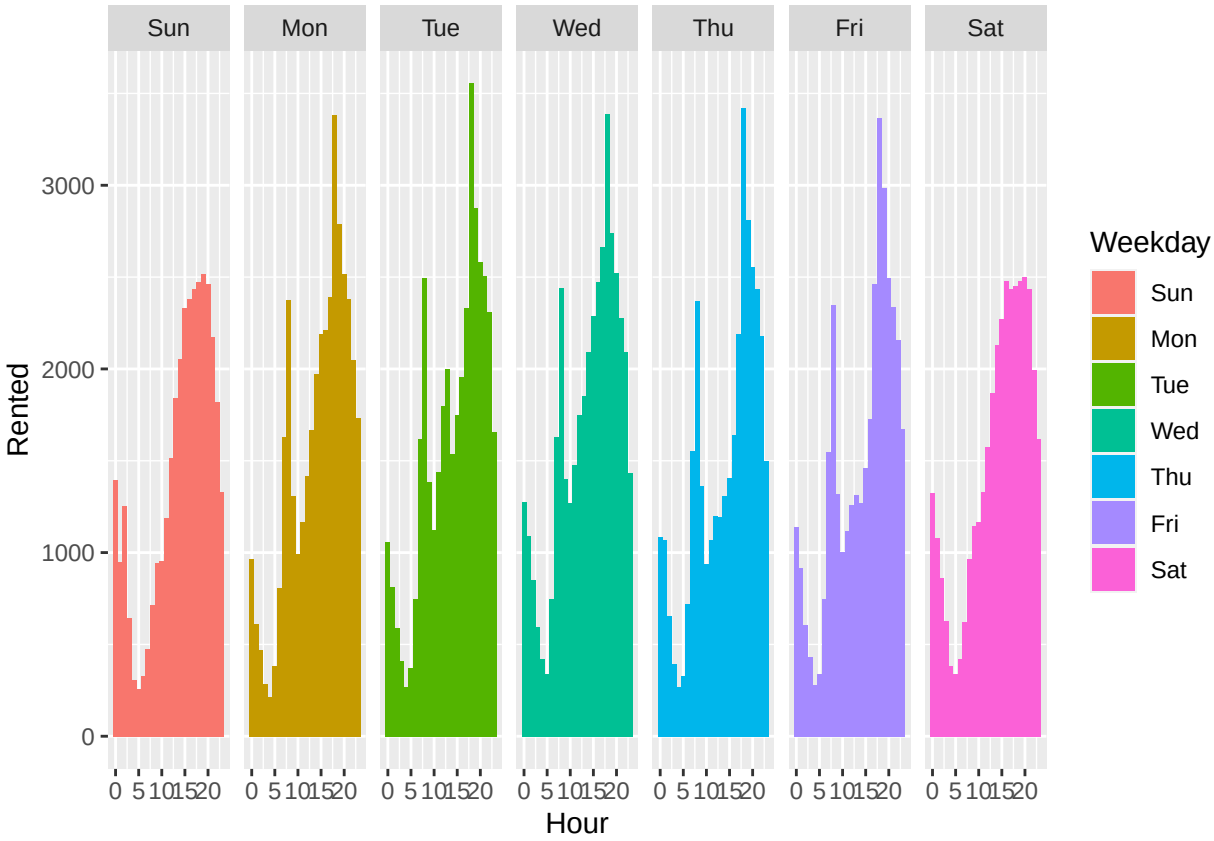

Generally speaking, we have lower demands on Saturday and Sunday, while other weekdays have similar higher demand.

```
plot(Rented ~ as.factor(Hour), xlab = "Hour",data = bike)
```



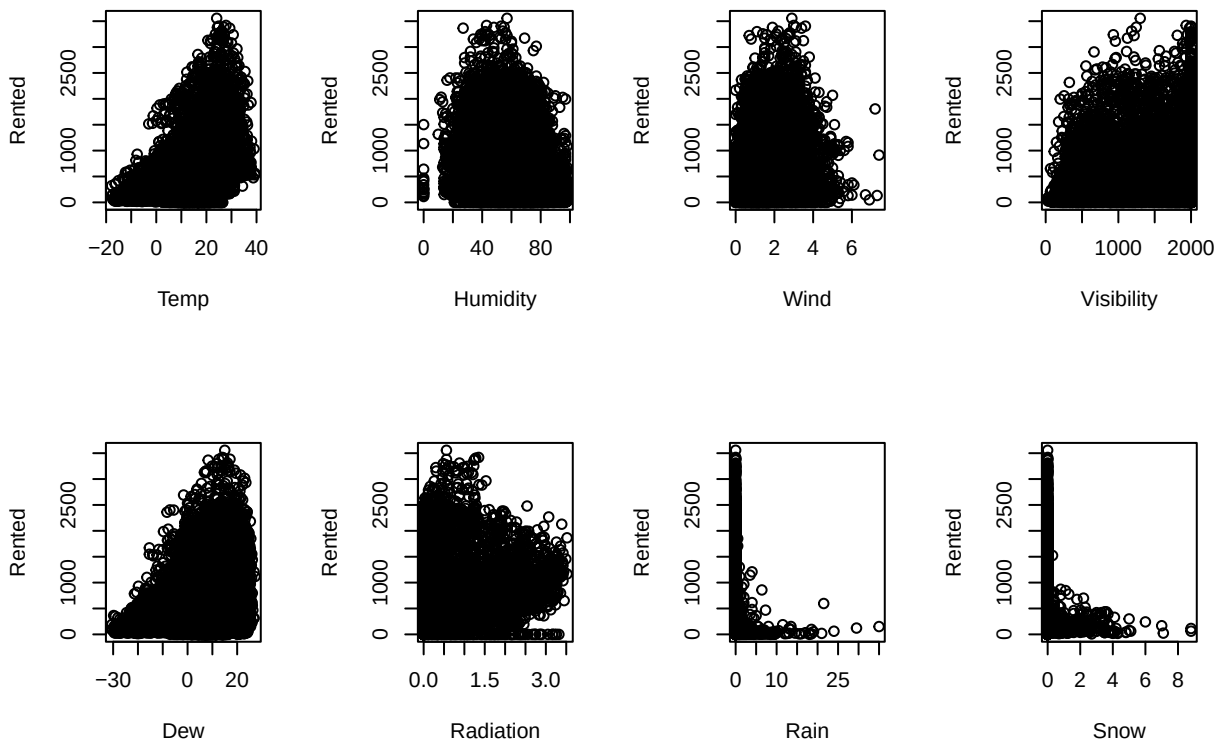

We can see two peaks on the rental bike count vs hour chart: one at 8 AM and the other one at 6 PM, which correspond with the peak commute hours. This supports our assumption earlier that people may use rented bikes more as a commute method.

```
ggplot(bike, aes(x = Hour, y = Rented, fill = Weekday)) +
      facet_grid(. ~ Weekday) + geom_bar(stat = "identity", position = "dodge")
```




Combine Weekday and Hour together, we can see that on weekdays, the hourly trend is similar - a small peak in the morning, and then the demand grows stronger in the afternoon and continues into the evening. During weekends, the demand peaks around evening time.

```
par(mfrow=c(2, 4))

plot(Rented ~ Temp,       data = bike)
plot(Rented ~ Humidity,   data = bike)
plot(Rented ~ Wind,       data = bike)
plot(Rented ~ Visibility, data = bike)
plot(Rented ~ Dew,        data = bike)
plot(Rented ~ Radiation,  data = bike)
plot(Rented ~ Rain,       data = bike)
plot(Rented ~ Snow,       data = bike)
```




Rented bike counts generally increase as temperature and dew point temperature rise, but decrease quickly once they pass the optimal range. For humidity and wind speed, there also exist an obvious optimal range that lead to highest rented bike counts. The better the visibility, the higher the rented bike count is. Rainfall and Snowfall cause a sharply decreased demand of rental bikes. Since they are highly skewed, we may try to transform these two variables later in the modeling process.

