Commute Convergence Vignette

David Burton

December 14, 2017

Basic Commute Analysis

So, you've been driving to the same place more and more often, but it's eating you alive inside that you don't know when it will take you the least time to get there.

This R package is for you. *commuteconvergence* is a simple package that has a very simple requirement: you must record the day of the week and total commute time each day, or when you leave home and arrive at work, in the columns of a Google Sheet.

commuteconvergence will harness the power of the great *googlesheets* and *ggplot2* packages to work for you!

Step 0: Collect data

It is *extremely* important that you record your data into a Google Sheet as instructed. At the bare minimum it must include two columns: Day and Total(drive time each day). Day is a group by which you want to compare your times, so you can use other groups than day of the week, such as what you listen to in the car. However, you also have the option to have the package calculate drive time for you if you will supply a Leave time and Arrive time instead of Total. Leave and Arrive must be formatted in HH:MM:SS; such as 10:23:00 AM, for 10:23 AM. Googlesheets seems to do this automatically, but be careful to double check if you report afternoon and morning times. Using actual times also provides you more in depth, and probably, more useful analysis.

Step 1: Read in data

Get *commuteconvergence* going using *library()*. After you publish your Google sheet to the web for viewing, via the file menu in Google Sheets, use the share button to get a url. Paste it in the function *read_sheet()*. If unsure of how to share via a link in Google Sheets please consult a Google help resource, or view documentation related to the *googleheets* package. You have to place your own quotation marks around the sharing link. Assign the value of *read_sheet* to the data frame name you have chosen.

I have included a link to an example minimal sheet which is available for public viewing on Google Sheets. Please replace it with your own data once you understand the process.

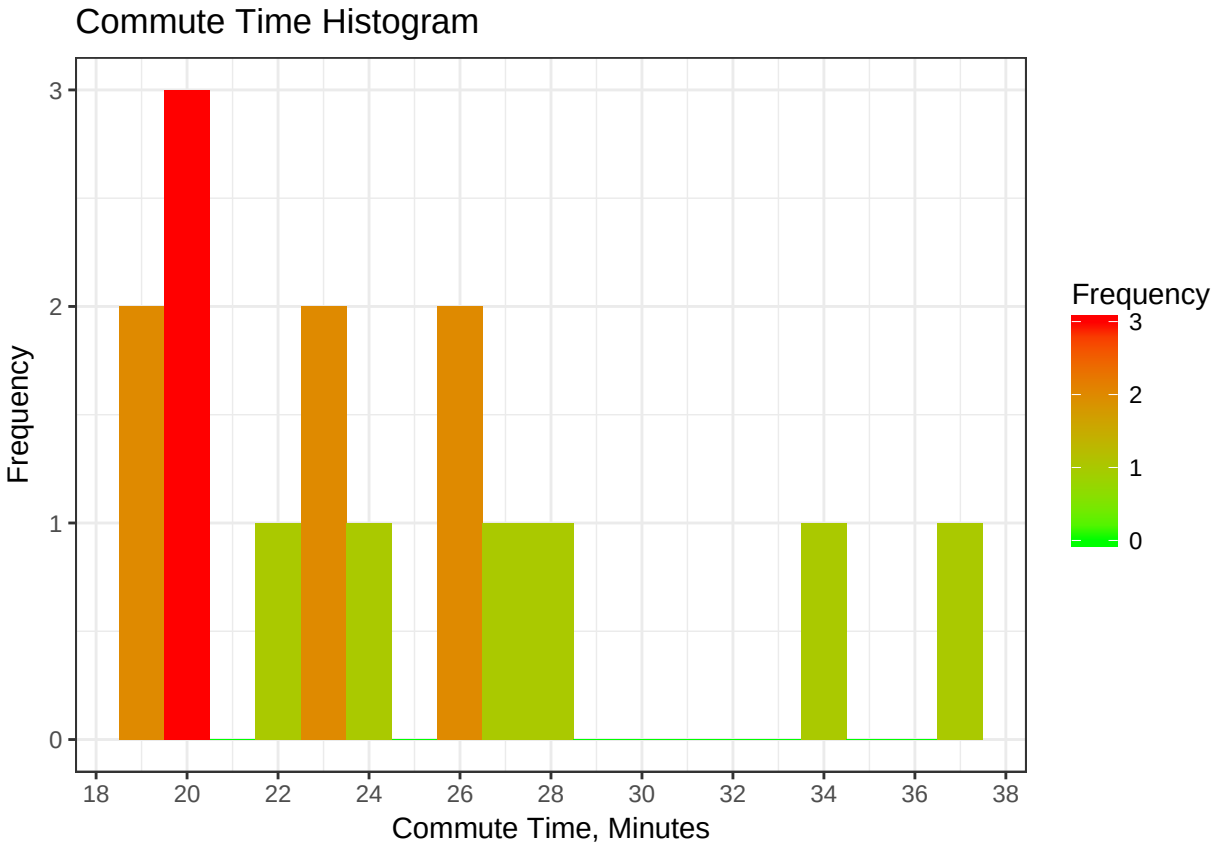
```
library(commuteconvergence)
commute <- read_sheet(
"https://docs.google.com/spreadsheets/d/14TN5tRLf2HQq3m8Fot8VhEydobYqDW_7hBWqLWPcQzw/edit?usp=sharing")
```

