Analysis of VEO Bike Rental Data

Getting the Data

Loading Data into R

Some libraries you may want!

```
install.packages(c("jsonlite", "lubridate",
    "wesanderson", "sp", "osmdata"))
```

First, let's load the data file which is direct from the API in JSON format, `veo_tripData_fromAPI_11-06-22_to_11-14-22.j`

```
# with jsonlite
library(jsonlite)
jsonData <- read_json(
    "veo_tripData_fromAPI_11-06-22_to_11-14-22.json",
    simplifyVector = TRUE)
```

More typically with R, you may load datasets with `read.table()` or `read.csv()`, but APIs will often return complex hierarchical data sets as JSON format, which then becomes a list or a data-frame.

Reviewing the Data

Let's look at the first trip entry.

```
jsonData[1,]
```

```
##                                 trip_id                          provider_id
## 1 9af1faa7-0cda-5c0c-959b-5bda1c64c3f2 6e39f9db-751b-5cea-ae7e-486f579a56bc
##   provider_name   start_time     end_time propulsion_types trip_distance
## 1  VeoRide INC. 1.667801e+12 1.667801e+12         electric             0
##   trip_duration vehicle_id vehicle_type        route.type
## 1            30 1000005123      bicycle FeatureCollection
##                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                route.
## 1 Feature, Feature, 1667800962896, 1667800962896, Point, Point, -96.344284, 30.612522, -96.344284, 3
##                                device_id accuracy
## 1 aaf6e89d-17d1-5fea-9d62-30960a250b45        1
```

So this has all sorts of interesting data... Some variables like `vehicle_type` are the same for all, some aren't like `propulsion_types`.

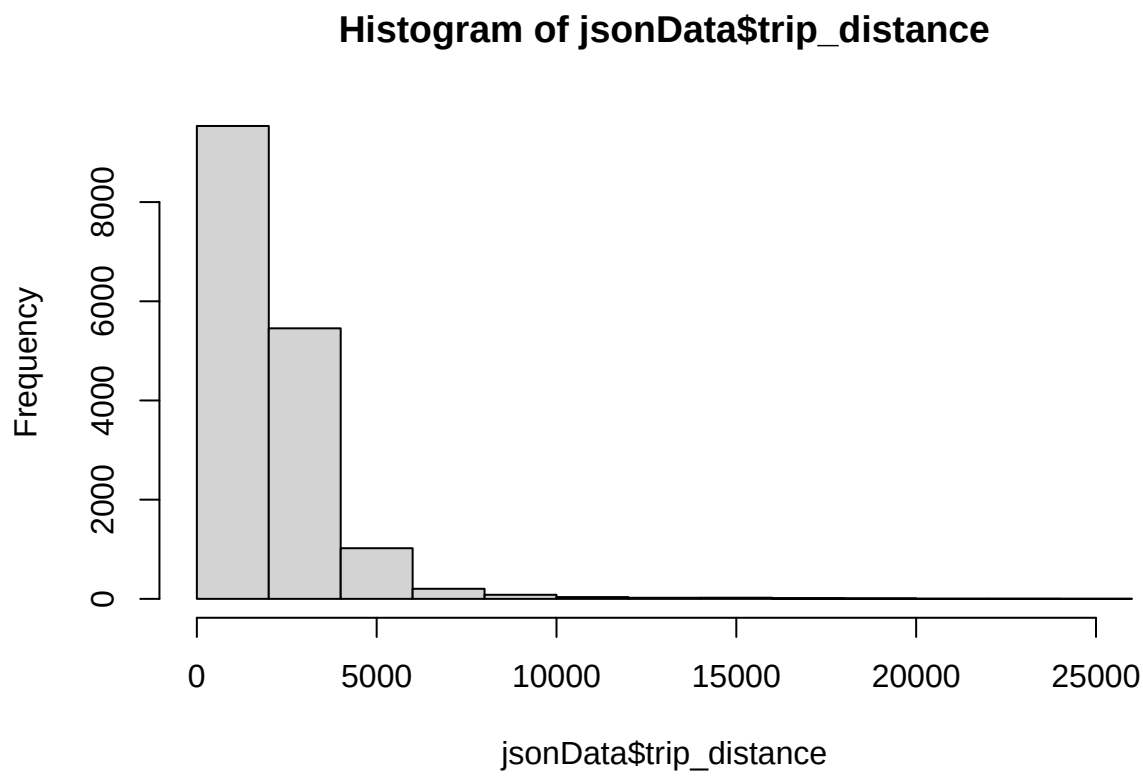
```r
table(unlist(jsonData$propulsion_types))
```

```
##
## electric    human
##    14441     1971
```

I didn't even know there were non-electronic VEOs.

Let's look at the `trip_distance` variable.

```r
hist(jsonData$trip_distance)
```




```r
# look at how many trips are 0 distance?
sum(jsonData$trip_distance == 0)
```

```
## [1] 2470
```

```r
# quite a chunk of the dataset
sum(jsonData$trip_distance == 0)/nrow(jsonData)
```

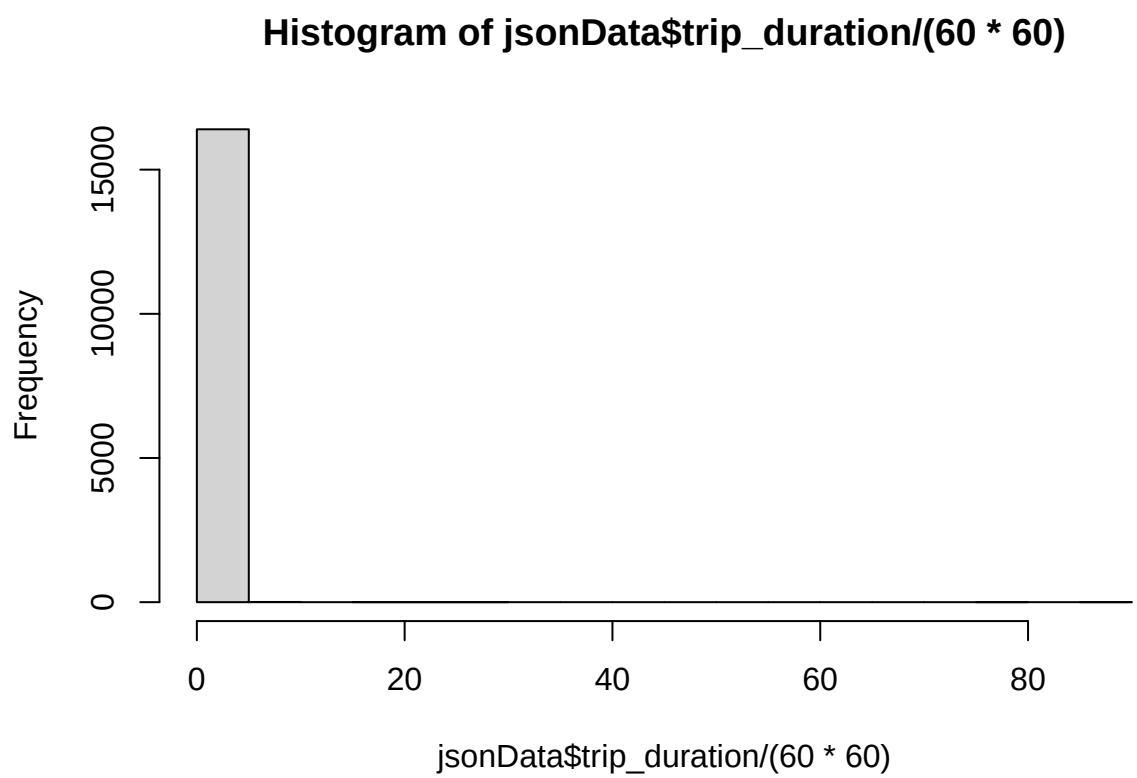
```
## [1] 0.1504996
```

```r
# seems unlikely
```

```r
# look at duration

# are there trip durations equal to zero?
sum(jsonData$trip_duration/360 == 0)
```

```
## [1] 46
```

```r
# get duration in hour units (is in seconds)
hist(jsonData$trip_duration/(60*60))
```




```r
# a 24+ bicycle trip seems unlikely
quantile(jsonData$trip_duration/(60*60),0.999)
```

```
##    99.9%
## 3.534646
```

```r
# 99.9% of trips end after 3.5 hours
```

Cleaning the Data

```r
# let's clean out 0 distance trips and trips with durations over 3.5 hours
filter <- (jsonData$trip_duration/360 < 3.5) & (jsonData$trip_distance > 0)
filter <- filter & (jsonData$trip_duration != 0)
jsonDataClean <- jsonData[filter,]
```

