Language Translations: An Introduction to R

Shukry Zablah (szablah20@amherst.edu)

06 July, 2018

There are many tools that accomplish the same jobs, but each tool has its advantages and disadvantages. This is the first part of the Language Translations tutorial series that aims to introduce the learner to R. R is a powerful and free software environment for statistics and data science that is interpreted on the fly, providing powerful computational power with relatively simple syntax. R is free available for download at https://www.r-project.org/ .

Overview and Motivation

In this tutorial we will guide the learner through a case study in which we take a friendly dataset from read in to analysis. The learner will be able to familiarize with the R workflow and be able to have a glimpse of its capabilities. In the end, the learner will extract insights from the dataset and visualize them.

Dataset

If need be, the learner can find more information on the dataset in our codebook or at https://www.kaggle.com/starbucks/starbucks-menu/home .

Import libraries

R has thousands of free user created packages that expand on the core functionality of the language. These packages often provide specialized functionality and other aids that ease the task of analyzing data.

Run the following command if you don't have these packages installed on your computer:

```
#install.packages(c("tidyverse", "mosaic", "xtable"))
```

Afterwards, we will load the packages into the environment in order to get access to their functions.

```
library(tidyverse)
library(mosaic)
library(xtable)
```

Now that we are all set, let's start the process of data analysis by ingesting a friendly dataset on Starbucks items and their nutritional information.

Read in data

The file is easily read into an R dataframe (a data structure that holds tabular information) with one command. We will call our dataset "Starbucks".

```
Starbucks <- read_csv("../resources/Starbucks.csv")
```

Global characteristics and overview

Now that we have our data in R we can take a look at it.

```
glimpse(Starbucks)
```

```
## Observations: 242
## Variables: 19
## $ X1                  <int> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13,...
## $ beverageCategory    <chr> "Coffee", "Coffee", "Coffee", "Coffee", "C...
## $ beverage            <chr> "Brewed Coffee", "Brewed Coffee", "Brewed ...
## $ beveragePrep        <chr> "Short", "Tall", "Grande", "Venti", "Short...
## $ calories            <int> 3, 4, 5, 5, 70, 100, 70, 100, 150, 110, 13...
## $ totalFatG           <chr> "0.1", "0.1", "0.1", "0.1", "0.1", "3.5", ...
## $ transFatG           <dbl> 0.0, 0.0, 0.0, 0.0, 0.1, 2.0, 0.4, 0.2, 3....
## $ saturatedFatG       <dbl> 0.0, 0.0, 0.0, 0.0, 0.0, 0.1, 0.0, 0.0, 0....
## $ sodiumMg            <int> 0, 0, 0, 0, 5, 15, 0, 5, 25, 0, 5, 30, 0, ...
## $ totalCarbohydratesG <int> 5, 10, 10, 10, 75, 85, 65, 120, 135, 105, ...
## $ cholesterolMg       <int> 0, 0, 0, 0, 10, 10, 6, 15, 15, 10, 19, 19,...
## $ dietaryFibreG       <int> 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 1, 0, ...
## $ sugarsG             <int> 0, 0, 0, 0, 9, 9, 4, 14, 14, 6, 18, 17, 8,...
## $ proteinG            <dbl> 0.3, 0.5, 1.0, 1.0, 6.0, 6.0, 5.0, 10.0, 1...
## $ vitaminAPercentDv   <chr> "0%", "0%", "0%", "0%", "10%", "10%", "6%"...
## $ vitaminCPercentDv   <chr> "0%", "0%", "0%", "0%", "0%", "0%", "0%", ...
## $ calciumPercentDv    <chr> "0%", "0%", "0%", "2%", "20%", "20%", "20%...
## $ ironPercentDv       <chr> "0%", "0%", "0%", "0%", "0%", "0%", "8%", ...
## $ caffeineMg          <chr> "175", "260", "330", "410", "75", "75", "7...
```

We quickly got a glimpse of what we are dealing with here. Our dataset has 242 observations and 19 variables.

