Assignment 1: Bike Sharing

2024-09-16

```
knitr::opts_chunk$set(echo = TRUE)

library(ggplot2)
library(vtable)
```

```
## Loading required package: kableExtra
```

Introduction

For this assignment, I chose the Bike Sharing dataset, which provides a comprehensive log of bike-sharing rentals collected from the Capital Bikeshare system in Washington D.C. over two years, 2011 and 2012. The rental data is combined with environmental and seasonal factors that influence bike usage. Variables like temperature, humidity, wind speed, weather conditions, and whether the day is a holiday or weekend are provided.

```
# Download the dataset
download.file('https://archive.ics.uci.edu/static/public/275/bike+sharing+dataset.zip',
              'bike-sharing.zip')
unzip('bike-sharing.zip', files = 'day.csv')

# Load the dataset
df <- read.csv('day.csv', header = TRUE)

# Remove zip and csv files
file.remove(list.files(pattern = "\\.zip$|\\.csv$"))
```

Problem 1. Summary Statistics Table

```
st(df)
```

The dataset contains 16 columns and 731 obs. Below are a brief overview of the columns

Variable	Role	Type	Description
instant	Feature	Integer	Record index
dteday	Feature	Date	Date of the record
season	Feature	Categorical	Season (1 = Spring, 2 = Summer, 3 = Fall, 4 = Winter)
yr	Feature	Categorical	Year (0 = 2011, 1 = 2012)

Variable	Role	Type	Description
mnth	Feature	Categorical	Month (1 to 12)
holiday	Feature	Categorical	Whether the day is a holiday (1 = Yes, 0 = No)
weekday	Feature	Categorical	Day of the week (0 = Sunday, 1 = Monday, ..., 6 = Saturday)
workingday	Feature	Categorical	Whether the day is a working day (1 = Yes, 0 = No)
weathersit	Feature	Categorical	Weather situation (1 = Clear, 2 = Mist, 3 = Light Snow/Rain, 4 = Heavy Rain/Snow)
temp	Feature	Continuous	Normalized temperature in Celsius (values divided by 41)
atemp	Feature	Continuous	Normalized feeling temperature in Celsius (values divided by 50)
hum	Feature	Continuous	Normalized humidity (values divided by 100)
windspeed	Feature	Continuous	Normalized wind speed (values divided by 67)
casual	Other	Integer	Count of casual (unregistered) users
registered	Other	Integer	Count of registered users
cnt	Target	Integer	Total count of bike rentals, including both casual and registered users

This dataset is suitable for both classification and regression tasks. Key questions it can help address include:

- How can bike-sharing systems be optimized by identifying peak usage periods?
- How can planners prepare for special events or holidays?
- How does the weather impact bike rentals?

Problem 2. Bad Data Visualization

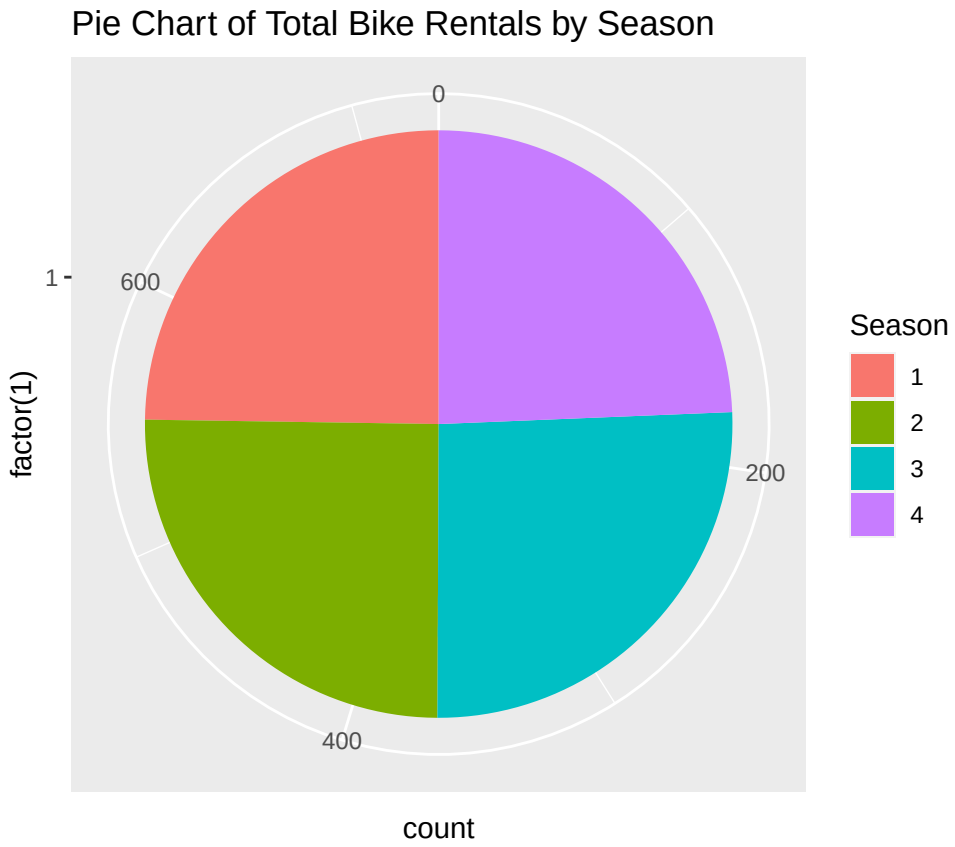
2.1. Categorical Variable (season)

