Overview

Background

With increasing urban population, traffic congestion and saturation and/or lack of public transportation bike sharing proved to be an ingenious environment friendly solution for daily commuters. There has been steady increase in the number of bike share programs worldwide reaching 1608 bike share programs with a fleet of 18.2 million bikes in 2018.

Despite the steady growth in bike sharing programs one of the key challenges faced by aggregators is to estimate the demand for bikes and allocate resources accordingly as the usage rates vary from around three to eight trips per bicycle per day globally2. The variation in usage could be due to multitude of factors one of which we believe are the prevalent weather conditions. We can expect that passengers are more likely to choose bike rides on days when the weather is pleasant without snowfall and/or heavy winds. Another important factor is time during the day. The demand is more during morning and evening peak traffic hours, and lesser during other times of the day.

Further, a study carried out by Bowman Cutter and Matthew Neidell's on the effect of voluntary information disclosure of information on air quality urging people to reduce ozone emissions found that there is an increase in people choosing alternate methods of transportation on days such warnings are issued, supporting the idea that weather parameters have an effect on individual's behavior and choices.

Data Description

The response variable is:
Y (Cnt): Total bikes rented by both casual & registered users together

The predicting variables are:
X_1 (Instant): Record index
X_2 (Dteday): Day on which the observation is made
X_3 (Season): Season which the observation is made (1 = Winter, 2 = Spring, 3 = Summer, 4 = Fall)
X_4 (Yr): Year on which the observation is made
X_5 (Mnth): Month on which the observation is made
X_6 (Hr): Day on which the observation is made (0 through 23)
X_7 (Holiday): Indictor of a public holiday or not (1 = public holiday, 0 = not a public holiday)
X_8 (Weekday): Day of week (0 through 6)
X_9 (Working day): Indicator of a working day (1 = working day, 0 = not a working day)
X_{10} (Weathersit): Weather condition (1 = Clear, Few clouds, Partly cloudy, Partly cloudy, 2 = Mist & Cloudy, Mist & Broken clouds, Mist & Few clouds, Mist, 3 = Light Snow, Light Rain, Thunderstorm & Scattered clouds, Light Rain & Scattered clouds, 4 = Heavy Rain, Ice Pallets, Thunderstorm & Mist, Snow & Fog)
X_{11} (Temp): Normalized temperature in Celsius
X_{12} (Atemp): Normalized feeling temperature in Celsius
X_{13} (Hum): Normalized humidity
X_{14} (Windspeed): Normalized wind speed
X_{15} (Casual): Bikes rented by casual users in that hour
X_{16} (Registered): Bikes rented by registered users in that hour

Reading data

```r
# clear memory
rm(list=ls())
# Set colors
gtblue = rgb(0, 48, 87, maxColorValue = 255)
```

```r
techgold = rgb(179, 163, 105, maxColorValue = 255)
buzzgold = rgb(234, 170, 0, maxColorValue = 255)
bobbyjones = rgb(55, 113, 23, maxColorValue = 255)
# Read the data using read.csv
setwd('C:/NSerban/Courses/Regression Analysis/')
data = read.csv("Bikes.csv")
# Show the number of observations
obs = nrow(data)
cat("There are", obs, "observations in the data")
```

```
## There are 17379 observations in the data
```

Preparing the Data

```r
# Set a seed for reproducibility
set.seed(9)

# Remove the irrelevant columns
clean_data = data[-c(1,2,9,15,16)]

# Convert the numerical categorical variables to predictors
clean_data$season = as.factor(clean_data$season)
clean_data$yr = as.factor(clean_data$yr)
clean_data$mnth = as.factor(clean_data$mnth)
clean_data$hr = as.factor(clean_data$hr)
clean_data$holiday = as.factor(clean_data$holiday)
clean_data$weekday = as.factor(clean_data$weekday)
clean_data$weathersit = as.factor(clean_data$weathersit)

# 80% Train 20% Test split
sample_size = floor(0.8*nrow(clean_data))
picked = sample(seq_len(nrow(clean_data)), size=sample_size)
train = clean_data[picked,]
test = clean_data[-picked,]
```

