London Final Project DSC 520

Jennifer Ruiz

May 27, 2020

```
library(tidyr)
library(ggplot2)
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(readr)
library(gtools)
library(tidyverse)

## -- Attaching packages --------------------------------------- tidyverse 1.
3.0 --

## v tibble  3.0.1     v stringr 1.4.0
## v purrr   0.3.4     v forcats 0.5.0

## -- Conflicts ------------------------------------------ tidyverse_conflict
s() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()

library(lubridate)

##
## Attaching package: 'lubridate'

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

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

london <- read.csv("london.csv", stringsAsFactors = FALSE)
```

```r
glimpse(london)
```

```
## Rows: 17,414
## Columns: 10
## $ timestamp    <chr> "1/4/2015 0:00", "1/4/2015 1:00", "1/4/2015 2:00", "1
/...
## $ cnt          <int> 182, 138, 134, 72, 47, 46, 51, 75, 131, 301, 528, 727
,...
## $ t1           <dbl> 3.0, 3.0, 2.5, 2.0, 2.0, 2.0, 1.0, 1.0, 1.5, 2.0, 3.0
,...
## $ t2           <dbl> 2.0, 2.5, 2.5, 2.0, 0.0, 2.0, -1.0, -1.0, -1.0, -0.5,
...
## $ hum          <dbl> 93.0, 93.0, 96.5, 100.0, 93.0, 93.0, 100.0, 100.0, 96
....
## $ wind_speed   <dbl> 6.0, 5.0, 0.0, 0.0, 6.5, 4.0, 7.0, 7.0, 8.0, 9.0, 12.
0...
## $ weather_code <int> 3, 1, 1, 1, 1, 1, 4, 4, 4, 3, 3, 3, 4, 3, 3, 3, 3, 3,
...
## $ is_holiday   <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
...
## $ is_weekend   <int> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
...
## $ season       <int> 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
...
```

```r
summary(london)
```

```
##   timestamp              cnt              t1               t2
##  Length:17414       Min.   :   0   Min.   :-1.50   Min.   :-6.00
##  Class :character   1st Qu.: 257   1st Qu.: 8.00   1st Qu.: 6.00
##  Mode  :character   Median : 844   Median :12.50   Median :12.50
##                     Mean   :1143   Mean   :12.47   Mean   :11.52
##                     3rd Qu.:1672   3rd Qu.:16.00   3rd Qu.:16.00
##                     Max.   :7860   Max.   :34.00   Max.   :34.00
##       hum            wind_speed     weather_code      is_holiday
##  Min.   : 20.50   Min.   : 0.00   Min.   : 1.000   Min.   :0.00000
##  1st Qu.: 63.00   1st Qu.:10.00   1st Qu.: 1.000   1st Qu.:0.00000
##  Median : 74.50   Median :15.00   Median : 2.000   Median :0.00000
##  Mean   : 72.32   Mean   :15.91   Mean   : 2.723   Mean   :0.02205
##  3rd Qu.: 83.00   3rd Qu.:20.50   3rd Qu.: 3.000   3rd Qu.:0.00000
##  Max.   :100.00   Max.   :56.50   Max.   :26.000   Max.   :1.00000
##    is_weekend         season
##  Min.   :0.0000   Min.   :0.000
##  1st Qu.:0.0000   1st Qu.:0.000
##  Median :0.0000   Median :1.000
##  Mean   :0.2854   Mean   :1.492
##  3rd Qu.:1.0000   3rd Qu.:2.000
##  Max.   :1.0000   Max.   :3.000
```

```r
Times <- format(as.POSIXct(strptime(london$timestamp,"%m/%d/%Y %H:%M",tz=""))
,format = "%H:%M")
```