### Outlier diagnostics

Let's check if our data-set has outliers.

```
diag = diagnose_numeric(bike)
formattable(diag, digits = 2)
```

variables

min

Q1

mean

median

Q3

max

zero

minus

outlier

Rented

0

191.0

704.602

504.50

1065.25

3556.0

295

0

157

Hour

0

5.8

11.500

11.50

17.25

23.0

365

0

0

Temp

-18

3.5

12.883

13.70

22.50

39.4

21

1433

0

Humidity

0

42.0

58.226

57.00

74.00

98.0

17

0

0

Wind

0

0.9

1.725

1.50

2.30

7.4

74

0

161

Visibility

27

940.0

1436.826

1698.00

2000.00

2000.0

0

0

0

Dew

-31

-4.7

4.074

5.10

14.80

27.2

60

3138

0

Radiation

0

0.0

0.569

0.01

0.93

3.5

4300

0

641

Rain

0

0.0

0.149

0.00

0.00

35.0

8232

0

528

Snow

0

0.0

0.075

0.00

0.00

8.8

8317

0

443

Month

1

4.0

6.526

7.00

10.00

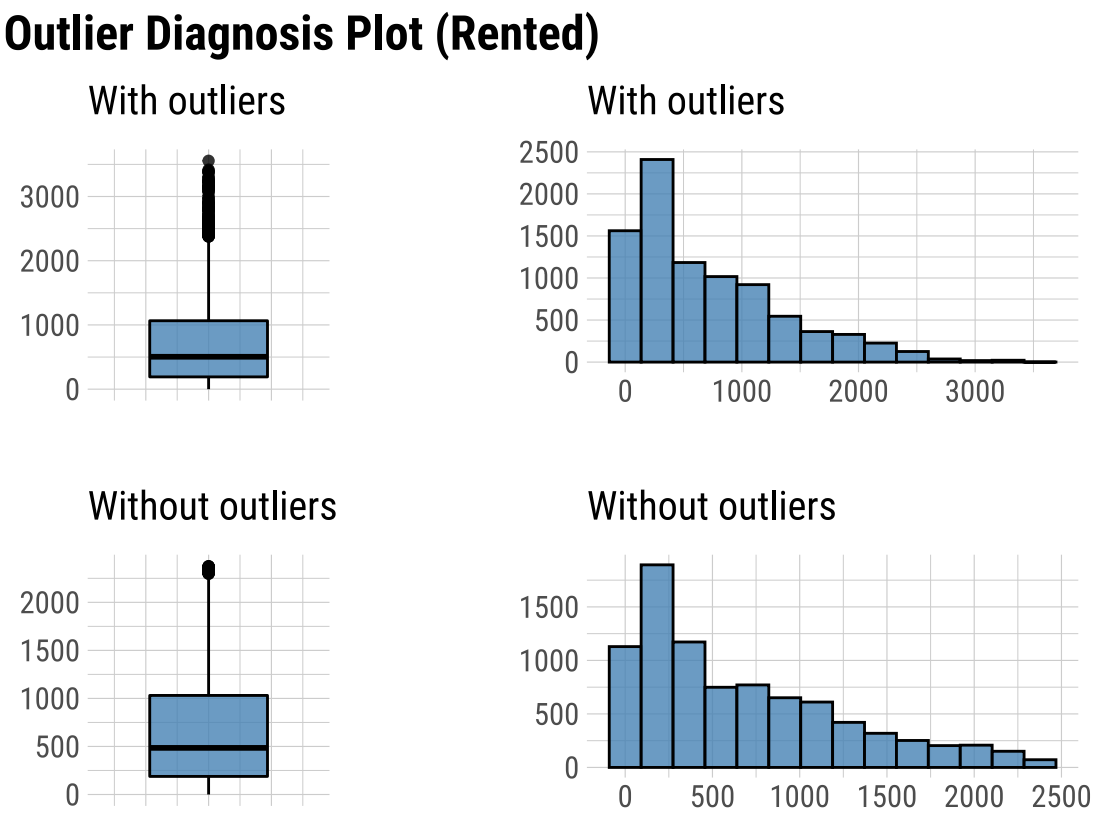
12.0

0

0

0

```
bike %>% plot_outlier(Rented)
```




We do have outliers for Rented, Wind, Radiation, Rain and Snow. This also supports our previous observations from the EDA charts that some of these variables are highly skewed. If we remove outliers from the response variable, the distribution becomes a bit more evenly spread.

But by examining the boundaries of these variables, we understand that these are extreme weather conditions, valid data instead of data errors. These outliers don't necessary have a big impact on the prediction. We decide to keep them in the dataset.

