Lesson 7

Multivariate Data

Notes:

Moira Perceived Audience Size Colored by Age

Notes:

###Reading in Data

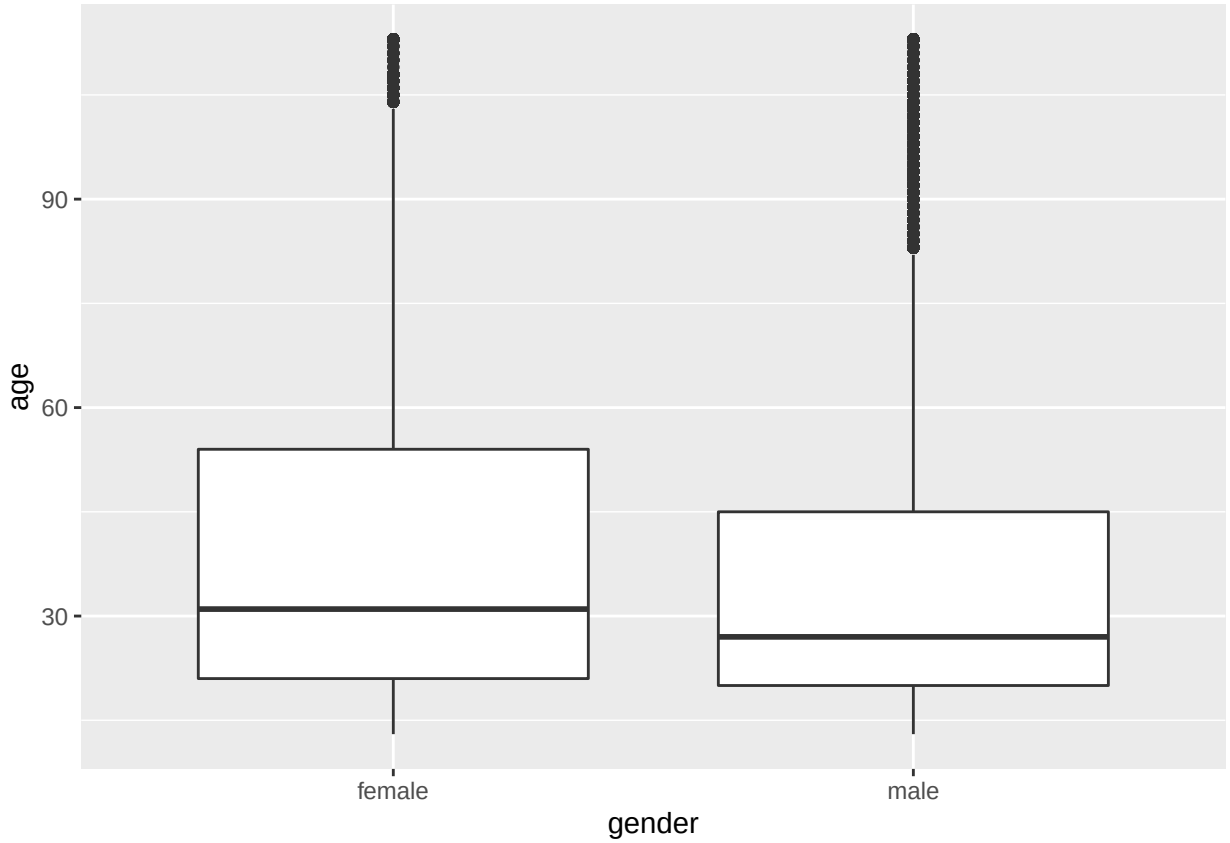
```
pf=read.csv('pseudo_facebook.tsv',sep='\t')
names(pf)
```

```
##  [1] "userid"                "age"                   "dob_day"
##  [4] "dob_year"              "dob_month"             "gender"
##  [7] "tenure"                "friend_count"          "friendships_initiated"
## [10] "likes"                 "likes_received"        "mobile_likes"
## [13] "mobile_likes_received" "www_likes"             "www_likes_received"
```

Third Qualitative Variable

Notes:

```
library(ggplot2)

ggplot(aes(x = gender, y = age),
       data = subset(pf, !is.na(gender))) +
  geom_boxplot()
```

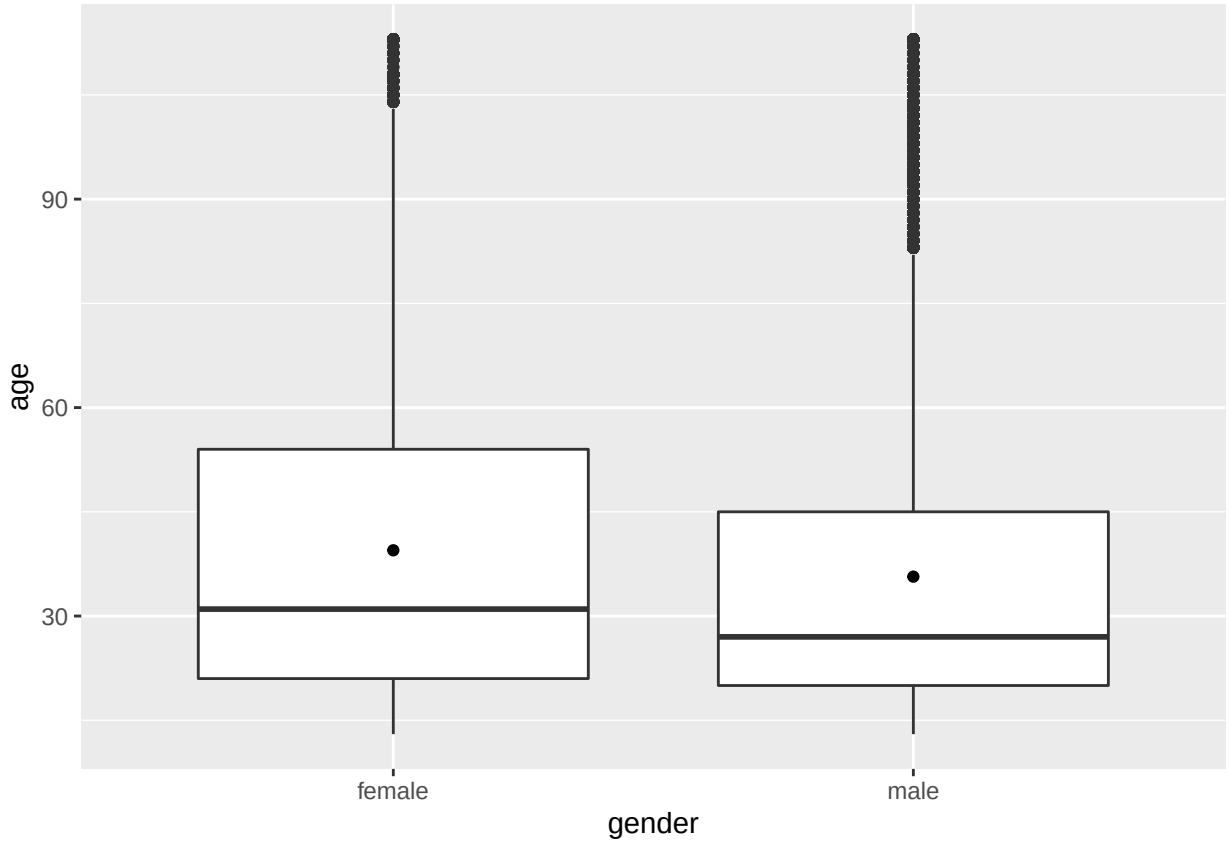



```
ggsave('boxplot1.png')
```

```
## Saving 6.5 x 4.5 in image
```

```
ggplot(aes(x = gender, y = age),
       data = subset(pf, !is.na(gender))) +
  geom_boxplot() +
  stat_summary(fun = mean, geom = 'point', shap = 4)
```