If we make a Pie Chart of Total Bike Rentals by Season (category variable), it becomes difficult to compare exact differences between the seasons by comparing angles, which is harder for the human eye than comparing lengths (as in bar charts). In contrast, a bar chart would clearly show the differences by the heights of the bars, refer 3.1

```
ggplot(df, aes(x = factor(1), fill = factor(season))) +
  geom_bar(width = 1) +
  coord_polar(theta = "y") +
  labs(title = "Pie Chart of Total Bike Rentals by Season", fill = "Season")
```

Table 1: Summary Statistics

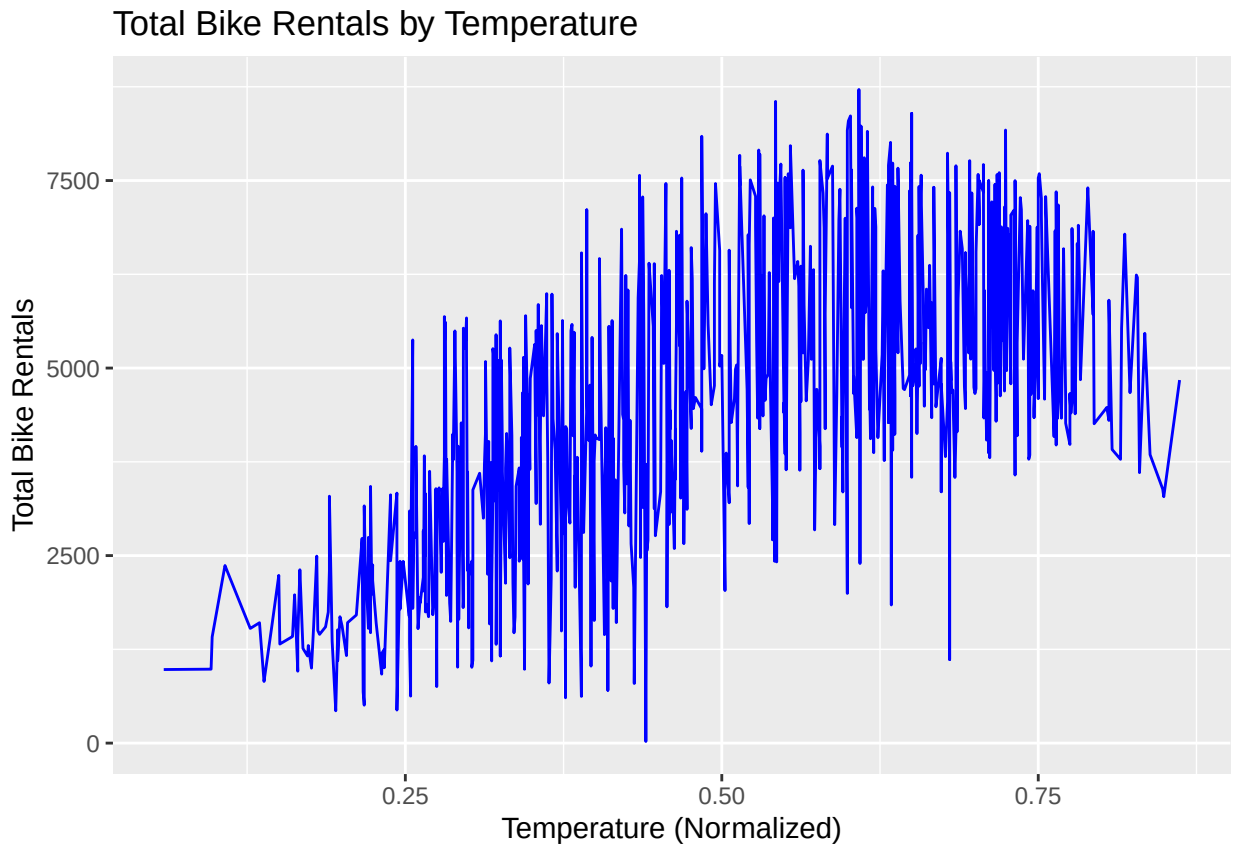
Variable	N	Mean	Std. Dev.	Min	Pctl. 25	Pctl. 75	Max
instant	731	366	211	1	184	548	731
season	731	2.5	1.1	1	2	3	4
yr	731	0.5	0.5	0	0	1	1
mnth	731	6.5	3.5	1	4	10	12
holiday	731	0.029	0.17	0	0	0	1
weekday	731	3	2	0	1	5	6
workingday	731	0.68	0.47	0	0	1	1
weathersit	731	1.4	0.54	1	1	2	3
temp	731	0.5	0.18	0.059	0.34	0.66	0.86
atemp	731	0.47	0.16	0.079	0.34	0.61	0.84
hum	731	0.63	0.14	0	0.52	0.73	0.97
windspeed	731	0.19	0.077	0.022	0.13	0.23	0.51
casual	731	848	687	2	316	1096	3410
registered	731	3656	1560	20	2497	4776	6946
cnt	731	4504	1937	22	3152	5956	8714