### Modeling

**A naive additive model** First of all, we take a look at the most basic model - an additive model using all the predictors in their original format.

```
mod_naive = lm(Rented ~ ., data = bike)
summary(mod_naive)
```

```
##
```

```
## Call:
## lm(formula = Rented ~ ., data = bike)
##
## Residuals:
##    Min     1Q Median     3Q    Max
##  -1157   -275    -56    206   2226
##
## Coefficients: (1 not defined because of singularities)
##                     Estimate Std. Error t value Pr(>|t|)
## (Intercept)         2.65e+04   3.17e+03    8.36  < 2e-16 ***
## Date               -1.50e+00   1.78e-01   -8.44  < 2e-16 ***
## Hour                2.73e+01   7.29e-01   37.44  < 2e-16 ***
## Temp                1.84e+01   3.65e+00    5.04  4.8e-07 ***
## Humidity           -1.08e+01   1.03e+00  -10.56  < 2e-16 ***
## Wind                1.73e+01   5.07e+00    3.42  0.00063 ***
## Visibility          2.21e-03   9.84e-03    0.22  0.82204
## Dew                 9.44e+00   3.82e+00    2.47  0.01340 *
## Radiation          -8.33e+01   7.55e+00  -11.04  < 2e-16 ***
## Rain               -5.73e+01   4.24e+00  -13.53  < 2e-16 ***
## Snow                3.29e+01   1.12e+01    2.94  0.00327 **
## SeasonSpring       -4.02e+02   3.84e+01  -10.47  < 2e-16 ***
## SeasonSummer       -2.94e+02   2.53e+01  -11.59  < 2e-16 ***
## SeasonWinter       -7.64e+02   5.38e+01  -14.20  < 2e-16 ***
## HolidayNo Holiday   1.27e+02   2.15e+01    5.91  3.6e-09 ***
## FunctioningYes      9.52e+02   2.66e+01   35.75  < 2e-16 ***
## Month               1.94e+00   1.82e+00    1.06  0.28734
## WeekdayMon          8.50e+01   1.72e+01    4.93  8.2e-07 ***
## WeekdayTue          1.12e+02   1.73e+01    6.49  8.9e-11 ***
## WeekdayWed          1.37e+02   1.73e+01    7.96  1.9e-15 ***
## WeekdayThu          1.09e+02   1.72e+01    6.33  2.6e-10 ***
## WeekdayFri          1.40e+02   1.71e+01    8.14  4.4e-16 ***
## WeekdaySat          7.18e+01   1.72e+01    4.17  3.1e-05 ***
## WeekendYes                NA         NA      NA       NA
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 429 on 8737 degrees of freedom
## Multiple R-squared:  0.559,  Adjusted R-squared:  0.558
## F-statistic:  503 on 22 and 8737 DF,  p-value: <2e-16
```

The result is not too bad. We got an adjusted $R^2$ of 0.56 and an extremely small p-value. It looks like this base model can explain more than 55% of the variance in the response variable. We also notice that we obviously have a variable that can be completely derived from another variable - Weekend, so it's redundant. Let's try to improve the model.

**Variable Selection and Transformations** Let's drop some variables: - Date: Too many distinct values for a categorical variable. - Weekend: Created for data exploration purposes but all the information can be derived from Weekday. - Season: Can be derived from Month.

```
bike_cln = subset(bike, select = -c(Date, Weekend, Season))
```

```
mod_additive = lm(Rented ~ ., data = bike_cln)
summary(mod_additive)$adj.r.squared
```

**A basic additive model on the cleaner data-set**

```
## [1] 0.53
```

The above gives us an adjusted $R^2$ score at 0.53. It's lower than the previous adjusted $R^2$ score since we removed some predictors.

```
library(faraway)
vif(mod_additive)[vif(mod_additive) > 5]
```

```
##     Temp Humidity      Dew
##       88       21      117
```

We do have high variance inflation factors. We decide to keep them since they don't usually impact our predictions much.

**A basic additive model with factor variables** Let's try to convert Hour and Month to factor variables - although they are numeric numbers now, they can only have certain values and we can't say the difference in average rented bike counts between Hour 1 and 2 will be the same as the difference in average rented bike counts between Hour 17 and 18.

```
# Copy dataset to test Hour and Month as factor variables
bike_factor       = data.frame(bike_cln)
bike_factor$Hour  = as.factor(bike_factor$Hour)
bike_factor$Month = as.factor(bike_factor$Month)

# Build the model using Hour and Month as factor variables
mod_additive_factor = lm(Rented ~ ., data = bike_factor)
summary(mod_additive_factor)$adj.r.squared
```

```
## [1] 0.7
```

After conversion, the model gives a better adjusted $R^2$ score at 0.7 now.

```
vif(mod_additive_factor)[vif(mod_additive_factor) > 5]
```

```
##     Temp Humidity      Dew   Month6   Month7   Month8   Month9
##     99.6     21.4    126.5      6.1      8.4      9.0      6.2
```

We can see Temperature and some Month variables are high variance inflation factors. This makes sense since Temperature usually has some correlation with seasonality / month. Again we decide to keep them since they won't impact our predictions much.

```
mod_interact = lm(Rented ~ . ^ 2, data = bike_cln)
summary(mod_interact)$adj.r.squared
```

**An interaction model using the data-set without the factor variables conversion**

```
## [1] 0.68
```

The adjusted $R^2$ score ( 0.68 ) is better than the additive model using the same dataset but worse than the additive model using the dataset with Hour and Month as factor variables.

```
mod_interact_factor = lm(Rented ~ . ^ 2, data = bike_factor)
summary(mod_interact_factor)$adj.r.squared
```

**An interaction model using the dataset with the factor variables conversion**

```
## [1] 0.91
```

The adjusted $R^2$ score has been improved greatly to 0.91.

```
anova(mod_additive, mod_interact)
```

**ANOVA: Comparison of additive and interaction models**

```
## Analysis of Variance Table
##
## Model 1: Rented ~ Hour + Temp + Humidity + Wind + Visibility + Dew + Radiation +
##     Rain + Snow + Holiday + Functioning + Month + Weekday
## Model 2: Rented ~ (Hour + Temp + Humidity + Wind + Visibility + Dew +
##     Radiation + Rain + Snow + Holiday + Functioning + Month +
##     Weekday)^2
##   Res.Df      RSS  Df Sum of Sq    F Pr(>F)
## 1   8741 1.70e+09
## 2   8605 1.16e+09 136 537500402 29.3 <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
anova(mod_additive_factor, mod_interact_factor)
```

```
## Analysis of Variance Table
##
## Model 1: Rented ~ Hour + Temp + Humidity + Wind + Visibility + Dew + Radiation +
##     Rain + Snow + Holiday + Functioning + Month + Weekday
## Model 2: Rented ~ (Hour + Temp + Humidity + Wind + Visibility + Dew +
##     Radiation + Rain + Snow + Holiday + Functioning + Month +
```

```
##     Weekday)^2
##   Res.Df      RSS  Df Sum of Sq    F Pr(>F)
## 1   8709 1.09e+09
## 2   7837 2.85e+08 872 801994254 25.3 <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

By comparing the additive model and the interaction model using the two datasets (with or without factor variables conversion), we can see the p-value is extremely small in both cases. So we prefer the interaction model on the dataset with factor variables conversion based on the Adjusted R-Squared value.

**Influential observations with cook's distance** We use the interaction model with factor variables from previous discussion for this evaluation.

```
mod_interact_factor_cd = cooks.distance(mod_interact_factor)
(count_cd = sum(mod_interact_factor_cd > 4 / length(mod_interact_factor_cd)))
```

```
## [1] 566
```

There are 566 influential observations out of total 8760 observations. Lets remove these and fit an interaction model with factor variables.

```
mod_interact_factor_no_ol = lm(Rented ~ . ^ 2, data = bike_factor,
                               subset = mod_interact_factor_cd
                                        <= 4 / length(mod_interact_factor_cd))
summary(mod_interact_factor_no_ol)$adj.r.squared
```

```
## [1] 0.96
```

We see that the adjusted $R^2$ value has greatly increased to 0.96, which naturally makes sense since we removed these outliers and the model can explain more variance in the response variable.

Next, lets try to understand if the original model fit with the new dataset without the first outliers eliminated has outliers as evaluated by cook's distance.