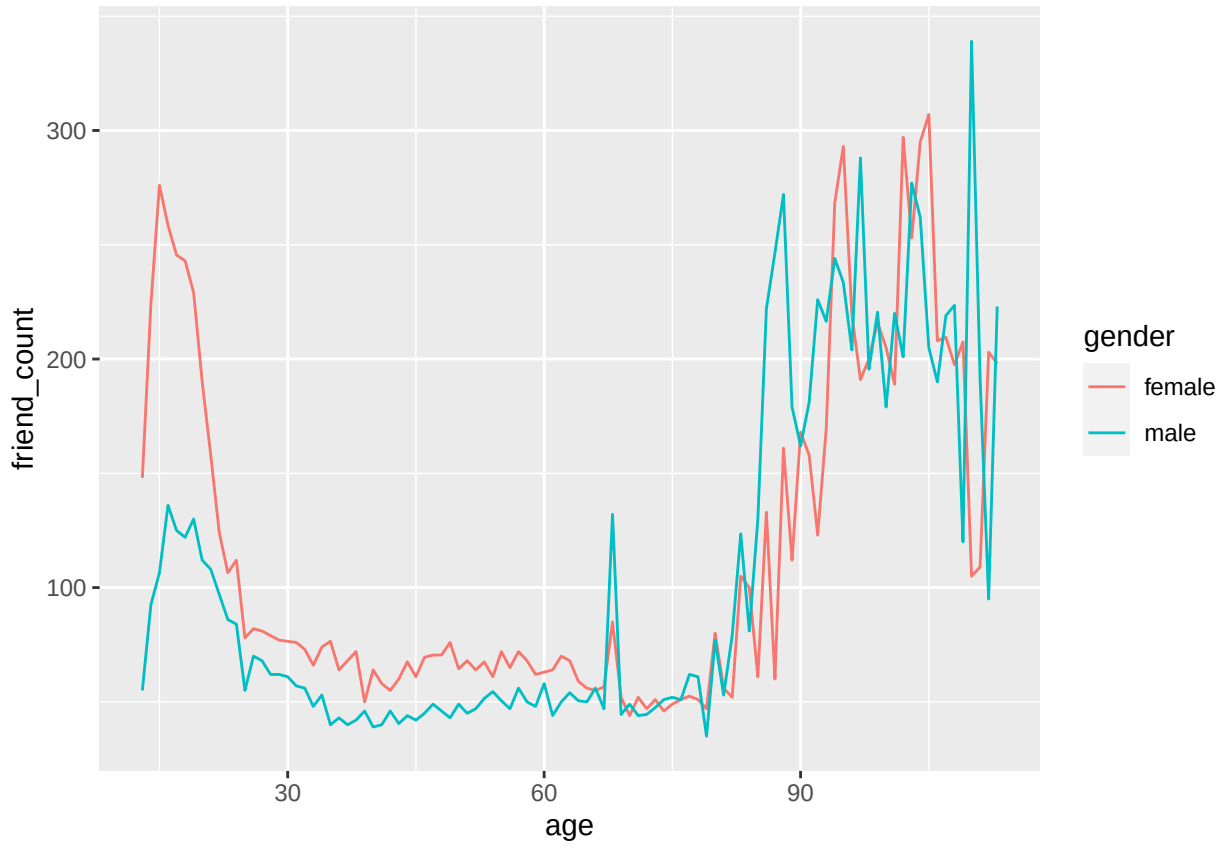
```
## Warning: Ignoring unknown parameters: shap
```




```
ggsave('boxplot2.png')
```

```
## Saving 6.5 x 4.5 in image
```

```
ggplot(aes(x = age, y = friend_count),
       data = subset(pf, !is.na(gender))) +
  geom_line(aes(color = gender), stat = 'summary', fun = median)
```




```
ggsave('lineplot3.png')
```

```
## Saving 6.5 x 4.5 in image
```

```
# Write code to create a new data frame,
# called 'pf.fc_by_age_gender', that contains
# information on each age AND gender group.

# The data frame should contain the following variables:

#    mean_friend_count,
#    median_friend_count,
#    n (the number of users in each age and gender grouping)

# Here is an example of the structure of your data frame. Your
# data values will be different. Note that if you are grouping by
# more than one variable, you will probably need to call the
# ungroup() function.

#   age gender mean_friend_count median_friend_count    n
# 1  13 female          247.2953                 150  207
# 2  13   male          184.2342                  61  265
# 3  14 female          329.1938                 245  834
# 4  14   male          157.1204                  88 1201
```

```
# ENTER YOUR CODE BELOW THIS LINE.
# ============================================================
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
```

```
pf.fc_by_age_gender = pf %>%
  filter(!is.na(gender)) %>%
  group_by(age, gender) %>%
  summarise(mean_friend_count = mean(friend_count),
            median_friend_count = median(friend_count),
            n = n()) %>%
  ungroup() %>%
  arrange(age)
head(pf.fc_by_age_gender, 10)
```

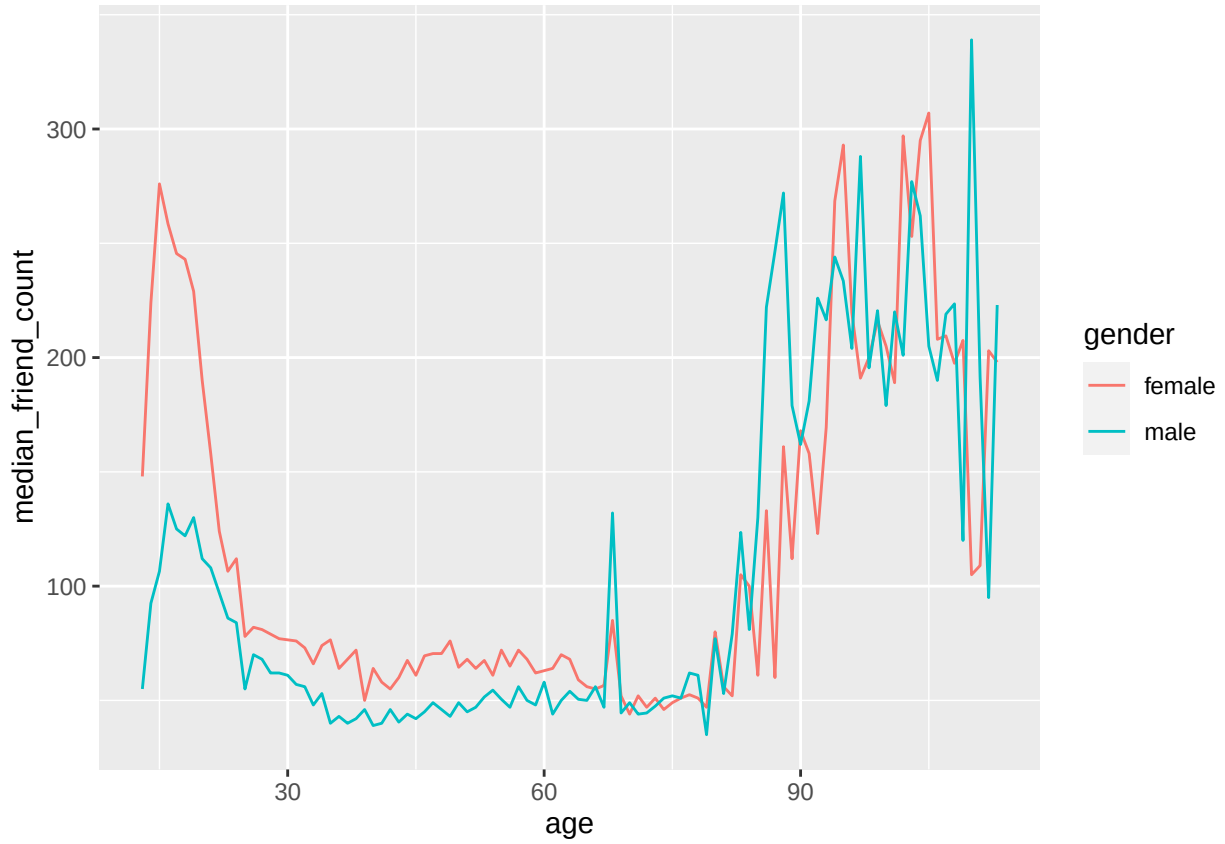
```
## # A tibble: 10 x 5
##      age gender mean_friend_count median_friend_count     n
##    <int> <fct>              <dbl>               <dbl> <int>
##  1    13 female              259.                148     193
##  2    13 male                102.                 55     291
##  3    14 female              362.                224     847
##  4    14 male                164.                 92.5  1078
##  5    15 female              539.                276    1139
##  6    15 male                201.                106.   1478
##  7    16 female              520.                258.   1238
##  8    16 male                240.                136    1848
##  9    17 female              539.                246.   1236
## 10    17 male                236.                125    2045
```

Plotting Conditional Summaries

Notes:

```
# Create a line graph showing the
# median friend count over the ages
# for each gender. Be sure to use
# the data frame you just created,
# pf.fc_by_age_gender.
```

```
ggplot(aes(x = age, y = median_friend_count),
       data = pf.fc_by_age_gender) +
  geom_line(aes(color = gender))
```




```
ggsave('lineplot4.png')
```

```
## Saving 6.5 x 4.5 in image
```

Thinking in Ratios

Notes:

Wide and Long Format

Notes:

Reshaping Data

Notes:

```
#install.packages('reshape2')
library(reshape2)
#install.packages("tidyr")
head(pf.fc_by_age_gender, 5)
```

```
## # A tibble: 5 x 5
##     age gender mean_friend_count median_friend_count     n
##   <int> <fct>              <dbl>               <dbl> <int>
## 1    13 female              259.                148     193
## 2    13 male                102.                 55     291
## 3    14 female              362.                224     847
## 4    14 male                164.                 92.5  1078
## 5    15 female              539.                276    1139
```

```
library(tidyr)
```

```
##
## Attaching package: 'tidyr'
```

```
## The following object is masked from 'package:reshape2':
##
##     smiths
```

```
spread(subset(pf.fc_by_age_gender,
       select = c('gender', 'age', 'median_friend_count')),
       gender, median_friend_count)
```

```
## # A tibble: 101 x 3
##      age female  male
##    <int>  <dbl> <dbl>
##  1    13   148   55
##  2    14   224   92.5
##  3    15   276  106.
##  4    16   258. 136
##  5    17   246. 125
##  6    18   243  122
##  7    19   229  130
##  8    20   190  112
##  9    21   158  108
## 10    22   124   97
## # ... with 91 more rows
```

```
pf.fc_by_age_gender.wide = dcast(pf.fc_by_age_gender,
                                 age ~ gender,
                                 value.var = 'median_friend_count')
head(pf.fc_by_age_gender.wide, 6)
```

```
##   age female  male
## 1  13  148.0  55.0
## 2  14  224.0  92.5
## 3  15  276.0 106.5
## 4  16  258.5 136.0
## 5  17  245.5 125.0
## 6  18  243.0 122.0
```

```
library(dplyr)
pf.fc_by_age_gender.wide = subset(pf.fc_by_age_gender[c('age', 'gender', 'median_friend_count')], !is.na
  spread(gender, median_friend_count) %>%
  mutate(ratio= female / male)
head(pf.fc_by_age_gender.wide, 8)
```