Creating the Model

```r
# Create a Poisson regression model
model1 = glm(cnt ~ ., data=train, family='poisson')
summary(model1)
```

```
##
## Call:
## glm(formula = cnt ~ ., family = "poisson", data = train)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -24.6089   -3.7805   -0.8685    3.0436   22.6553
##
## Coefficients:
##               Estimate Std. Error  z value Pr(>|z|)
## (Intercept)  2.936590   0.007629  384.941   <2e-16 ***
```

```
## season2      0.265486   0.004129   64.298   <2e-16 ***
## season3      0.255689   0.004730   54.059   <2e-16 ***
## season4      0.448706   0.004582   97.918   <2e-16 ***
## yr1          0.468400   0.001289  363.518   <2e-16 ***
## mnth2        0.115282   0.004247   27.143   <2e-16 ***
## mnth3        0.235149   0.004422   53.179   <2e-16 ***
## mnth4        0.210302   0.005857   35.909   <2e-16 ***
## mnth5        0.271895   0.006138   44.295   <2e-16 ***
## mnth6        0.223900   0.006247   35.840   <2e-16 ***
## mnth7        0.121634   0.006787   17.921   <2e-16 ***
## mnth8        0.222757   0.006617   33.667   <2e-16 ***
## mnth9        0.309161   0.006082   50.831   <2e-16 ***
## mnth10       0.213624   0.006047   35.328   <2e-16 ***
## mnth11       0.075836   0.005951   12.744   <2e-16 ***
## mnth12       0.074580   0.005259   14.182   <2e-16 ***
## hr1         -0.479025   0.009065  -52.844   <2e-16 ***
## hr2         -0.845300   0.010406  -81.231   <2e-16 ***
## hr3         -1.545448   0.013822 -111.810   <2e-16 ***
## hr4         -2.107926   0.017546 -120.139   <2e-16 ***
## hr5         -0.979853   0.011115  -88.157   <2e-16 ***
## hr6          0.374431   0.007476   50.083   <2e-16 ***
## hr7          1.419890   0.006360  223.257   <2e-16 ***
## hr8          1.887954   0.006124  308.268   <2e-16 ***
## hr9          1.388284   0.006355  218.470   <2e-16 ***
## hr10         1.129249   0.006513  173.376   <2e-16 ***
## hr11         1.261107   0.006451  195.485   <2e-16 ***
## hr12         1.429817   0.006346  225.322   <2e-16 ***
## hr13         1.417036   0.006371  222.430   <2e-16 ***
## hr14         1.353248   0.006444  210.014   <2e-16 ***
## hr15         1.394908   0.006423  217.177   <2e-16 ***
## hr16         1.615193   0.006294  256.631   <2e-16 ***
## hr17         2.020408   0.006141  329.000   <2e-16 ***
## hr18         1.968755   0.006134  320.942   <2e-16 ***
## hr19         1.667622   0.006226  267.860   <2e-16 ***
## hr20         1.362250   0.006365  214.029   <2e-16 ***
## hr21         1.109951   0.006542  169.659   <2e-16 ***
## hr22         0.842977   0.006791  124.130   <2e-16 ***
## hr23         0.474647   0.007194   65.974   <2e-16 ***
## holiday1    -0.163996   0.004166  -39.361   <2e-16 ***
## weekday1     0.051117   0.002406   21.249   <2e-16 ***
## weekday2     0.056214   0.002345   23.968   <2e-16 ***
## weekday3     0.061986   0.002348   26.400   <2e-16 ***
## weekday4     0.057992   0.002337   24.818   <2e-16 ***
## weekday5     0.092797   0.002323   39.941   <2e-16 ***
## weekday6     0.068880   0.002349   29.322   <2e-16 ***
## weathersit2 -0.075864   0.001589  -47.755   <2e-16 ***
## weathersit3 -0.488664   0.003226 -151.481   <2e-16 ***
## temp         0.195809   0.020820    9.405   <2e-16 ***
## atemp        0.880985   0.021633   40.723   <2e-16 ***
## hum         -0.209942   0.004613  -45.514   <2e-16 ***
## windspeed   -0.102402   0.005446  -18.802   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

```
## (Dispersion parameter for poisson family taken to be 1)
##
##     Null deviance: 2311056  on 13902  degrees of freedom
## Residual deviance:  458653  on 13851  degrees of freedom
## AIC: 547438
##
## Number of Fisher Scoring iterations: 5
```

Finding Insignificant Variables