Another thing we can do is get a vector of the variable names.

```
names(Starbucks)
```

```
##  [1] "X1"                  "beverageCategory"    "beverage"
##  [4] "beveragePrep"        "calories"            "totalFatG"
##  [7] "transFatG"           "saturatedFatG"       "sodiumMg"
## [10] "totalCarbohydratesG" "cholesterolMg"       "dietaryFibreG"
## [13] "sugarsG"             "proteinG"            "vitaminAPercentDv"
## [16] "vitaminCPercentDv"   "calciumPercentDv"    "ironPercentDv"
## [19] "caffeineMg"
```

We will try to extract insights about the proteinG (how many grams of protein) the items have, but first we have to uncover the distributions of some of the variables in the dataset to familiarize ourselves with the dataset and get exposed to some useful functions in R.

Univariate Analysis

The main purpose of univariate analysis is to begin to describe the dataset without worrying about relationships or causes. This step is important because it will help guide our questions and the way we will answer them later on.

Now, let's take a look at some variables:

```
mosaic::tally(~ beverageCategory, data = Starbucks)
```

```
## beverageCategory
```

```
##           Classic Espresso Drinks                                Coffee
##                                58                                     4
##       Frappuccino® Blended Coffee        Frappuccino® Blended Crème
##                                36                                    13
## Frappuccino® Light Blended Coffee             Shaken Iced Beverages
##                                12                                    18
##         Signature Espresso Drinks                            Smoothies
##                                40                                     9
##                 Tazo® Tea Drinks
##                                52
```