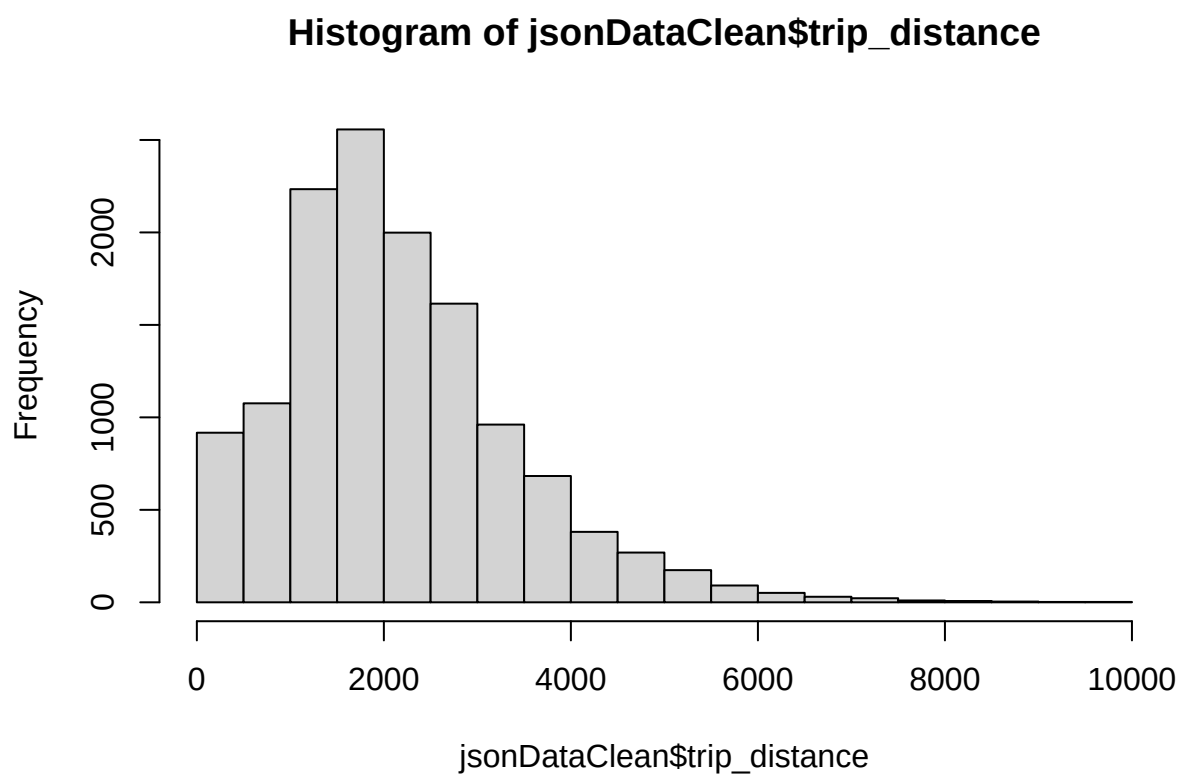
```r
# did it work?
# trip distance
summary(jsonData$trip_distance)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##       0     846    1755    1967    2698   24785
```

```r
summary(jsonDataClean$trip_distance)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##       8    1302    1953    2144    2792    9805
```

```r
hist(jsonDataClean$trip_distance)
```



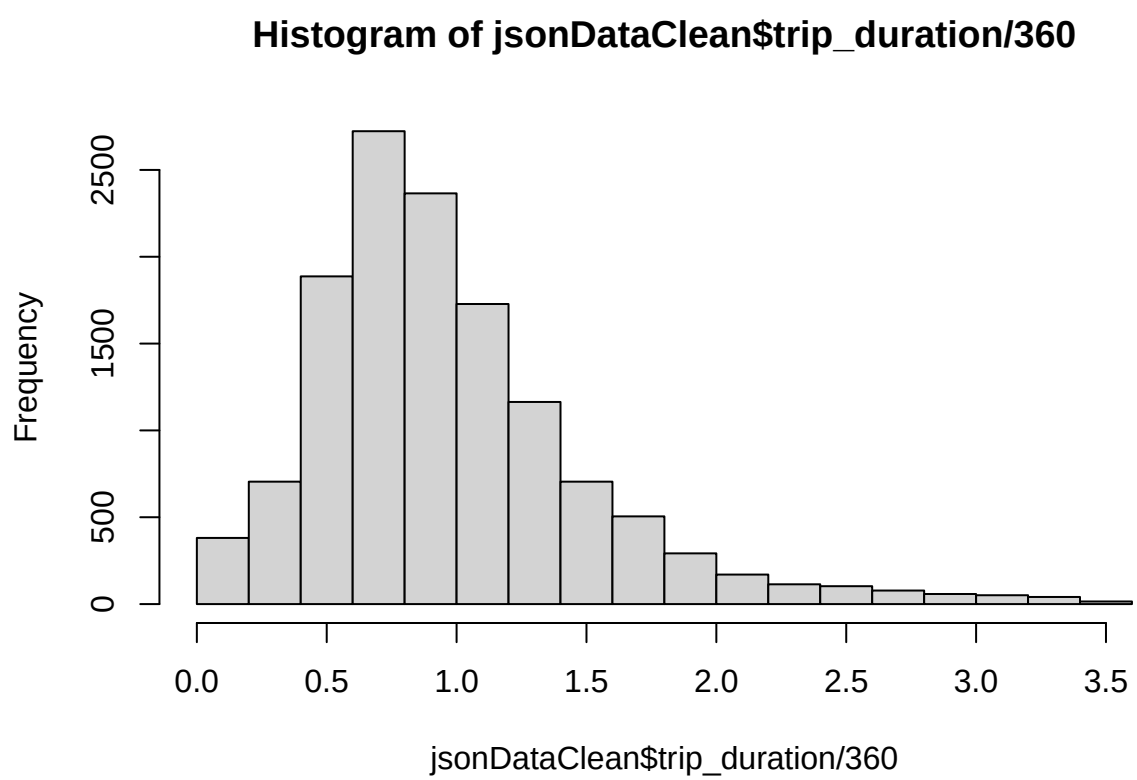

```r
# trip duration
summary(jsonData$trip_duration/360)
```

```
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
##   0.0000   0.5944   0.8694   1.5321   1.2778 884.8500
```

```r
summary(jsonDataClean$trip_duration/360)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## 0.01667 0.62500 0.86389 0.97331 1.20278 3.49167
```

```r
hist(jsonDataClean$trip_duration/360)
```




```r
# how much data was lost?
nrow(jsonData)
```

```
## [1] 16412
```

```r
nrow(jsonDataClean)
```

```
## [1] 13085
```

```r
# what proportion?
nrow(jsonDataClean)/nrow(jsonData)
```

```
## [1] 0.7972825
```

Get the Times for each Trip

Use package `lubridate` to convert the milliseconds reported by the API to dates, by considering them relative to Jan 1st, 1970 (UTC), and then converting that date-time to Central Standard Time (`US/Central`).

```r
# convert start and end times
library(lubridate)
```

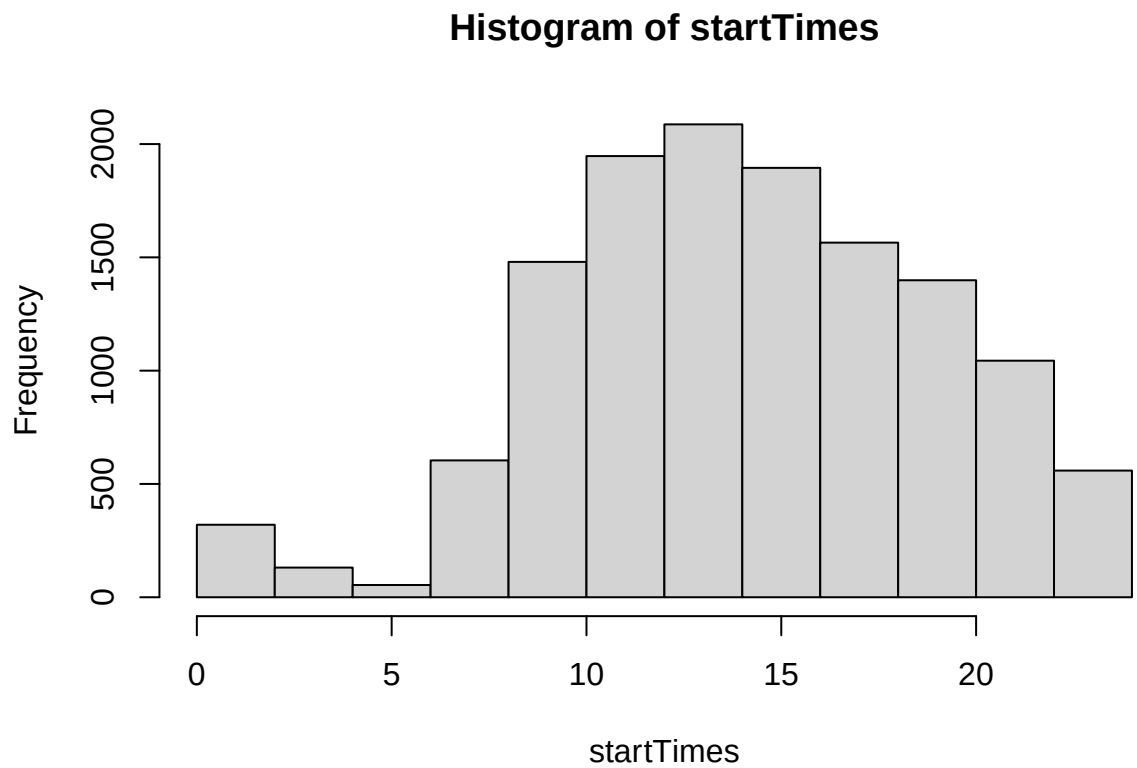
```
##
## Attaching package: 'lubridate'

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

```r
startTimes <- as_datetime(jsonDataClean$start_time/1000)
endTimes <- as_datetime(jsonDataClean$end_time/1000)
startTimes <- with_tz(startTimes,"US/Central")
endTimes <- with_tz(endTimes,"US/Central")
```

```r
# get weekday it started
weekday <- as.character(wday(startTimes, label = TRUE))
```

```r
# get just the hour-time of the start, end and weekday
startTimes <- hour(startTimes) + minute(startTimes)/60
endTimes <- hour(endTimes) + minute(endTimes)/60
```