```r
which(summary(model1)$coeff[,4]>0.05)
```

```
## named integer(0)
```

Evaluate goodness-of-fit

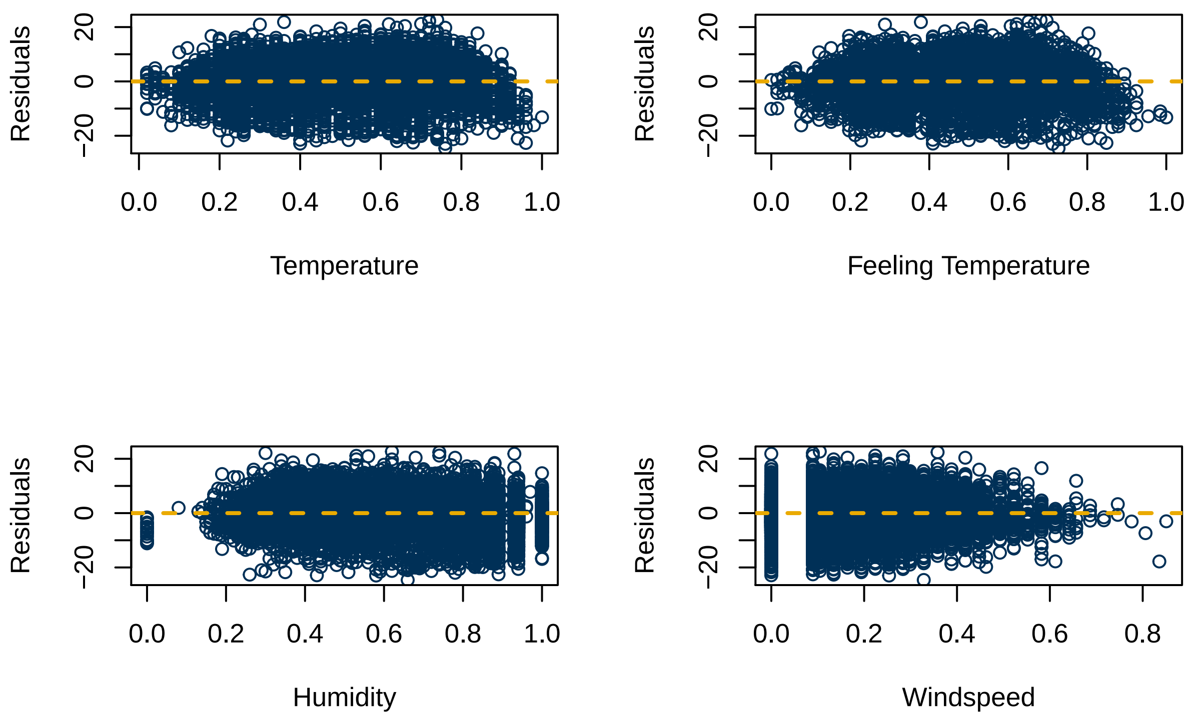
```r
with(model1, cbind(res.deviance = deviance, df = df.residual,
               p = pchisq(deviance, df.residual, lower.tail=FALSE)))
```