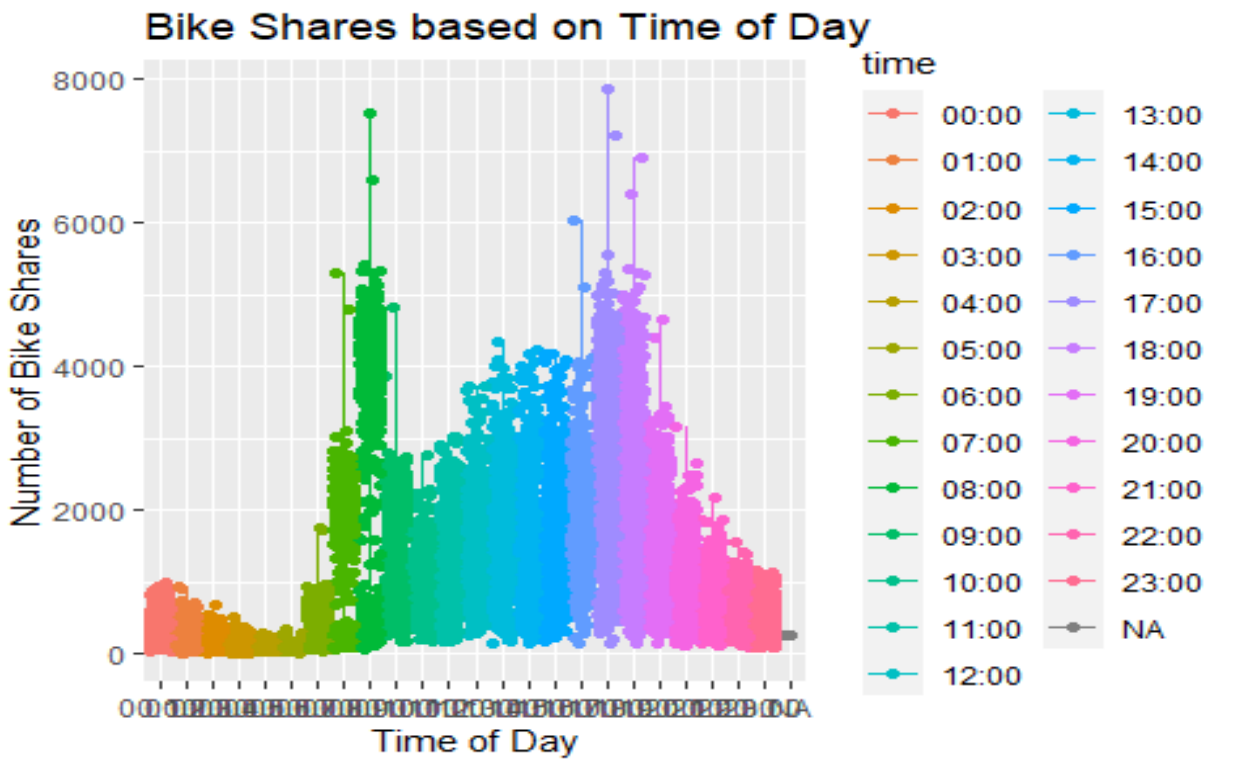
```r
Dates <- format(as.POSIXct(strptime(london$timestamp,"%m/%d/%Y %H:%M",tz=""))
,format = "%m/%d/%Y")
## had to separate out date and time to properly work with the variable and c
ombine datasets

london$time <- Times
london$date <- Dates

head(london)
```

```
##       timestamp cnt  t1  t2   hum wind_speed weather_code is_holiday is_we
ekend
## 1 1/4/2015 0:00 182 3.0 2.0  93.0        6.0            3          0
1
## 2 1/4/2015 1:00 138 3.0 2.5  93.0        5.0            1          0
1
## 3 1/4/2015 2:00 134 2.5 2.5  96.5        0.0            1          0
1
## 4 1/4/2015 3:00  72 2.0 2.0 100.0        0.0            1          0
1
## 5 1/4/2015 4:00  47 2.0 0.0  93.0        6.5            1          0
1
## 6 1/4/2015 5:00  46 2.0 2.0  93.0        4.0            1          0
1
##   season  time       date
## 1      3 00:00 01/04/2015
## 2      3 01:00 01/04/2015
## 3      3 02:00 01/04/2015
## 4      3 03:00 01/04/2015
## 5      3 04:00 01/04/2015
## 6      3 05:00 01/04/2015
```

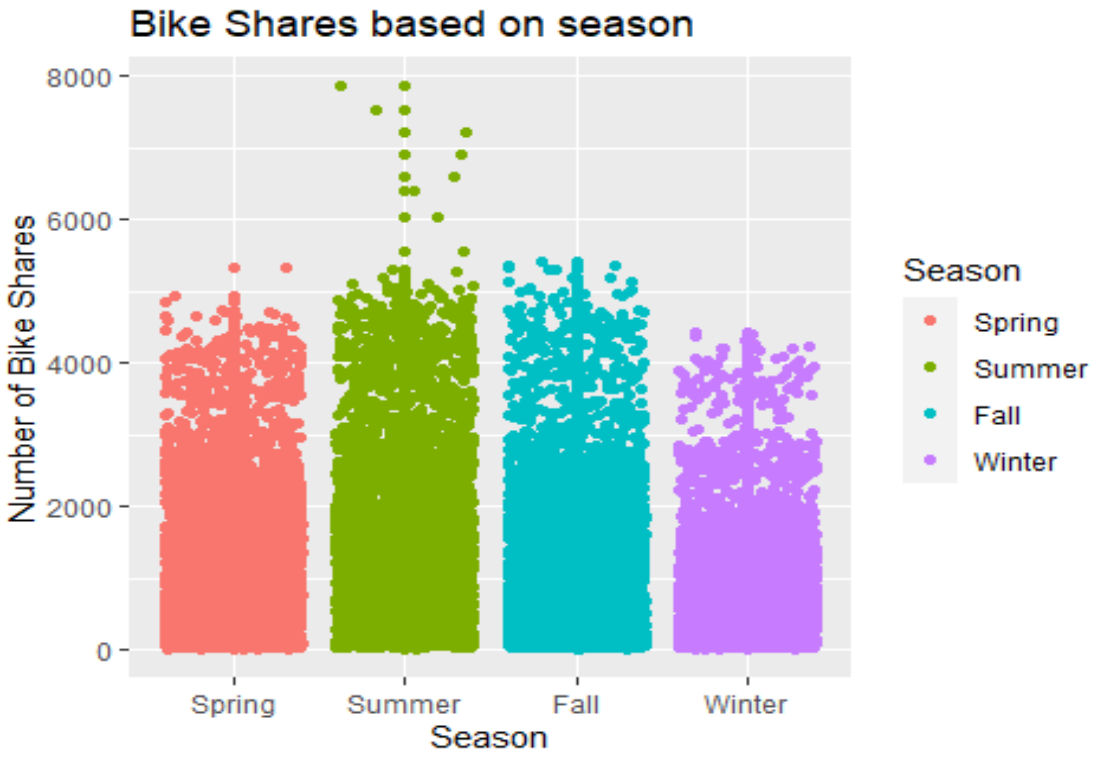
```r
london %>%
  rename(
    actual_temp = t1,
    feels_like = t2
    )


# removing timestamp column from dataset after separating out date and time i
nformation
london$timestamp <- NULL

london$weather_code <- as.factor(london$weather_code)
london$is_holiday <- as.factor(london$is_holiday)
london$is_weekend <- as.factor(london$is_weekend)
london$season <- as.factor(london$season)
```

