Week 1 - Homework

STAT 420, Summer 2019, D. Unger

Exercise 1 (Subsetting and Statistics)

For this exercise, we will use the `msleep` dataset from the `ggplot2` package.

(a) Install and load the `ggplot2` package. **Do not** include the installation command in your `.Rmd` file. (If you do it will install the package every time you knit your file.) **Do** include the command to load the package into your environment.

Solution:

```
# Include the first line if you want to suppress the messages that result from executing the library()
suppressMessages(library(ggplot2))
library(ggplot2)
```

(b) Note that this dataset is technically a `tibble`, not a data frame. How many observations are in this dataset? How many variables? What are the observations in this dataset?

Solution:

```
msleep
```

```
## # A tibble: 83 x 11
##    name  genus vore  order conservation sleep_total sleep_rem sleep_cycle
##    <chr> <chr> <chr> <chr> <chr>              <dbl>     <dbl>       <dbl>
##  1 Chee~ Acin~ carni Carn~ lc                  12.1      NA        NA
##  2 Owl ~ Aotus omni  Prim~ <NA>                17         1.8      NA
##  3 Moun~ Aplo~ herbi Rode~ nt                  14.4       2.4      NA
##  4 Grea~ Blar~ omni  Sori~ lc                  14.9       2.3       0.133
##  5 Cow   Bos   herbi Arti~ domesticated         4         0.7       0.667
##  6 Thre~ Brad~ herbi Pilo~ <NA>                14.4       2.2       0.767
##  7 Nort~ Call~ carni Carn~ vu                   8.7       1.4       0.383
##  8 Vesp~ Calo~ <NA>  Rode~ <NA>                 7        NA        NA
##  9 Dog   Canis carni Carn~ domesticated        10.1       2.9       0.333
## 10 Roe ~ Capr~ herbi Arti~ lc                   3        NA        NA
## # ... with 73 more rows, and 3 more variables: awake <dbl>, brainwt <dbl>,
## #   bodywt <dbl>
```

```
?msleep
```

We find there are 83 observations and 11 variables that describe **mammals**.

(c) What is the mean hours of REM sleep of individuals in this dataset?

Solution:

```r
any(is.na(msleep$sleep_rem))
```

```
## [1] TRUE
```

```r
anyNA(msleep$sleep_rem)
```

```
## [1] TRUE
```

```r
mean(msleep$sleep_rem, na.rm = TRUE)
```

```
## [1] 1.87541
```

Notice that we need to deal with some missing data. We only remove observations with missing data from the variable of interest. Had we instead removed any observation with missing data, we would have less data to calculate this statistic.

(d) What is the standard deviation of brain weight of individuals in this dataset?

Solution:

```r
sd(msleep$brainwt, na.rm = TRUE)
```

```
## [1] 0.9764137
```

We again deal with missing data using a similar strategy to the mean calculation.

(e) Which observation (provide the `name`) in this dataset gets the most REM sleep?

Solution:

```r
msleep$name[which.max(msleep$sleep_rem)]
```

```
## [1] "Thick-tailed opposum"
```

(f) What is the average bodyweight of carnivores in this dataset?

Solution:

```r
mean(subset(msleep, vore == "carni")$bodywt, na.rm = TRUE)
```

```
## [1] 90.75111
```

Exercise 2 (Plotting)

For this exercise, we will use the `birthwt` dataset from the `MASS` package.

(a) Note that this dataset is a data frame and all of the variables are numeric. How many observations are in this dataset? How many variables? What are the observations in this dataset?

```
library(MASS)
library(tibble)
as_tibble(birthwt)
```

```
## # A tibble: 189 x 10
##      low   age   lwt  race smoke   ptl    ht    ui   ftv   bwt
##    <int> <int> <int> <int> <int> <int> <int> <int> <int> <int>
##  1     0    19   182     2     0     0     0     1     0  2523
##  2     0    33   155     3     0     0     0     0     3  2551
##  3     0    20   105     1     1     0     0     0     1  2557
##  4     0    21   108     1     1     0     0     1     2  2594
##  5     0    18   107     1     1     0     0     1     0  2600
##  6     0    21   124     3     0     0     0     0     0  2622
##  7     0    22   118     1     0     0     0     0     1  2637
##  8     0    17   103     3     0     0     0     0     1  2637
##  9     0    29   123     1     1     0     0     0     1  2663
## 10     0    26   113     1     1     0     0     0     0  2665
## # ... with 179 more rows
```