```
mod_interact_factor_cd_2 = cooks.distance(mod_interact_factor_no_ol)
(count_cd_2 = sum(mod_interact_factor_cd_2 > 4/length(mod_interact_factor_no_ol)))
```

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

We see that there are 0 influential points with the outliers removed and model refitted. We eventually did not end up using this model and retained all the original observations. This was because using the factor/sq based models we were unable to complete the "Step" and "Exhaustive Search" operations for the best AIC Model in R.

### Response Variable Transformations

**Square root of the response variable** We will use Hour and Month as factor variables going forward based on previous discussions.

```
# add 0.001 to response variable to avoid errors in log
bike_factor$Rented = bike_factor$Rented + 0.001
```

```
mod_interact_sq = lm(sqrt(Rented) ~ . ^ 2, data = bike_factor)
summary(mod_interact_sq)$adj.r.squared
```

```
## [1] 0.93
```

After taking square root of the response variable, the adjusted $R^2$ is further improved to 0.93 now.

```
mod_interact_log = lm(log(Rented) ~ . ^ 2, data = bike_factor)
summary(mod_interact_log)$adj.r.squared
```

**Log transformation on the response variable**

```
## [1] 0.98
```

The adjusted $R^2$ is at 0.98 now.

**Model selection with AIC backwards** Let's run AIC backward searching on the interaction model (It takes too long to run step wise model selection on the other models with transformations). We commented out the code which did the actual selection work to speed up the compilation of the final report, and pasted the selected model directly below.

```
# mod_int_aic = step(mod_interact, direction = "backward", trace = 0)
mod_int_aic = lm(formula = Rented ~ Hour + Temp + Humidity + Wind + Visibility +
    Dew + Radiation + Rain + Snow + Holiday + Functioning + Month +
    Weekday + Hour:Humidity + Hour:Visibility + Hour:Dew + Hour:Radiation +
    Hour:Snow + Hour:Holiday + Hour:Functioning + Hour:Month +
    Hour:Weekday + Temp:Humidity + Temp:Wind + Temp:Visibility +
    Temp:Dew + Temp:Radiation + Temp:Rain + Temp:Snow + Temp:Month +
    Temp:Weekday + Humidity:Wind + Humidity:Visibility + Humidity:Dew +
    Humidity:Radiation + Humidity:Rain + Humidity:Functioning +
    Humidity:Weekday + Wind:Visibility + Wind:Radiation + Wind:Rain +
    Wind:Month + Wind:Weekday + Visibility:Dew + Visibility:Radiation +
    Visibility:Rain + Visibility:Snow + Visibility:Holiday +
    Visibility:Month + Visibility:Weekday + Dew:Radiation + Dew:Rain +
    Dew:Holiday + Dew:Functioning + Dew:Month + Dew:Weekday +
    Radiation:Holiday + Radiation:Functioning + Radiation:Month +
    Radiation:Weekday + Rain:Holiday + Rain:Functioning + Rain:Month +
    Snow:Month + Snow:Weekday + Holiday:Weekday + Month:Weekday,
    data = bike_cln)
summary(mod_int_aic)$adj.r.squared
```

```
## [1] 0.68
```

The AIC backward model has an adjusted $R^2$ score of 0.68. We know fewer predictors will lead to a decreased adjusted $R^2$ score but a simpler model with easier interpretation.

**Log transformations on select variables** Now we will try to do log transformations on those highly skewed variables accordingly to our previous EDA.

```
mod_select = lm(log(Rented) ~ Hour + Temp + Humidity + log(Wind + 0.0001) +
                  Visibility + Dew + log(Radiation + 0.0001) +
                  log(Rain + 0.0001) + log(Snow + 0.0001) + Month + Weekday +
                  Functioning + Holiday, data = bike_factor)
summary(mod_select)$adj.r.squared
```

```
## [1] 0.96
```

```
mod_int_select = lm(log(Rented) ~ (Hour + Temp + Humidity + log(Wind + 0.0001) +
                                     Visibility + Dew + log(Radiation + 0.0001) +
                                     log(Rain + 0.0001) + log(Snow + 0.0001) +
                                     Month + Weekday + Functioning + Holiday) ^ 2,
                    data = bike_factor)
summary(mod_int_select)$adj.r.squared
```

```
## [1] 0.98
```

The interaction model with log transformations on skewed variables achieved the best adjusted $R^2$ score so far.

## Results

In the previous section, we mostly looked at adjusted $R^2$ scores. Now let's use some other evaluation metrics too, such as Cross-validated RMSE and AIC.

```
calc_loocv_rmse = function(model) {
  sqrt(mean((resid(model) / (1 - hatvalues(model))) ^ 2))
}
```

```
result = data.frame(matrix(ncol = 3, nrow = 10))
colnames(result) = c("Adjusted R_2 Score", "Cross-validated RMSE", "AIC")

result[1, ] = c(summary(mod_naive)$adj.r.squared,
                calc_loocv_rmse(mod_naive),
                AIC(mod_naive))
result[2, ] = c(summary(mod_additive)$adj.r.squared,
                calc_loocv_rmse(mod_additive),
                AIC(mod_additive))
result[3, ] = c(summary(mod_additive_factor)$adj.r.squared,
                calc_loocv_rmse(mod_additive_factor),
                AIC(mod_additive_factor))
result[4, ] = c(summary(mod_interact)$adj.r.squared,
                calc_loocv_rmse(mod_interact),
                AIC(mod_interact))
result[5, ] = c(summary(mod_interact_factor)$adj.r.squared,
                calc_loocv_rmse(mod_interact_factor),
                AIC(mod_interact_factor))
result[6, ] = c(summary(mod_interact_sq)$adj.r.squared,
```

```
                0,
                AIC(mod_interact_sq))
result[7, ] = c(summary(mod_interact_log)$adj.r.squared,
                0,
                AIC(mod_interact_log))
result[8, ] = c(summary(mod_int_aic)$adj.r.squared,
                calc_loocv_rmse(mod_int_aic),
                AIC(mod_int_aic))
result[9, ] = c(summary(mod_select)$adj.r.squared,
                0,
                AIC(mod_select))
result[10, ] = c(summary(mod_int_select)$adj.r.squared,
                0,
                AIC(mod_int_select))

row.names(result) = c("mod_naive",
                      "mod_additive",
                      "mod_additive_factor",
                      "mod_interact",
                      "mod_interact_factor",
                      "mod_interact_sq",
                      "mod_interact_log",
                      "mod_int_aic",
                      "mod_select",
                      "mod_int_select")
```

Note that the 0 in the table below means "Not Available", instead of meaning a perfect score.