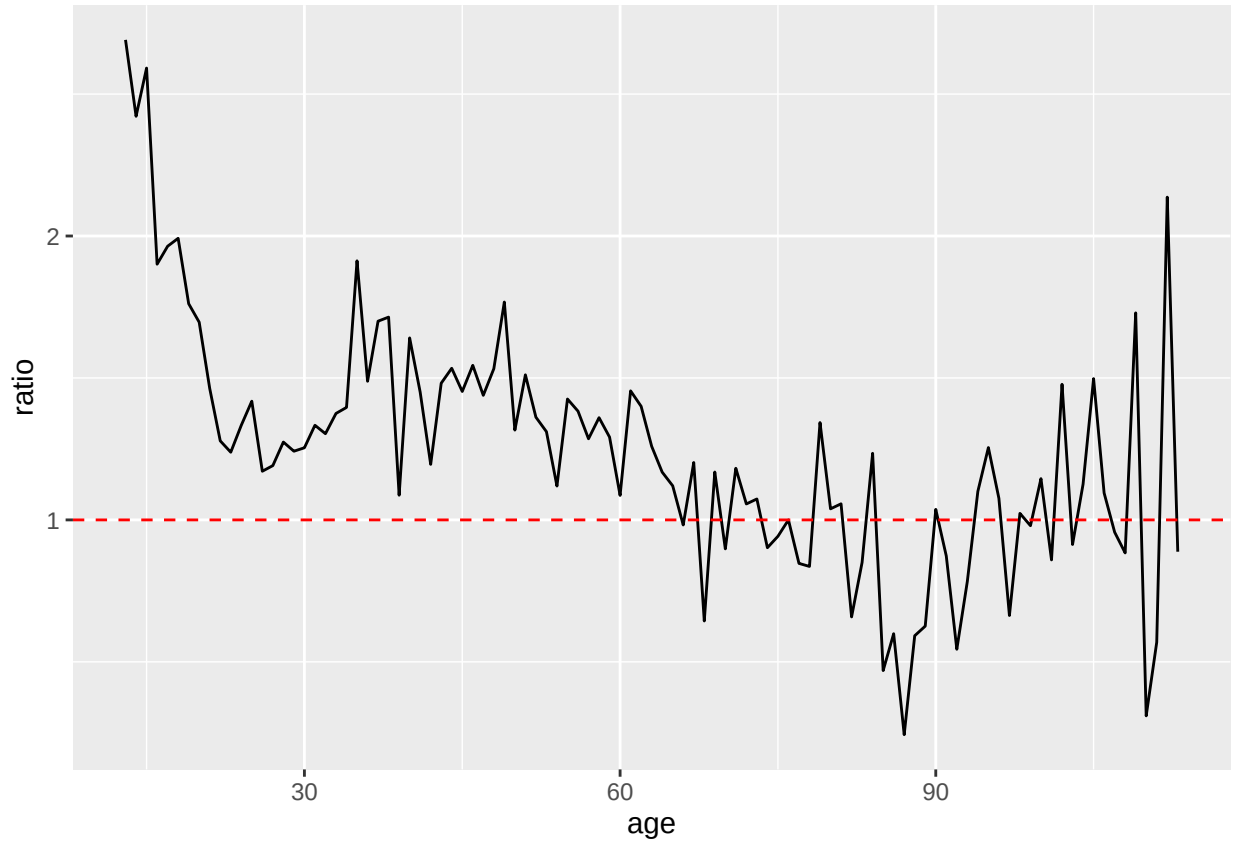
```
## # A tibble: 8 x 4
##     age female  male ratio
##   <int>  <dbl> <dbl> <dbl>
## 1    13   148   55    2.69
## 2    14   224   92.5  2.42
## 3    15   276  106.   2.59
## 4    16   258. 136    1.90
## 5    17   246. 125    1.96
## 6    18   243  122    1.99
## 7    19   229  130    1.76
## 8    20   190  112    1.70
```

Ratio Plot

Notes:

```
# Plot the ratio of the female to male median
# friend counts using the data frame
# pf.fc_by_age_gender.wide.

# Think about what geom you should use.
# Add a horizontal line to the plot with
# a y intercept of 1, which will be the
# base line. Look up the documentation
# for geom_hline to do that. Use the parameter
# linetype in geom_hline to make the
# line dashed.

# The linetype parameter can take the values 0-6:
# 0 = blank, 1 = solid, 2 = dashed
# 3 = dotted, 4 = dotdash, 5 = longdash
# 6 = twodash

# This assignment is not graded and
# will be marked as correct when you submit.
```

```
# ENTER YOUR CODE BELOW THIS LINE
# ================================================

library(ggplot2)
ggplot(aes(x = age, y = ratio), data = pf.fc_by_age_gender.wide) +
  geom_line() +
  geom_hline(yintercept = 1, linetype = 2, color = 'red')
```




```
ggsave('ratio.png')
```

```
## Saving 6.5 x 4.5 in image
```

Third Quantitative Variable

Notes:

```
# Create a variable called year_joined
# in the pf data frame using the variable
# tenure and 2014 as the reference year.

# The variable year joined should contain the year
```

```
# that a user joined facebook.

# See the Instructor Notes for three hints if you get
# stuck. Scroll down slowly to see one hint at a time
# if you would like some guidance.

# This programming exercise WILL BE automatically graded.

pf$year_joined = 2014 - ceiling(pf$tenure / 365)
```

Cut a Variable

Notes:

```
summary(pf$year_joined)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
##    2005    2012    2012    2012    2013    2014       2
```

```
table(pf$year_joined)
```

```
##
##  2005  2006  2007  2008  2009  2010  2011  2012  2013  2014
##     9    15   581  1507  4557  5448  9860 33366 43588    70
```

```
# Create a new variable in the data frame
# called year_joined.bucket by using
# the cut function on the variable year_joined.

# You need to create the following buckets for the
# new variable, year_joined.bucket

#        (2004, 2009]
#        (2009, 2011]
#        (2011, 2012]
#        (2012, 2014]

# Note that a parenthesis means exclude the year and a
# bracket means include the year.
pf$year_joined.bucket = cut(pf$year_joined,
                            c(2004, 2009, 2011, 2012, 2014))
table(pf$year_joined.bucket)
```

```
##
## (2004,2009] (2009,2011] (2011,2012] (2012,2014]
##        6669       15308       33366       43658
```

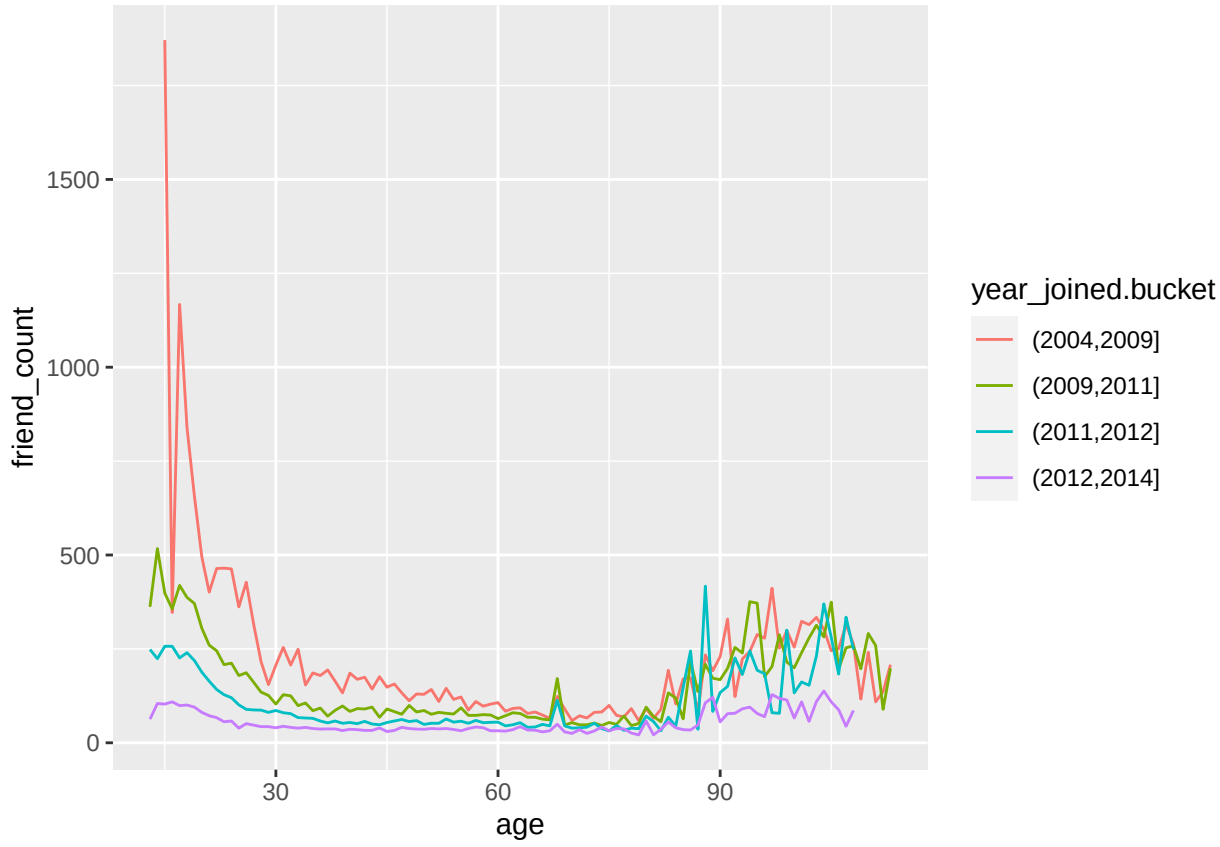
Plotting it All Together

Notes:

```
table(pf$year_joined.bucket, useNA = 'ifany')
```

```
##
## (2004,2009] (2009,2011] (2011,2012] (2012,2014]        <NA>
##        6669       15308       33366       43658           2
```

```
# Create a line graph of friend_count vs. age
# so that each year_joined.bucket is a line
# tracking the median user friend_count across
# age. This means you should have four different
# lines on your plot.

# You should subset the data to exclude the users
# whose year_joined.bucket is NA.

ggplot(aes(x = age, y = friend_count),
       data = subset(pf, !is.na(year_joined.bucket))) +
  stat_summary(aes(color = year_joined.bucket),
               fun = median,
               geom = "line")
```