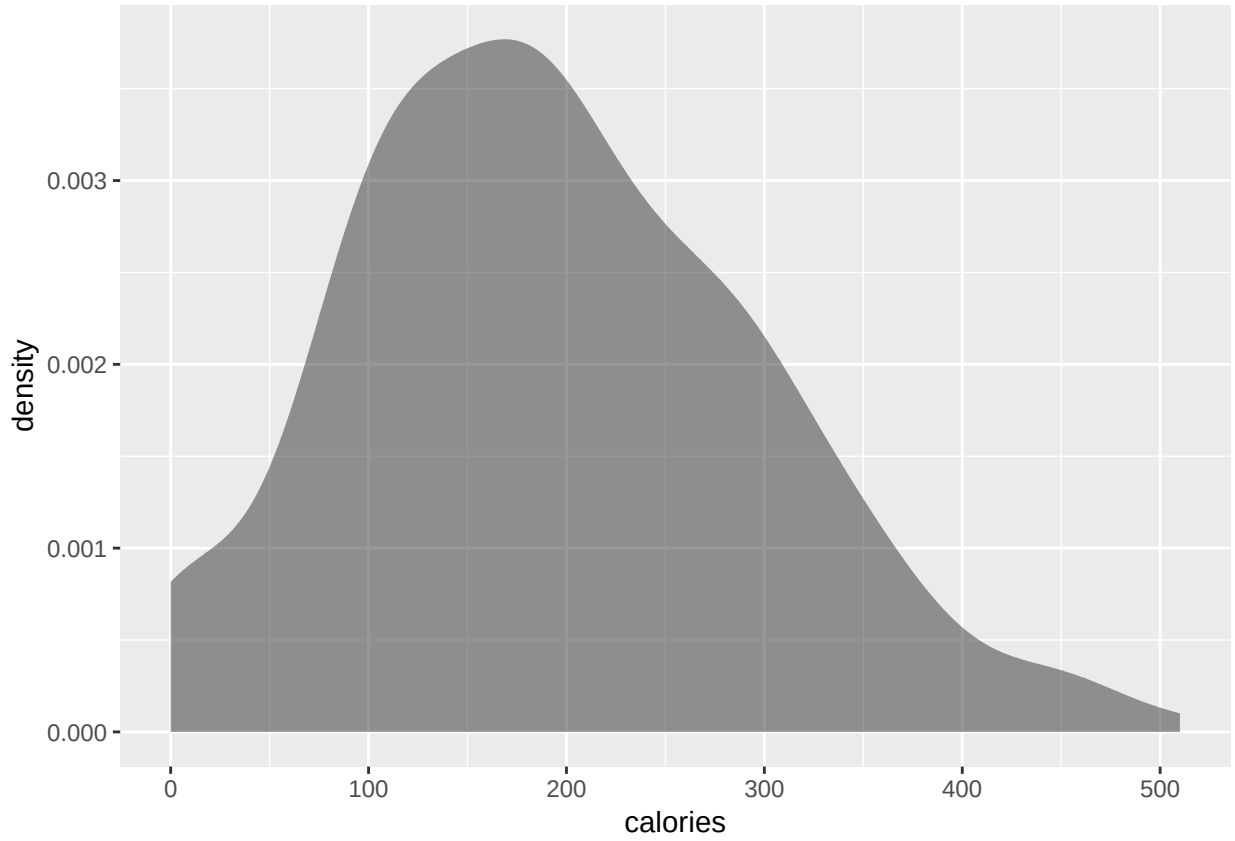
```
favstats(~ calories, data = Starbucks)
```

```
##  min  Q1 median  Q3 max     mean       sd   n missing
##    0 120    185 260 510 193.8719 102.8633 242       0
```

```
with(Starbucks, stem(calories))
```

```
##
##   The decimal point is 2 digit(s) to the right of the |
##
##   0 | 00000011111123
##   0 | 55666677788888888999999
##   1 | 000000000011111111122222222223333333333344444
##   1 | 555555555556666666677777777788888888888999999999999
##   2 | 00000000001111111222222233333444444444
##   2 | 55556666666677778888888999999999
##   3 | 0011111111222334444
##   3 | 555556777899
##   4 | 023
##   4 | 5566
##   5 | 1
```

```
gf_density(~ calories, data = Starbucks)
```




```
summary(Starbucks)
```

```
##        X1         beverageCategory     beverage         beveragePrep
##  Min.   :  1.00   Length:242         Length:242         Length:242
##  1st Qu.: 61.25   Class :character   Class :character   Class :character
##  Median :121.50   Mode  :character   Mode  :character   Mode  :character
##  Mean   :121.50
##  3rd Qu.:181.75
##  Max.   :242.00
##     calories      totalFatG           transFatG     saturatedFatG
##  Min.   :  0.0   Length:242         Min.   :0.000   Min.   :0.0000
##  1st Qu.:120.0   Class :character   1st Qu.:0.100   1st Qu.:0.0000
##  Median :185.0   Mode  :character   Median :0.500   Median :0.0000
##  Mean   :193.9                      Mean   :1.307   Mean   :0.0376
##  3rd Qu.:260.0                      3rd Qu.:2.000   3rd Qu.:0.1000
##  Max.   :510.0                      Max.   :9.000   Max.   :0.3000
##     sodiumMg      totalCarbohydratesG cholesterolMg   dietaryFibreG
##  Min.   : 0.000   Min.   :  0.0       Min.   : 0.00   Min.   :0.0000
##  1st Qu.: 0.000   1st Qu.: 70.0       1st Qu.:21.00   1st Qu.:0.0000
##  Median : 5.000   Median :125.0       Median :34.00   Median :0.0000
##  Mean   : 6.364   Mean   :128.9       Mean   :35.99   Mean   :0.8058
##  3rd Qu.:10.000   3rd Qu.:170.0       3rd Qu.:50.75   3rd Qu.:1.0000
##  Max.   :40.000   Max.   :340.0       Max.   :90.00   Max.   :8.0000
##     sugarsG         proteinG      vitaminAPercentDv  vitaminCPercentDv
##  Min.   : 0.00   Min.   : 0.000   Length:242         Length:242
##  1st Qu.:18.00   1st Qu.: 3.000   Class :character   Class :character
```

```
##  Median :32.00   Median : 6.000   Mode  :character   Mode  :character
##  Mean   :32.96   Mean   : 6.979
##  3rd Qu.:43.75   3rd Qu.:10.000
##  Max.   :84.00   Max.   :20.000
##  calciumPercentDv   ironPercentDv       caffeineMg
##  Length:242         Length:242         Length:242
##  Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character
##
##
##
```