```
# options(knitr.kable.NA = "0")

knitr::kable(result, digits = 2, align = "lccr") %>%
  column_spec(2, color = ifelse(result$`Adjusted R_2 Score` ==
                          max(result$`Adjusted R_2 Score`), "green", "black"),
              background = ifelse(result$`Adjusted R_2 Score` ==
                          max(result$`Adjusted R_2 Score`), "lightgray", "white")) %>%
  column_spec(3, color = ifelse(result$`Cross-validated RMSE` ==
                          sort(unique(na.omit(result$`Cross-validated RMSE`)))[2], "green", "black"),
              background = ifelse(result$`Cross-validated RMSE` ==
                          sort(unique(na.omit(result$`Cross-validated RMSE`)))[2], "lightgray", "white")
  column_spec(4, color = ifelse(result$AIC == min(result$AIC), "green", "black"),
              background = ifelse(result$AIC == min(result$AIC), "lightgray", "white")) %>%
  kable_styling()
```

The interaction model with log transformation of skewed variables has the best adjusted $R^2$ score and AIC score. We can see the interaction model with the factor variables has the lowest cross-validated RMSE. However, we can't easily apply this function to the models with transformed response variables.

#### RMSE on test data-set

The previous evaluation metrics are all for seen data and we are not sure if the model with a good score actually over fits. Let's split data into train/test to test how the models perform on unseen data. We are only using the data-set with the Hour and Month factor variables now, since we know the factor variables greatly

| | Adjusted R_2 Score | Cross-validated RMSE | AIC |
|---|---|---|---|
| mod_naive | 0.56 | 430 | 131081 |
| mod_additive | 0.53 | 441 | 131551 |
| mod_additive_factor | 0.70 | 355 | 127706 |
| mod_interact | 0.68 | 371 | 128489 |
| mod_interact_factor | 0.91 | 211 | 117721 |
| mod_interact_sq | 0.93 | 0 | 46724 |
| mod_interact_log | 0.98 | 0 | 8945 |
| mod_int_aic | 0.68 | 369 | 128451 |
| mod_select | 0.96 | 0 | 13865 |
| mod_int_select | 0.98 | 0 | 6987 |

boosted the model performance. The train dataset will contain 6132 observations, which is approximately 70% of the total observations.

```
set.seed(420)
bike_idx = sample(1:nrow(bike_factor), 6132)
bike_trn = bike_factor[bike_idx, ]
bike_tst = bike_factor[-bike_idx, ]
```

Define the function to calculate RMSE.

```
RMSE <- function(model, data, trans = "") {
  n = nrow(data)
  y_hat = predict(model, data)
  if(trans=="log") {
      resid = data$Rented - exp(y_hat)
  } else if (trans=="sqrt"){
      resid = data$Rented - y_hat ^ 2
  } else {
      resid = data$Rented - y_hat
  }
  sqrt(sum(resid ^ 2) / n)
}
```

```
mod_additive_trn = lm(Rented ~ ., data = bike_trn)
mod_interact_trn = lm(Rented ~ . ^ 2, data = bike_trn)
mod_interact_sq_trn = lm(sqrt(Rented) ~ . ^ 2, data = bike_trn)
mod_interact_log_trn = lm(log(Rented) ~ . ^ 2, data = bike_trn)
mod_select_trn = lm(log(Rented) ~ Hour + Temp + Humidity + log(Wind + 0.0001) +
                      Visibility + Dew + log(Radiation + 0.0001) +
                      log(Rain + 0.0001) + log(Snow + 0.0001) + Month +
                      Weekday + Functioning + Holiday,
                    data = bike_trn)
mod_int_select_trn = lm(log(Rented) ~ (Hour + Temp + Humidity +
                                         log(Wind + 0.0001) + Visibility + Dew +
                                         log(Radiation + 0.0001) +
                                         log(Rain + 0.0001) + log(Snow + 0.0001) +
                                         Month + Weekday + Functioning + Holiday) ^ 2,
                        data = bike_trn)

test_rmse = data.frame(matrix(ncol = 1, nrow = 0))
```

| | Test Dataset RMSE |
|---|---|
| mod_additive_trn | 365 |
| mod_interact_trn | 217 |
| **mod_interact_sq_trn** | **193** |
| mod_interact_log_trn | 256 |
| mod_select_trn | 320 |
| mod_int_select_trn | 200 |

```
colnames(test_rmse) = c("Test Dataset RMSE")

test_rmse[1, ] = RMSE(mod_additive_trn, bike_tst)
test_rmse[2, ] = RMSE(mod_interact_trn, bike_tst)
test_rmse[3, ] = RMSE(mod_interact_sq_trn, bike_tst, trans = "sqrt")
test_rmse[4, ] = RMSE(mod_interact_log_trn, bike_tst, trans = "log")
test_rmse[5, ] = RMSE(mod_select_trn, bike_tst, trans = "log")
test_rmse[6, ] = RMSE(mod_int_select_trn, bike_tst, trans = "log")

row.names(test_rmse) = c("mod_additive_trn",
                         "mod_interact_trn",
                         "mod_interact_sq_trn",
                         "mod_interact_log_trn",
                         "mod_select_trn",
                         "mod_int_select_trn")
```

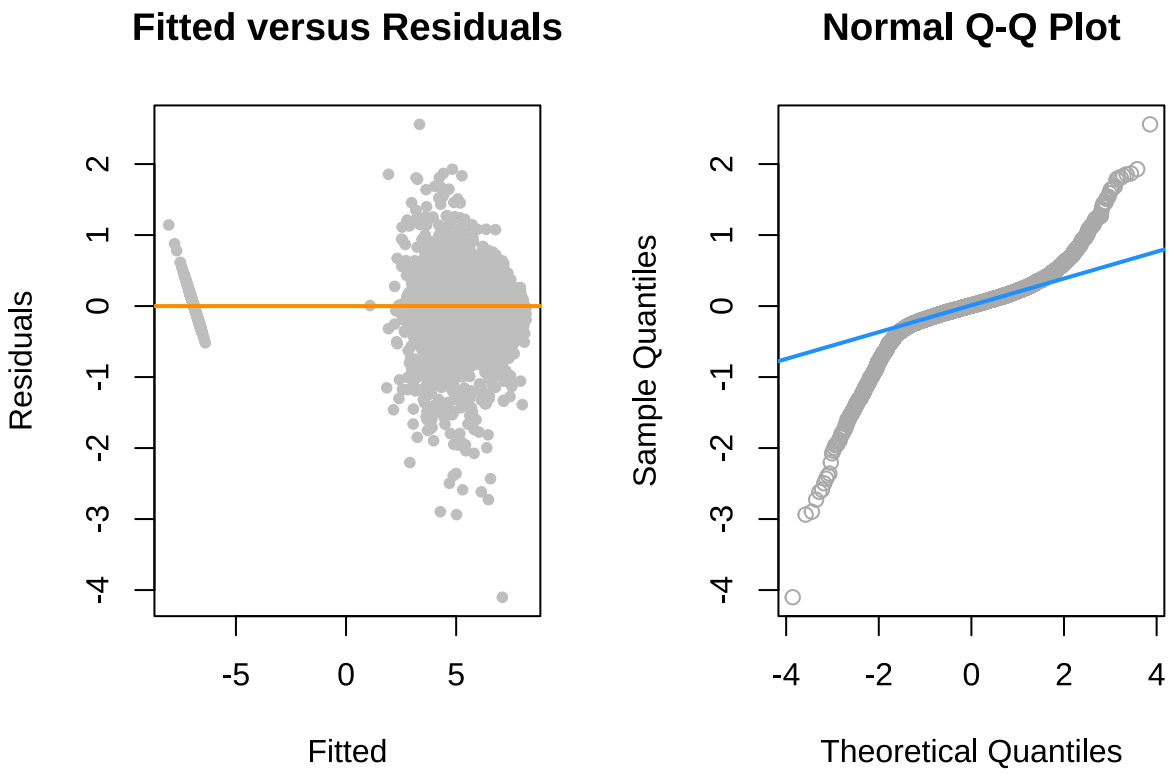
```
knitr::kable(test_rmse, align = "lc", digits = 2) %>%
  kable_styling() %>%
  row_spec(which.min(test_rmse$`Test Dataset RMSE`), bold = TRUE, color = "white", background = "green")
```

We can see both mod_interact_sq_trn and mod_int_select_trn have very lowest RMSE on the test dataset. Considering mod_int_select_trn also has the best metrics in previous evaluation, we would like to choose this model as the best prediction model.

### Residual diagnostics

Check model assumptions for the best model selected from the metrics evaluation section:

```
par(mfrow = c(1, 2))

plot(fitted(mod_int_select), resid(mod_int_select), col = "grey", pch = 20,
     xlab = "Fitted", ylab = "Residuals", main = "Fitted versus Residuals")
abline(h = 0, col = "darkorange", lwd = 2)

qqnorm(resid(mod_int_select), main = "Normal Q-Q Plot", col = "darkgrey")
qqline(resid(mod_int_select), col = "dodgerblue", lwd = 2)
```




On the Fitted vs Residual chart, the values more or less center around 0, so linear assumption is valid. The spread of the data is also relatively equal along the x-axis, so constant variance assumption looks valid too. The normality assumption is violated accordingly to the QQ-Plot. Maybe there is another family of model that is better suited for our problem. For now, we are happy that our model can achieve good predictions.

## Discussion

In order to understand the factors that affect rented bike counts and make accurate predictions on the hourly rental bike demand, we chose Parametric Model Family and Multiple Linear Regression Models for the fit.

We applied multiple model building techniques we studied in this course:

- Data exploration analysis and outlier diagnostics. We performed feature engineering and created additional variables based on existing variables. We converted categorical variables into proper factor variables.
- Used Multiple linear regression , ANOVA for p-value analysis, predictor Interaction ,Residual diagnostics ,Transformations and Step-wise model selection with Variable selection
- We evaluated the models' performances using several metrics: Adjusted $R^2$ Score, Cross-validated RMSE, AIC and RMSE on test data-set.
- The final model we chose is: **lm(log(Rented) ~ (Hour + Temp + Humidity + log(Wind + 0.0001) + Visibility + Dew + log(Radiation + 0.0001) + log(Rain + 0.0001) + log(Snow + 0.0001) + Month + Weekday + Functioning + Holiday) ^ 2, data = bike_factor)**. This model has great performances on Adjusted $R^2$ Score, AIC and RMSE on test data-set, which means it predict well on both train and test data-sets, no under-fitting or over-fitting, and is not too complex either.