2.2. Continuous Variable (temp)

In below example, Line chart of Total Bike Rentals by Temperature gives the false impression of a sequential or time-based relationship and it can not show a clear trend, so this visualization offers little meaningful insight into how bike rentals respond to temperature changes.

```
ggplot(df, aes(x = temp, y = cnt)) +
  geom_line(color = "blue") +
  labs(title = "Total Bike Rentals by Temperature",
       x = "Temperature (Normalized)",
       y = "Total Bike Rentals")
```



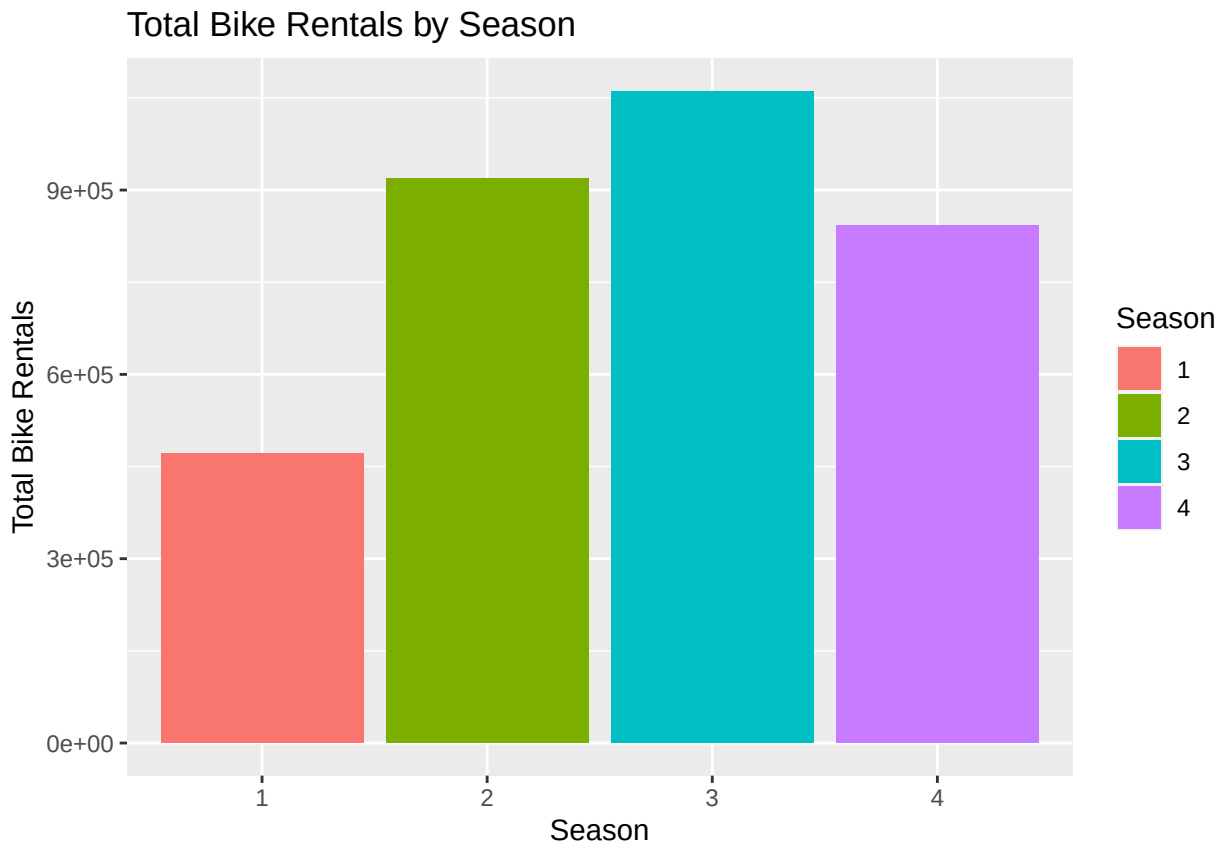
Problem 3. Good Data Visualization

3.1. Categorical Variable