Starting Data Visualizations

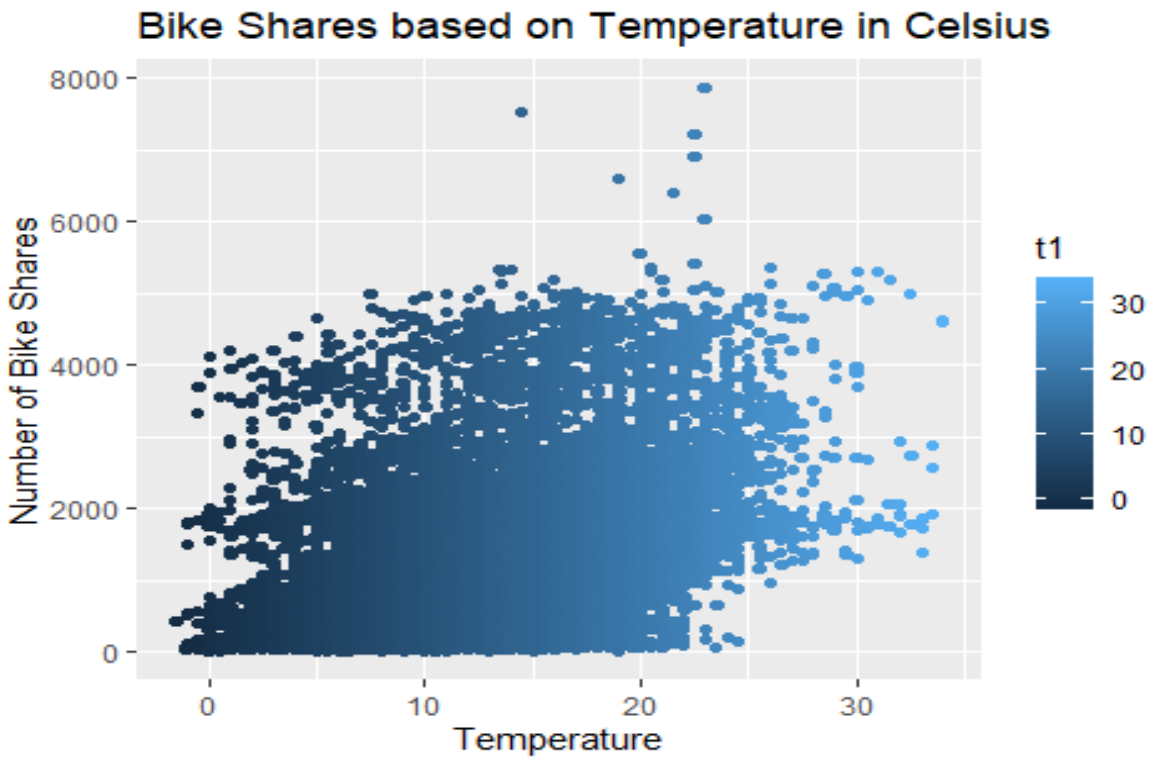
```
# visualizing number of bike shares by time
ggplot(london, aes(x=time, y =cnt, color = time)) + geom_line() + geom_jitter
() +xlab("Time of Day") +ylab("Number of Bike Shares") + ggtitle("Bike Shares
based on Time of Day")
```