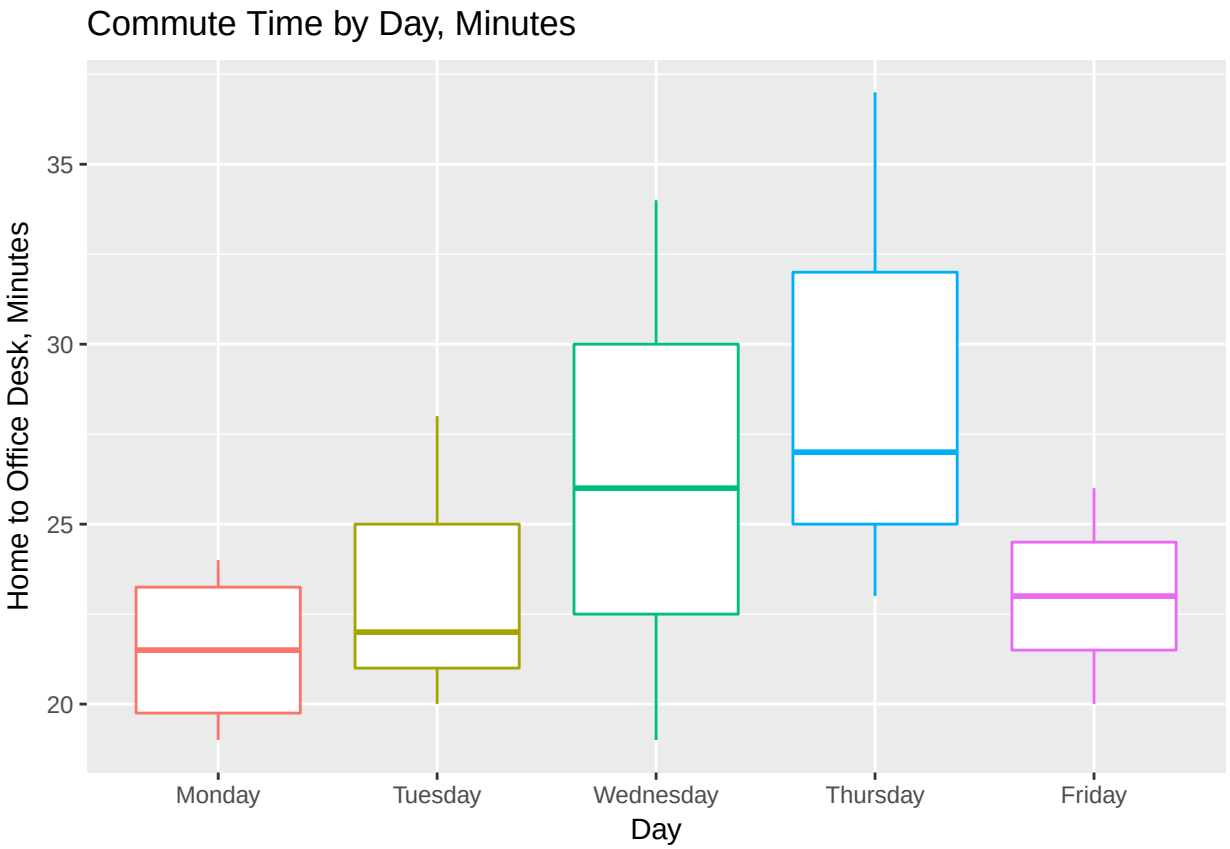
Step 2: View the data

Using the *commuteplots* function with your newly minted dataframe, one line of code will net you two relatively, well formatted graphs of your untransformed commute data.

