CMTH 642 Data Analytics: Advanced Methods

Assignment 1 (10%)
[Daria Yip]
[DHA - Student#500721106]

---

1. Read the csv files in the folder. (3 points)

```
csvdata1 <- read.csv("C:/Users/samda/Documents/Ryerson/CIND 642 -
R/Macronutrients1.csv", header = T, sep =",")
csvdata2 <- read.csv("C:/Users/samda/Documents/Ryerson/CIND 642 -
R/USDA_Micronutrients2.csv", header = T, sep =",")
```

2. Merge the data frames using the variable "ID". Name the Merged Data Frame "USDA". (6 points)

```
USDA <- merge(csvdata1, csvdata2, by = "ID") #merge function to merge two
dataframes by a common variable

head(USDA)

##     ID              Description Calories Protein TotalFat Carbohydrate
Sodium
## 1 1001         BUTTER,WITH SALT      717    0.85    81.11         0.06
714
## 2 1002 BUTTER,WHIPPED,WITH SALT      717    0.85    81.11         0.06
827
## 3 1003     BUTTER OIL,ANHYDROUS      876    0.28    99.48         0.00
2
## 4 1004              CHEESE,BLUE      353   21.40    28.74         2.34
1,395
## 5 1005             CHEESE,BRICK      371   23.24    29.68         2.79
560
## 6 1006              CHEESE,BRIE      334   20.75    27.68         0.45
629
##   Cholesterol Sugar Calcium Iron Potassium VitaminC VitaminE VitaminD
## 1         215  0.06      24 0.02        24        0     2.32      1.5
## 2         219  0.06      24 0.16        26        0     2.32      1.5
## 3         256  0.00       4 0.00         5        0     2.80      1.8
## 4          75  0.50     528 0.31       256        0     0.25      0.5
## 5          94  0.51     674 0.43       136        0     0.26      0.5
## 6         100  0.45     184 0.50       152        0     0.24      0.5
```

3. Check the datatypes of the attributes. Delete the commas in the Sodium and Potasium records. Assign Sodium and Potasium as numeric data types. (6 points)

```
str(USDA) #Check the datatypes of the attributes, you can also use
sapply(USDA, class)
```

```
## 'data.frame':    7057 obs. of  15 variables:
##  $ ID         : int  1001 1002 1003 1004 1005 1006 1007 1008 1009 1010
...
##  $ Description : chr  "BUTTER,WITH SALT" "BUTTER,WHIPPED,WITH SALT"
"BUTTER OIL,ANHYDROUS" "CHEESE,BLUE" ...
##  $ Calories    : int  717 717 876 353 371 334 300 376 403 387 ...
##  $ Protein     : num  0.85 0.85 0.28 21.4 23.24 ...
##  $ TotalFat    : num  81.1 81.1 99.5 28.7 29.7 ...
##  $ Carbohydrate: num  0.06 0.06 0 2.34 2.79 0.45 0.46 3.06 1.28 4.78 ...
##  $ Sodium      : chr  "714" "827" "2" "1,395" ...
##  $ Cholesterol : int  215 219 256 75 94 100 72 93 105 103 ...
##  $ Sugar       : num  0.06 0.06 0 0.5 0.51 0.45 0.46 NA 0.52 NA ...
##  $ Calcium     : int  24 24 4 528 674 184 388 673 721 643 ...
##  $ Iron        : num  0.02 0.16 0 0.31 0.43 0.5 0.33 0.64 0.68 0.21 ...
##  $ Potassium   : chr  "24" "26" "5" "256" ...
##  $ VitaminC    : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ VitaminE    : num  2.32 2.32 2.8 0.25 0.26 0.24 0.21 NA 0.29 NA ...
##  $ VitaminD    : num  1.5 1.5 1.8 0.5 0.5 0.5 0.4 NA 0.6 NA ...

USDA$Sodium <- as.numeric(gsub(",","",USDA$Sodium)) #delete commas in Sodium
and assign
USDA$Potassium <- as.numeric(gsub(",","", USDA$Potassium)) #delete commas in
Potassium & #assign to numeric

class(USDA$Sodium) #check class

## [1] "numeric"

class(USDA$Potassium) #check class

## [1] "numeric"

head(USDA) #check that commas have been removed from both

##     ID              Description Calories Protein TotalFat Carbohydrate
Sodium
## 1 1001         BUTTER,WITH SALT      717    0.85    81.11         0.06
714
## 2 1002 BUTTER,WHIPPED,WITH SALT      717    0.85    81.11         0.06
827
## 3 1003     BUTTER OIL,ANHYDROUS      876    0.28    99.48         0.00
2
## 4 1004              CHEESE,BLUE      353   21.40    28.74         2.34
1395
## 5 1005             CHEESE,BRICK      371   23.24    29.68         2.79
560
## 6 1006              CHEESE,BRIE      334   20.75    27.68         0.45
629
##   Cholesterol Sugar Calcium Iron Potassium VitaminC VitaminE VitaminD
## 1         215  0.06      24 0.02        24        0     2.32      1.5
## 2         219  0.06      24 0.16        26        0     2.32      1.5
## 3          256  0.00        4 0.00          5          0      2.80       1.8
## 4           75  0.50      528 0.31        256          0      0.25       0.5
## 5           94  0.51      674 0.43        136          0      0.26       0.5
## 6          100  0.45      184 0.50        152          0      0.24       0.5
```