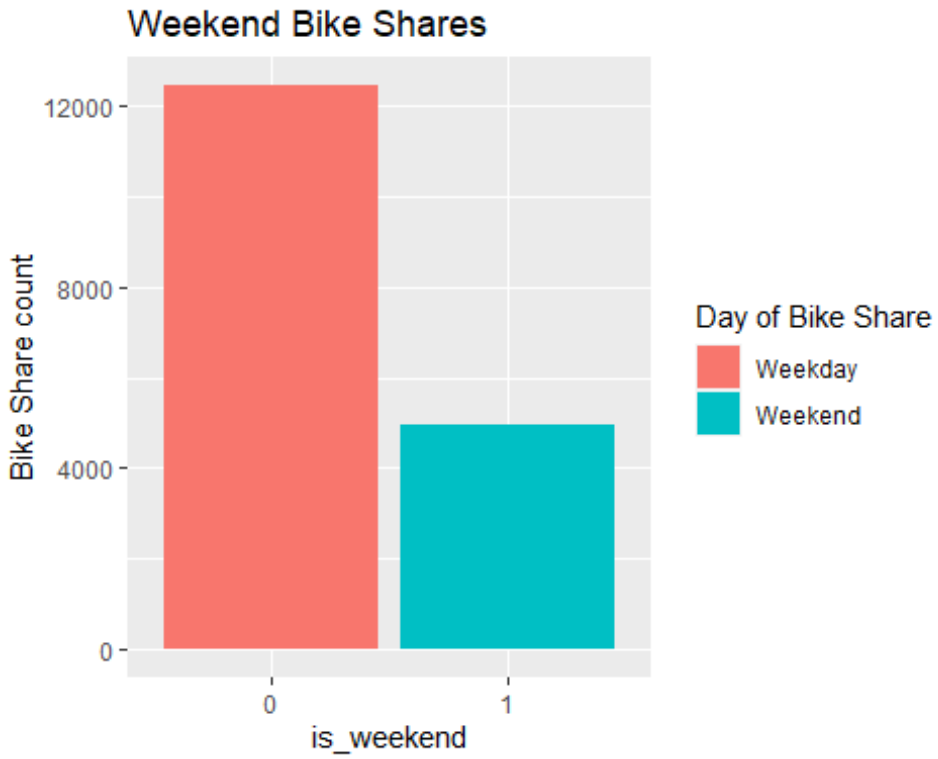

```
 Season_info <- c("Spring", "Summer", "Fall", "Winter")

# visualizing number of bike shares by seaon
ggplot(london, aes(x=season, y = cnt, color = season)) +geom_point() + geom_j
itter() + xlab("Season") + ylab("Number of Bike Shares") + ggtitle("Bike Shar
es based on season") + scale_color_discrete(name = "Season", labels = c("Spri
ng", "Summer", "Fall", "Winter")) + scale_x_discrete(labels= Season_info)
```




```
#Visualizing bike shares based on season
ggplot(london, aes(x=t1, y = cnt, color=t1)) +geom_point() + geom_jitter() +
xlab("Temperature") + ylab("Number of Bike Shares") + ggtitle("Bike Shares ba
sed on Temperature in Celsius")
```



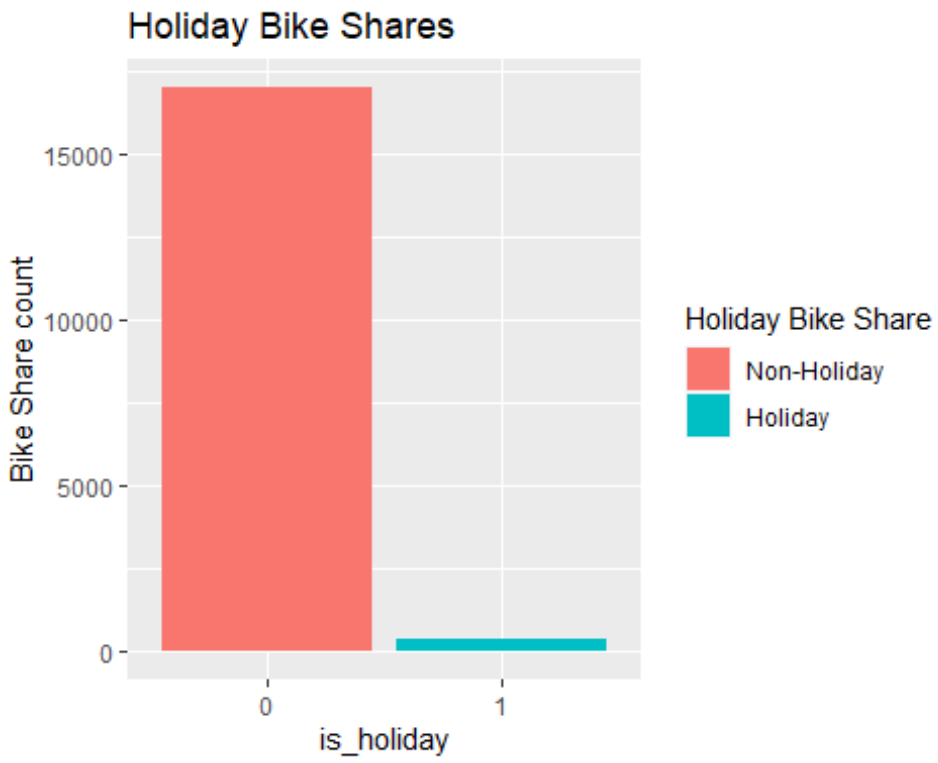

```r
#visualizing weekend vs weekday data
ggplot(london, aes(x=is_weekend, fill= is_weekend)) + geom_bar() +labs(y= "Bi
ke Share count") + ggtitle("Weekend Bike Shares") + scale_fill_discrete(name
= "Day of Bike Share", labels = c("Weekday", "Weekend"))
```