Data wrangling

```
Starbucks <- Starbucks %>%
  mutate(beverageCategory = factor(beverageCategory),
         beverage = factor(beverage),
         beveragePrep = factor(beveragePrep),
         totalFatG = parse_number(totalFatG),
         vitaminAPercentDv = parse_number(vitaminAPercentDv),
         vitaminCPercentDv = parse_number(vitaminCPercentDv),
         calciumPercentDv = parse_number(calciumPercentDv),
         ironPercentDv = parse_number(ironPercentDv),
         caffeineMg = parse_number(caffeineMg)
         ) %>%
  select(-X1)
```

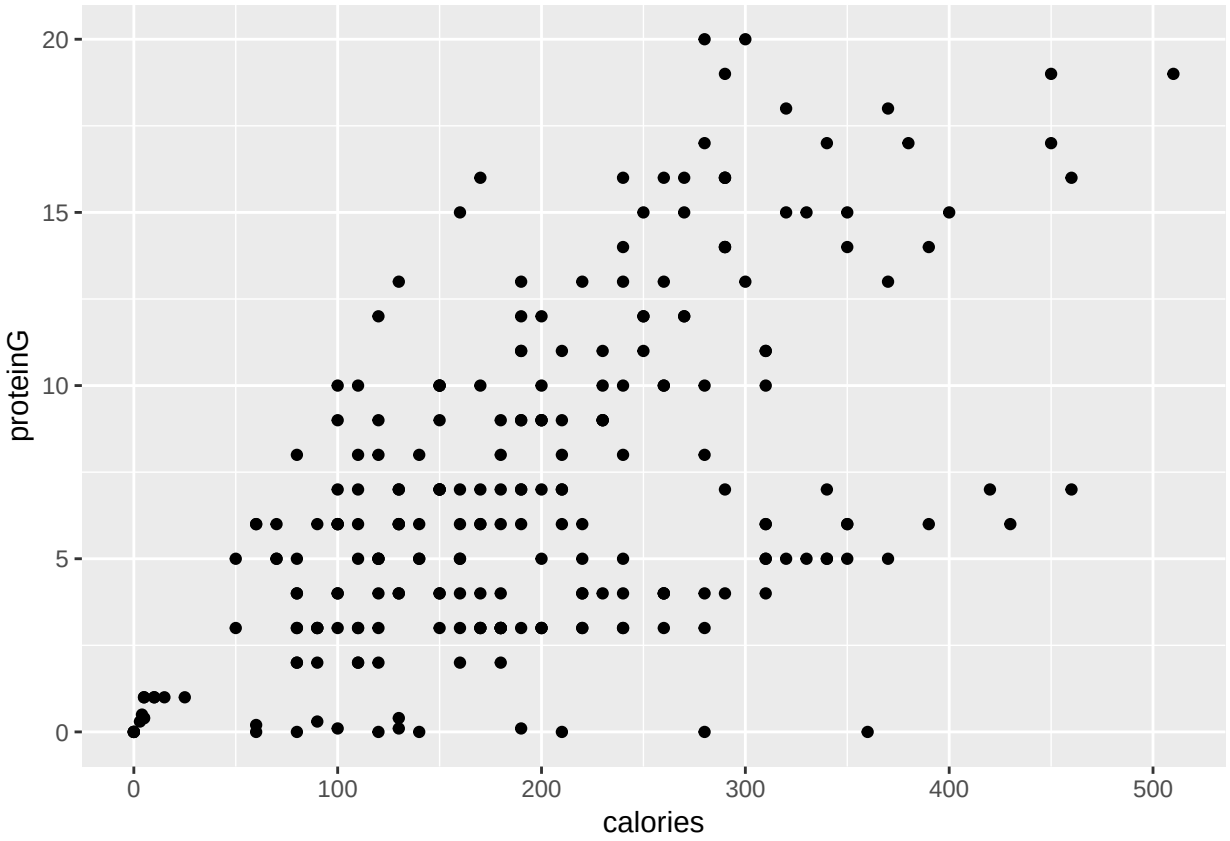
```
summary(Starbucks)
```

```
##                     beverageCategory
##  Classic Espresso Drinks    :58
##  Tazo® Tea Drinks           :52
##  Signature Espresso Drinks  :40
##  Frappuccino® Blended Coffee:36
##  Shaken Iced Beverages      :18
##  Frappuccino® Blended Crème :13
##  (Other)                    :25
##                                    beverage               beveragePrep
##  Caffè Latte                           : 12   Soymilk           :66
##  Caffè Mocha (Without Whipped Cream)   : 12   2% Milk           :50
##  Cappuccino                            : 12   Grande Nonfat Milk:26
##  Caramel Macchiato                     : 12   Tall Nonfat Milk  :23
##  Coffee                                : 12   Venti Nonfat Milk :22
##  Hot Chocolate (Without Whipped Cream): 12   Whole Milk        :16
##  (Other)                               :170   (Other)           :39
##     calories       totalFatG        transFatG     saturatedFatG
##  Min.   :  0.0   Min.   : 0.000   Min.   :0.000   Min.   :0.0000
##  1st Qu.:120.0   1st Qu.: 0.200   1st Qu.:0.100   1st Qu.:0.0000
##  Median :185.0   Median : 2.500   Median :0.500   Median :0.0000
##  Mean   :193.9   Mean   : 2.904   Mean   :1.307   Mean   :0.0376
##  3rd Qu.:260.0   3rd Qu.: 4.500   3rd Qu.:2.000   3rd Qu.:0.1000
##  Max.   :510.0   Max.   :15.000   Max.   :9.000   Max.   :0.3000
##
```