4. Remove records (rows) with missing values in more than 4 attributes (columns). How many records remain in the data frame? (6 points)

```
missing <- rowSums(is.na(USDA))
USDA <- USDA[!missing > 4,]
nrow(USDA)

## [1] 6887

#Therefore, there are 6887 records that remain in the dataframe #that do not
have rows with missing values more than 4 #attributes.
```

5. For records with missing values for Sugar, Vitamin E and Vitamin D, replace missing values with mean value for the respective variable. (6 points)

```
mean.sugar <- mean(USDA$Sugar, na.rm = TRUE)
USDA$Sugar[is.na(USDA$Sugar)] = mean.sugar

mean.vitd <- mean(USDA$VitaminD, na.rm=TRUE)
USDA$VitaminD[is.na(USDA$VitaminD)] = mean.vitd

mean.vite <- mean(USDA$VitaminE, na.rm=TRUE)
USDA$VitaminE[is.na(USDA$VitaminE)] = mean.vite

check <- sum(is.na(USDA$Sugar)) + sum(is.na(USDA$VitaminE)) +
sum(is.na(USDA$VitaminD)) #double check #no more missing values in these
columns
check

## [1] 0
```

6. With a single line of code, remove all remaining records with missing values. Name the new Data Frame "USDAclean". How many records remain in the data frame? (6 points)

```
USDAclean <- na.omit(USDA) #single line of code to remove all remaining
missing values
nrow(USDAclean) #remaining is 6310 from the original 6887 in Question 4

## [1] 6310
```