The model does have some limitations which may affect the interpretation of the coefficients:

- The normality assumption is violated.
- We discussed high variance inflation factors and influential observations, but didn't attempt to remove them in the final models.
- Since we don't plan to use the coefficients for any important decisions, we are fine with the final model which gives good predictions.
- Though we used linear regression for the prediction problem, we acknowledge the fact that the response variable is actually integer. Strictly speaking, poisson regression may suit this problem better. We didn't attempt that since it is beyond the scope of this course.

We can use this model to predict the hourly rental bike demand in Seoul and help the Bike Sharing System plan inventory in advance. The system may be able to use this demand information to offer flexible rates to attract more users during the slow hours and make more revenue during peak hours. By knowing that people usually rent more bikes on weekdays during commute hours, the Bike Sharing System may work with the other public transit system to better coordinate the city's transportation network, or work with tourism agency to promote rental bikes during weekends and holidays.

## Appendix

### Normality assumption of observations

During our data exploration analysis, we looked into the normality assumption of each variable. Since it takes more spaces, we are showing the result here.

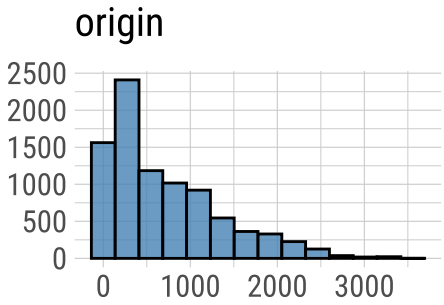
```
normality(bike_cln)
```

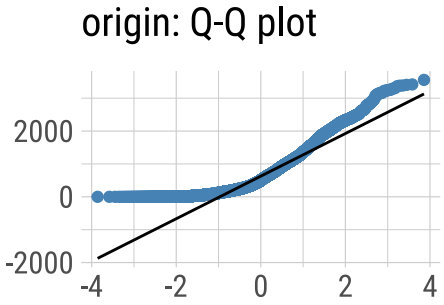
```
## # A tibble: 11 x 4
##    vars       statistic  p_value sample
##    <chr>          <dbl>    <dbl>  <dbl>
##  1 Rented         0.883 6.77e-52   5000
##  2 Hour           0.951 5.73e-38   5000
##  3 Temp           0.981 1.83e-25   5000
##  4 Humidity       0.981 2.05e-25   5000
##  5 Wind           0.950 2.54e-38   5000
##  6 Visibility     0.839 1.47e-57   5000
##  7 Dew            0.966 7.39e-33   5000
##  8 Radiation      0.705 3.82e-69   5000
##  9 Rain           0.115 5.48e-93   5000
## 10 Snow           0.163 1.10e-91   5000
## 11 Month          0.942 2.00e-40   5000
```