For the chart shown in section 2.1, bar charts are much better for comparing categorical variables like seasons, as they show the total bike rentals for each season clearly. And we can compare the height of the bars to see which season has the highest bike rentals. In addition, the y-axis starts at zero, providing an accurate sense of scale and preventing any distortion.

```
ggplot(df, aes(x = factor(season), y = cnt, fill = factor(season))) +
  geom_bar(stat = "summary", fun = "sum") +
```

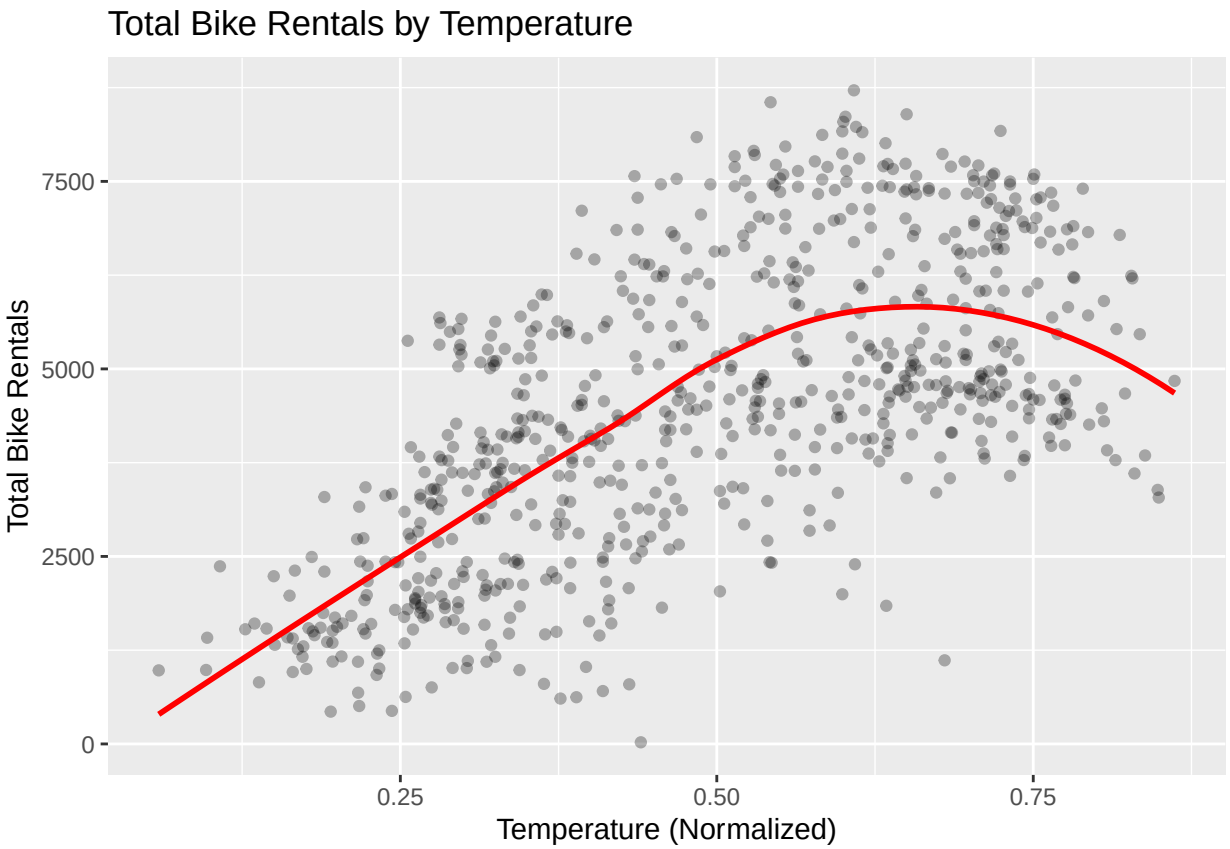
```
  labs(title = "Total Bike Rentals by Season",
       x = "Season",
       y = "Total Bike Rentals",
       fill = "Season")
```