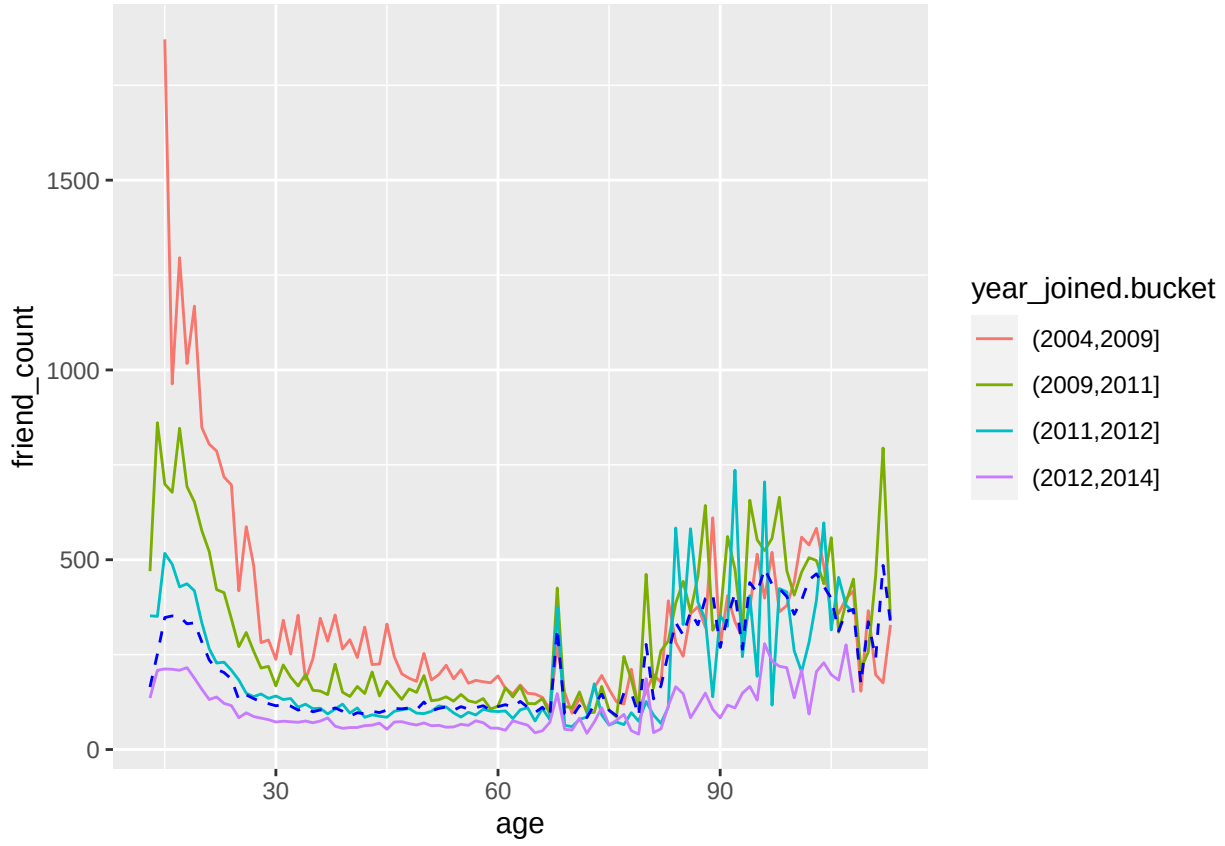
```
ggsave('year_joined.bucket1.png')
```

```
## Saving 6.5 x 4.5 in image
```

Plot the Grand Mean

Notes:

```
# Write code to do the following:

# (1) Add another geom_line to code below
# to plot the grand mean of the friend count vs age.

# (2) Exclude any users whose year_joined.bucket is NA.

# (3) Use a different line type for the grand mean.

# As a reminder, the parameter linetype can take the values 0-6:

# 0 = blank, 1 = solid, 2 = dashed
# 3 = dotted, 4 = dotdash, 5 = longdash
# 6 = twodash

# The code from the last programming exercise should
# be your starter code!


ggplot(aes(x = age, y = friend_count),
       data = subset(pf, !is.na(year_joined.bucket))) +
  stat_summary(aes(color = year_joined.bucket),
               fun = mean,
               geom = "line") +
  stat_summary(fun = mean, geom = 'line', linetype = 2, color = 'blue')
```




```
ggsave('year_joined.bucket2.png')
```

```
## Saving 6.5 x 4.5 in image
```

Friending Rate

Notes: What is the median friend rate? Response: 0.2205 What is the maximum friend rate? Response: 417

```
summary(pf$tenure)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
##     0.0   226.0   412.0   537.9   675.0  3139.0       2
```

```
friending_rate = subset(pf, tenure > 0)
summary(friending_rate$friend_count / friending_rate$tenure)
```

```
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
##   0.0000   0.0775   0.2205   0.6096   0.5658 417.0000
```

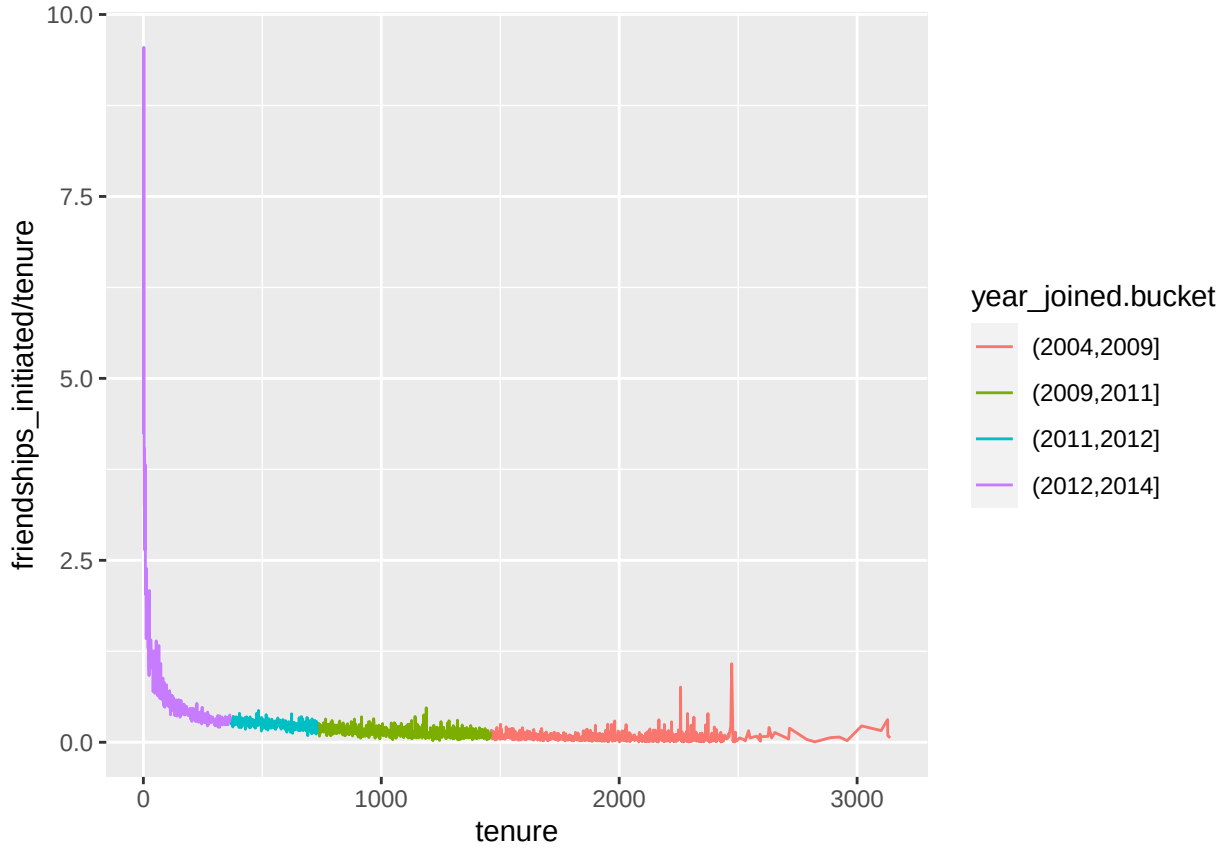
```
# alternate code:
# with(subset(pf, tenure > 0), summary(friend_count / tenure))
```

Friendships Initiated

Notes:

What is the median friend rate? Response: 0.2205 What is the maximum friend rate? Response: 417

```
# Create a line graph of mean of friendships_initiated per day (of tenure)
# vs. tenure colored by year_joined.bucket.

# You need to make use of the variables tenure,
# friendships_initiated, and year_joined.bucket.

# You also need to subset the data to only consider user with at least
# one day of tenure.

ggplot(aes(x = tenure, y = friendships_initiated / tenure),
       data = subset(pf, tenure >= 1)) +
  stat_summary(aes(color = year_joined.bucket),
               fun = mean,
               geom = "line")
```



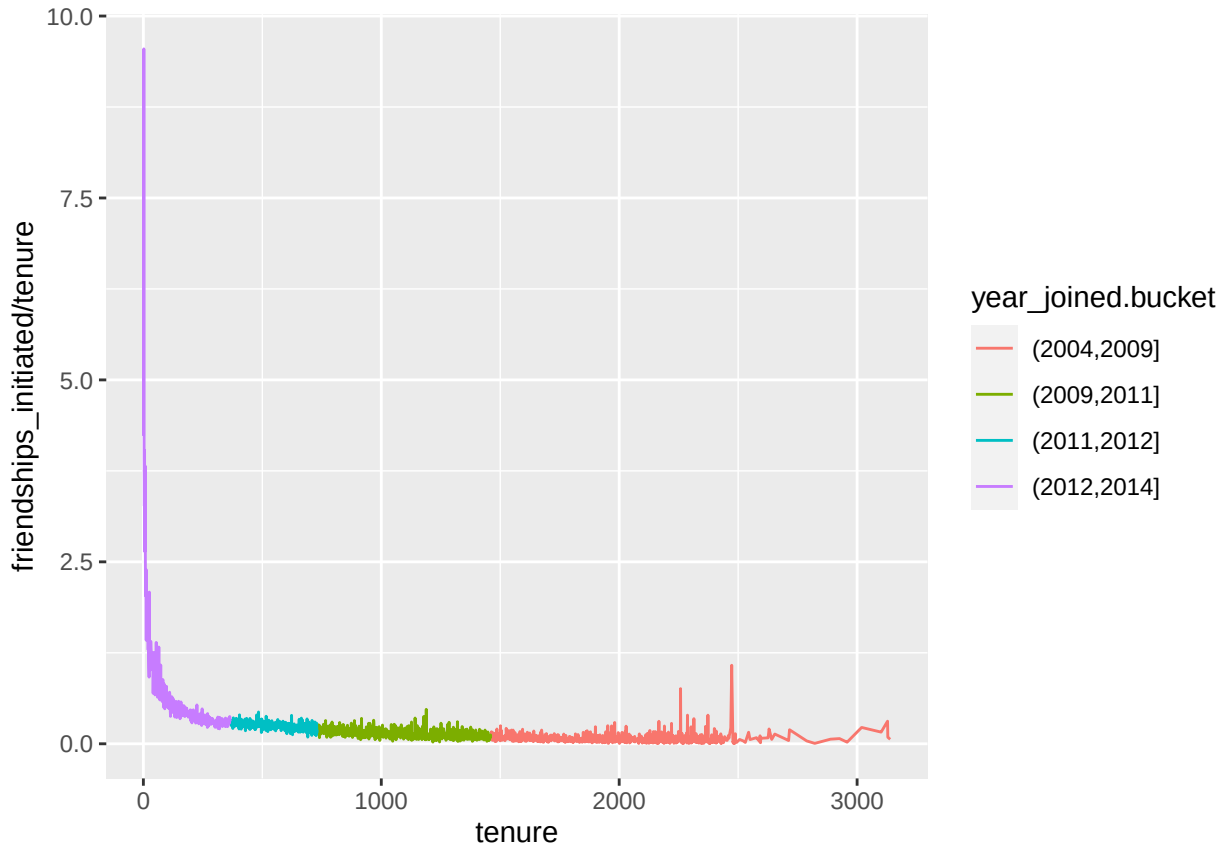
```
ggsave('year_joined.bucket3.png')
```

```
## Saving 6.5 x 4.5 in image
```