The defaults assume that you want to look at your data comparing total drive time by day of the week. Thus, the *Time* variable = "Total", and the *Group* variable ="Day".

```
commuteplots(commute)
```

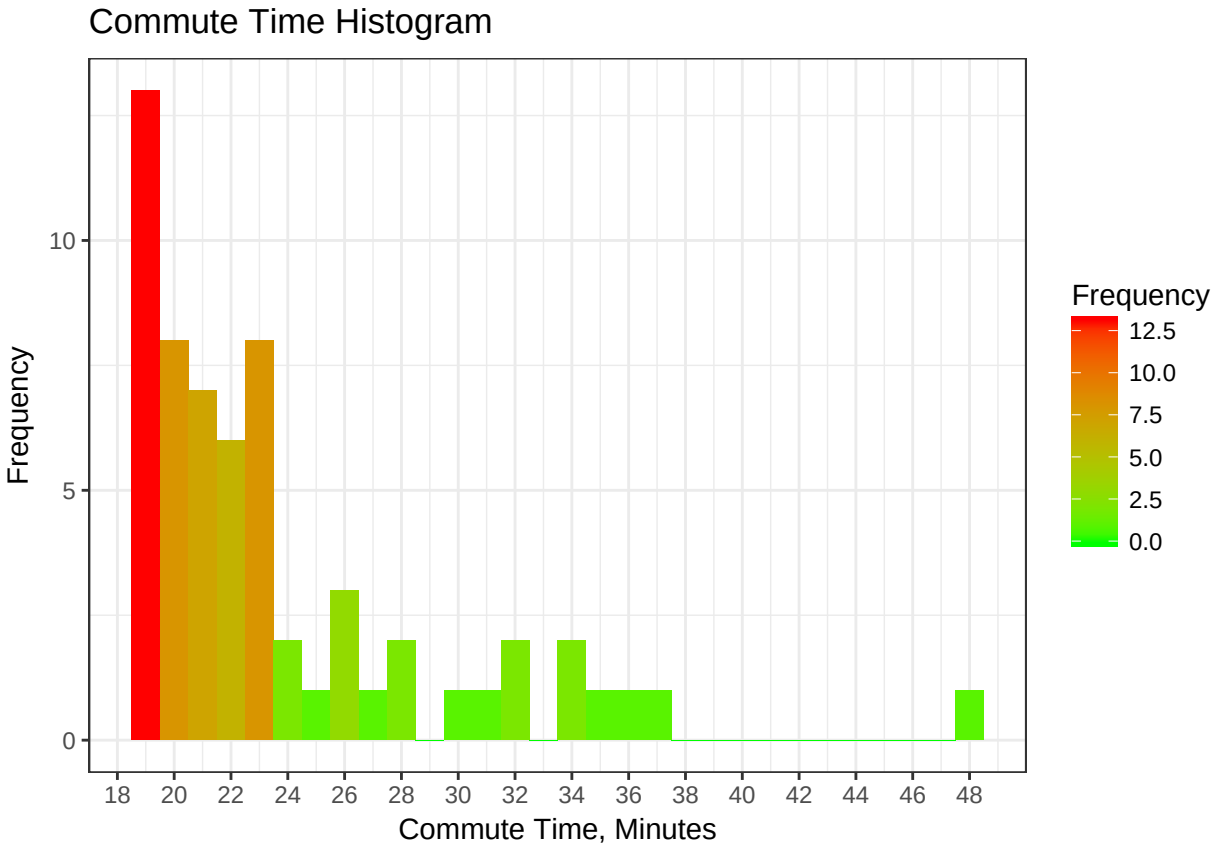


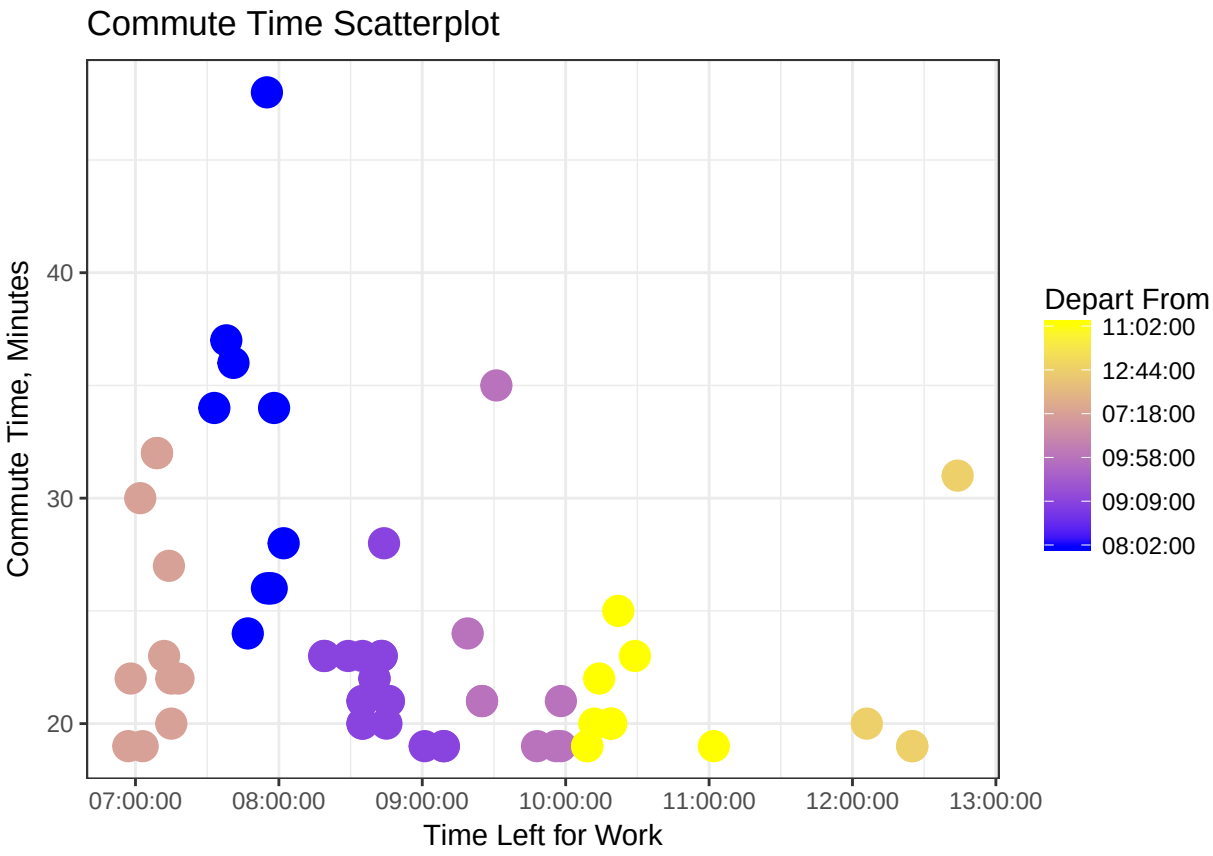


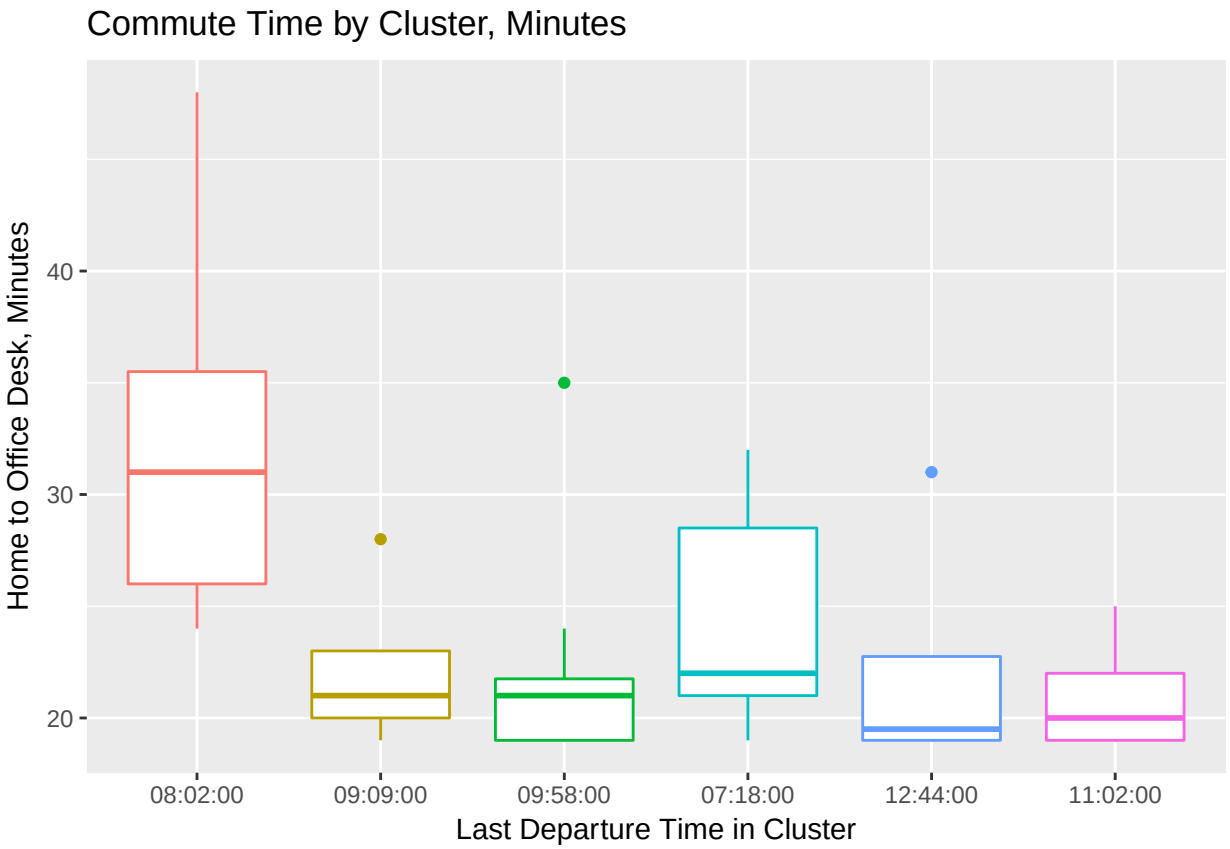
If you don't want to use days of the week, just subsitute in for *Group*. For instance, here I have run the plots based on comparing my total drive times by what I was listening to that morning. I also used actual times instead of a total drive time. This will net you another plot of total drive time explained by what time you left.