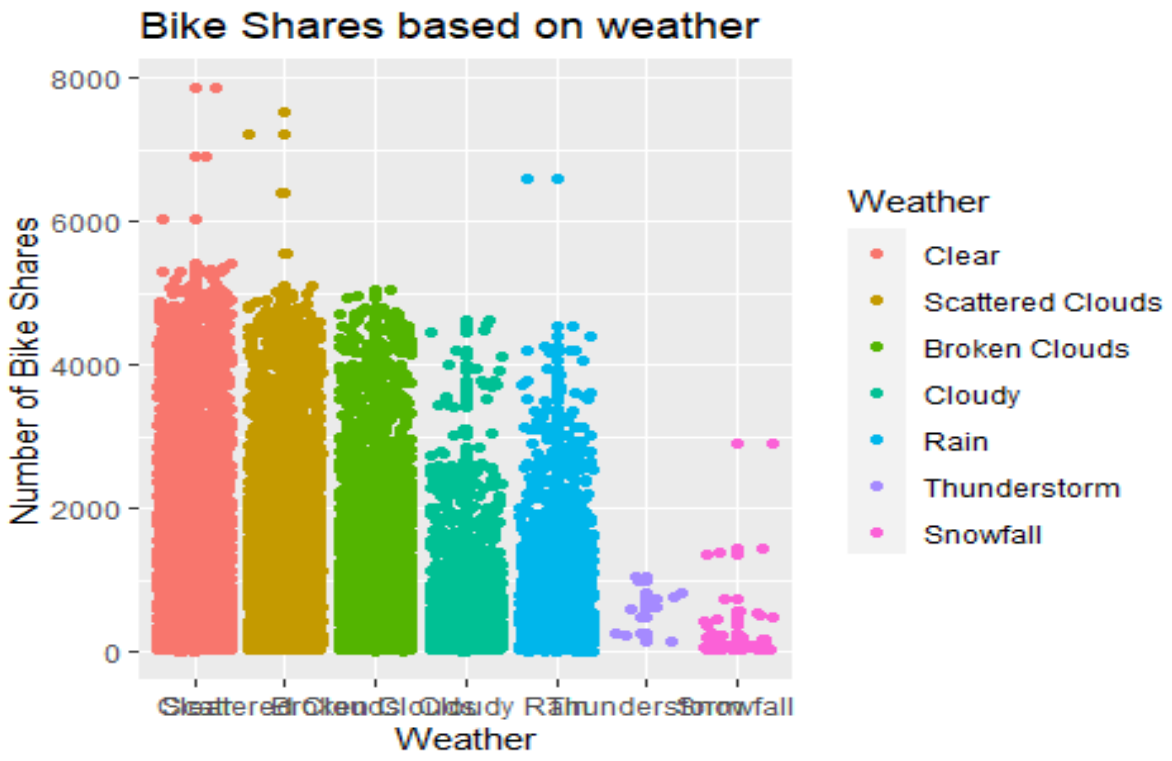

```
#Examining the probability of when rides occur
prop.table(table(london$is_weekend))

##
##         0         1
## 0.7145975 0.2854025
```

Probability table indicates that 71% of ride shares occur during Weekdays and only 29% occur on the weekends.

```
#visualizing holiday data
ggplot(london, aes(x=is_holiday, fill= is_holiday)) + geom_bar() +labs(y= "Bi
ke Share count") + ggtitle("Holiday Bike Shares") + scale_fill_discrete(name
= "Holiday Bike Share", labels = c("Non-Holiday", "Holiday"))
```




```
 weather_info <- c("Clear", "Scattered Clouds", "Broken Clouds", "Cloudy", "R
ain", "Thunderstorm", "Snowfall", "Freezing Fog")

# visualizing number of bike shares by seaon
ggplot(london, aes(x=weather_code, y = cnt, color = weather_code)) +geom_poin
t() + geom_jitter() + xlab("Weather") + ylab("Number of Bike Shares") + ggtit
le("Bike Shares based on weather") + scale_color_discrete(name = "Weather", l
abels = weather_info) + scale_x_discrete(labels= weather_info)
```




```
# Looking at Correlations within the data
london_num = london[,c(1:5)]

res <- cor(london_num)
round(res, 2)
```

```
##              cnt    t1    t2   hum wind_speed
## cnt         1.00  0.39  0.37 -0.46       0.12
## t1          0.39  1.00  0.99 -0.45       0.15
## t2          0.37  0.99  1.00 -0.40       0.09
## hum        -0.46 -0.45 -0.40  1.00      -0.29
## wind_speed  0.12  0.15  0.09 -0.29       1.00
```

```
res
```

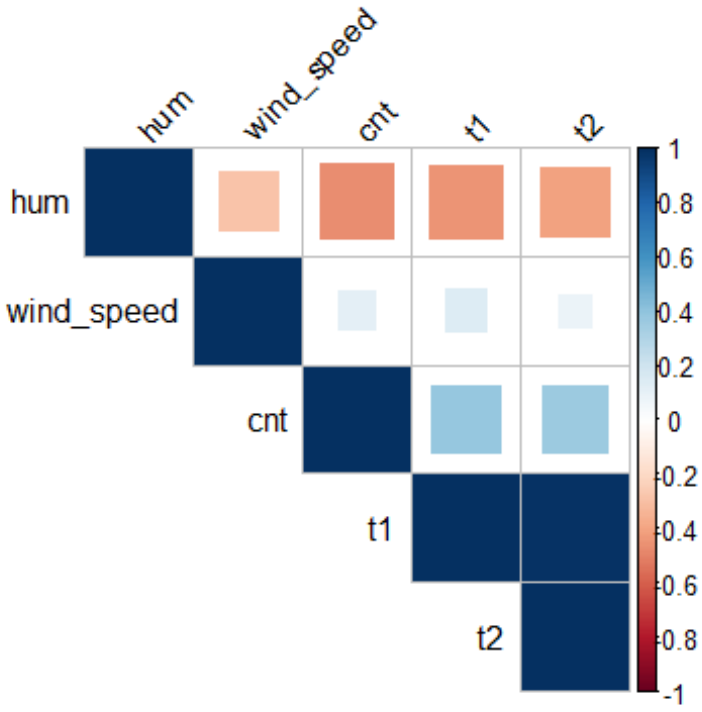
```
##                  cnt         t1          t2        hum  wind_speed
## cnt         1.0000000  0.3887985  0.36903479 -0.4629010  0.11629523
## t1          0.3887985  1.0000000  0.98834422 -0.4477810  0.14547097
## t2          0.3690348  0.9883442  1.00000000 -0.4034951  0.08840854
## hum        -0.4629010 -0.4477810 -0.40349514  1.0000000 -0.28778917
## wind_speed  0.1162952  0.1454710  0.08840854 -0.2877892  1.00000000
```

```
#install.packages("Hmisc")
library(Hmisc)
```