```r
hist(startTimes)
```



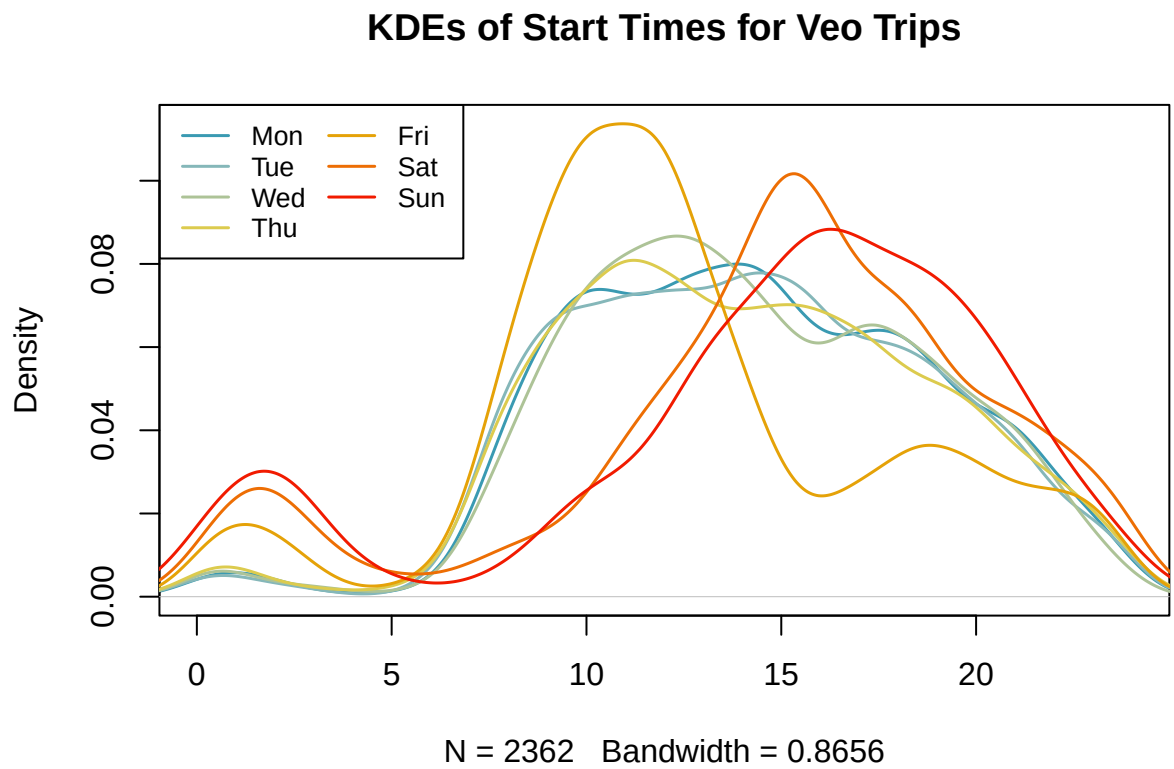
Comparing Kernal Density Estimates (KDEs) of Start Time for Different Week Days

```
# effectively shifted 0 to be at 11 AM...

# let's use kernals to look at how each day of the week differs
kde_mon <- density(x = startTimes[weekday == "Mon"])
kde_tue <- density(x = startTimes[weekday == "Tue"])
kde_wed <- density(x = startTimes[weekday == "Wed"])
kde_thu <- density(x = startTimes[weekday == "Thu"])
kde_fri <- density(x = startTimes[weekday == "Fri"])
kde_sat <- density(x = startTimes[weekday == "Sat"])
kde_sun <- density(x = startTimes[weekday == "Sun"])
```

```
#get max density estimate
max_kde_y <- max(
    c(kde_mon$y, kde_tue$y, kde_wed$y,
      kde_thu$y, kde_fri$y,
      kde_sat$y, kde_sun$y))
```

```
library(wesanderson)
# wes_palettes()
steveZissou <- wes_palette(name = "Zissou1", n = 7, type = "continuous")
```

```r
# now plot it!

plot(kde_mon, col = steveZissou[1],
    lwd = 1.5, sub = "",
    xlim = c(0,24), ylim = c(0, max_kde_y),
    main = "KDEs of Start Times for Veo Trips")
lines(kde_tue, lwd = 1.5, col = steveZissou[2])
lines(kde_wed, lwd = 1.5, col = steveZissou[3])
lines(kde_thu, lwd = 1.5, col = steveZissou[4])
lines(kde_fri, lwd = 1.5, col = steveZissou[5])
lines(kde_sat, lwd = 1.5, col = steveZissou[6])
lines(kde_sun, lwd = 1.5, col = steveZissou[7])

legend(x = "topleft", ncol = 2,
       legend = c("Mon", "Tue", "Wed", "Thu", "Fri", "Sat", "Sun"),
       col = steveZissou, lty = 1, lwd = 1.5, cex = 0.8)
```



Get the Route Coordinates for each Trip

```r
routeFeatures <- jsonDataClean$route$features

startLat <- startLong <- endLat <- endLong <- numeric()
```

```r
for(i in 1:length(routeFeatures)){
    latlongs <- routeFeatures[[i]]$geometry$coordinates
    startLat[i] <- latlongs[[1]][2]
    startLong[i] <- latlongs[[1]][1]
    endLat[i] <- latlongs[[length(latlongs)]][2]
    endLong[i] <- latlongs[[length(latlongs)]][1]
    #
    }

summary(c(startLat,endLat))
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   30.59   30.61   30.61   30.61   30.62   30.63
```

```r
summary(c(startLong,endLong))
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  -96.37  -96.35  -96.34  -96.34  -96.34  -96.32
```

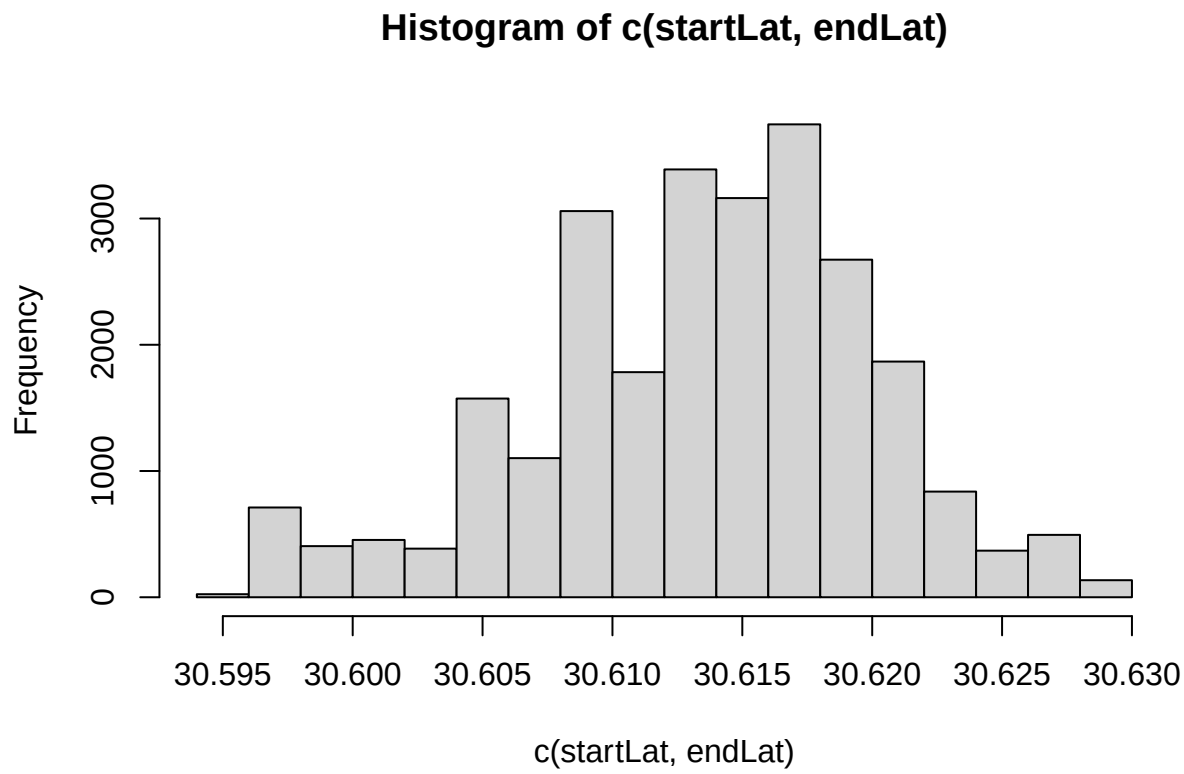
```r
# how often do bicycles start and in exact same place
sum(startLat == endLat)/length(startLat)
```