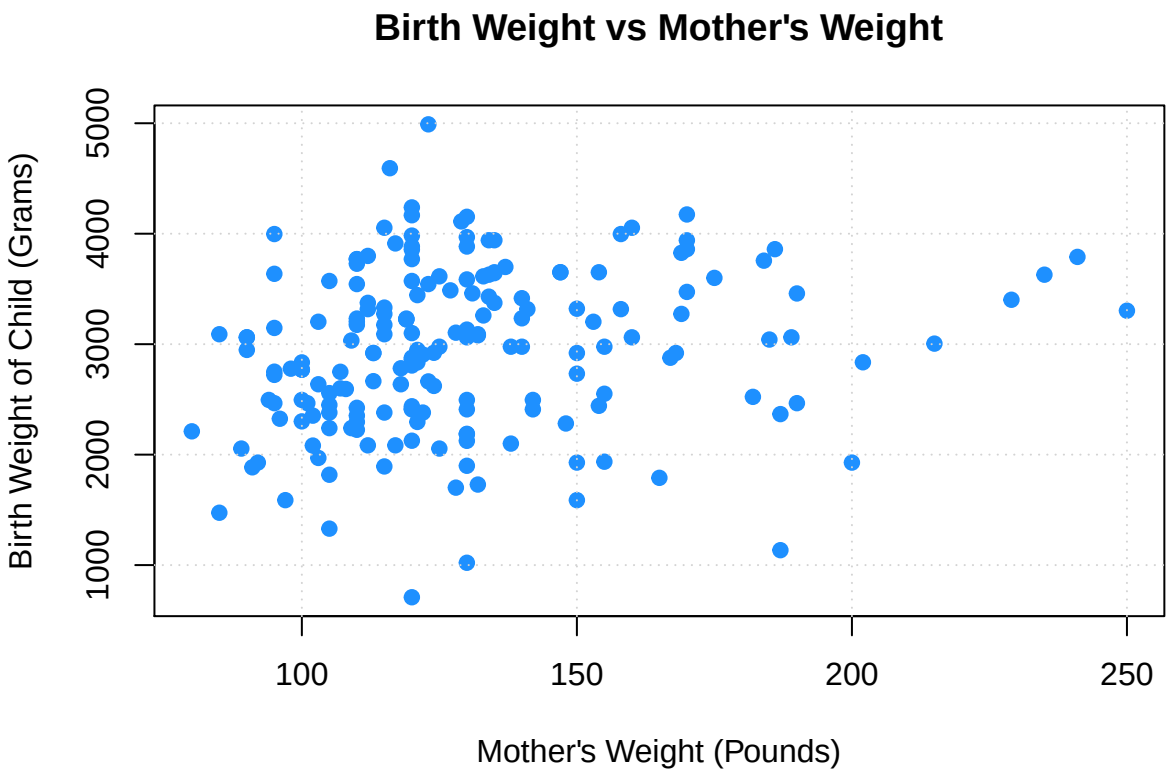
```
?birthwt
```

We find there are 189 observations and 10 variables that describe mothers and their newborn infants at Baystate Medical Center, Springfield, Mass in 1986.

(b) Create a scatter plot of birth weight (y-axis) vs mother's weight before pregnancy (x-axis). Use a non-default color for the points. (Also, be sure to give the plot a title and label the axes appropriately.) Based on the scatter plot, does there seem to be a relationship between the two variables? Briefly explain.