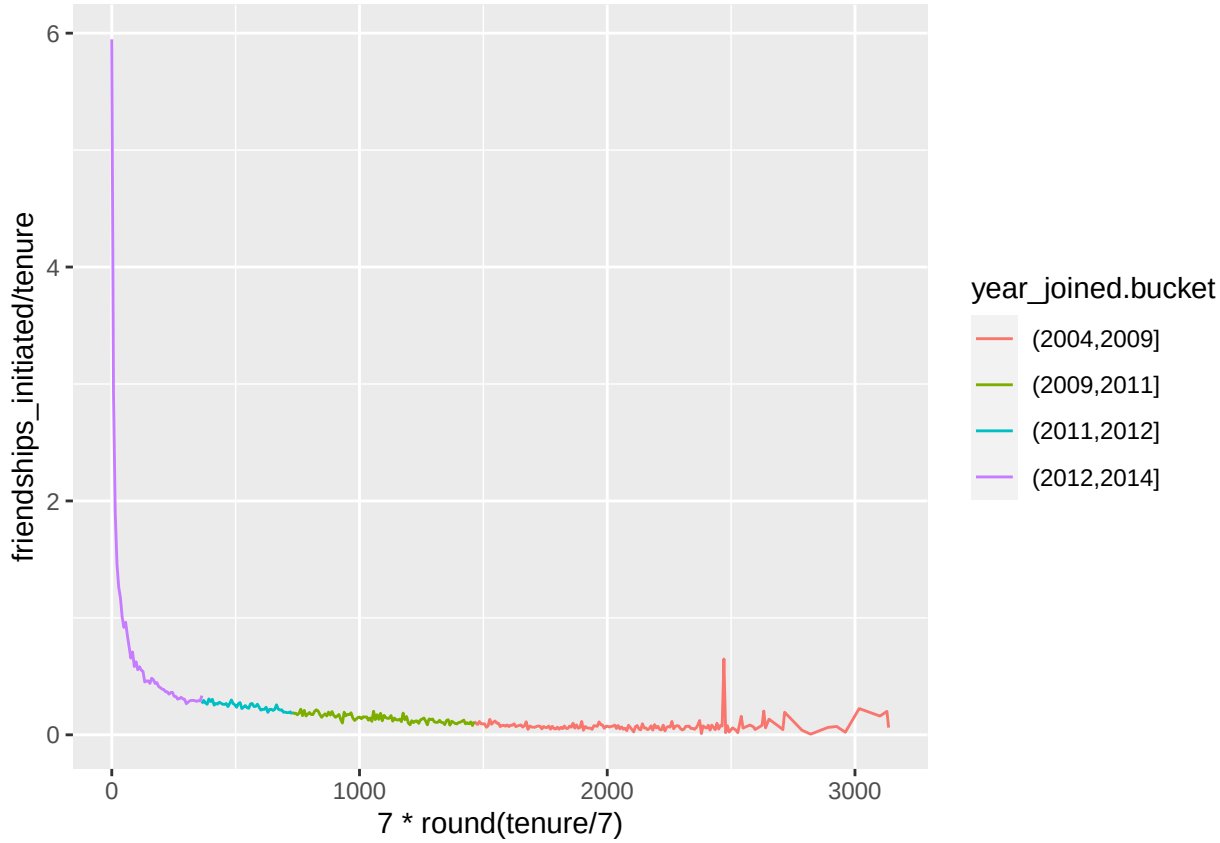
Bias-Variance Tradeoff Revisited

Notes:

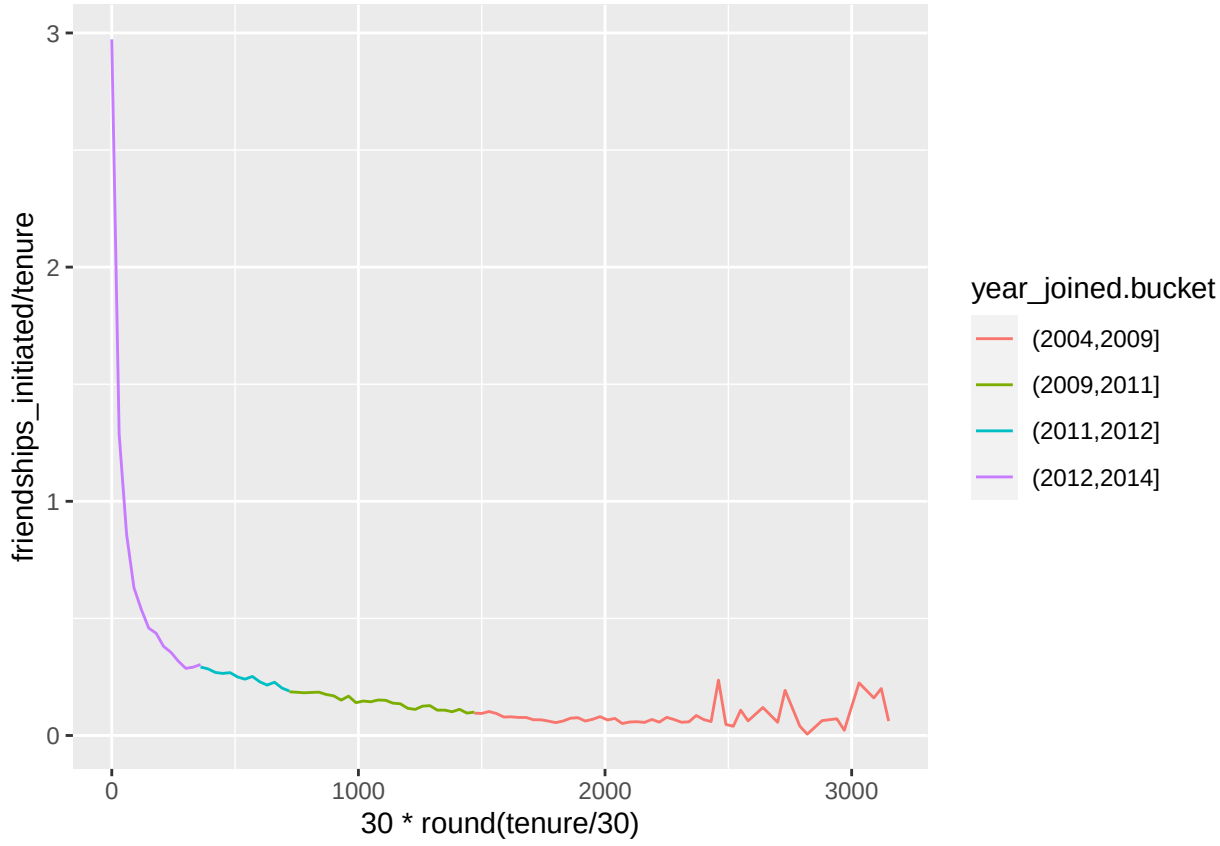
```
ggplot(aes(x = tenure, y = friendships_initiated / tenure),
       data = subset(pf, tenure >= 1)) +
  stat_summary(aes(color = year_joined.bucket),
               fun = mean,
               geom = "line")
```