```
## [1] 0.0003056935
```

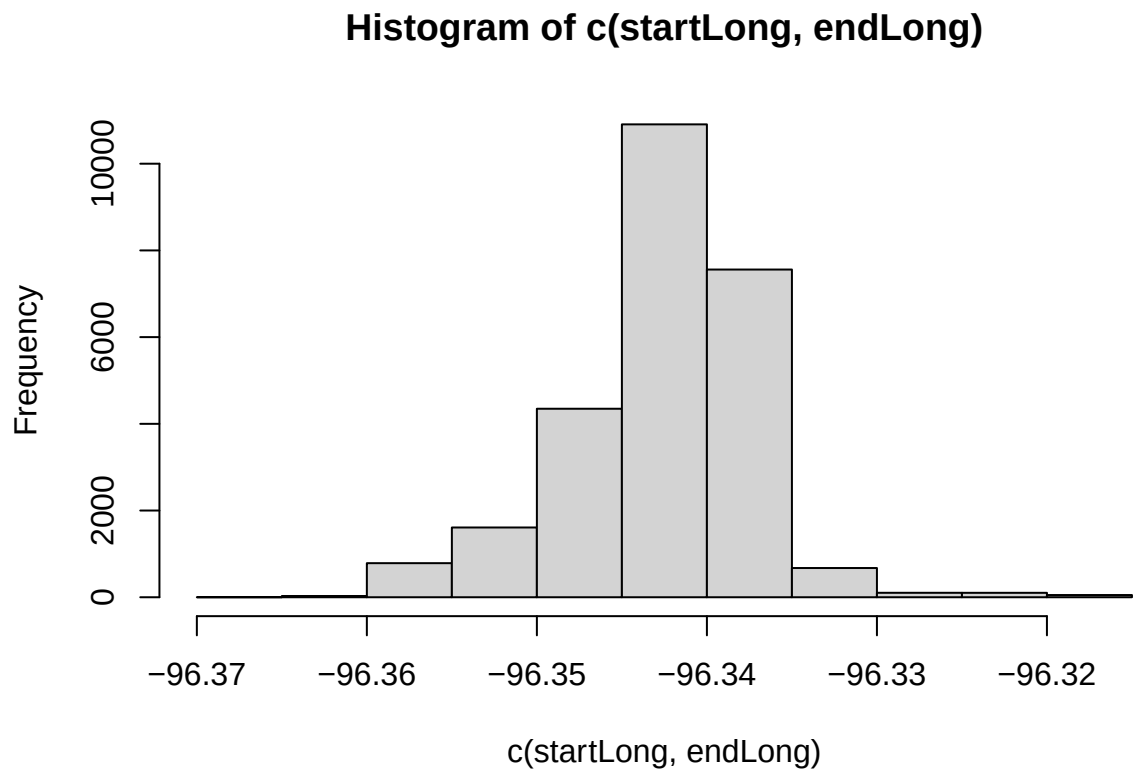
```r
sum(startLong == endLong)/length(startLong)
```

```
## [1] 0.0003056935
```

```r
# very rarely!

hist(c(startLat,endLat))
```




```r
hist(c(startLong,endLong))
```



Plot Start and End Times on a Map

```
# map

library(osmdata)
```

```
## Data (c) OpenStreetMap contributors, ODbL 1.0. https://www.openstreetmap.org/copyright
```

```
coords <- c( -96.394813, 30.578354 , -96.300863 , 30.657674 )
coordsLim <- c( -96.37, 30.59 , -96.32 , 30.63 )
cstx_query <- opq(bbox = coords)
# get primary and secondayr roads
highway_query <- add_osm_feature(cstx_query, key = 'highway')
highway_sp <- osmdata_sp(highway_query)
```

```
# plot it
oldPar <- par(no.readonly = TRUE)
par(mar=rep(0.1,4))

sp::plot(highway_sp$osm_lines,
    xlim = coordsLim[c(1,3)],
    ylim = coordsLim[c(2,4)])
# can we plot start and end lat/longs?
points(startLong, startLat, cex = 0.3, col = "red")
```

```r
sp::plot(highway_sp$osm_lines,
    xlim = coordsLim[c(1,3)],
    ylim = coordsLim[c(2,4)])
points(endLong, endLat, cex = 0.3, col = "purple")
```

```r
par(oldPar)
```

Multivariate Analyses

Cluster Analysis

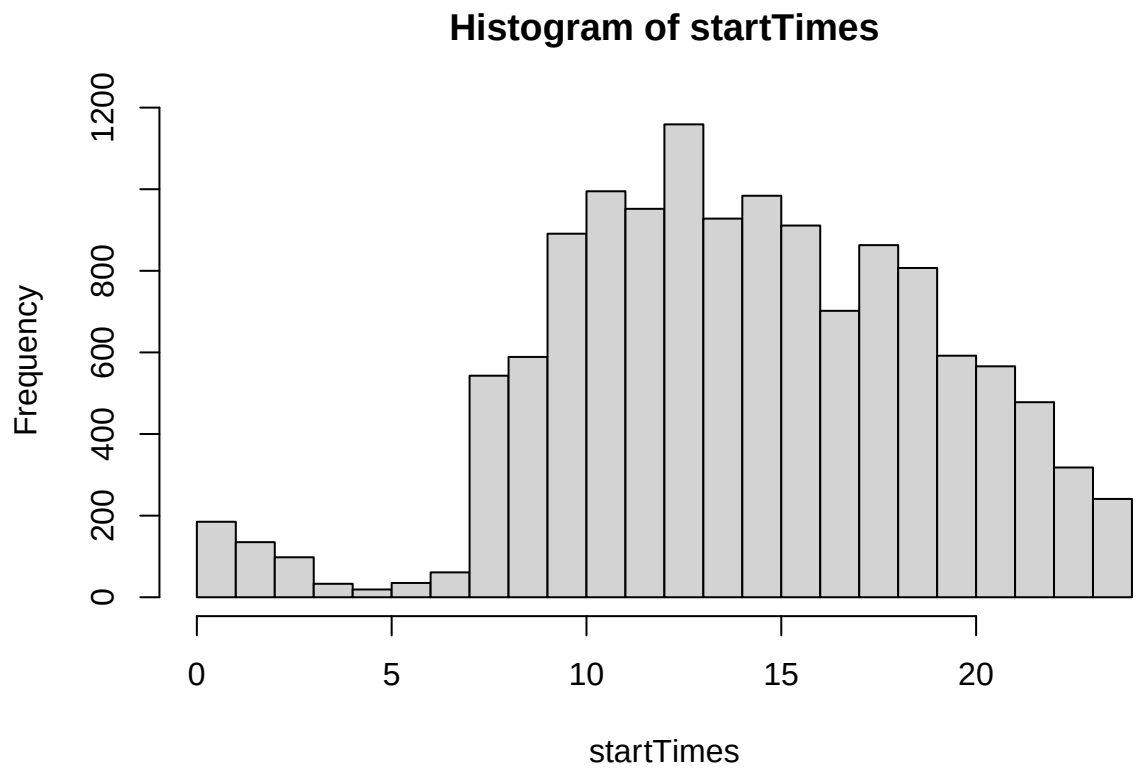
The hardest step of multivariate analysis isn't running the analysis but getting your ducks in a row – making sure your data is all of the right form.

Making the Timings More Continuous

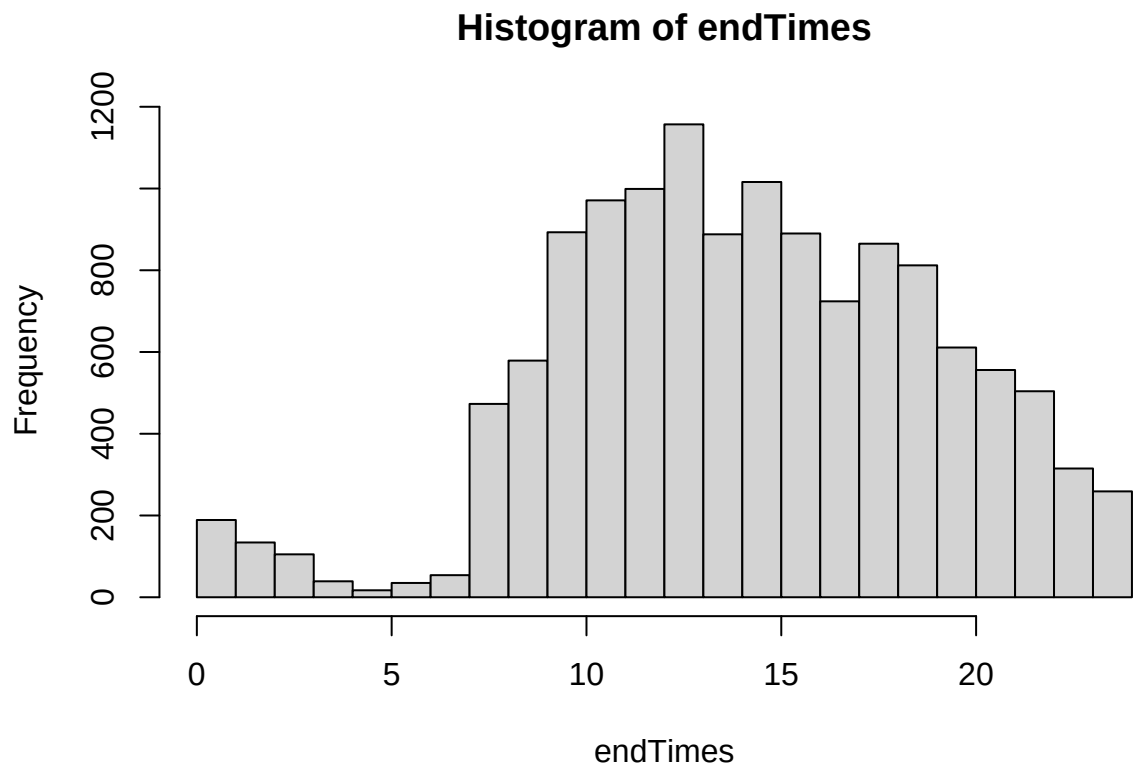
Cluster analysis based on Euclidean distance assumes that data will be distributed similarly, and that difference in values reflects difference in association. That may seem obvious, but things get complicated with values like time-of-day (which is cyclical), just like with other 'circular' measures, like compass orientation of the wind, strike/dip of rocks, etc.

In our case, for the starting time-of-day for each trip, it would probably be ideal to cut weekdays at 5:00 AM as that is the time of lowest traffic, and stick the hours of midnight to 5:00 AM to the previous day. This is in practice a bit tricky, but let's try it.