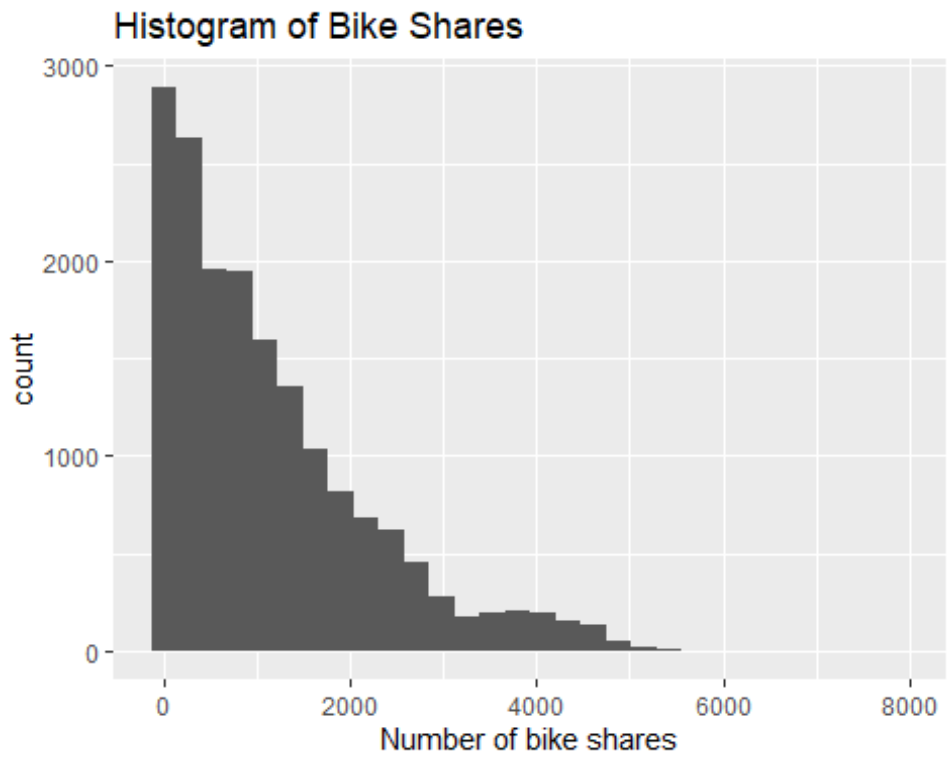
```
## Loading required package: lattice
```

```
## Loading required package: survival
```

```
## Loading required package: Formula

##
## Attaching package: 'Hmisc'

## The following objects are masked from 'package:dplyr':
##
##     src, summarize

## The following objects are masked from 'package:base':
##
##     format.pval, units

flattenCorrMatrix <- function(cormat, pmat) {
  ut <- upper.tri(cormat)
  data.frame(
    row = rownames(cormat)[row(cormat)[ut]],
    column = rownames(cormat)[col(cormat)[ut]],
    cor  =(cormat)[ut],
    p = pmat[ut]
    )
}

res2<-rcorr(as.matrix(res))
flattenCorrMatrix(res2$r, res2$P)

##    row     column         cor            p
## 1  cnt         t1  0.60563108 2.790364e-01
## 2  cnt         t2  0.58394783 3.012289e-01
## 3   t1         t2  0.99815988 9.472925e-05
## 4  cnt        hum -0.83528686 7.823374e-02
## 5   t1        hum -0.82367014 8.649454e-02
## 6   t2        hum -0.79093824 1.110797e-01
## 7  cnt wind_speed  0.15477614 8.037226e-01
## 8   t1 wind_speed  0.09639120 8.774612e-01
## 9   t2 wind_speed  0.04001319 9.490672e-01
## 10 hum wind_speed -0.50394852 3.866541e-01

library(corrplot)

## corrplot 0.84 loaded

corrplot(res, type = "upper", order = "hclust",
         tl.col = "black", tl.srt = 45, method = "square")
```




```
# Histogram for cnt (number of bike shares)

ggplot(london, aes(x=cnt)) + geom_histogram() + labs(x= "Number of bike share
s") + ggtitle("Histogram of Bike Shares")

## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```




```
## creating models

mod <- lm(cnt ~ t1, data = london)

summary(mod)

##
## Call:
## lm(formula = cnt ~ t1, data = london)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1930.4  -680.0  -227.9   426.8  6234.0
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   199.04      18.57   10.72   <2e-16 ***
## t1             75.72       1.36   55.69   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 999.8 on 17412 degrees of freedom
## Multiple R-squared:  0.1512, Adjusted R-squared:  0.1511
## F-statistic:  3101 on 1 and 17412 DF,  p-value: < 2.2e-16

#finding the correlation coefficient of the model
sqrt(0.15)
```

```
## [1] 0.3872983
```

The correlation coefficient of the model is 0.39. This indicates a mild, positive correlation between the variables.