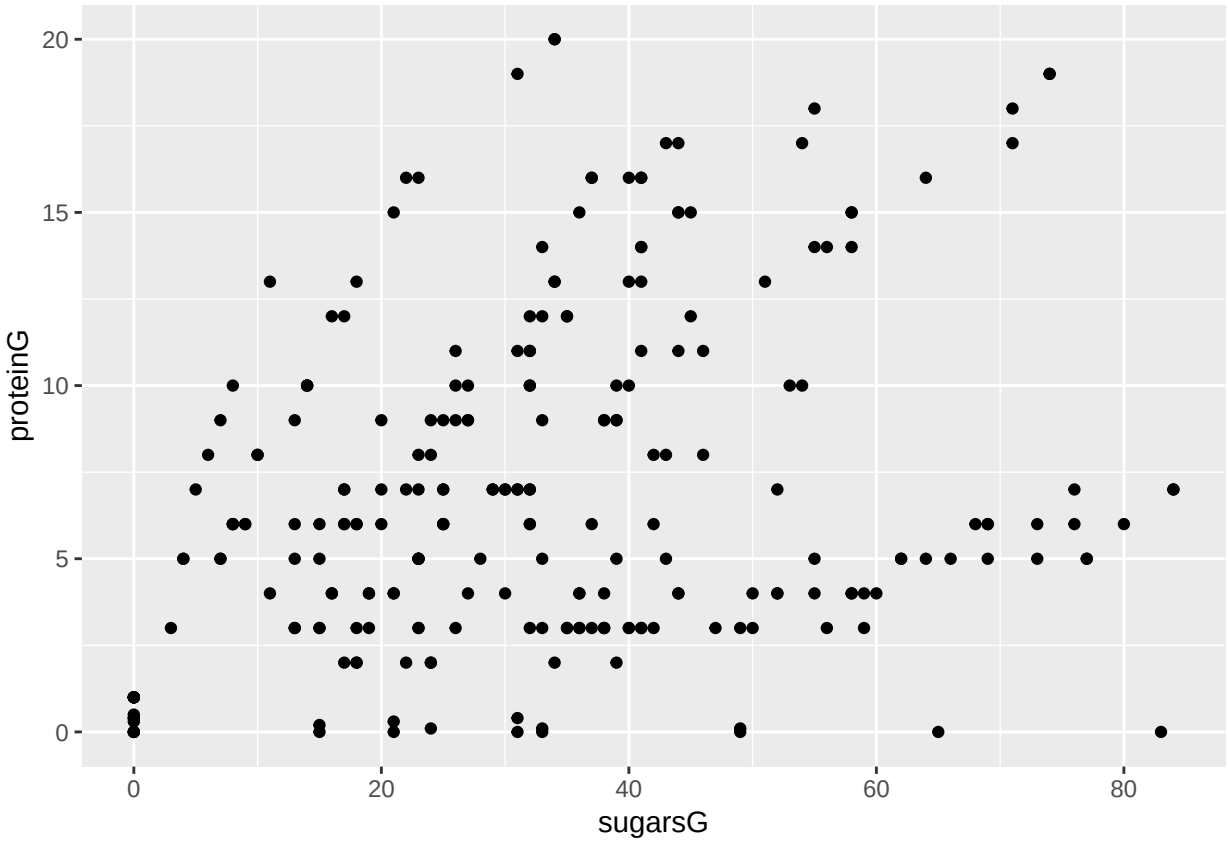
```
##     sodiumMg      totalCarbohydratesG cholesterolMg   dietaryFibreG
##  Min.   : 0.000   Min.   :  0.0       Min.   : 0.00   Min.   :0.0000
##  1st Qu.: 0.000   1st Qu.: 70.0       1st Qu.:21.00   1st Qu.:0.0000
##  Median : 5.000   Median :125.0       Median :34.00   Median :0.0000
##  Mean   : 6.364   Mean   :128.9       Mean   :35.99   Mean   :0.8058
##  3rd Qu.:10.000   3rd Qu.:170.0       3rd Qu.:50.75   3rd Qu.:1.0000
##  Max.   :40.000   Max.   :340.0       Max.   :90.00   Max.   :8.0000
##
##     sugarsG         proteinG      vitaminAPercentDv vitaminCPercentDv
##  Min.   : 0.00   Min.   : 0.000   Min.   : 0.000    Min.   :  0.000
##  1st Qu.:18.00   1st Qu.: 3.000   1st Qu.: 4.000    1st Qu.:  0.000
##  Median :32.00   Median : 6.000   Median : 8.000    Median :  0.000
##  Mean   :32.96   Mean   : 6.979   Mean   : 9.831    Mean   :  3.649
##  3rd Qu.:43.75   3rd Qu.:10.000   3rd Qu.:15.000    3rd Qu.:  0.000
##  Max.   :84.00   Max.   :20.000   Max.   :50.000    Max.   :100.000
##
##  calciumPercentDv ironPercentDv      caffeineMg
##  Min.   : 0.00    Min.   : 0.000   Min.   :  0.00
##  1st Qu.:10.00    1st Qu.: 0.000   1st Qu.: 50.00
##  Median :20.00    Median : 2.000   Median : 75.00
##  Mean   :20.76    Mean   : 7.446   Mean   : 89.52
##  3rd Qu.:30.00    3rd Qu.:10.000   3rd Qu.:142.50
##  Max.   :60.00    Max.   :50.000   Max.   :410.00
##                                    NA's   :23
```