3.2. Continuous Variable

The scatter plot is better suited for continuous data like temperature and rentals. It shows the spread and density of the data points without implying a sequential connection. On the other hand, adding a smoothing line (LOESS) or trend line gives a clear indication of the relationship between temperature and total bike rentals, helping to highlight any trends in the data.

```
ggplot(data = df, aes(x = temp, y = cnt)) +
  geom_point(alpha = 0.3) +
  geom_smooth(formula = y ~ x, method = "loess", color = "red", se = FALSE) +
  labs(title = "Total Bike Rentals by Temperature",
       x = "Temperature (Normalized)",
       y = "Total Bike Rentals")
```



Problem 4. Simple analysis

```
y <- df$cnt
X <- df[, c('season', 'mnth', 'holiday', 'weekday', 'workingday',
            'weathersit', 'temp', 'atemp', 'hum', 'windspeed')]
```

```
cor(X)
```

```
##                 season         mnth      holiday       weekday   workingday
## season      1.000000000  0.831440114 -0.010536659 -0.0030798813  0.012484963
## mnth        0.831440114  1.000000000  0.019190895  0.0095093129 -0.005900951
## holiday    -0.010536659  0.019190895  1.000000000 -0.1019602689 -0.253022700
## weekday    -0.003079881  0.009509313 -0.101960269  1.0000000000  0.035789674
## workingday  0.012484963 -0.005900951 -0.253022700  0.0357896736  1.000000000
## weathersit  0.019211028  0.043528098 -0.034626841  0.0310874694  0.061200430
## temp        0.334314856  0.220205335 -0.028555535 -0.0001699624  0.052659810
## atemp       0.342875613  0.227458630 -0.032506692 -0.0075371318  0.052182275
## hum         0.205444765  0.222203691 -0.015937479 -0.0522321004  0.024327046
## windspeed  -0.229046337 -0.207501752  0.006291507  0.0142821241 -0.018796487
##              weathersit          temp         atemp         hum    windspeed
## season       0.01921103  0.3343148564  0.342875613  0.20544476 -0.229046337
## mnth         0.04352810  0.2202053352  0.227458630  0.22220369 -0.207501752
```

```
## holiday    -0.03462684 -0.0285555350 -0.032506692 -0.01593748  0.006291507
## weekday     0.03108747 -0.0001699624 -0.007537132 -0.05223210  0.014282124
## workingday  0.06120043  0.0526598102  0.052182275  0.02432705 -0.018796487
## weathersit  1.00000000 -0.1206022365 -0.121583354  0.59104460  0.039511059
## temp       -0.12060224  1.0000000000  0.991701553  0.12696294 -0.157944120
## atemp      -0.12158335  0.9917015532  1.000000000  0.13998806 -0.183642967
## hum         0.59104460  0.1269629390  0.139988060  1.00000000 -0.248489099
## windspeed   0.03951106 -0.1579441204 -0.183642967 -0.24848910  1.000000000
```