```
# building second model

mod2<- lm(cnt ~ season + weather_code + wind_speed, data = london)

summary(mod2)

##
## Call:
## lm(formula = cnt ~ season + weather_code + wind_speed, data = london)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1836.9  -704.0  -244.1   457.7  6476.7
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)     856.085     23.326  36.700  < 2e-16 ***
## season1         338.411     21.836  15.498  < 2e-16 ***
## season2         107.719     22.093   4.876 1.09e-06 ***
## season3        -254.819     22.027 -11.568  < 2e-16 ***
## weather_code2   272.282     21.179  12.856  < 2e-16 ***
## weather_code3     5.643     21.995   0.257   0.7975
## weather_code4  -443.212     29.946 -14.800  < 2e-16 ***
## weather_code7  -475.673     26.120 -18.211  < 2e-16 ***
## weather_code10 -685.223    273.355  -2.507   0.0122 *
## weather_code26 -742.988    132.939  -5.589 2.32e-08 ***
## wind_speed       17.164      1.024  16.761  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1021 on 17403 degrees of freedom
## Multiple R-squared:  0.1145, Adjusted R-squared:  0.114
## F-statistic:   225 on 10 and 17403 DF,  p-value: < 2.2e-16

#finding the correlation coefficient
sqrt(0.11)

## [1] 0.3316625
```

The correlation coefficient is 0.33, which indicates a mild, positive correlation between the variables.

```
#nstall.packages("lm.beta")
#nstall.packages("lmtest")
```

```r
## calculating the standardized betas for the multiple regression mode

library(lm.beta)
lm.beta(mod2)
```

```
##
## Call:
## lm(formula = cnt ~ season + weather_code + wind_speed, data = london)
##
## Standardized Coefficients::
##    (Intercept)        season1        season2        season3  weather_code2
##    0.000000000    0.135391582    0.042819033   -0.101504895    0.105865714
##  weather_code3  weather_code4  weather_code7 weather_code10 weather_code26
##    0.002095351   -0.113345311   -0.143952861   -0.017898297   -0.040123458
##     wind_speed
##    0.124876271
```

```r
# calculating the confidence intervals
confint(mod2)
```

```
##                       2.5 %     97.5 %
## (Intercept)      810.36287  901.80672
## season1          295.61065  381.21171
## season2           64.41501  151.02300
## season3         -297.99458 -211.64345
## weather_code2    230.76854  313.79478
## weather_code3    -37.46860   48.75464
## weather_code4   -501.90988 -384.51459
## weather_code7   -526.87054 -424.47624
## weather_code10 -1221.02720 -149.41947
## weather_code26 -1003.56206 -482.41460
## wind_speed        15.15704   19.17143
```

```r
# performing anova to compare models
anova(mod, mod2)
```

```
## Analysis of Variance Table
##
## Model 1: cnt ~ t1
## Model 2: cnt ~ season + weather_code + wind_speed
##   Res.Df        RSS Df  Sum of Sq F Pr(>F)
## 1  17412 1.7404e+10
## 2  17403 1.8155e+10  9 -751674100
```

This Model 2 does not significantly improve the model fit compared to Model 1.

```r
# Creating 3rd model
mod3 <- lm(cnt ~ t1 + time + is_weekend, data = london)

summary(mod3)
```

```
##
## Call:
## lm(formula = cnt ~ t1 + time + is_weekend, data = london)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2934.5  -274.1   -23.2   273.0  4514.3
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -226.3574    25.0640  -9.031  < 2e-16 ***
## t1            51.8466     0.8743  59.303  < 2e-16 ***
## time01:00    -76.3240    32.3324  -2.361   0.0183 *
## time02:00   -130.3525    32.3906  -4.024 5.74e-05 ***
## time03:00   -159.7166    32.3712  -4.934 8.13e-07 ***
## time04:00   -171.5881    32.3744  -5.300 1.17e-07 ***
## time05:00   -128.8010    32.3767  -3.978 6.97e-05 ***
## time06:00    222.7867    32.3189   6.893 5.64e-12 ***
## time07:00   1204.4450    32.3124  37.275  < 2e-16 ***
## time08:00   2586.1271    32.3318  79.987  < 2e-16 ***
## time09:00   1312.7295    32.3091  40.630  < 2e-16 ***
## time10:00    680.6213    32.3585  21.034  < 2e-16 ***
## time11:00    730.8413    32.3724  22.576  < 2e-16 ***
## time12:00    985.9136    32.3855  30.443  < 2e-16 ***
## time13:00   1039.7388    32.4233  32.068  < 2e-16 ***
## time14:00    995.0007    32.4408  30.671  < 2e-16 ***
## time15:00   1086.9471    32.4308  33.516  < 2e-16 ***
## time16:00   1400.7974    32.4057  43.227  < 2e-16 ***
## time17:00   2379.6090    32.3997  73.445  < 2e-16 ***
## time18:00   2200.5399    32.3715  67.978  < 2e-16 ***
## time19:00   1248.9232    32.3538  38.602  < 2e-16 ***
## time20:00    686.3624    32.3287  21.231  < 2e-16 ***
## time21:00    394.0702    32.3237  12.191  < 2e-16 ***
## time22:00    267.4307    32.3259   8.273  < 2e-16 ***
## time23:00    132.4982    32.3552   4.095 4.24e-05 ***
## is_weekend1 -231.1552    10.3239 -22.390  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 615.2 on 17386 degrees of freedom
##   (2 observations deleted due to missingness)
## Multiple R-squared:  0.6791, Adjusted R-squared:  0.6786
## F-statistic:  1472 on 25 and 17386 DF,  p-value: < 2.2e-16
```