Solution:

```
plot(bwt ~ lwt, data = birthwt,
     xlab = "Mother's Weight (Pounds)",
     ylab = "Birth Weight of Child (Grams)",
     main = "Birth Weight vs Mother's Weight",
     pch  = 20,
     cex  = 1.5,
     col  = "dodgerblue")
grid()
```

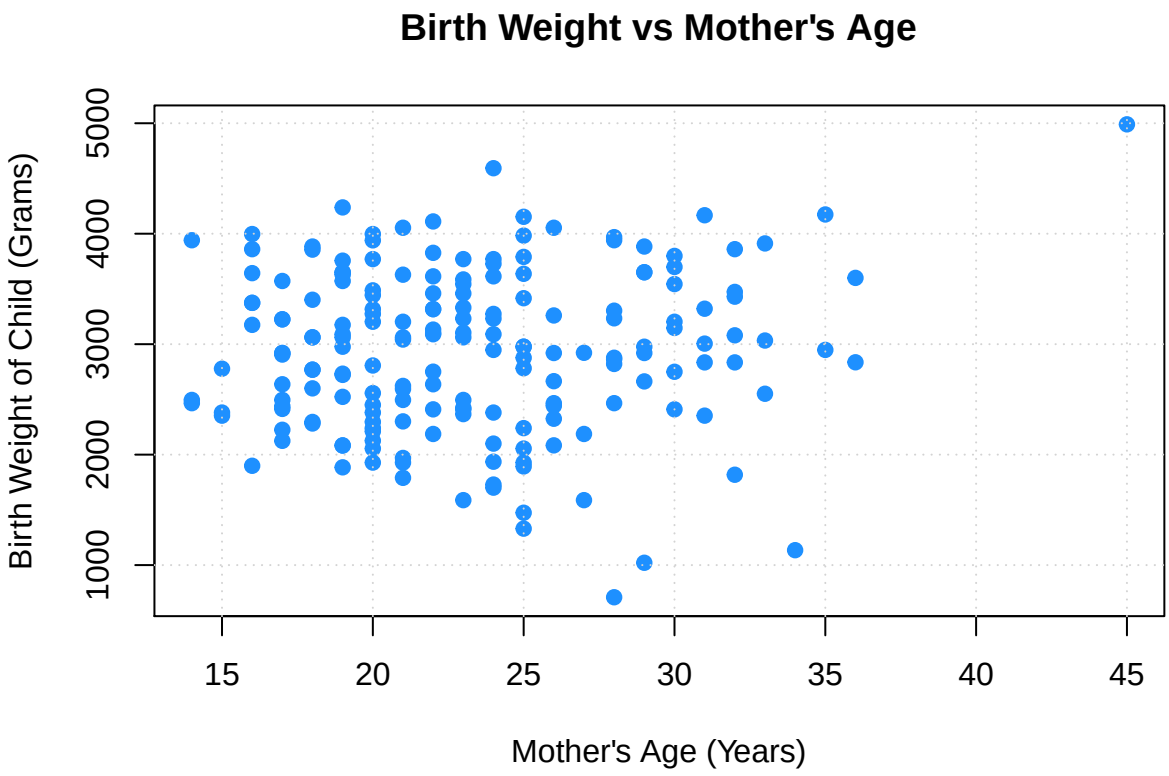


We see very little pattern in the data. Perhaps heavier mothers give birth to slightly heavier babies, but there is also less data on heavier mothers.

(c) Create a scatter plot of birth weight (y-axis) vs mother's age (x-axis). Use a non-default color for the points. (Also, be sure to give the plot a title and label the axes appropriately.) Based on the scatter plot, does there seem to be a relationship between the two variables? Briefly explain.

Solution:

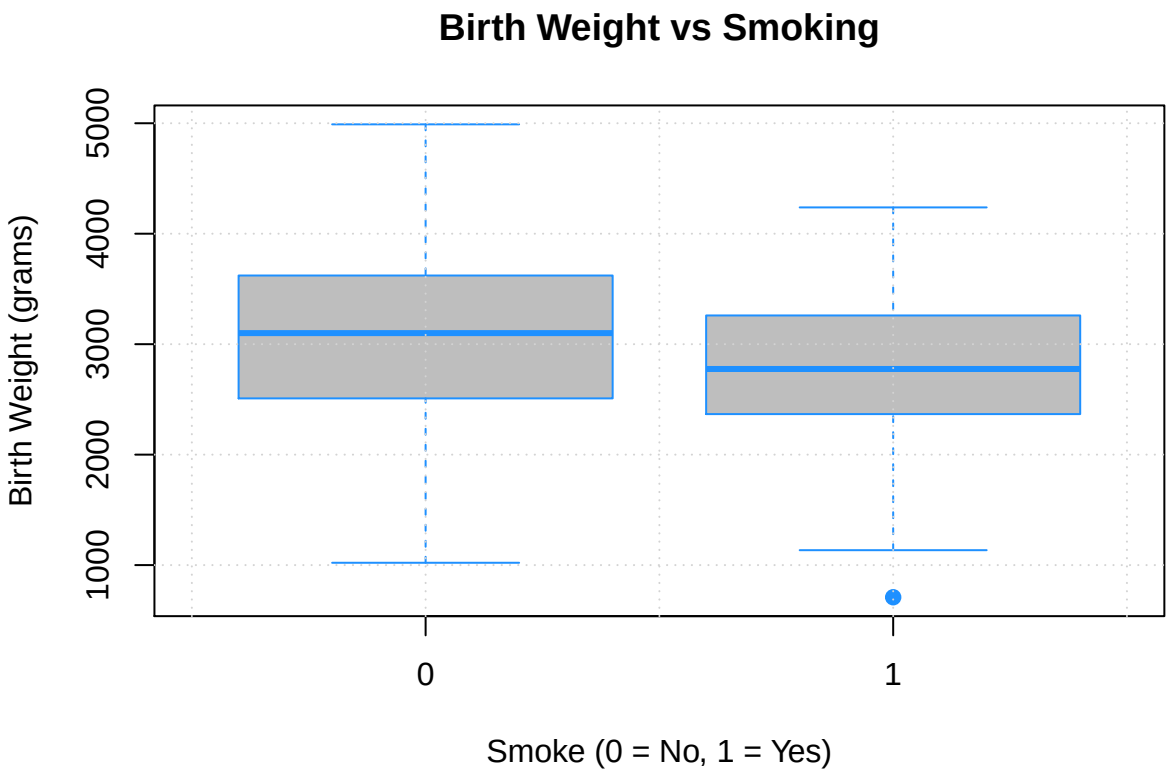
```
plot(bwt ~ age, data = birthwt,
     xlab = "Mother's Age (Years)",
     ylab = "Birth Weight of Child (Grams)",
     main = "Birth Weight vs Mother's Age",
     pch  = 20,
     cex  = 1.5,
     col  = "dodgerblue")
grid()
```



Overall, we see a slight decreasing trend in weight with increasing age. That is, it seems that older mothers can have slightly lighter babies. Also, there is a 5-kilogram baby born to a 45-year-old mother that seems suspicious given the remainder of the data.

(d) Create side-by-side boxplots for birth weight grouped by smoking status. Use non-default colors for the plot. (Also, be sure to give the plot a title and label the axes appropriately.) Based on the boxplot, does there seem to be a difference in birth weight for mothers who smoked? Briefly explain.

```
boxplot(bwt ~ smoke, data = birthwt,
     xlab = "Smoke (0 = No, 1 = Yes)",
     ylab = "Birth Weight (grams)",
     main = "Birth Weight vs Smoking",
     pch  = 20,
     cex  = 1.5,
     col    = "grey",
     border = "dodgerblue")
grid()
```



While babies born to mothers who smoke seem to on average weigh less, there is too much variation within both groups to easily decide if the difference is significant. (It probably is.)

Exercise 3 (Importing Data, More Plotting)

For this exercise we will use the data stored in `nutrition-2018.csv`. It contains the nutritional values per serving size for a large variety of foods as calculated by the USDA in 2018. It is a cleaned version totaling 5956 observations and is current as of April 2018.