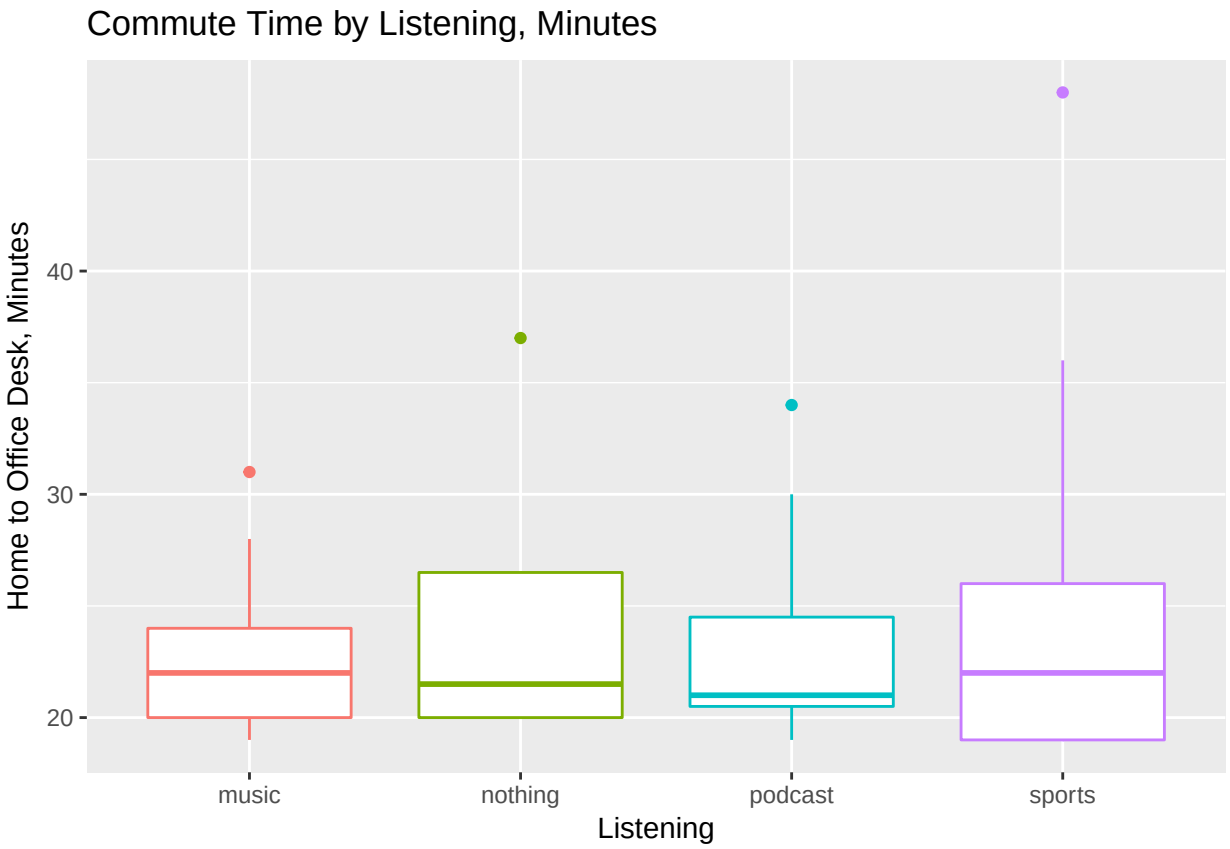
```
davscommute <- read_sheet(
  "https://docs.google.com/spreadsheets/d/12-8vjn46tIZ2Xr98_o_AcUzX5qeMAV878h63wFPo4d4/edit?usp=sharing

commuteplots(davscommute,Leave = "Left",Arrive="ArriveDesk",Group="Listening")
```

```
## Warning in groupLabels[i] <- as.character(maxdepart[, 3]): number of items
## to replace is not a multiple of replacement length

## Warning in groupLabels[i] <- as.character(maxdepart[, 3]): number of items
## to replace is not a multiple of replacement length
```