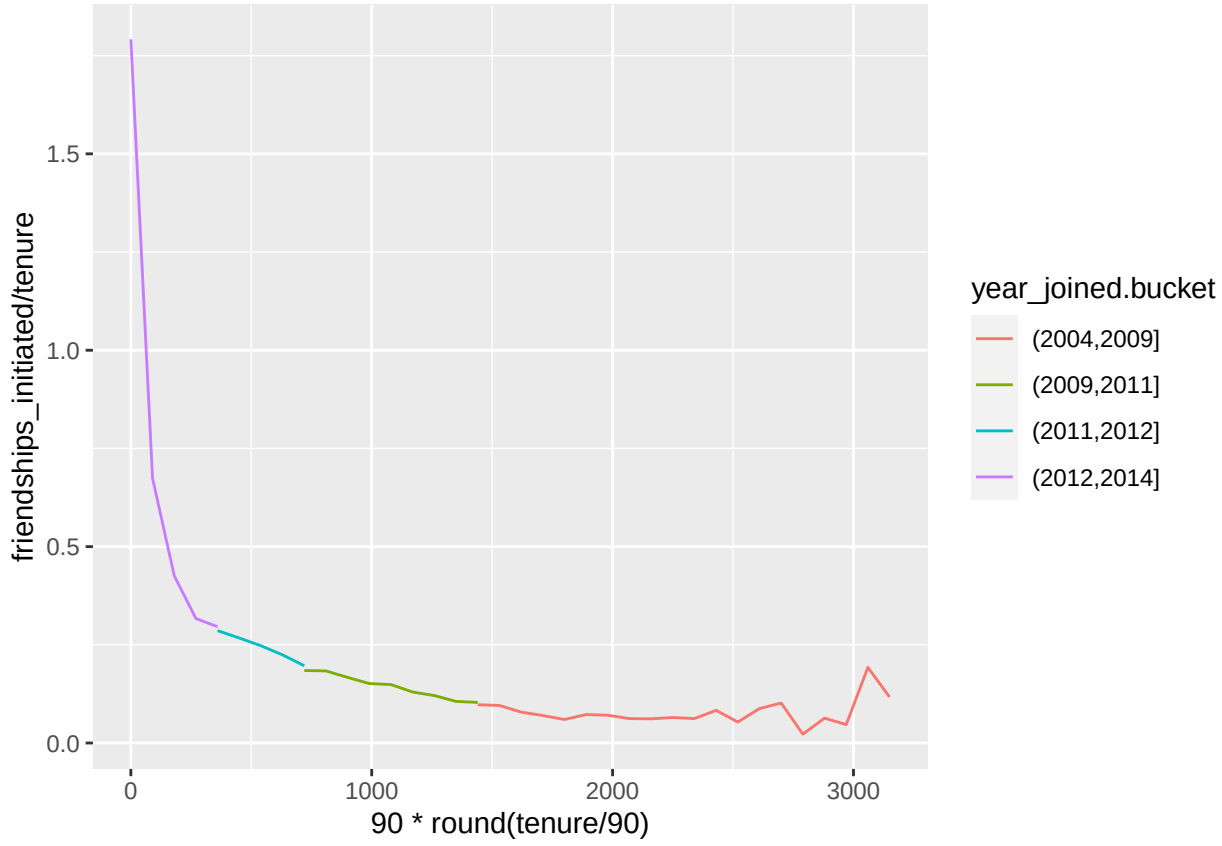

```
ggplot(aes(x = 7 * round(tenure / 7), y = friendships_initiated / tenure),
       data = subset(pf, tenure >= 1)) +
  stat_summary(aes(color = year_joined.bucket),
               fun = mean,
               geom = "line")
```




```
ggplot(aes(x = 30 * round(tenure / 30), y = friendships_initiated / tenure),
       data = subset(pf, tenure >= 1)) +
  stat_summary(aes(color = year_joined.bucket),
               fun = mean,
               geom = "line")
```

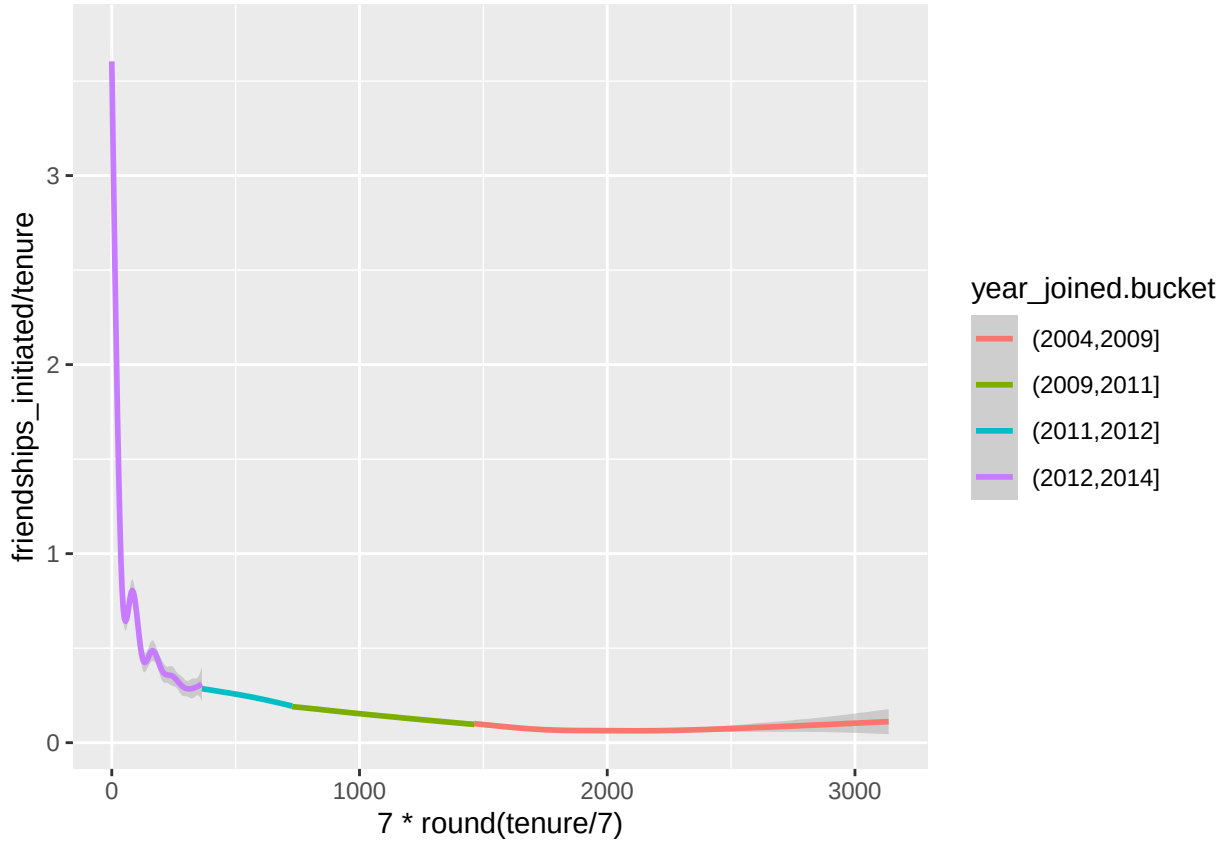



```
ggplot(aes(x = 90 * round(tenure / 90), y = friendships_initiated / tenure),
       data = subset(pf, tenure >= 1)) +
  stat_summary(aes(color = year_joined.bucket),
               fun = mean,
               geom = "line")
```




```
# Instead of geom_line(), use geom_smooth() to add a smoother to the plot.
# You can use the defaults for geom_smooth() but do color the line
# by year_joined.bucket

ggplot(aes(x = 7 * round(tenure / 7), y = friendships_initiated / tenure),
       data = subset(pf, tenure >= 1)) +
  geom_smooth(aes(color = year_joined.bucket))
```

```
## `geom_smooth()` using method = 'gam' and formula 'y ~ s(x, bs = "cs")'
```



Sean's NFL Fan Sentiment Study

Notes:

Introducing the Yogurt Data Set

Notes:

Histograms Revisited

Notes:

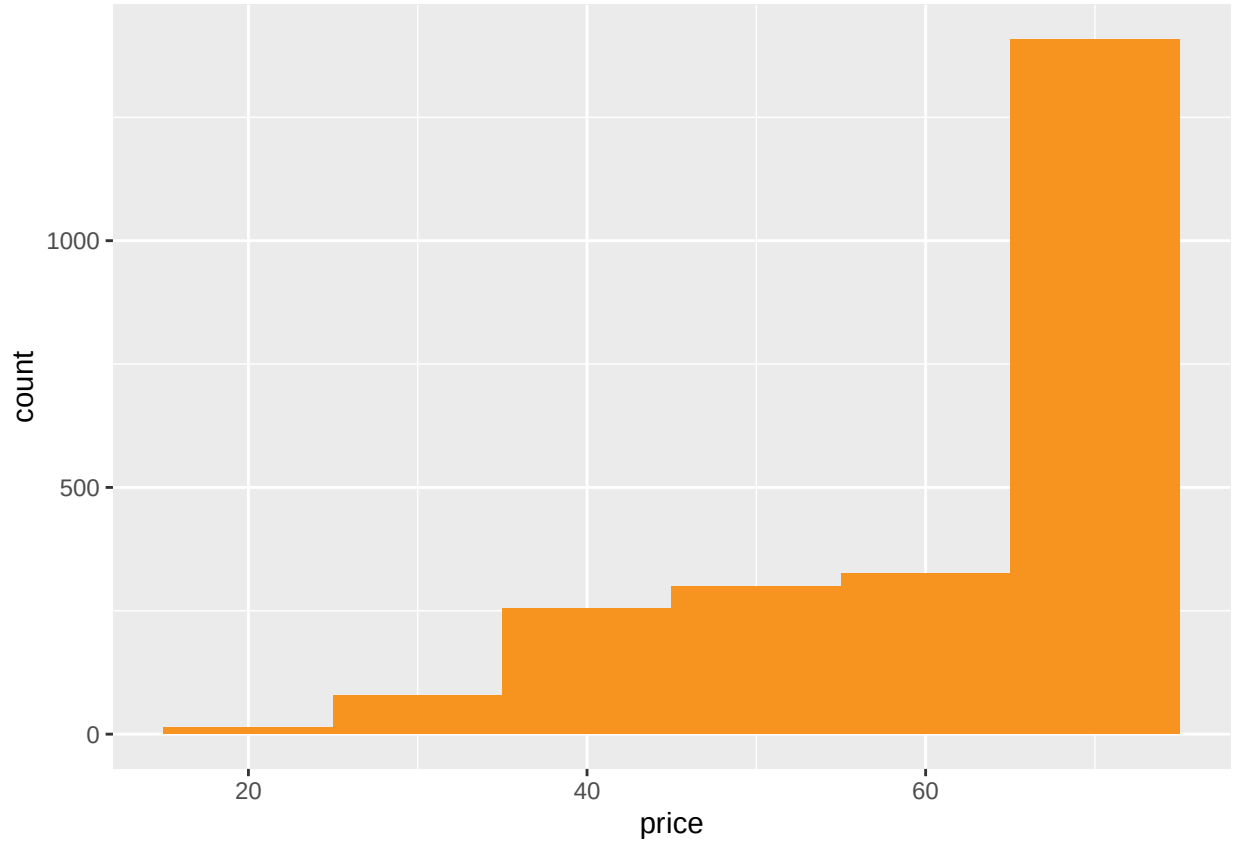
```
yo = read.csv('yogurt.csv')
str(yo)
```

```
## 'data.frame':    2380 obs. of  9 variables:
##  $ obs        : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ id         : int  2100081 2100081 2100081 2100081 2100081 2100081 2100081 2100081 2100081 2100081
##  $ time       : int  9678 9697 9825 9999 10015 10029 10036 10042 10083 10091 ...
##  $ strawberry : int  0 0 0 0 1 1 0 0 0 0 ...
##  $ blueberry  : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ pina.colada: int  0 0 0 0 1 2 0 0 0 0 ...
##  $ plain      : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ mixed.berry: int  1 1 1 1 1 1 1 1 1 1 ...
##  $ price      : num  59 59 65 65 49 ...
```

```
yo$id = factor(yo$id)
str(yo)
```

```
## 'data.frame':    2380 obs. of  9 variables:
##  $ obs        : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ id         : Factor w/ 332 levels "2100081","2100370",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ time       : int  9678 9697 9825 9999 10015 10029 10036 10042 10083 10091 ...
##  $ strawberry : int  0 0 0 0 1 1 0 0 0 0 ...
##  $ blueberry  : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ pina.colada: int  0 0 0 0 1 2 0 0 0 0 ...
##  $ plain      : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ mixed.berry: int  1 1 1 1 1 1 1 1 1 1 ...
##  $ price      : num  59 59 65 65 49 ...
```

```
library(ggplot2)
ggplot(aes(x = price), data = yo) +
  geom_histogram(binwidth = 10, fill = '#F79420')
```




```
ggsave('hist1.png')
```

```
## Saving 6.5 x 4.5 in image
```