```r
# where is good cut point?
hist(startTimes,breaks = 24)
```



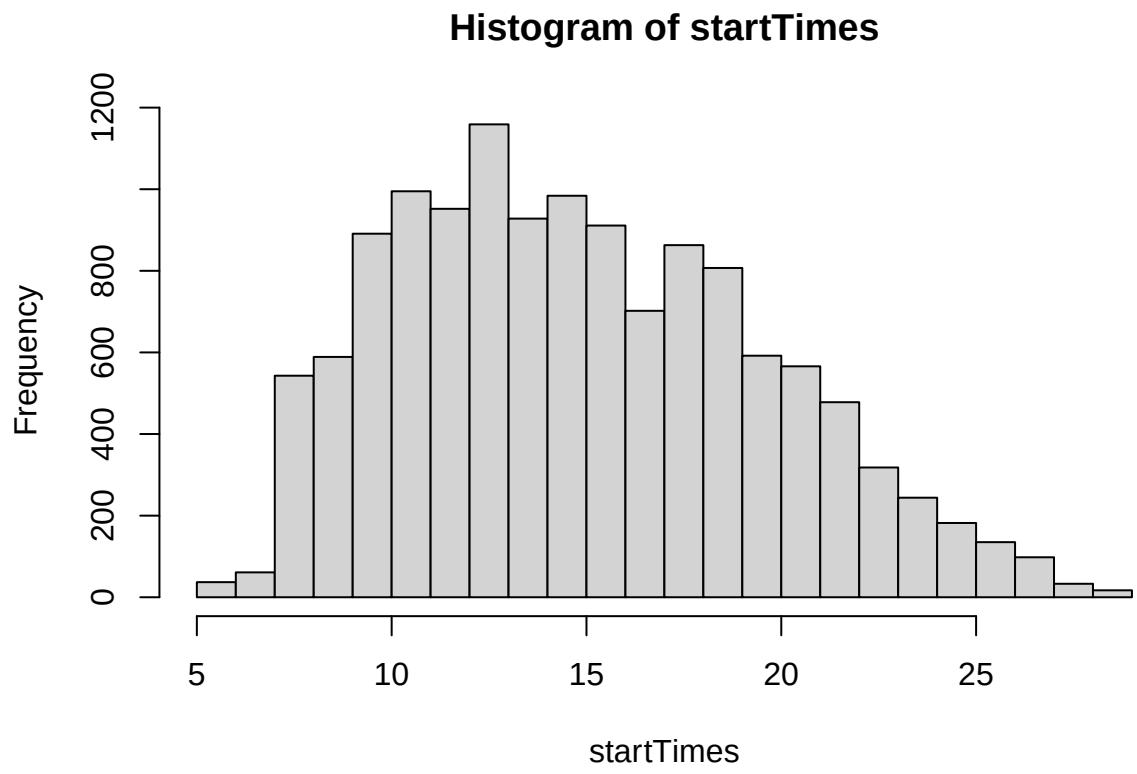

```r
hist(endTimes, breaks = 24)
```



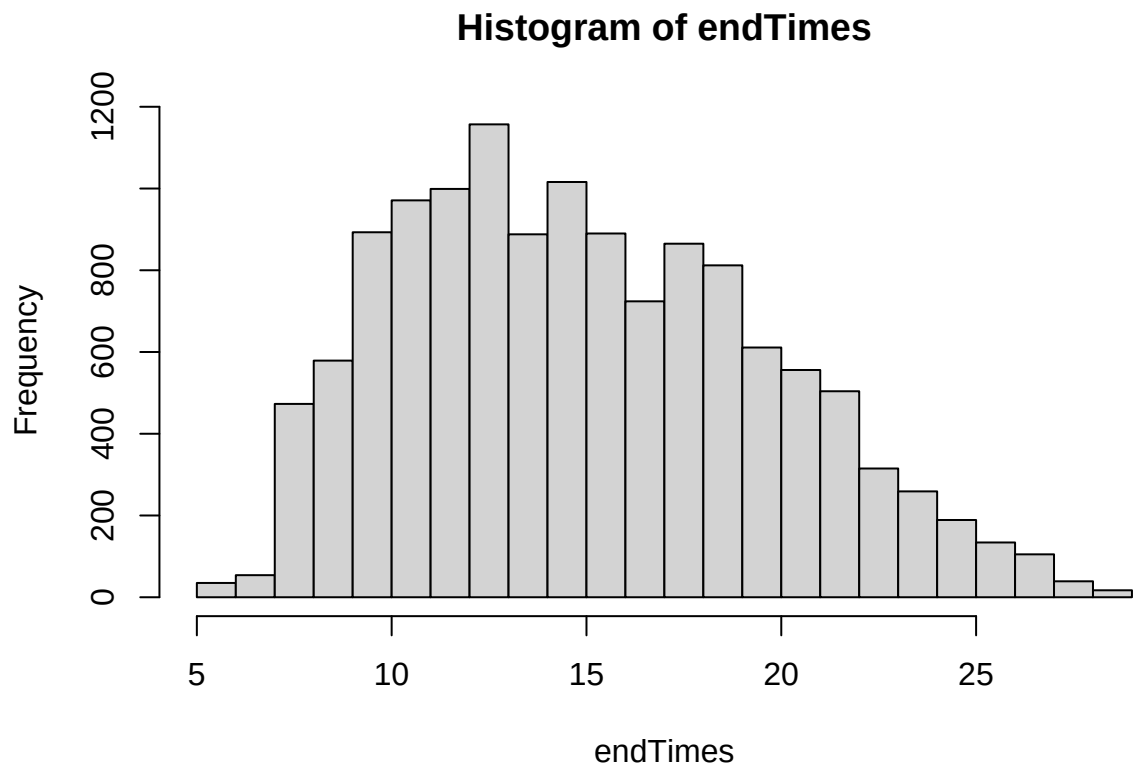
Okay, so we need to move the hours of 0-5 to 24-29 on the 24h schedule.

```
# shift cut point to 5
startTimes <- ifelse(startTimes < 5,
    startTimes + 24, startTimes)

endTimes <- ifelse(endTimes < 5,
    endTimes + 24, endTimes)
```

```
# did our cut points work well?
hist(startTimes,breaks = 24)
```




```r
hist(endTimes, breaks = 24)
```



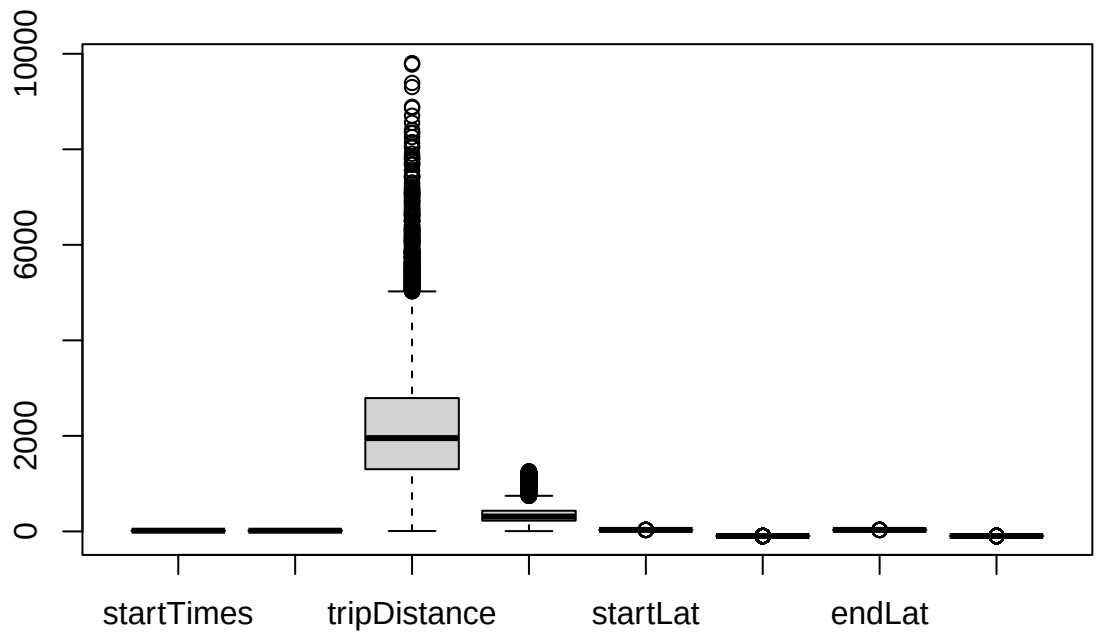
Then we need to shift the weekday for every trip-start timestamp over 24, as it is best classified as the 'previous day'.