```
#calculating correlation coefficient of model 3
sqrt(0.68)
```

```
## [1] 0.8246211
```

The correlation coefficient is 0.82 indicating a strong positive correlation.

```
lm.beta(mod3)
```

```
##
## Call:
## lm(formula = cnt ~ t1 + time + is_weekend, data = london)
##
## Standardized Coefficients::
## (Intercept)          t1   time01:00   time02:00   time03:00   time04:00
##  0.00000000  0.26620766 -0.01404155 -0.02390196 -0.02932523 -0.03150492
##   time05:00   time06:00   time07:00   time08:00   time09:00   time10:00
## -0.02364888  0.04104084  0.22187785  0.47577742  0.24198480  0.12529860
##   time11:00   time12:00   time13:00   time14:00   time15:00   time16:00
##  0.13472119  0.18197939  0.19178850  0.18353619  0.20062810  0.25872796
##   time17:00   time18:00   time19:00   time20:00   time21:00   time22:00
##  0.43893873  0.40590796  0.23022294  0.12652208  0.07259397  0.04923251
##   time23:00 is_weekend1
##  0.02434386 -0.09619599
```

```
confint(mod3)
```

```
##                   2.5 %     97.5 %
## (Intercept) -275.48530 -177.22956
## t1            50.13296   53.56025
## time01:00   -139.69885  -12.94916
## time02:00   -193.84131  -66.86372
## time03:00   -223.16733  -96.26589
## time04:00   -235.04523 -108.13092
## time05:00   -192.26263  -65.33933
## time06:00    159.43838  286.13505
## time07:00   1141.10956 1267.78053
## time08:00   2522.75357 2649.50068
## time09:00   1249.40036 1376.05868
## time10:00    617.19537  744.04726
## time11:00    667.38823  794.29436
## time12:00    922.43471 1049.39244
## time13:00    976.18587 1103.29170
## time14:00    931.41350 1058.58797
## time15:00   1023.37953 1150.51463
## time16:00   1337.27900 1464.31589
## time17:00   2316.10233 2443.11559
## time18:00   2137.08845 2263.99142
## time19:00   1185.50659 1312.33989
## time20:00    622.99488  749.72989
## time21:00    330.71256  457.42784
## time22:00    204.06883  330.79266
## time23:00     69.07881  195.91767
## is_weekend1 -251.39117 -210.91922
```

These variables are good predictors as they have tight confidence intervals that do not cross zero. This also indicates that they are representative of the values of the population.

```
library(car)
```

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

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

```
## The following object is masked from 'package:purrr':
##
##     some
```

```
## The following object is masked from 'package:gtools':
##
##     logit
```

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

```
library(ROCR)
library(dplyr)
library(class)
library(caret)
```

```
##
## Attaching package: 'caret'
```

```
## The following object is masked from 'package:survival':
##
##     cluster
```

```
## The following object is masked from 'package:purrr':
##
##     lift
```

```
library(caTools)
```

```
#create random number that is 90% of the total number of rows
ran <- sample(1:nrow(london), 0.9 * nrow(london))
#creating the normalization function
nor <- function(x) { (x-min(x))/ (max(x)-min(x))}
london_norm <- as.data.frame(lapply(london[,c(1,2,3)], nor))
summary(london_norm)
```

```
##       cnt                t1                t2
##  Min.   :0.0000   Min.   :0.0000   Min.   :0.0000
##  1st Qu.:0.0327   1st Qu.:0.2676   1st Qu.:0.3000
##  Median :0.1074   Median :0.3944   Median :0.4625
```

```
##  Mean   :0.1454   Mean   :0.3935   Mean   :0.4380
##  3rd Qu.:0.2127   3rd Qu.:0.4930   3rd Qu.:0.5500
##  Max.   :1.0000   Max.   :1.0000   Max.   :1.0000
```

```
#training set
london_train <- london_norm[ran,]

#test set
london_test <- london_norm[-ran,]
#extract 1st column of train and test datasets because it will be used as "cl
" argument
london_target_category <- london[ran, 1]

london_test_category <- london[-ran,1]
# used the formula k = sqrt(n)/2 to find k value. Used training dataset n for
value
pr <- knn(london_train, london_test, cl=london_target_category, k=6)
#create the confusion matrix
tab <- table(pr,london_test_category)
accuracy <- function (x) {sum(diag(x)/ (sum(rowSums(x)))) * 100}
accuracy(tab)
```

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

End of Project!!!