Commute Time Histogram
Frequency
10
5
0
18
20
22
24
26
28
30
32
34
36
38
40
42
44
46
48
Commute Time, Minutes
Frequency
12.5
10.0
7.5
5.0
2.5
0.0


Commute Time Scatterplot
Commute Time, Minutes
40
30
20
07:00:00
08:00:00
09:00:00
10:00:00
11:00:00
12:00:00
13:00:00
Time Left for Work
Depart From
11:02:00
12:44:00
07:18:00
09:58:00
09:09:00
08:02:00


Commute Time by Cluster, Minutes
Home to Office Desk, Minutes
40
30
20
08:02:00
09:09:00
09:58:00
07:18:00
12:44:00
11:02:00
Last Departure Time in Cluster



Step 3: Find out if times are significantly different.

Using the *commuteanova* function on your data frame will let you know if there are any significant differences based on the day you are driving. If no actual times are used, assumptions of ANOVA are ignored, and an ANOVA is run based on day of the week.

```
commuteanova(commute)
```

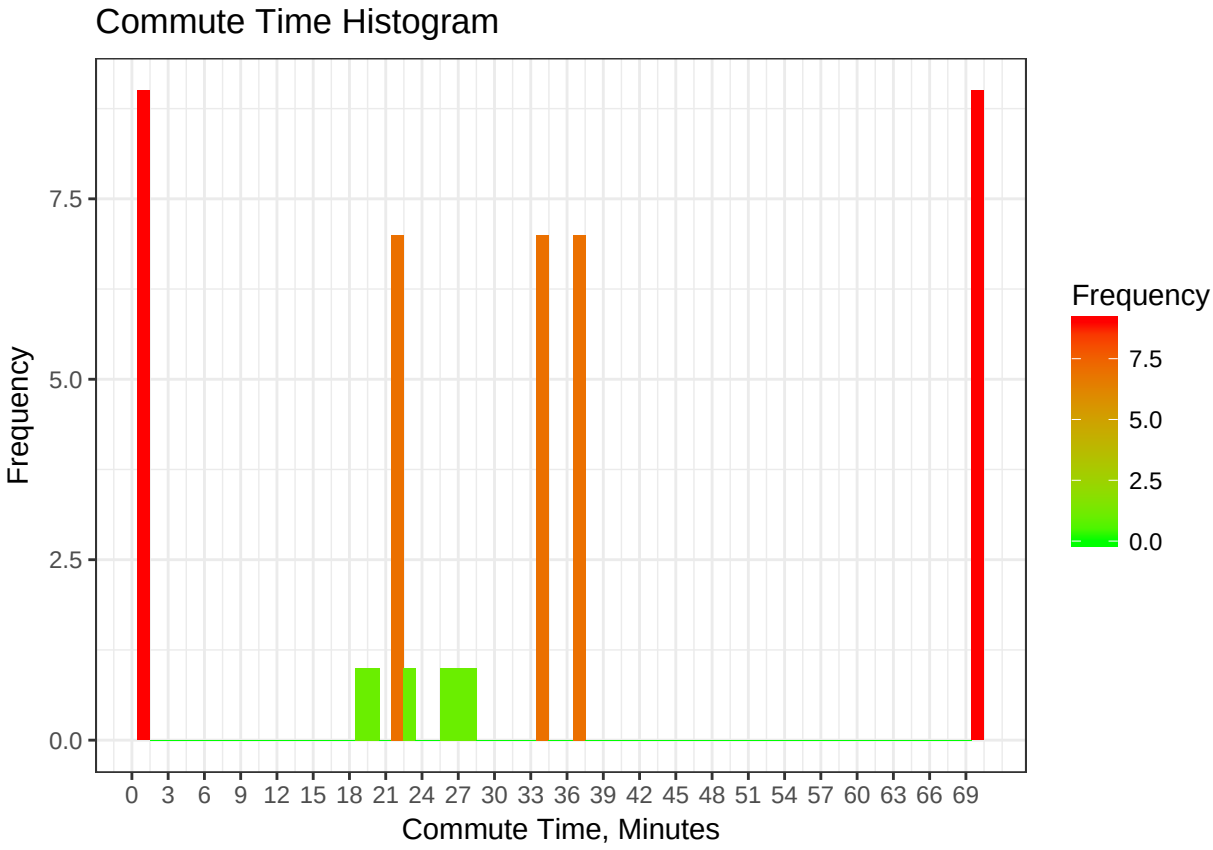
```
## [1] "Your commutes are not significantly different between one another based on Day."
## [1] "Monday is the group with the shortest average drive time."
```