```
for(i in 1:length(startTimes)){
    if(startTimes[i] > 24){
        if(weekday[i] == "Mon"){weekday[i] <- "Sun"}
        if(weekday[i] == "Tue"){weekday[i] <- "Mon"}
        if(weekday[i] == "Wed"){weekday[i] <- "Tue"}
        if(weekday[i] == "Thu"){weekday[i] <- "Wed"}
        if(weekday[i] == "Fri"){weekday[i] <- "Thu"}
        if(weekday[i] == "Sat"){weekday[i] <- "Fri"}
        if(weekday[i] == "Sun"){weekday[i] <- "Sat"}
        }
    }
```

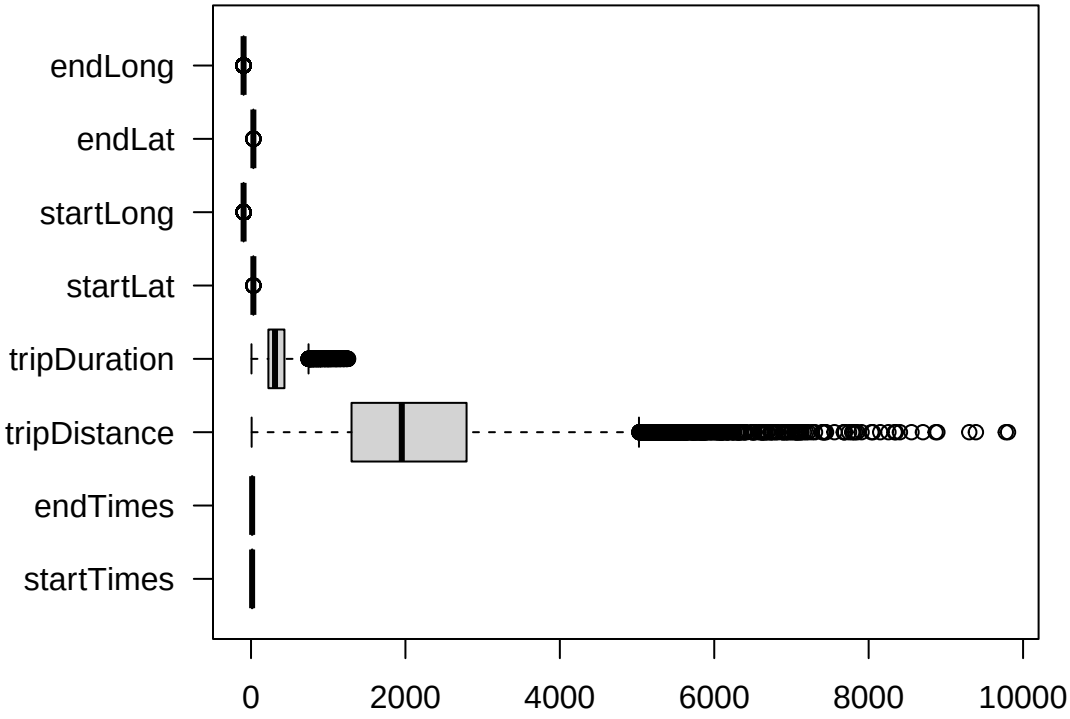
Assembling a Multivariate Data Set

```
# do a cluster analysis

tripData <- data.frame(
    #vehicleType = jsonDataClean$vehicle_type,
    #propulsion = jsonDataClean$propulsion_types,
    startTimes = startTimes, endTimes = endTimes,
    tripDistance = jsonDataClean$trip_distance,
    tripDuration = jsonDataClean$trip_duration,
```

```
    startLat = startLat, startLong = startLong,
    endLat = endLat, endLong = endLong
    )

boxplot(tripData)
```

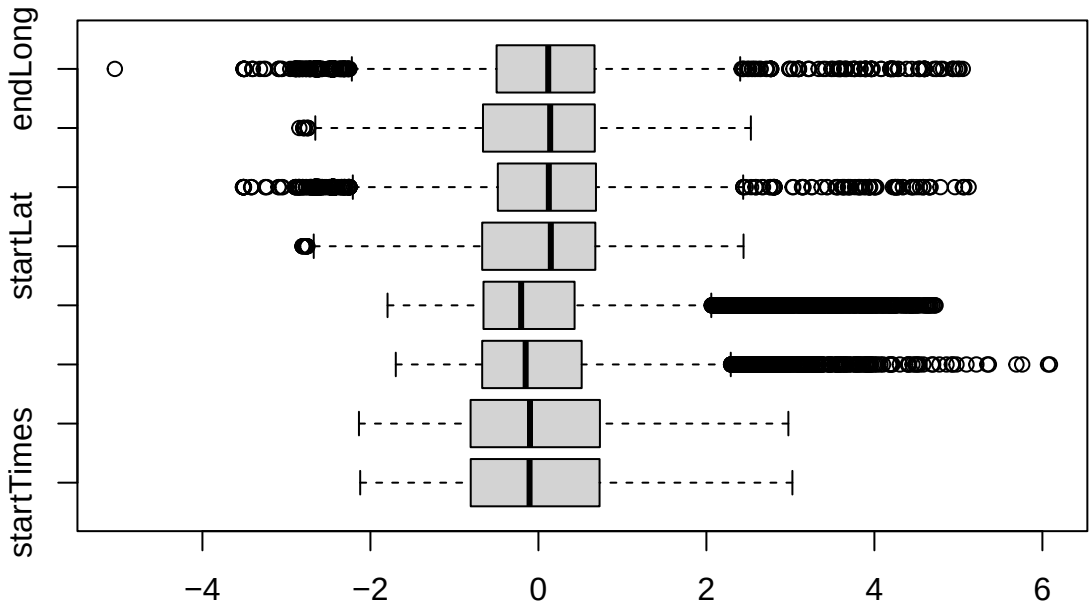



```
par(mar = c(5,10,1,1), las = 1)
boxplot(tripData, horizontal = TRUE)
```

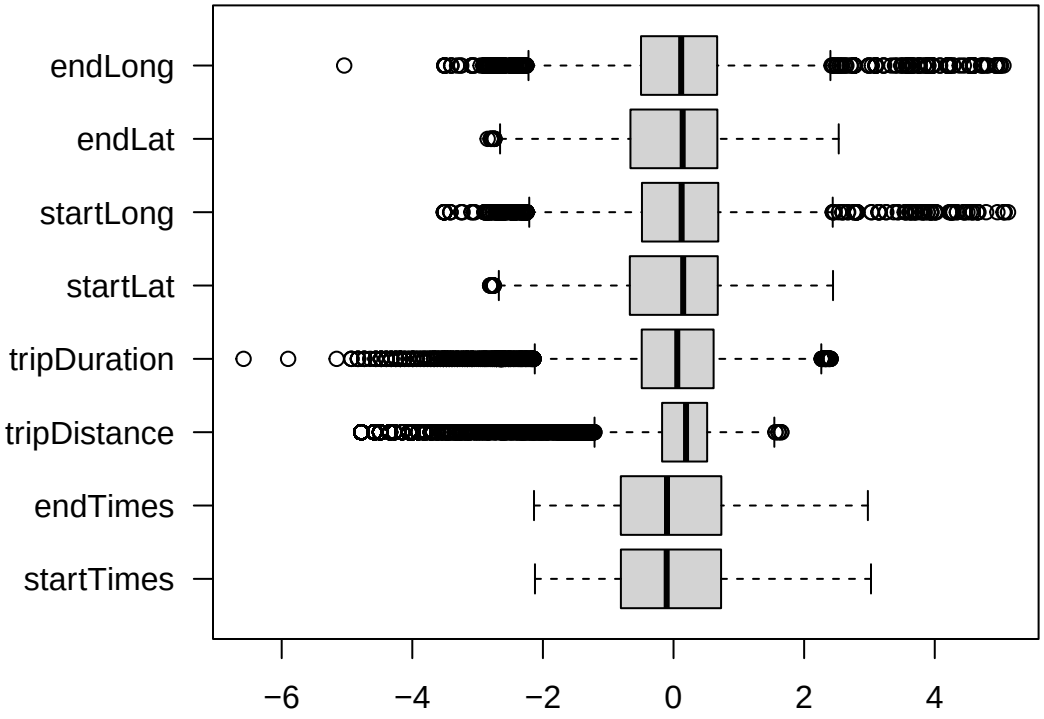


Try a z-score transformation using `scale`, which subtracts the mean of each variable from each variable, and divides by that variable's variance, thus ensuring each variable has a new mean of 0 and a variance of 1.

```
boxplot(scale(tripData), horizontal = TRUE)
```




```
# lets log duration and distance too

tripData <- scale(data.frame(
    startTimes = startTimes, endTimes = endTimes,
    tripDistance = log(jsonDataClean$trip_distance),
    tripDuration = log(jsonDataClean$trip_duration),
    startLat = startLat, startLong = startLong,
    endLat = endLat, endLong = endLong
    ))

par(mar = c(5,10,1,1), las = 1)
boxplot(tripData, horizontal = TRUE)
```

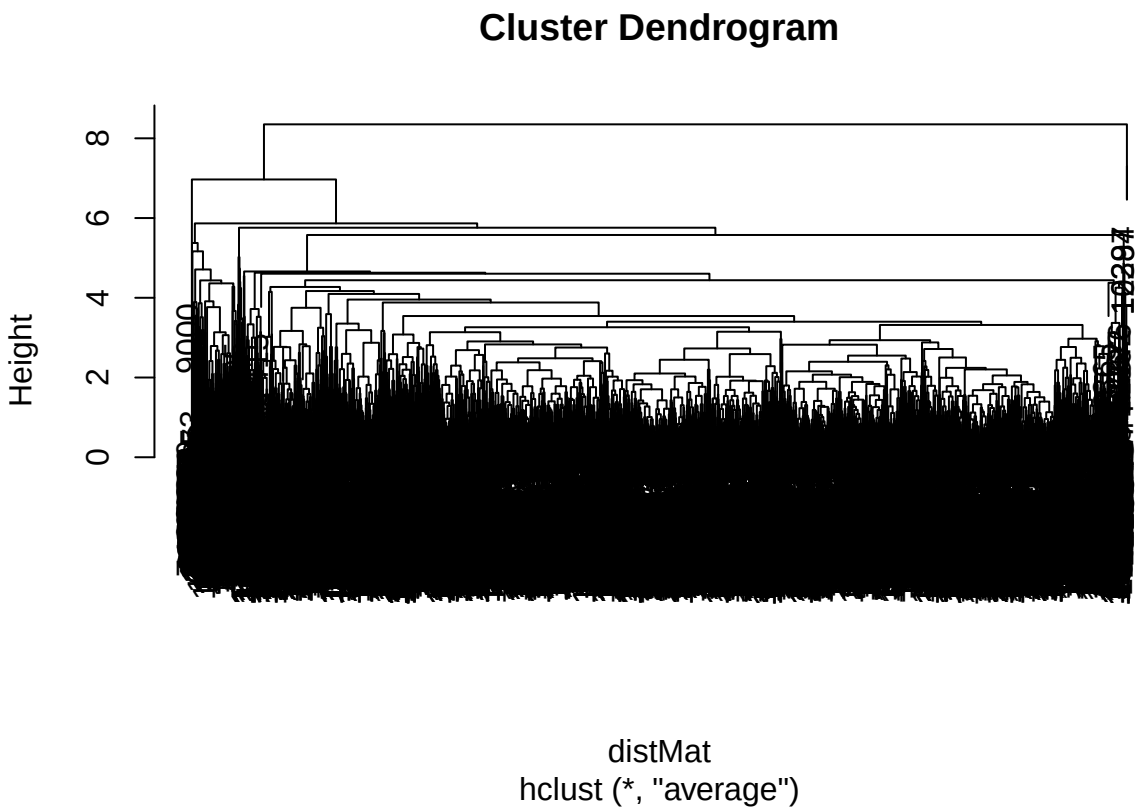