The variables in the dataset are:

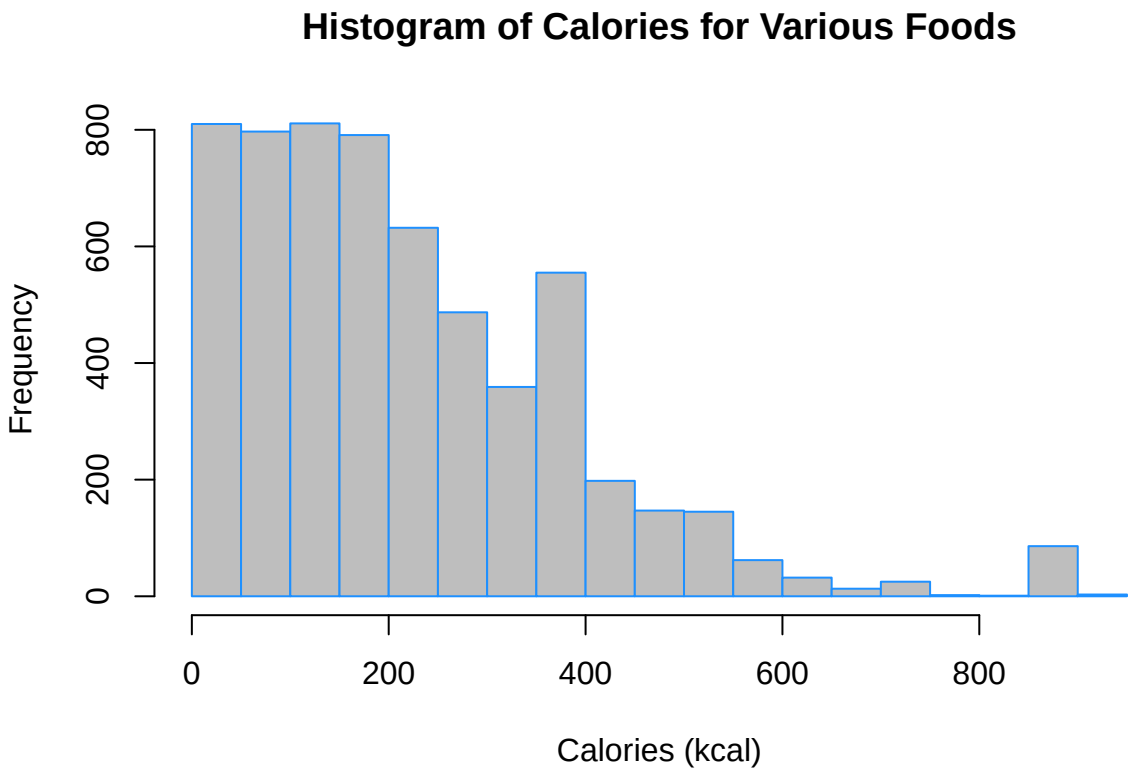
- `ID`
- `Desc` - short description of food
- `Water` - in grams
- `Calories` - in kcal
- `Protein` - in grams
- `Fat` - in grams
- `Carbs` - carbohydrates, in grams
- `Fiber` - in grams
- `Sugar` - in grams
- `Calcium` - in milligrams
- `Potassium` - in milligrams
- `Sodium` - in milligrams

- `VitaminC` - vitamin C, in milligrams
- `Chol` - cholesterol, in milligrams
- `Portion` - description of standard serving size used in analysis

(a) Create a histogram of `Calories`. Do not modify `R`'s default bin selection. Make the plot presentable. Describe the shape of the histogram. Do you notice anything unusual?

Solution:

```
library(readr)
nutrition = read_csv("nutrition-2018.csv")
hist(nutrition$Calories,
     xlab = "Calories (kcal)",
     main = "Histogram of Calories for Various Foods",
     border = "dodgerblue",
     col  = "grey")
```