```
##      res.deviance    df p
## [1,]     458653.4 13851 0
```

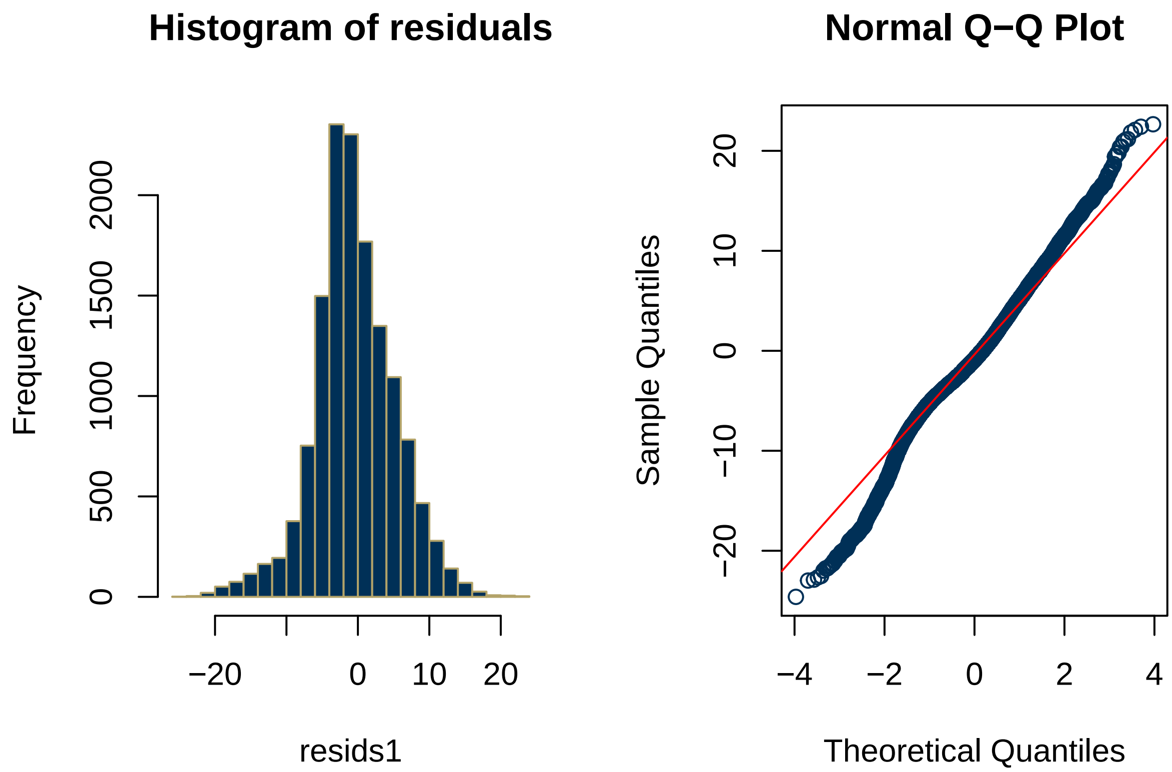
Model Assessment

```r
# Extract the standardized residuals
resids1 = resid(model1,type="deviance")


par(mfrow=c(2,2))
# Plot the standardized residuals against
# temperature
plot(train$temp, resids1,
     xlab="Temperature",
     ylab="Residuals",
     main="",
     col=gtblue)
abline(0, 0,
       col=buzzgold,
       lty=2, lwd=2)
plot(train$atemp, resids1,
     xlab="Feeling Temperature",
     ylab="Residuals",
     main="",
     col=gtblue)
abline(0, 0,
       col=buzzgold,
       lty=2, lwd=2)
plot(train$hum, resids1,
     xlab="Humidity",
     ylab="Residuals",
     main="",
     col=gtblue)
abline(0, 0,
       col=buzzgold,
```

```r
        lty=2, lwd=2)
plot(train$windspeed, resids1,
      xlab="Windspeed",
      ylab="Residuals",
      main="",
      col=gtblue)
abline(0, 0,
        col=buzzgold,
        lty=2, lwd=2)
```




```r
par(mfrow=c(1,2))
# Plot histogram of std residuals
hist(resids1,
      nclass=20,
      col=gtblue,
      border=techgold,
      main="Histogram of residuals")

# qq plot of std residuals
qqnorm(resids1,
        col=gtblue)
qqline(resids1,
        col="red")
```



Accuracy

Model1 Accuracy

```
## Save Predictions to compare with observed data
test.pred1 = predict(model1, test, type='response')

# Mean Squared Prediction Error (MSPE)
mean((test.pred1-test$cnt)^2)
```

```
## [1] 8060.083
```

```
# Mean Absolute Prediction Error (MAE)
mean(abs(test.pred1-test$cnt))
```

```
## [1] 59.96461
```

```
# Mean Absolute Percentage Error (MAPE)
mean(abs(test.pred1-test$cnt)/test$cnt)
```

```
## [1] 0.8214892
```

```
# Precision Measure (PM)
sum((test.pred1-test$cnt)^2)/sum((test$cnt-mean(test$cnt))^2)
```