4.1. Linear regression model with all available variables

```
full.model <- lm(y ~ ., data = X)
summary(full.model)
```

```
##
## Call:
## lm(formula = y ~ ., data = X)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -4278.2  -973.0  -190.1  1056.6  4006.7
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  3033.31     360.95   8.404 2.30e-16 ***
## season        482.12      84.11   5.732 1.46e-08 ***
## mnth          -29.27      26.23  -1.116   0.2649
## holiday      -479.95     308.83  -1.554   0.1206
## weekday        61.35      25.04   2.450   0.0145 *
## workingday    112.39     110.62   1.016   0.3099
## weathersit   -494.64     120.24  -4.114 4.35e-05 ***
## temp         2381.14    2156.21   1.104   0.2698
## atemp        3634.09    2441.58   1.488   0.1371
## hum         -2234.15     478.65  -4.668 3.63e-06 ***
## windspeed   -3145.35     700.33  -4.491 8.25e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1340 on 720 degrees of freedom
## Multiple R-squared:  0.528,  Adjusted R-squared:  0.5214
## F-statistic: 80.53 on 10 and 720 DF,  p-value: < 2.2e-16
```

4.2. Model Selection Process

Initial full and null model

```
library(MASS)

full.model <- lm(y ~ ., data = as.data.frame(X))
null.model <- lm(y ~ 1, data = as.data.frame(X))
```

4.2.1. Backward Elimination

- Starting with the full model
- Removing non-significant variables step by step

```
model.backward.aic <- stepAIC(object = full.model, scope = null.model, direction = 'backward')
```

```
summary(model.backward.aic)
```

```
##
## Call:
## lm(formula = y ~ season + holiday + weekday + weathersit + atemp +
##     hum + windspeed, data = as.data.frame(X))
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -4393.3  -957.0  -183.5  1074.3  3885.6
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  3008.14     341.53   8.808  < 2e-16 ***
## season        404.91      48.79   8.298 5.16e-16 ***
## holiday      -562.77     298.79  -1.884   0.0600 .
## weekday        62.03      24.99   2.482   0.0133 *
## weathersit   -477.28     119.88  -3.981 7.55e-05 ***
## atemp        6369.91     334.62  19.036  < 2e-16 ***
## hum         -2332.37     475.52  -4.905 1.16e-06 ***
## windspeed   -3021.61     690.98  -4.373 1.41e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1341 on 723 degrees of freedom
## Multiple R-squared:  0.5256, Adjusted R-squared:  0.521
## F-statistic: 114.4 on 7 and 723 DF,  p-value: < 2.2e-16
```

4.2.2. Forward Selection

- Start with the null model (only intercept)
- Adding variables based on significance

```
model.forward.aic <- stepAIC(object = null.model, scope = full.model$terms, direction = 'forward')
```

```
summary(model.forward.aic)
```

```
##
## Call:
## lm(formula = y ~ atemp + weathersit + season + hum + windspeed +
##     weekday + holiday, data = as.data.frame(X))
##
## Residuals:
##     Min      1Q  Median      3Q     Max
```

```
## -4393.3  -957.0  -183.5  1074.3  3885.6
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  3008.14     341.53   8.808  < 2e-16 ***
## atemp        6369.91     334.62  19.036  < 2e-16 ***
## weathersit   -477.28     119.88  -3.981 7.55e-05 ***
## season        404.91      48.79   8.298 5.16e-16 ***
## hum         -2332.37     475.52  -4.905 1.16e-06 ***
## windspeed   -3021.61     690.98  -4.373 1.41e-05 ***
## weekday        62.03      24.99   2.482   0.0133 *
## holiday      -562.77     298.79  -1.884   0.0600 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1341 on 723 degrees of freedom
## Multiple R-squared:  0.5256, Adjusted R-squared:  0.521
## F-statistic: 114.4 on 7 and 723 DF,  p-value: < 2.2e-16
```