Bivariate Analysis

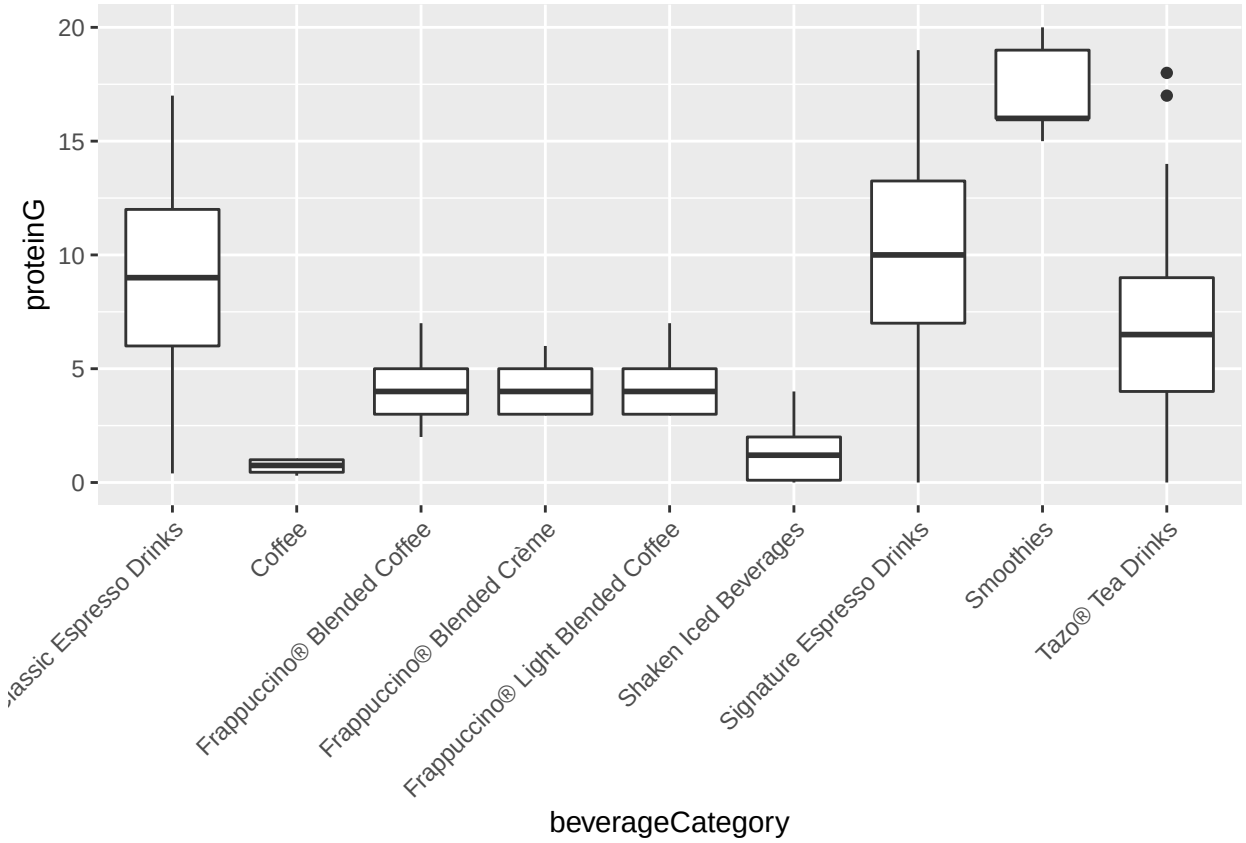
```
gf_point(proteinG ~ calories, data = Starbucks)
```




```
gf_point(proteinG ~ sugarsG, data = Starbucks)
```




```
gf_boxplot(proteinG ~ beverageCategory, data = Starbucks) + theme(axis.text.x=element_text(angle=45,hju
```