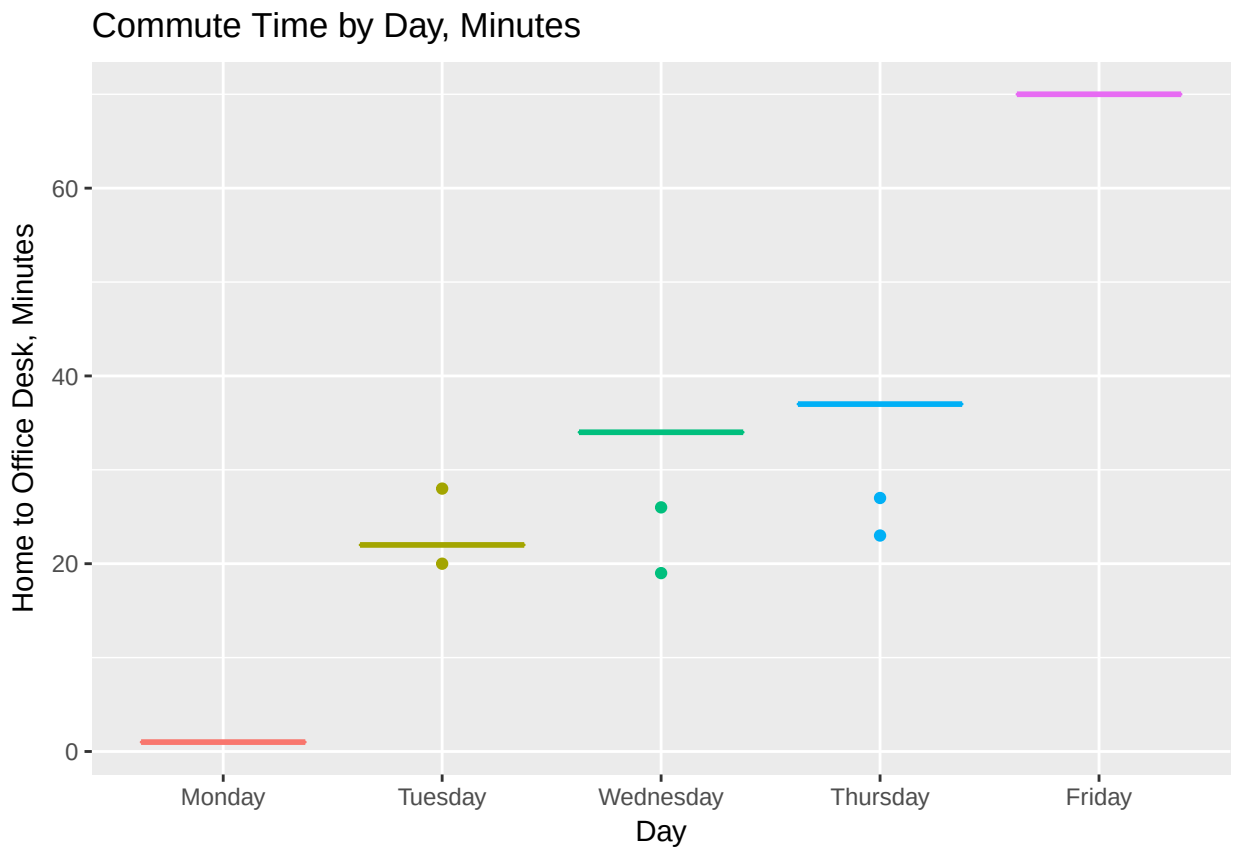
What about for days that are significantly different?

For that we'll need a Tukey Post-Hoc test and some different input to demonstrate.