```
## [1] 0.2425596
```

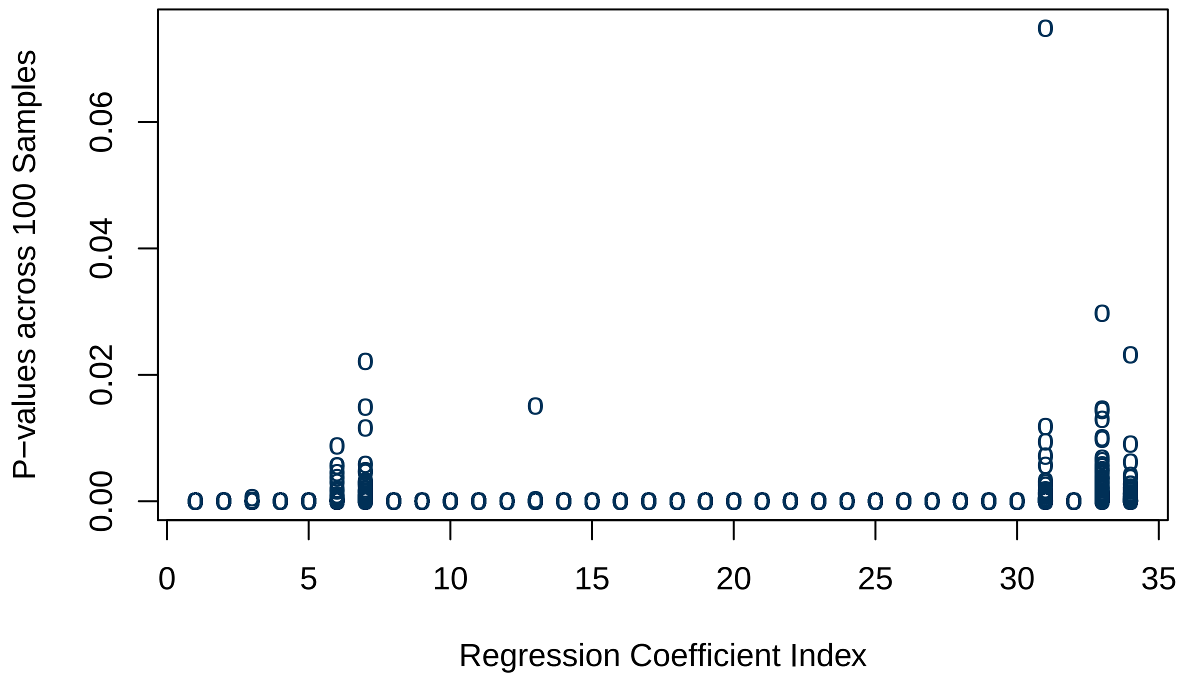
P-values and Large Sample Size

P-value Problem

```
# Approach: Subsample 40% of the initial data sample & repeat 100 times
count = 1
n = nrow(train)
B = 100
ncoef = dim(summary(model1)$coeff)[1]
pv_matrix = matrix(0,nrow = ncoef,ncol = B)

while (count <= B) {
  # 40% random sample of indices
  subsample = sample(n, floor(n*0.4), replace=FALSE)
  # Extract the random subsample data
  subdata = train[subsample,]
  # Fit the regression for each subsample
  submod = glm(round(cnt**0.5,0) ~ ., data=subdata, family='poisson')
  # Save the p-values
  pv_matrix[,count] = summary(submod)$coeff[,4]
  # Increment to the next subsample
  count = count + 1
}

# Count pvalues smaller than 0.01 across the 100 (sub)models
alpha = 0.01
pv_significant = rowSums(pv_matrix < alpha)
```

Statistically Significant Coefficients

```
# Which regression coefficients are statistically significant?
idx_scoef = which(pv_significant>=95)