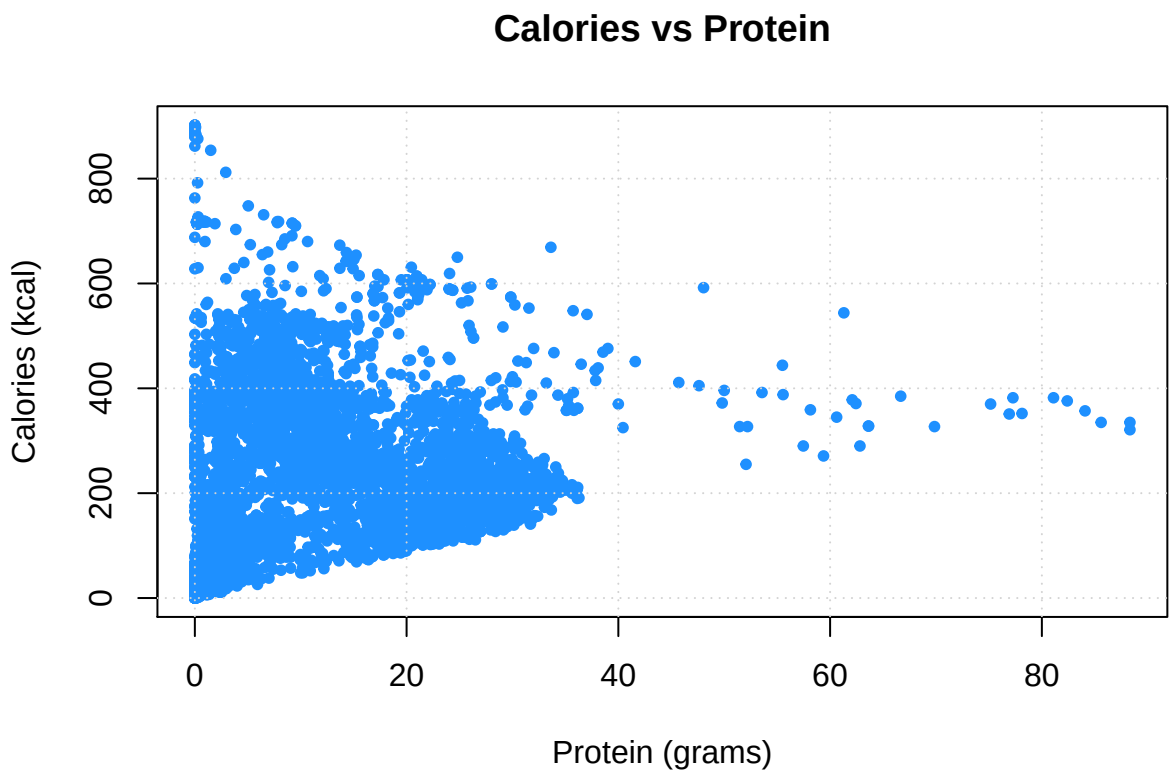


The distribution of `Calories` is right-skewed. There are two odd spikes, one around 400 kcal and one past 800 kcal. Perhaps some foods are being rounded to 400, or portion sizes are created with 400 kcal in mind. Also, perhaps there is an upper limit, and portion sizes are created to keep calories close to 900 but not above.

(b) Create a scatter plot of calories (y-axis) vs protein (x-axis). Make the plot presentable. Do you notice any trends? Do you think that knowing only the protein content of a food, you could make a good prediction of the calories in the food?

Solution:

```r
plot(Calories ~ Protein, data = nutrition,
     xlab = "Protein (grams)",
     ylab = "Calories (kcal)",
     main = "Calories vs Protein",
     pch  = 20,
     cex  = 1,
     col  = "dodgerblue")
grid()
```



While it is not very easy to see, as protein increases, the average calorie count increases. (Eventually we could summarize data like this with a model that makes this easier to see.)

Also, notice that the variability in calorie count decreases as protein increases.