Number of Purchases

Notes:

```
summary(yo)
```

```
##       obs              id            time         strawberry
##  Min.   :   1.0   2132290:  74   Min.   : 9662   Min.   : 0.0000
##  1st Qu.: 696.5   2130583:  59   1st Qu.: 9843   1st Qu.: 0.0000
##  Median :1369.5   2124073:  50   Median :10045   Median : 0.0000
##  Mean   :1367.8   2149500:  50   Mean   :10050   Mean   : 0.6492
##  3rd Qu.:2044.2   2101790:  47   3rd Qu.:10255   3rd Qu.: 1.0000
##  Max.   :2743.0   2129528:  39   Max.   :10459   Max.   :11.0000
##                   (Other):2061
##    blueberry        pina.colada          plain         mixed.berry
##  Min.   : 0.0000   Min.   : 0.0000   Min.   :0.0000   Min.   :0.0000
##  1st Qu.: 0.0000   1st Qu.: 0.0000   1st Qu.:0.0000   1st Qu.:0.0000
##  Median : 0.0000   Median : 0.0000   Median :0.0000   Median :0.0000
```

```
##  Mean   : 0.3571   Mean   : 0.3584   Mean   :0.2176   Mean   :0.3887
##  3rd Qu.: 0.0000   3rd Qu.: 0.0000   3rd Qu.:0.0000   3rd Qu.:0.0000
##  Max.   :12.0000   Max.   :10.0000   Max.   :6.0000   Max.   :8.0000
##
##      price
##  Min.   :20.00
##  1st Qu.:50.00
##  Median :65.04
##  Mean   :59.25
##  3rd Qu.:68.96
##  Max.   :68.96
##
```

```
unique(yo$price)
```

```
##  [1] 58.96 65.04 48.96 68.96 39.04 24.96 50.00 45.04 33.04 44.00 33.36 55.04
## [13] 62.00 20.00 49.60 49.52 33.28 63.04 33.20 33.52
```

```
length(unique(yo$price))
```

```
## [1] 20
```

```
table(yo$price)
```

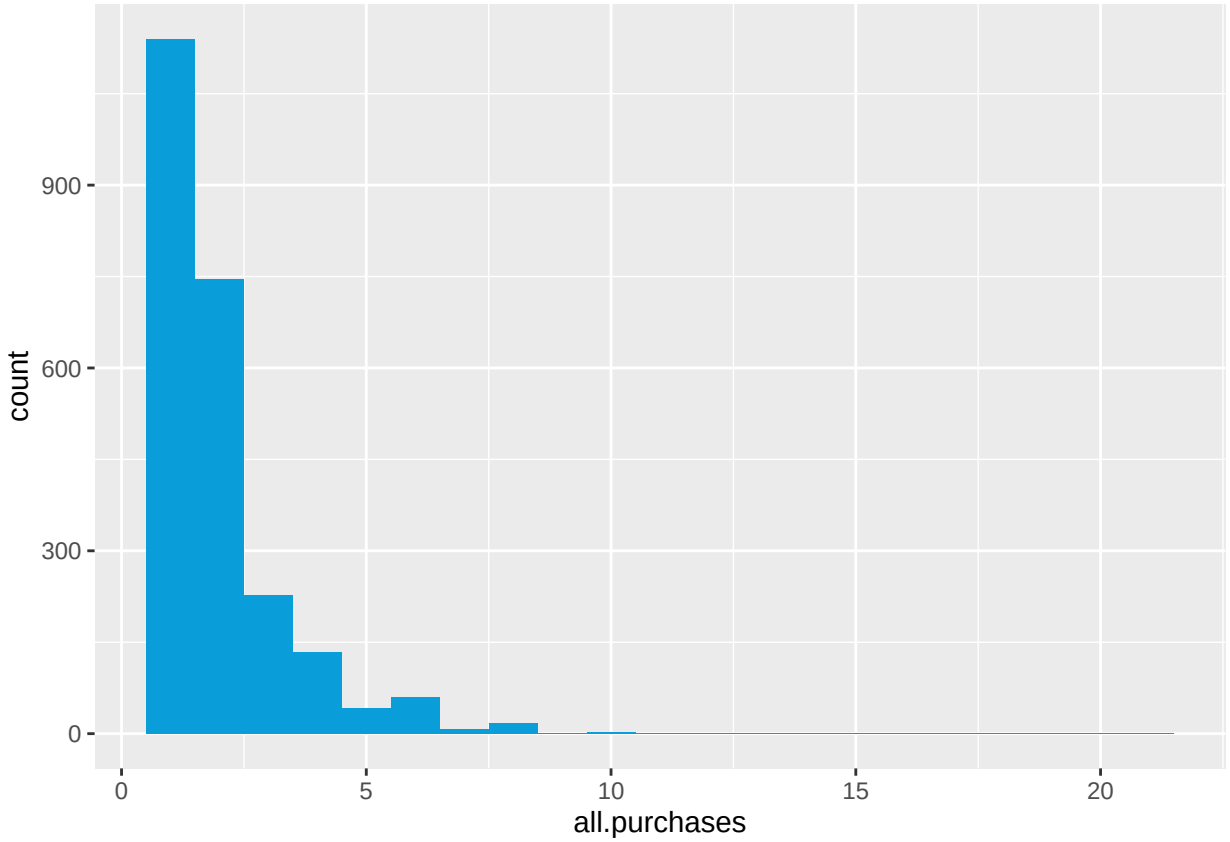
```
##
##    20 24.96 33.04  33.2 33.28 33.36 33.52 39.04    44 45.04 48.96 49.52  49.6
##     2    11    54     1     1    22     1   234    21    11    81     1     1
##    50 55.04 58.96    62 63.04 65.04 68.96
##   205     6   303    15     2   799   609
```

```
# Create a new variable called all.purchases,
# which gives the total counts of yogurt for
# each observation or household.

# One way to do this is using the transform
# function. You can look up the function transform
# and run the examples of code at the bottom of the
# documentation to figure out what it does.

# The transform function produces a data frame
# so if you use it then save the result to 'yo'!

# OR you can figure out another way to create the
# variable.
yo = transform(yo, all.purchases = strawberry + blueberry + pina.colada + plain + mixed.berry)
```

Prices over Time

Notes:

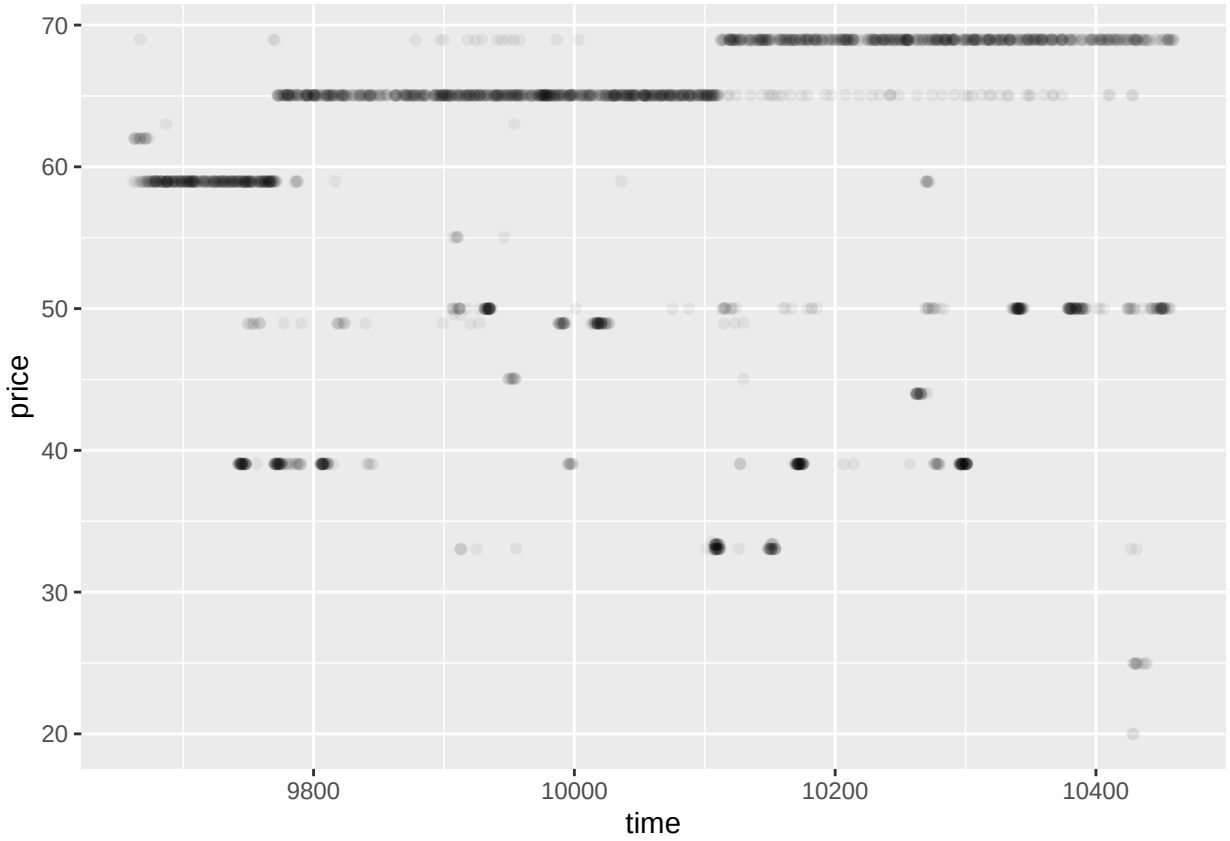
```
ggplot(aes(x = all.purchases), data = yo) +
  geom_histogram(binwidth = 1, fill = '#099DD9')
```




```
ggsave('hist2.png')
```

```
## Saving 6.5 x 4.5 in image
```

```
# Create a scatterplot of price vs time.

# This will be an example of a time series plot.

# Resolve overplotting issues by using
# techniques you learned in Lesson 4.

# What are some things that you notice?
ggplot(aes(x = time, y = price), data = yo) +
  geom_jitter(alpha = 1/20)
```




```
ggsave('scatter1.png')
```

```
## Saving 6.5 x 4.5 in image
```