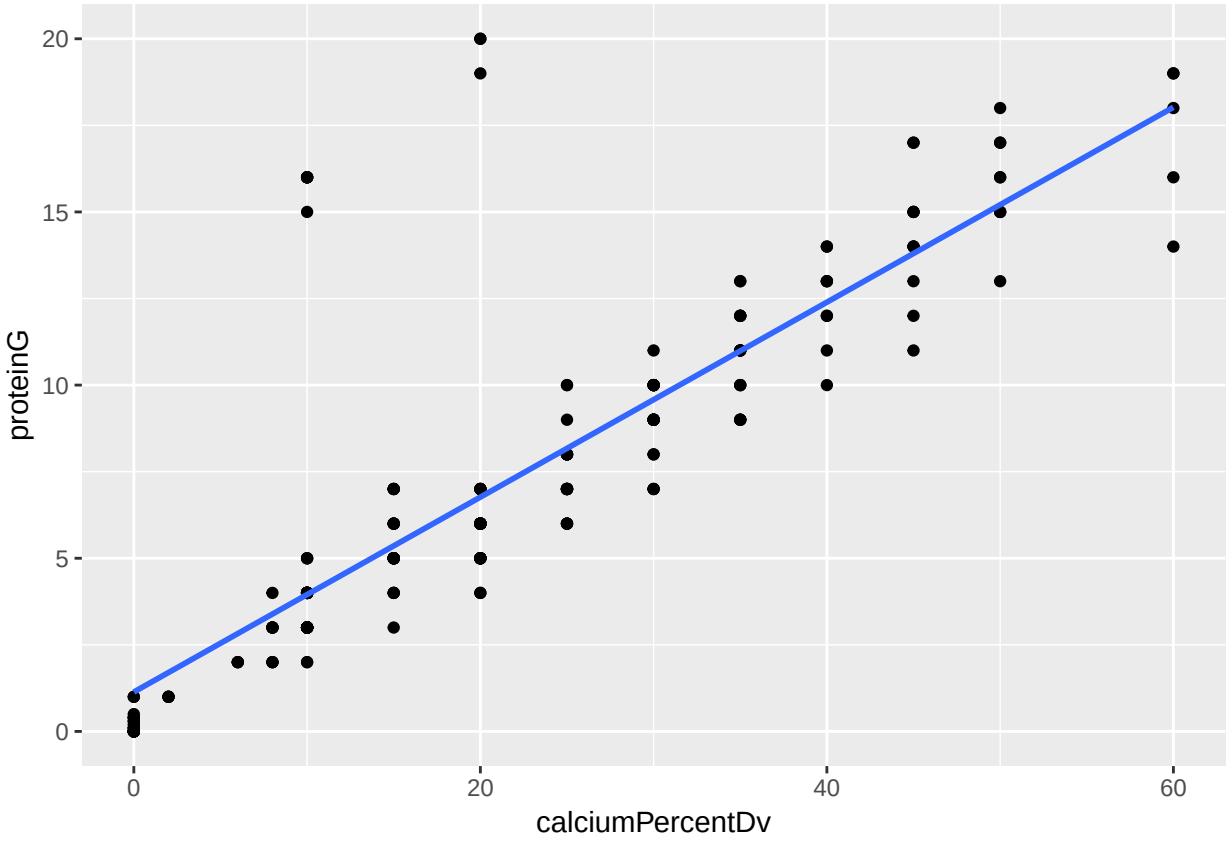


Linear regression

```
modProtein <- lm(proteinG ~ calciumPercentDv, data = Starbucks)
summary(modProtein)
```

```
##
## Call:
## lm(formula = proteinG ~ calciumPercentDv, data = Starbucks)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -4.0209 -1.0382 -0.5795  0.2256 13.2343
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)       1.13816    0.29720    3.83 0.000164 ***
## calciumPercentDv  0.28138    0.01173   23.98  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.649 on 240 degrees of freedom
## Multiple R-squared:  0.7055, Adjusted R-squared:  0.7043
## F-statistic: 574.9 on 1 and 240 DF,  p-value: < 2.2e-16
```

```r
gf_point(proteinG ~ calciumPercentDv, data = Starbucks) %>% gf_smooth(method = "lm")
```




```r
modProtein2 <- lm(proteinG ~ calciumPercentDv + transFatG, data = Starbucks)
summary(modProtein2)
```

```
##
## Call:
## lm(formula = proteinG ~ calciumPercentDv + transFatG, data = Starbucks)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.1980 -1.0846 -0.6342  0.2296 13.1562
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)       1.08458    0.29357   3.694 0.000273 ***
## calciumPercentDv  0.26301    0.01327  19.827  < 2e-16 ***
## transFatG         0.33267    0.11761   2.829 0.005071 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.611 on 239 degrees of freedom
## Multiple R-squared:  0.715,  Adjusted R-squared:  0.7127
## F-statistic: 299.9 on 2 and 239 DF,  p-value: < 2.2e-16
```

```
gf_point(proteinG ~ sugarsG, color = ~ beverageCategory, data = Starbucks) %>% gf_lm()
```