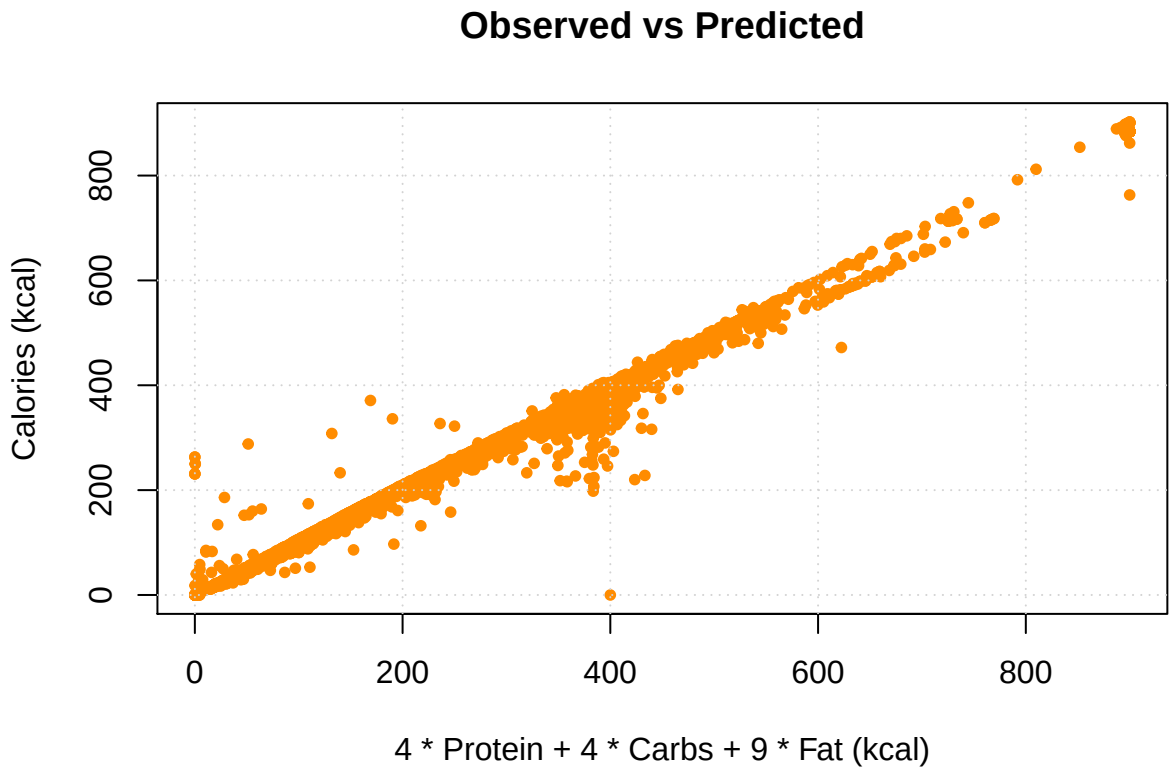
Since there is some trend, knowing the protein could help predict the calorie count, but not particularly well.

(c) Create a scatter plot of `Calories` (y-axis) vs `4 * Protein + 4 * Carbs + 9 * Fat` (x-axis). Make the plot presentable. You will either need to add a new variable to the data frame, or use the `I()` function in your formula in the call to `plot()`. If you are at all familiar with nutrition, you may realize that this formula calculates the calorie count based on the protein, carbohydrate, and fat values. You'd expect then that the result here is a straight line. Is it? If not, can you think of any reasons why it is not?

Solution:

```r
plot(Calories ~ I(4 * Protein + 4 * Carbs + 9 * Fat), data = nutrition,
     xlab = "4 * Protein + 4 * Carbs + 9 * Fat (kcal)",
     ylab = "Calories (kcal)",
```

```
     main = "Observed vs Predicted",
     pch  = 20,
     cex  = 1,
     col  = "darkorange")
grid()
```



The result is *not* a straight line. There could be any number of reasons:

- There are actually additional components that make up food energy that we are not considering. See Wikipedia: Food Energy.
- Rounding
- Measurement error

Exercise 4 (Writing and Using Functions)

For each of the following parts, use the following vectors:

```
a = 1:10
b = 10:1
c = rep(1, times = 10)
d = 2 ^ (1:10)
```

(a) Write a function called `sum_of_squares`.