```
par(oldPar)
```

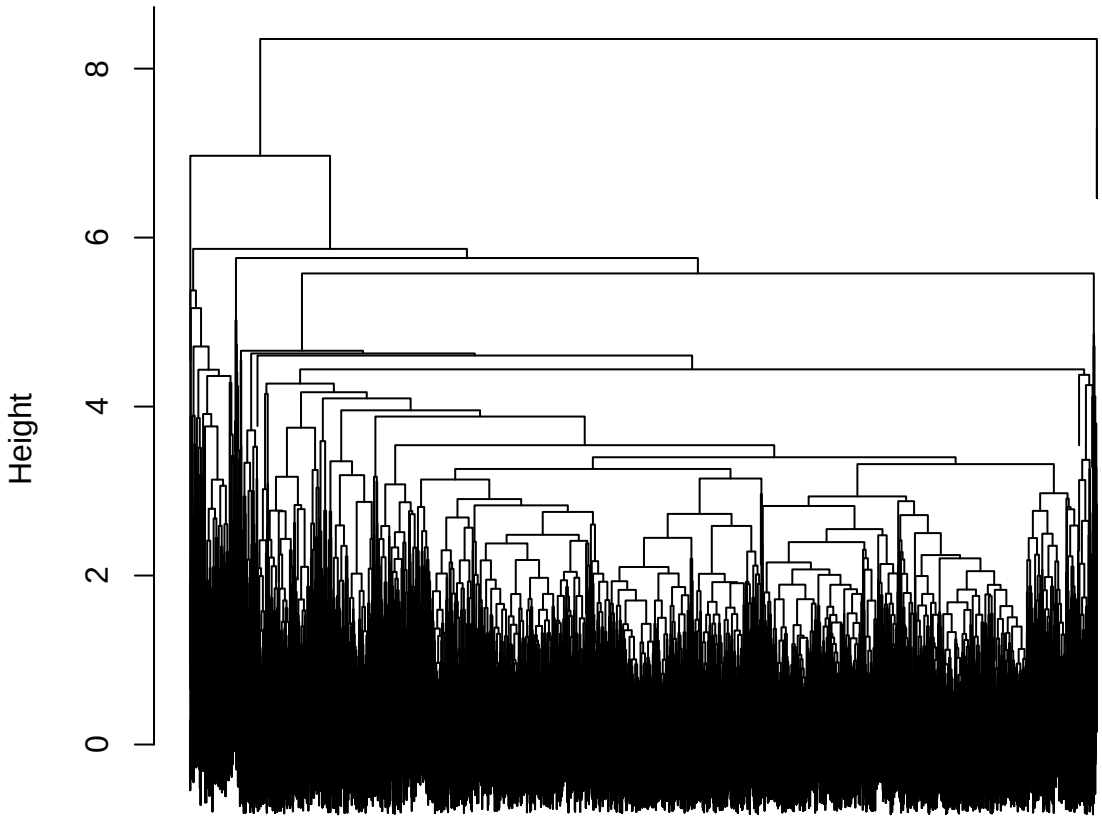
Onward to the Cluster Analysis of Trips

```
# need a pairwise distance matrix
distMat <- dist(tripData, method = "euclidean")
```

```
# cluster analysis with UPGMA / Average
clustOut <- hclust(distMat, method = "average")

# plot cluster analysis
plot(clustOut)
```



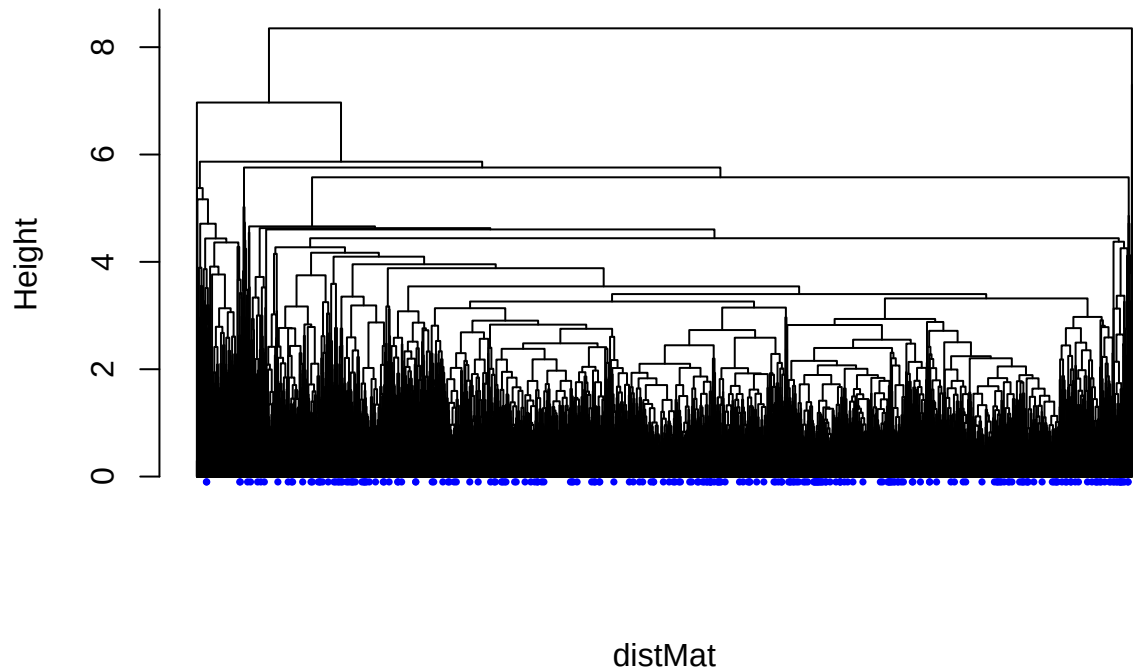

```r
# ugh! remove labels
par(mar = c(0,5,0,2))
plot(clustOut, labels = FALSE, main = "", sub = "")
```




```r
par(oldPar)
```

Are Human-Powered Veo Bike Rides more Clustered or Dispersed? Are Human-Powered Veo Bike Rides more Clustered or Dispersed?

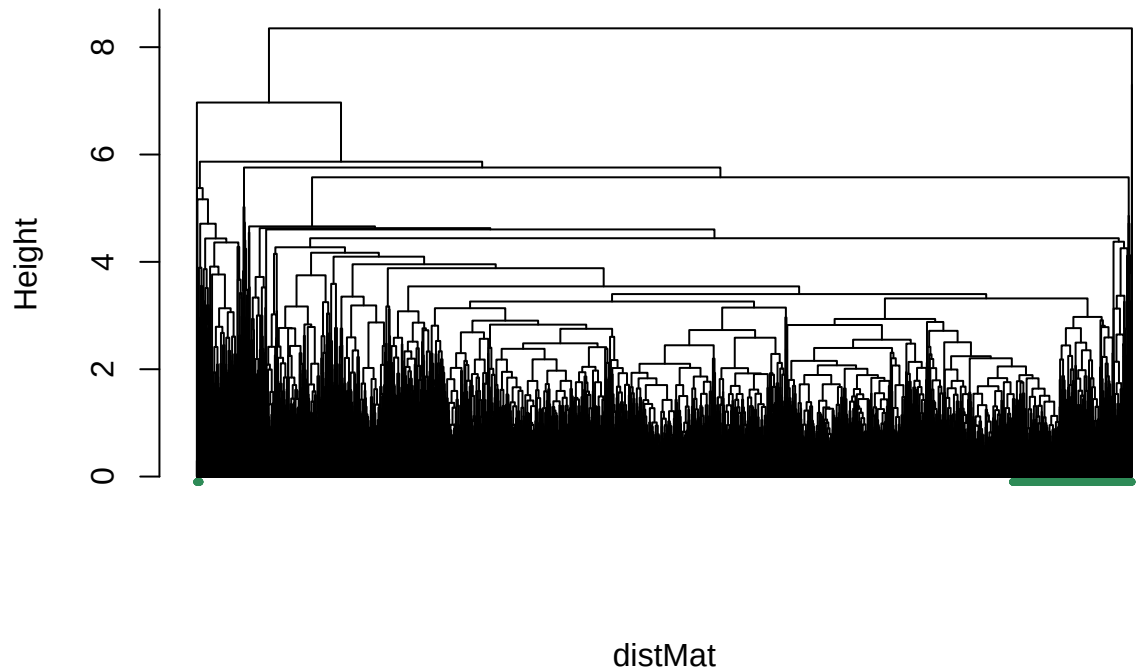
```r
# are human-powered veo bike rides clusters or dispersed?

# add symbols for propulsion type
plot(clustOut, labels = FALSE, main = "", sub = "",
    hang = -1)
# next let's add things to the plot

plotWhich <- jsonDataClean$propulsion_types[clustOut$order] == "human"
plotWhere <- clustOut$order[plotWhich]

points(
    x = plotWhere,
    y = rep(-0.1, sum(plotWhich)),
    pch =16,
    cex = 0.5,
    col = "blue")
```



Looks like the answer is they have relationship to the data structure we are looking at.

Are Weekend Rides more Clustered or Dispersed? **Are Weekend Rides more Clustered or Dispersed?**

```
# are weekend rides clusters or dispersed?