4.2.3. Stepwise Selection

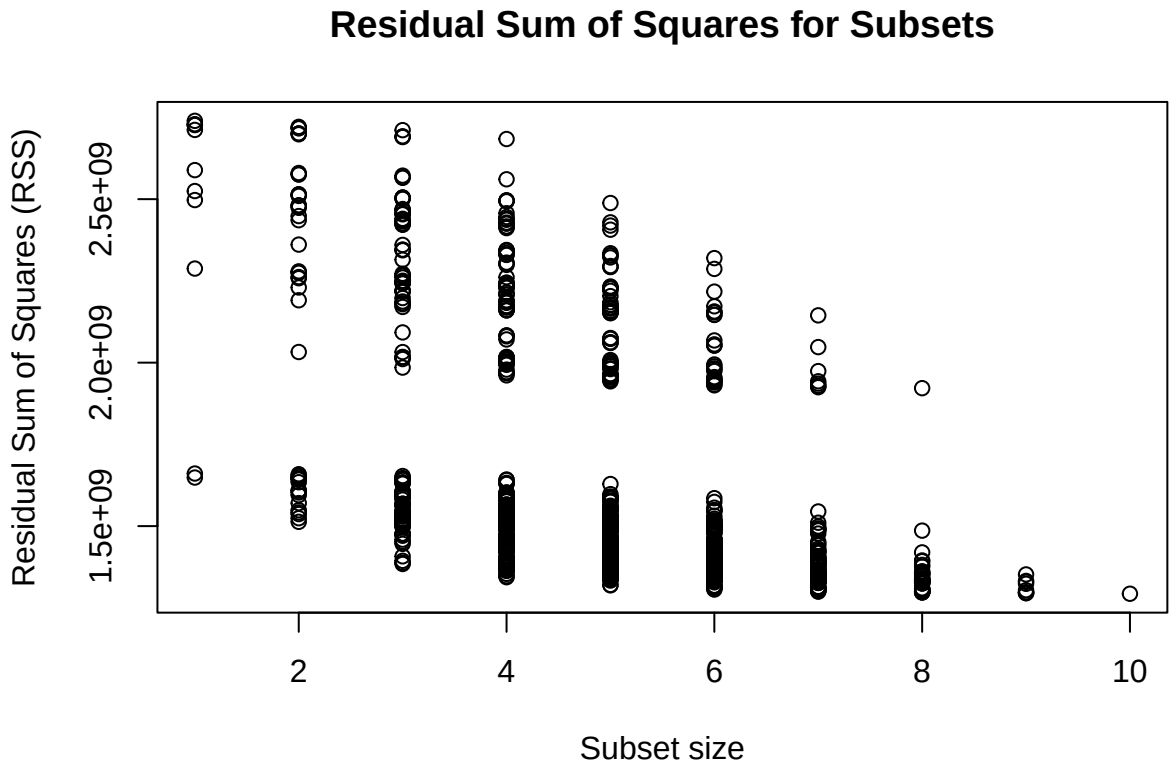
```
model.stepwise.aic <- stepAIC(object = null.model, scope = full.model$terms, direction = 'both')
```

```
summary(model.stepwise.aic)
```

```
##
## Call:
## lm(formula = y ~ atemp + weathersit + season + hum + windspeed +
##     weekday + holiday, data = as.data.frame(X))
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -4393.3  -957.0  -183.5  1074.3  3885.6
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  3008.14     341.53   8.808  < 2e-16 ***
## atemp        6369.91     334.62  19.036  < 2e-16 ***
## weathersit   -477.28     119.88  -3.981 7.55e-05 ***
## season        404.91      48.79   8.298 5.16e-16 ***
## hum         -2332.37     475.52  -4.905 1.16e-06 ***
## windspeed   -3021.61     690.98  -4.373 1.41e-05 ***
## weekday        62.03      24.99   2.482   0.0133 *
## holiday      -562.77     298.79  -1.884   0.0600 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1341 on 723 degrees of freedom
## Multiple R-squared:  0.5256, Adjusted R-squared:  0.521
## F-statistic: 114.4 on 7 and 723 DF,  p-value: < 2.2e-16
```