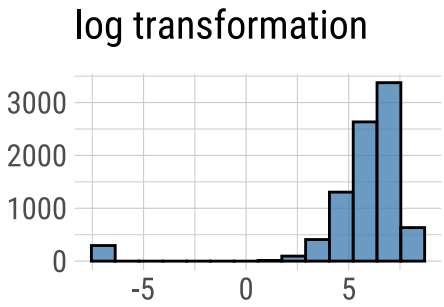
Check normality of each numeric variable - this provides valuable information about the skewness of each variable. We used this information and did log transformation on select variable and got the best model.

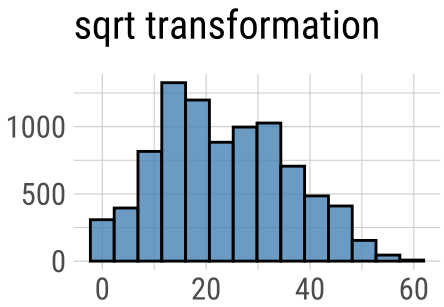
```
bike_factor %>% plot_normality(Rented, Temp, Humidity, Wind, Visibility,
                               Radiation, Rain, Snow)
```

## Normality Diagnosis Plot (Rented)

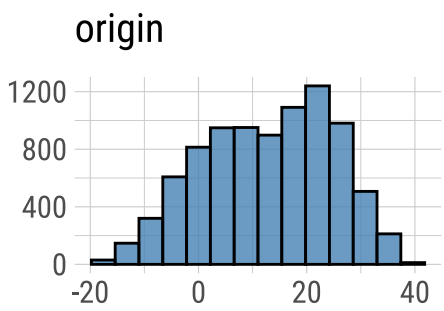


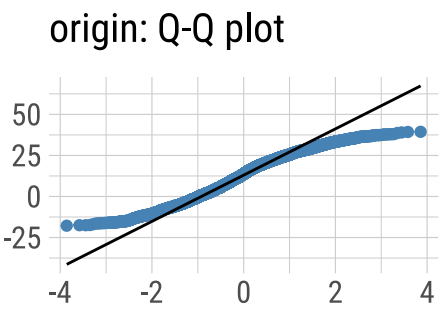



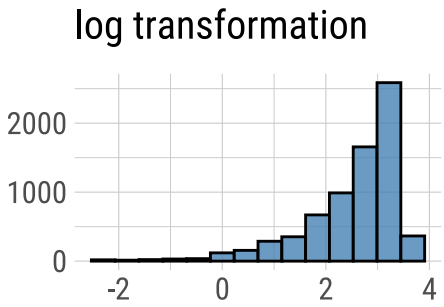



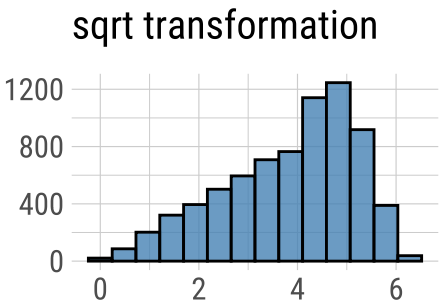



## Normality Diagnosis Plot (Temp)

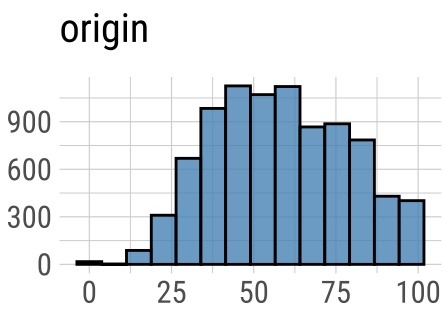


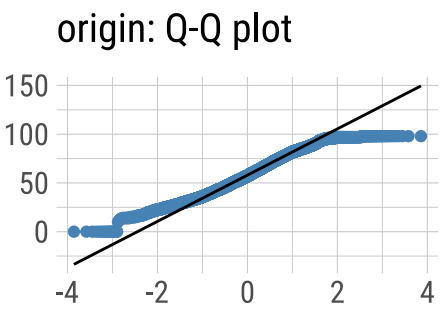



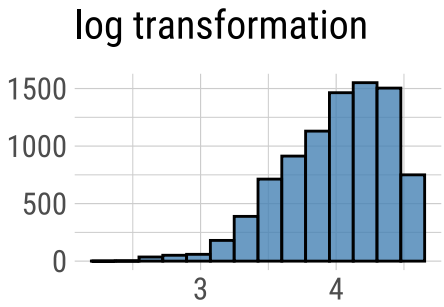



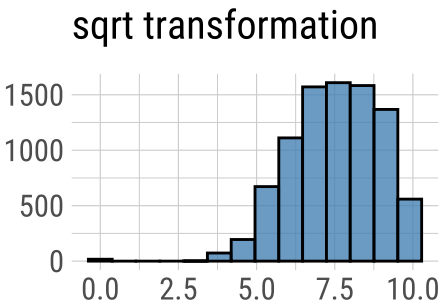



## Normality Diagnosis Plot (Humidity)

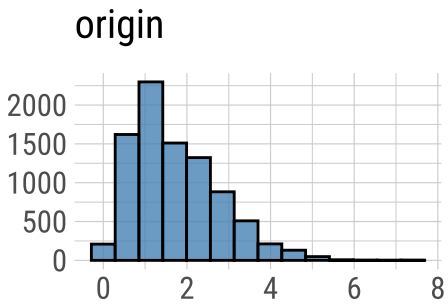


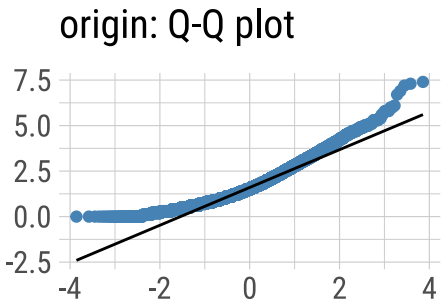



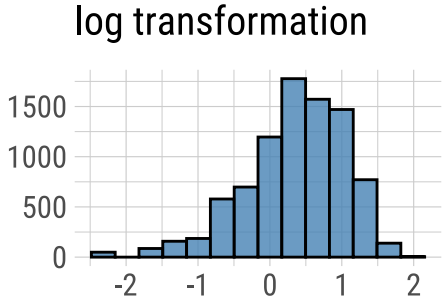



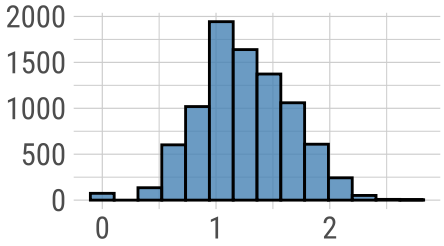



## Normality Diagnosis Plot (Wind)

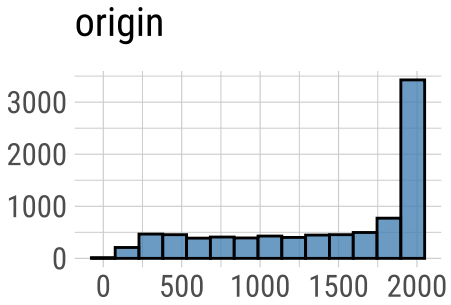


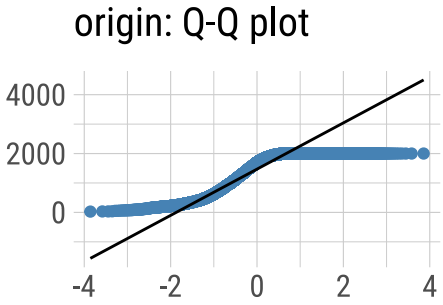



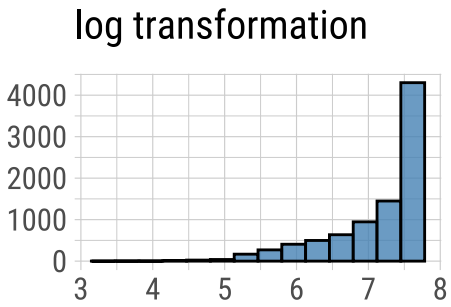



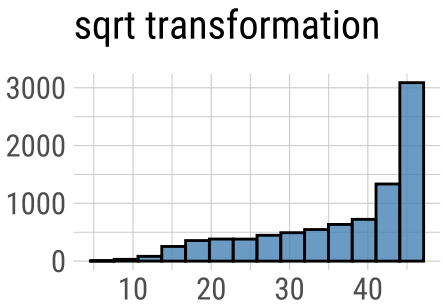



## Normality Diagnosis Plot (Visibility)

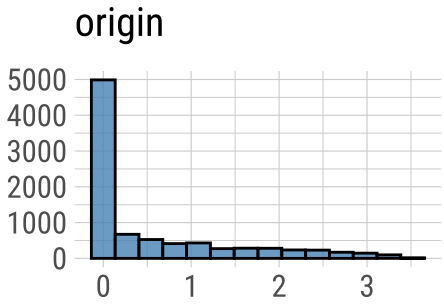


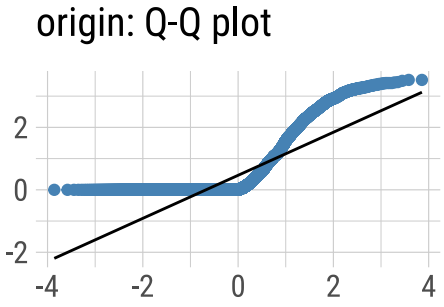



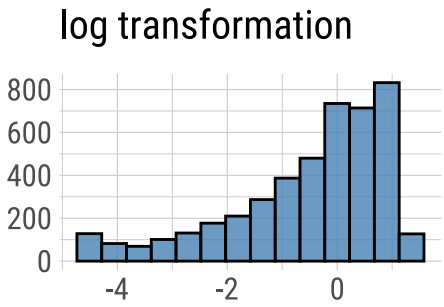



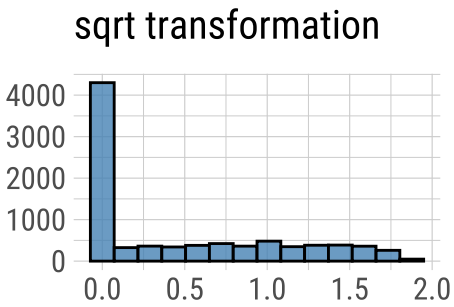



## Normality Diagnosis Plot (Radiation)

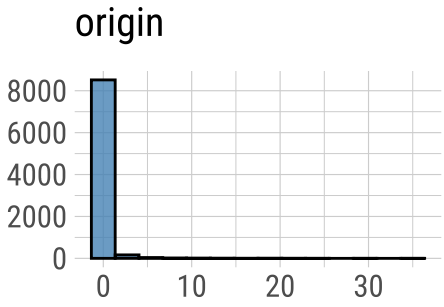


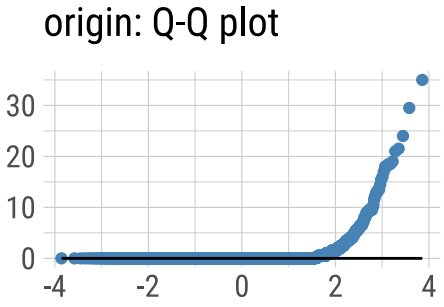



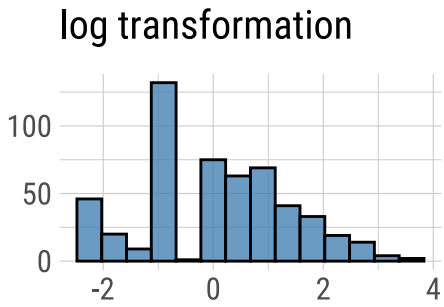



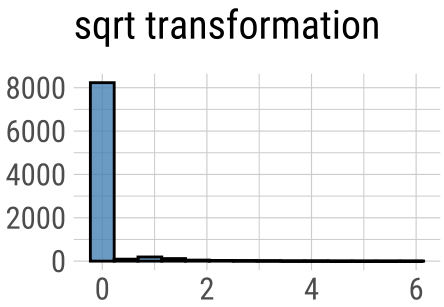



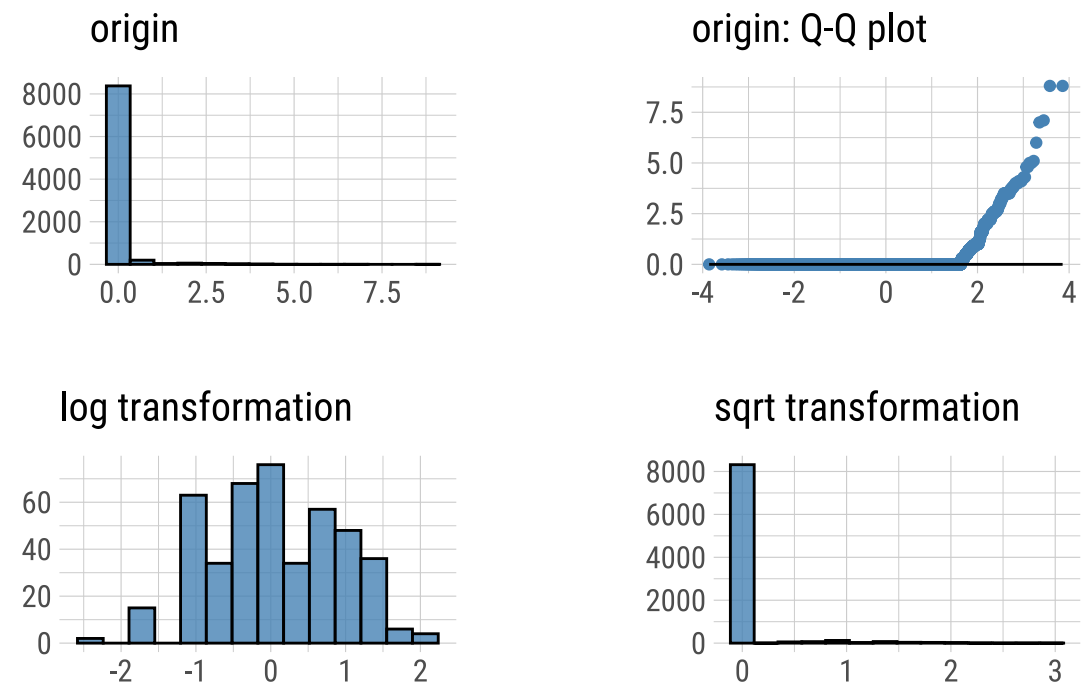
## Normality Diagnosis Plot (Rain)






**Authors**

Rui Zou (ruizou4), Ahmad Sadeed (asadeed2), Deepa Nemmili Veeravalli (deepan2)