# Plot the 100 p-values of the significant coefficients
matplot(pv_matrix[idx_scoef,],
     xlab="Regression Coefficient Index",
     ylab="P-values across 100 Samples",
     type="p",
     pch="o",
     col=gtblue)
```



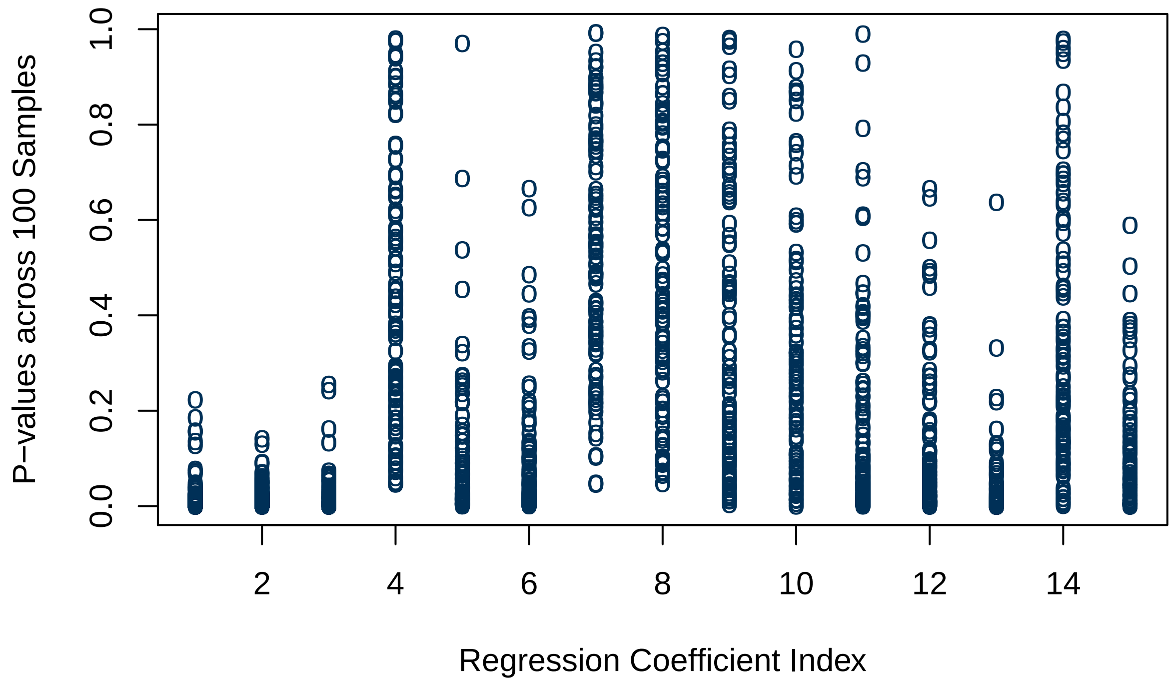

```r
# Show the p-values of the significant coefficients in model2
cbind(round(summary(model1)$coeff[idx_scoef,c(1,4)],3),
      Freq=pv_significant[idx_scoef])
```

```
##             Estimate Pr(>|z|) Freq
## (Intercept)    2.937        0  100
## season2        0.265        0  100
## season3        0.256        0  100
## season4        0.449        0  100
## yr1            0.468        0  100
## mnth3          0.235        0  100
## mnth5          0.272        0   97
## hr1           -0.479        0  100
## hr2           -0.845        0  100
## hr3           -1.545        0  100
## hr4           -2.108        0  100
## hr5           -0.980        0  100
## hr6            0.374        0   99
## hr7            1.420        0  100
## hr8            1.888        0  100
## hr9            1.388        0  100
## hr10           1.129        0  100
## hr11           1.261        0  100
## hr12           1.430        0  100
## hr13           1.417        0  100
## hr14           1.353        0  100
```

```
## hr15          1.395        0  100
## hr16          1.615        0  100
## hr17          2.020        0  100
## hr18          1.969        0  100
## hr19          1.668        0  100
## hr20          1.362        0  100
## hr21          1.110        0  100
## hr22          0.843        0  100
## hr23          0.475        0  100
## weekday5      0.093        0   98
## weathersit3  -0.489        0  100
## atemp         0.881        0   95
## hum          -0.210        0   99
```

Coefficients Not Statsitically Significant

```r
## Which regression coefficients are not statistically significant?
idx_icoef = which(pv_significant<85)

# Plot the 100 p-values of the coefficients not statistically
# significant
matplot(pv_matrix[idx_icoef,],
     xlab="Regression Coefficient Index",
     ylab="P-values across 100 Samples",
     type="p",
     pch="o",
     col=gtblue)
```




```r
# Show the p-values of coefficients not stastically significant
cbind(round(summary(model1)$coeff[idx_icoef,c(1,4)],3),
      Freq=pv_significant[idx_icoef])
```

```
##            Estimate Pr(>|z|) Freq
## mnth2         0.115        0   61
## mnth4         0.210        0   54
## mnth6         0.224        0   63
## mnth7         0.122        0    0
## mnth8         0.223        0   19
## mnth10        0.214        0   29
## mnth11        0.076        0    0
## mnth12        0.075        0    0
## weekday1      0.051        0    1
## weekday2      0.056        0    2
## weekday3      0.062        0    8
## weekday4      0.058        0   19
## weekday6      0.069        0   61
## temp          0.196        0    3
## windspeed    -0.102        0   20
```