4.2.4. Best Subset Model

```
num_vars <- ncol(X)

allCombinations <- sapply(1:num_vars, function(m) combn(x = 1:num_vars, m = m))

null.model <- lm(y ~ 1)
result.AIC <- extractAIC(null.model)
result.RSS <- cbind(1, deviance(null.model))

for (i in 1:num_vars) {  # For each number of variables
  for (j in 1:ncol(allCombinations[[i]])) {  # For each subset of size 'i'
    # Fit model using current subset of variables
    model <- lm(y ~ ., data = as.data.frame(X[, allCombinations[[i]][, j]]))

    # Store AIC and RSS for this model
    result.AIC <- rbind(result.AIC, extractAIC(model))
    result.RSS <- rbind(result.RSS, cbind(length(allCombinations[[i]][, j]), deviance(model)))
  }
}

# Plot RSS for all subset sizes
plot(result.RSS[, 1], result.RSS[, 2], main = 'Residual Sum of Squares for Subsets',
     xlab = 'Subset size', ylab = 'Residual Sum of Squares (RSS)')
```



```r
# identify the model with the smallest RSS
best.RSS <- result.RSS[which.min(result.RSS[, 2]), ]
index <- which.min(result.RSS[result.RSS[, 1] == best.RSS[1], 2])
variables.best.model <- allCombinations[[best.RSS[1]]][, index]

# final model
model.bestSubset.RSS <- lm(y ~ ., data = as.data.frame(X[, variables.best.model]))
summary(model.bestSubset.RSS)
```

```
##
## Call:
## lm(formula = y ~ ., data = as.data.frame(X[, variables.best.model]))
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -4278.2  -973.0  -190.1  1056.6  4006.7
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  3033.31     360.95   8.404 2.30e-16 ***
## season        482.12      84.11   5.732 1.46e-08 ***
## mnth          -29.27      26.23  -1.116   0.2649
## holiday      -479.95     308.83  -1.554   0.1206
## weekday        61.35      25.04   2.450   0.0145 *
## workingday    112.39     110.62   1.016   0.3099
## weathersit   -494.64     120.24  -4.114 4.35e-05 ***
## temp         2381.14    2156.21   1.104   0.2698
## atemp        3634.09    2441.58   1.488   0.1371
## hum         -2234.15     478.65  -4.668 3.63e-06 ***
## windspeed   -3145.35     700.33  -4.491 8.25e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1340 on 720 degrees of freedom
## Multiple R-squared:  0.528,  Adjusted R-squared:  0.5214
## F-statistic: 80.53 on 10 and 720 DF,  p-value: < 2.2e-16
```

4.3. Model Performance Comparison

```r
full.model <- lm(y ~ ., data = as.data.frame(X))
null.model <- lm(y ~ 1, data = as.data.frame(X))

# Forward Selection
scp <- full.model$terms

rss1 <- vector("numeric", num_vars)
for (j in 1:num_vars) {
  mdl <- stepAIC(object = null.model, scope = scp, direction =
                   'forward', k = 0, steps = j)
  rss1[j] <- sum((mdl$fitted.values-y)^2)
}
```

```r
# Backward Elimination
rss2 <- vector("numeric", num_vars)
for (j in 1:num_vars) {
  mdl <- stepAIC(object = full.model, scope = list(lower = null.model, upper = full.model),
                 direction = 'backward', k = 1e6, steps = j-1)
  rss2[j] <- sum((mdl$fitted.values - y)^2)
}
rss2 <- rev(rss2)

# Best Subset Selection
rss3 <- vector("numeric", num_vars)
for (j in 1:num_vars) {
  rss3[j] <- min(result.RSS[result.RSS[, 1] == j, 2])
}
```

```r
plot(0, 0, xlim = c(1, num_vars), ylim = c(min(rss1, rss2, rss3), max(rss1, rss2, rss3)),
     type = "n", xlab = "Number of Variables", ylab = "Residual Sum of Squares (RSS)")
lines(1:num_vars, rss1, col = "red", type = 'p', pch = 16, lty = 1)  # Forward
lines(1:num_vars, rss2, col = "green", type = 'p', pch = 16, lty = 2)  # Backward
lines(1:num_vars, rss3, col = "blue", type = 'p', pch = 16, lty = 3)  #Should be the minimum given that
legend("topright", legend = c("Forward", "Backward", "Best Subset"), col = c("red", "green", "blue"), p
```