Again, here is a practice link of data to use. It is artificically crafted to be significant for demonstration purposes.

```
different <- read_sheet(
"https://docs.google.com/spreadsheets/d/1d_KjDBLt-iSiRcGDvWsGQMNUlrPselEwjCN6OgwvfkM/edit?usp=sharing")

commuteplots(different)
```


Commute Time Histogram
Frequency
7.5
5.0
2.5
0.0
0 3 6 9 12 15 18 21 24 27 30 33 36 39 42 45 48 51 54 57 60 63 66 69
Commute Time, Minutes
Frequency
7.5
5.0
2.5
0.0




```
commuteanova(different)
```

```
## [1] "Your commutes are significantly different between these Day groups:"
## [1] "Monday-Friday"     "Thursday-Friday"   "Tuesday-Friday"
## [4] "Wednesday-Friday"  "Thursday-Monday"   "Tuesday-Monday"
## [7] "Wednesday-Monday"  "Tuesday-Thursday"  "Wednesday-Tuesday"
## [1] "Monday is the group with the shortest average drive time."
```

Here is an example using real data with actual times. This gives you a k-means clustering of the times you leave as well as an ANOVA to tell you if those clusters are significantly different from one another. *usertimebins* controls how many clusters are used.

For the sake of the vignette, it can not be run in a chunk, since the resulting error will stop the markdown from knitting.

commuteanova(davscommute,Leave="Left",Arrive="ArriveDesk",usertimebins=6)

[1] "Leave between 09:48:00 and 11:02:00 for the shortest average drive time based on descriptive st... Show Traceback Error in commuteanova(davscommute, Leave ="Left", Arrive ="ArriveDesk", : No formal statistical test was conducted as ANOVA assumptions were not met by the data.

Built in error messages, such as the one above, are in place to prevent an ANOVA being conducted when assumptions of an ANOVA test are not met. These include normally distributed data and homogeneity of variance.

Therein lies the problem though. Commute data seems likely to include ties, and in this case non-parameteric rank tests are unvailable due to ties. It seems then that descriptive statistics may be the only way to truly

analyze the data. In this case, the lowest mean and variance seem to be desirable, with mean probably having greater significance.

Step 4: Watch for future updates.

Future patches may include different types of statistical tests built to handle the data type better. Then you'll be extra certain you know which times or days you should drive instead of working at home. Naturally, though, you'll have some type of mandatory meeting that puts you on the road for your least optimal drive time, but this package can't do much about that! ;)

Recap

Leaving between 8:46 and 9:31 seemed to give me the best result. The mean was pretty low, and the variance of the time group was small with one outlier at 35 minutes. Other groups did seem to perform better, but there also needs to be some consideration given to when I needed to be at Saunders. Unless I wanted to leave very early in the morning, or arrive very late, this seemed to be the best compromise. Unfortunately, Probability Theory was at 9:15, and this guaranteed that at least two days a week I was on the road during the worst times, 7:18 to 8:02. Although, these interpretations are highly subjective since interpreting the boxplots is the only valid course of action at this point.

If I had it to do over again, I may have tried to synthesize some kind of test statistic based around variance and mean. It's pretty obvious now that this data didn't lend itself to my chosen method of analysis. If I were going to continue working on this, I would probably move in that direction. I may have tried to come up with another type of data to examine. My goals were so ambitious that there are a lot of logic checks built into the package that may have been unnecessary. They also took a lot of time.

```
sessionInfo()
```

```
## R version 3.4.1 (2017-06-30)
## Platform: x86_64-w64-mingw32/x64 (64-bit)
## Running under: Windows 10 x64 (build 16299)
##
## Matrix products: default
##
## locale:
## [1] LC_COLLATE=English_United States.1252
## [2] LC_CTYPE=English_United States.1252
## [3] LC_MONETARY=English_United States.1252
## [4] LC_NUMERIC=C
## [5] LC_TIME=English_United States.1252
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
## [1] bindrcpp_0.2                 commuteconvergence_0.0.0.9001
##
## loaded via a namespace (and not attached):
##  [1] Rcpp_0.12.13       xml2_1.1.1         knitr_1.17
##  [4] bindr_0.1          magrittr_1.5       hms_0.3
##  [7] googlesheets_0.2.2 munsell_0.4.3      colorspace_1.3-2
## [10] R6_2.2.2           rlang_0.1.2        plyr_1.8.4
## [13] stringr_1.2.0      httr_1.3.1         dplyr_0.7.4
## [16] tools_3.4.1        grid_3.4.1         gtable_0.2.0
```

```
## [19] htmltools_0.3.6    lazyeval_0.2.0     yaml_2.1.14
## [22] rprojroot_1.2      digest_0.6.12      assertthat_0.2.0
## [25] tibble_1.3.4       ggplot2_2.2.1      readr_1.1.1
## [28] purrr_0.2.4        curl_3.0           glue_1.1.1
## [31] evaluate_0.10.1    rmarkdown_1.8      labeling_0.3
## [34] stringi_1.1.5      compiler_3.4.1     cellranger_1.1.0
## [37] scales_0.5.0       backports_1.1.1    pkgconfig_2.0.1
```