# add symbols for propulsion type
plot(clustOut, labels = FALSE, main = "", sub = "",
    hang = -1)
# next let's add things to the plot

plotWhich <- (weekday[clustOut$order] == "Sun") |
    (weekday[clustOut$order] == "Sat")
plotWhere <- clustOut$order[plotWhich]

points(
    x = plotWhere,
    y = rep(-0.1, sum(plotWhich)),
    pch =16,
    cex = 0.5,
    col = "seagreen")
```

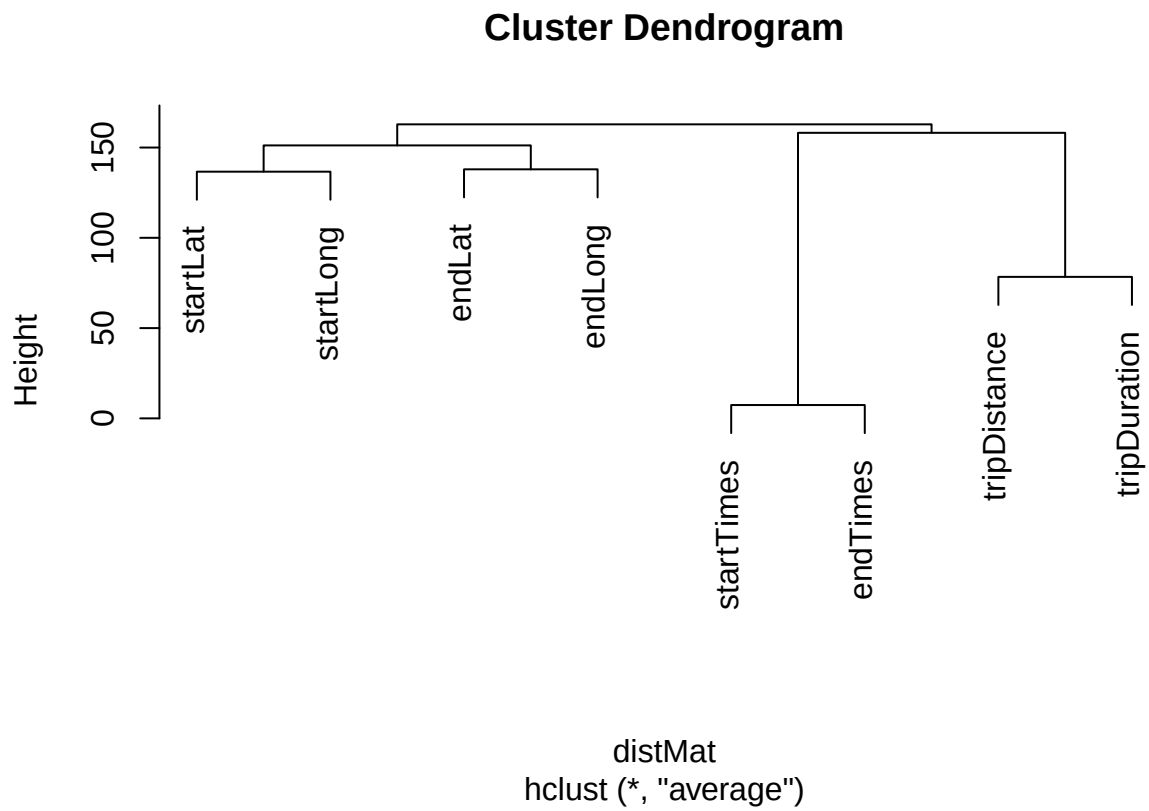



```r
par(oldPar)
```

Woah, very striking!

Cluster Analysis of Variables Rather than Samples (Trips)

```r
# cluster analysis of parameters?
distMat <- dist(t(tripData), method = "euclidean")

# cluster analysis with UPGMA / Average
clustOut <- hclust(distMat, method = "average")

# plot cluster analysis
plot(clustOut)
```



Multiple Regression and Model Selection

Do Trip Durations Change Over the Course of a Day?

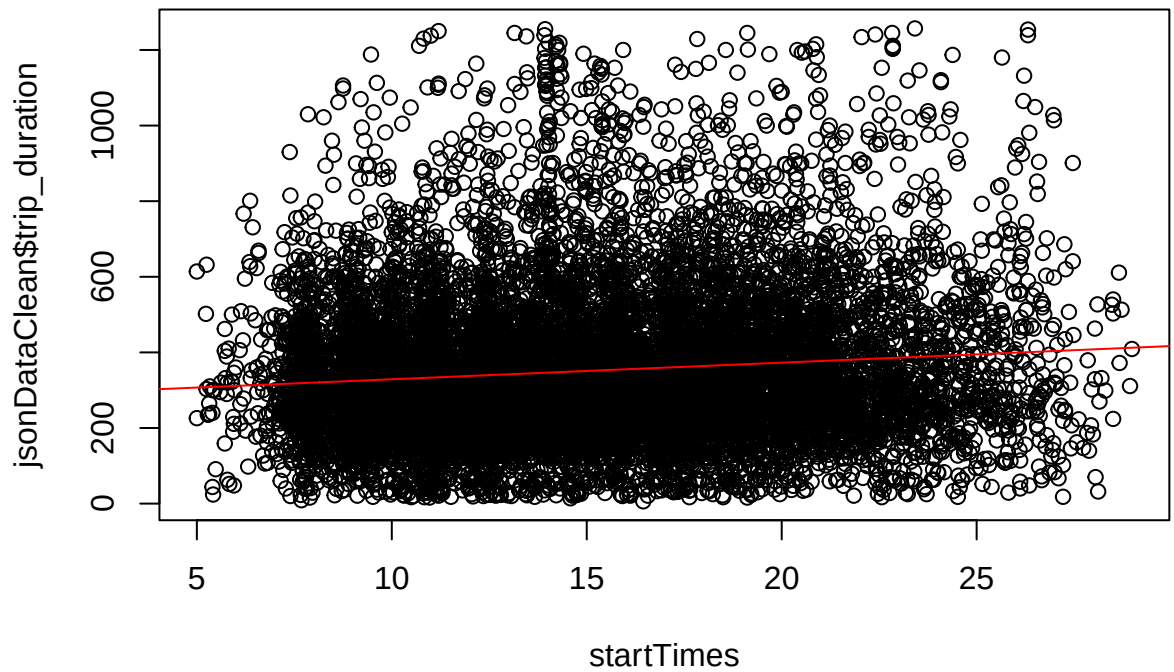
Do trip durations change in any significant way with starting time for a trip?

```
# do trip durations change over the course of a day?
regOut <- lm(jsonDataClean$trip_duration ~ startTimes)
summary(regOut)
```

```
##
## Call:
## lm(formula = jsonDataClean$trip_duration ~ startTimes)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -386.59 -125.09  -36.85   83.73  915.83
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 284.9231     5.5797   51.06   <2e-16 ***
## startTimes    4.3967     0.3576   12.29   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
##
## Residual standard error: 190.7 on 13083 degrees of freedom
## Multiple R-squared:  0.01142,    Adjusted R-squared:  0.01135
## F-statistic: 151.2 on 1 and 13083 DF,  p-value: < 2.2e-16
```

```r
# plot it
plot(startTimes, jsonDataClean$trip_duration)
abline(regOut, col = "red")
```




```r
AIC(regOut)
```

```
## [1] 174543.9
```

Do Trip Durations Depend on Trip Distance and Starting Time?

```r
# what if we add in trip distance?

# gets better if we include distance...
regOut <- lm(jsonDataClean$trip_duration ~ startTimes + jsonDataClean$trip_distance)
summary(regOut)
```

```
##
```

```
## Call:
## lm(formula = jsonDataClean$trip_duration ~ startTimes + jsonDataClean$trip_distance)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -862.34  -65.36  -30.42   23.42 1151.73
##
## Coefficients:
##                              Estimate Std. Error t value Pr(>|t|)
## (Intercept)                 5.564e+01  4.005e+00   13.89   <2e-16 ***
## startTimes                  3.178e+00  2.323e-01   13.68   <2e-16 ***
## jsonDataClean$trip_distance 1.154e-01  8.606e-04  134.09   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 123.7 on 13082 degrees of freedom
## Multiple R-squared:  0.5836, Adjusted R-squared:  0.5836
## F-statistic:  9169 on 2 and 13082 DF,  p-value: < 2.2e-16
```

```r
# better R-sq, AIC...
AIC(regOut)
```

```
## [1] 163231.2
```

Maybe Trip Duration is Just a Function of Trip Distance

```r
# but is it a negligible contribution to consider start time?
regOut <- lm(jsonDataClean$trip_duration ~ jsonDataClean$trip_distance)
summary(regOut)
```

```
##
## Call:
## lm(formula = jsonDataClean$trip_duration ~ jsonDataClean$trip_distance)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -870.31  -66.31  -32.86   23.63 1149.67
##
## Coefficients:
##                              Estimate Std. Error t value Pr(>|t|)
## (Intercept)                 1.020e+02  2.153e+00   47.36   <2e-16 ***
## jsonDataClean$trip_distance 1.159e-01  8.661e-04  133.78   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 124.6 on 13083 degrees of freedom
## Multiple R-squared:  0.5777, Adjusted R-squared:  0.5776
## F-statistic: 1.79e+04 on 1 and 13083 DF,  p-value: < 2.2e-16
```

```r
# slightly worse R-sq, higher AIC... actually, start time *WAS* informative
    # according to AIC
AIC(regOut)
```

```
## [1] 163415.1
```

AIC suggests no, the contribution of starting times to the model is negligible but shouldn't be ignored.