Sampling Observations

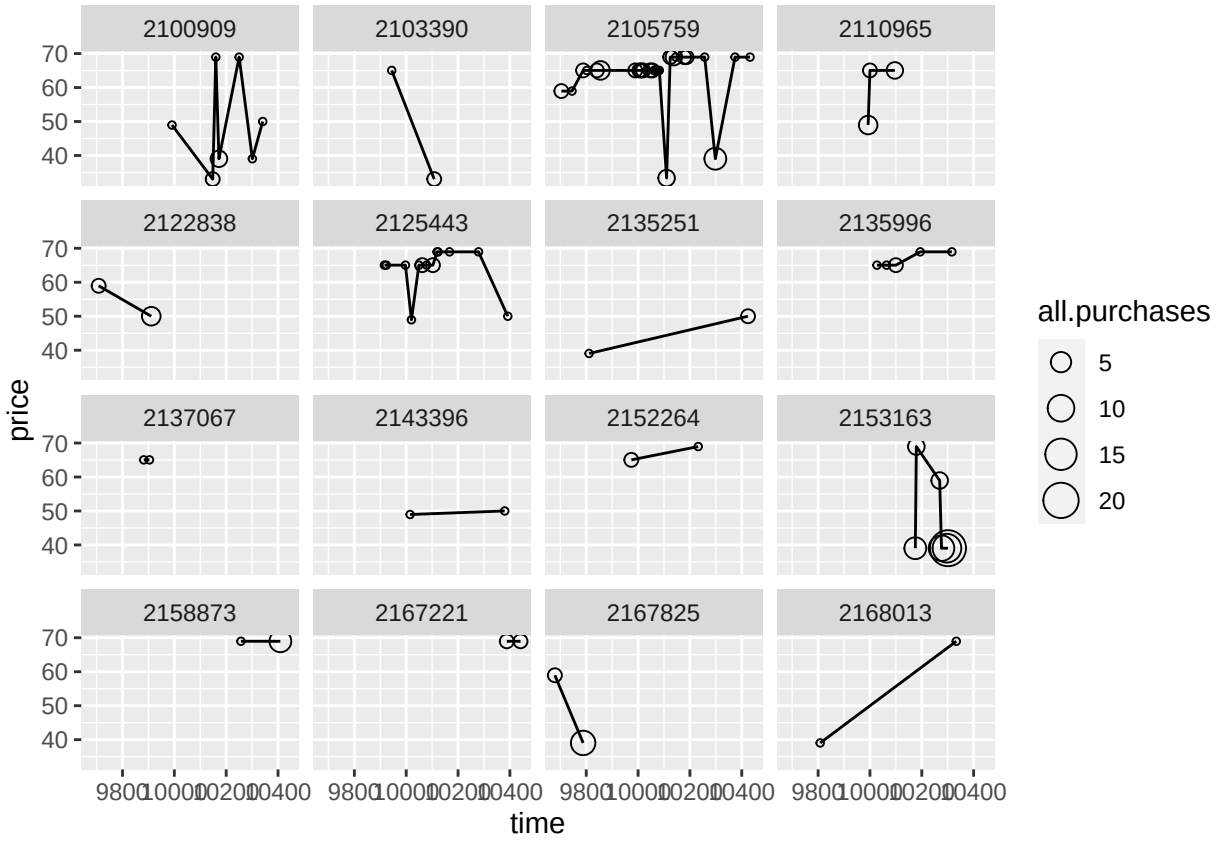
Notes:

Looking at Samples of Households

```
set.seed(6230)
sample.ids = sample(levels(yo$id), 16)
sample.ids
```

```
##  [1] "2110965" "2105759" "2137067" "2100909" "2167825" "2135251" "2125443"
##  [8] "2153163" "2122838" "2167221" "2158873" "2168013" "2103390" "2152264"
## [15] "2143396" "2135996"
```

```
ggplot(aes(x = time, y = price),
       data = subset(yo, id %in% sample.ids)) +
  facet_wrap( ~ id) +
  geom_line() +
  geom_point(aes(size = all.purchases), pch = 1)
```




```
ggsave('seed1.png')
```

```
## Saving 6.5 x 4.5 in image
```

The Limits of Cross Sectional Data

Notes:

Many Variables

Notes:

Scatterplot Matrix

Notes: “‘{r} scatterplot_Matrix} #install.packages(‘GGally’) library(GGally) theme_set(theme_minimal(20)) set.seed(1836) pf_subset = pf[, c(2:15)] ggpairs(pf_subset[sample.int(nrow(pf_subset), 1000),]) ggsave(‘scaMatrix1.png’)
“‘

Even More Variables

Notes:

```
nci = read.table('nci.tsv')
colnames(nci) = c(1:64)
```

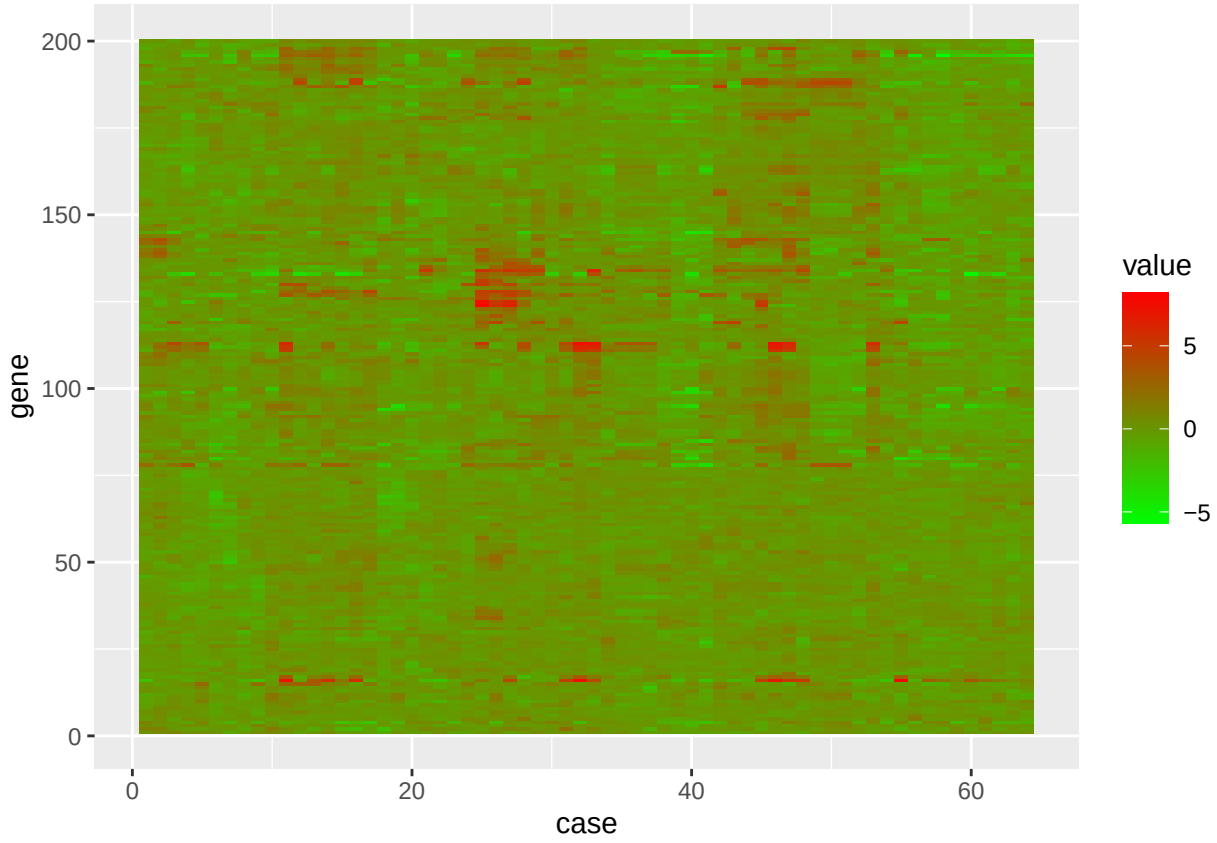
Heat Maps

Notes:

```
library(reshape2)
nci.long.samp = melt(as.matrix(nci[1:200,]))
names(nci.long.samp) = c("gene", "case", "value")
head(nci.long.samp)
```

```
##   gene case  value
## 1    1    1  0.300
## 2    2    1  1.180
## 3    3    1  0.550
## 4    4    1  1.140
## 5    5    1 -0.265
## 6    6    1 -0.070
```

```
ggplot(aes(y = gene, x = case, fill = value),
  data = nci.long.samp) +
  geom_tile() +
  scale_fill_gradientn(colours = colorRampPalette(c("green", "red"))(100))
```



Analyzing Three of More Variables

Reflection:

Click **KnitHTML** to see all of your hard work and to have an html page of this lesson, your answers, and your notes!