- Arguments:
  - A vector of numeric data `x`
- Output:
  - The sum of the squares of the elements of the vector $\sum_{i=1}^{n} x_i^2$

Provide your function, as well as the result of running the following code:

```
sum_of_squares(x = a)
sum_of_squares(x = c(c, d))
```

Solution:

```
sum_of_squares = function(x) {
  sum(x ^ 2)
}

sum_of_squares(x = a)
```

```
## [1] 385
```

```
sum_of_squares(x = c(c, d))
```

```
## [1] 1398110
```

(b) Using only your function `sum_of_squares()`, `mean()`, `sqrt()`, and basic math operations such as `+` and `-`, calculate

$$\sqrt{\frac{1}{n}\sum_{i=1}^{n}(x_i - 0)^2}$$

where the x vector is `d`.

Solution:

```
sqrt(sum_of_squares(d - 0) / length(d))
```

```
## [1] 373.9118
```

(c) Using only your function `sum_of_squares()`, `mean()`, `sqrt()`, and basic math operations such as `+` and `-`, calculate

$$\sqrt{\frac{1}{n}\sum_{i=1}^{n}(x_i - y_i)^2}$$

where the x vector is `a` and the y vector is `b`.

Solution:

```
sqrt(sum_of_squares(a - b) / length(a - b))
```

```
## [1] 5.744563
```

Exercise 5 (More Writing and Using Functions)

For each of the following parts, use the following vectors:

```
set.seed(42)
x = 1:100
y = rnorm(1000)
z = runif(150, min = 0, max = 1)
```

(a) Write a function called `list_extreme_values`.

- Arguments:
  - A vector of numeric data `x`
  - A positive constant, `k`, with a default value of `2`
- Output:
  - A list with two elements:
    - `small`, a vector of elements of `x` that are k sample standard deviations less than the sample mean. That is, the observations that are smaller than $\bar{x} - k \cdot s$.
    - `large`, a vector of elements of `x` that are k sample standard deviations greater than the sample mean. That is, the observations that are larger than $\bar{x} + k \cdot s$.

Provide your function, as well as the result of running the following code:

```
list_extreme_values(x = x, k = 1)
list_extreme_values(x = y, k = 3)
list_extreme_values(x = y, k = 2)
list_extreme_values(x = z, k = 1.5)
```

Solution:

```
list_extreme_values = function(x, k = 2) {
  x_bar = mean(x)
  s = sd(x)
  list(
    small = x[x < x_bar - k * s],
    large = x[x > x_bar + k * s]
  )
}

list_extreme_values(x = x, k = 1)
```

```
## $small
##  [1]  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21
##
## $large
##  [1]  80  81  82  83  84  85  86  87  88  89  90  91  92  93  94  95  96
## [18]  97  98  99 100
```

```
list_extreme_values(x = y, k = 3)
```

```
## $small
## [1] -3.371739
##
## $large
## [1] 3.229069 3.211199 3.495304
```

```
list_extreme_values(x = y, k = 2)
```

```
## $small
##  [1] -2.656455 -2.440467 -2.414208 -2.993090 -2.699930 -2.113200 -2.188835
##  [8] -2.071388 -2.138368 -2.461335 -2.170247 -3.017933 -2.192786 -2.253132
## [15] -2.277778 -2.292971 -2.206485 -2.553825 -2.082814 -2.958780 -2.136025
## [22] -2.183149 -3.371739
##
## $large
##  [1] 2.018424 2.286645 2.701891 2.059539 2.036972 2.049961 2.459594
##  [8] 2.212055 2.422163 2.019891 2.965865 2.098031 2.241904 2.041313
## [15] 3.229069 2.223534 3.211199 2.623495 2.727196 2.178668 3.495304
```

```
list_extreme_values(x = z, k = 1.5)
```

```
## $small
## [1] 0.001703130 0.077464589 0.047054933 0.060877148 0.009629518 0.004321658
## [7] 0.028495955 0.005327612 0.041129370
##
## $large
##  [1] 0.9899656 0.9521815 0.9741261 0.9474009 0.9586979 0.9756436 0.9954564
##  [8] 0.9517322 0.9342643 0.9310075
```

(b) Using only your function `list_extreme_values()`, `mean()`, and basic list operations, calculate the mean of observations that are greater than 1.5 standard deviation above the mean in the vector `y`.

Solution:

```
mean(list_extreme_values(x = y, k = 1.5)$large)
```

```
## [1] 1.970506
```