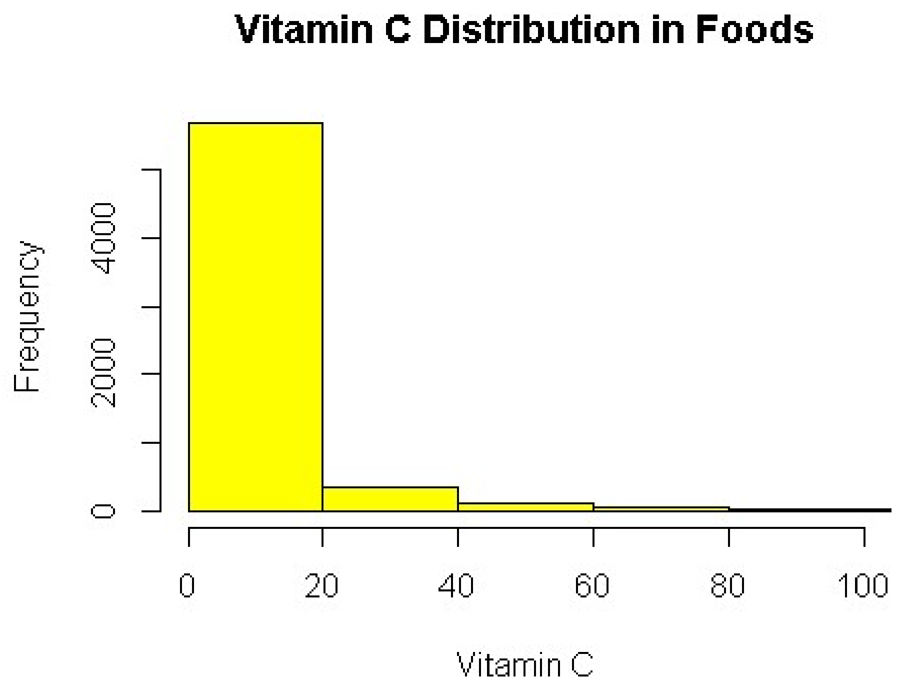
7. Which food has the highest sodium level? (6 points)

```
USDAclean[which(USDAclean$Sodium == max(USDAclean$Sodium)),]

##       ID Description Calories Protein TotalFat Carbohydrate Sodium
Cholesterol
## 265 2047  SALT,TABLE        0       0        0            0  38758
0
##     Sugar Calcium Iron Potassium VitaminC VitaminE VitaminD
## 265     0      24 0.33         8        0        0        0
```

```
#can also use the which.max #USDAclean[which.max(USDAclean$Sodium),]

#Therefore, salt has the highest sodium level.
```

8. Create a histogram of Vitamin C distribution in foods. (6 points)

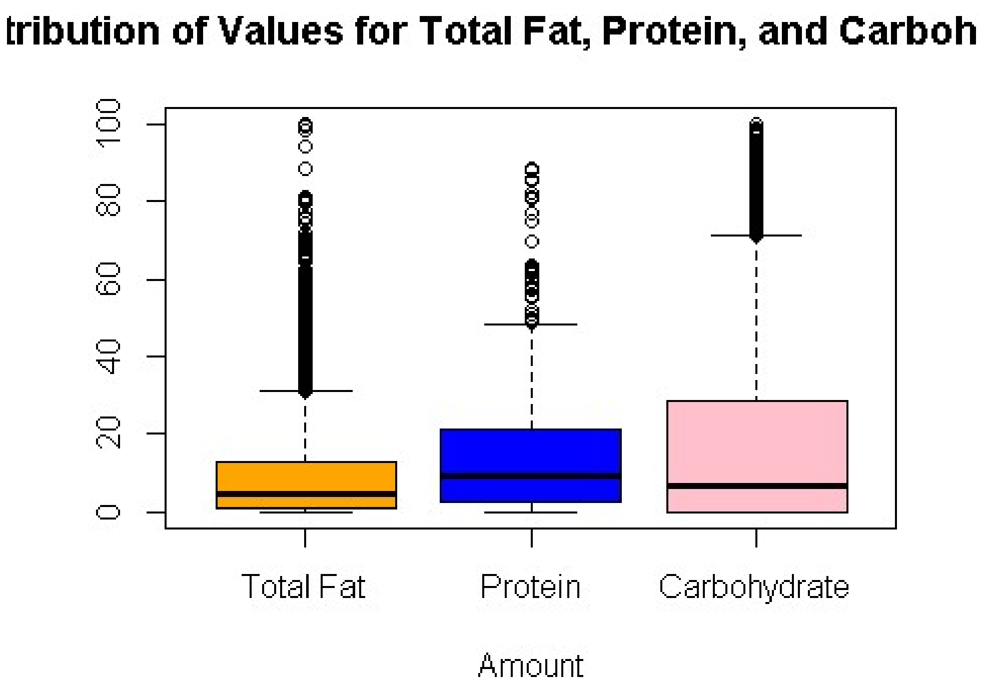
```
summary(USDAclean$VitaminC) #take a look at the column to see

##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
##    0.000    0.000    0.000    9.284    3.000 2400.000

#what kind of axis limits we need
hist(USDAclean$VitaminC, breaks = 100, xlim = c(0, 100),
     xlab = "Vitamin C", col = "yellow", main = "Vitamin C Distribution in
Foods")
```



```
#breaks is very important to include here so R knows how we
#want the histogram to be broken up
```

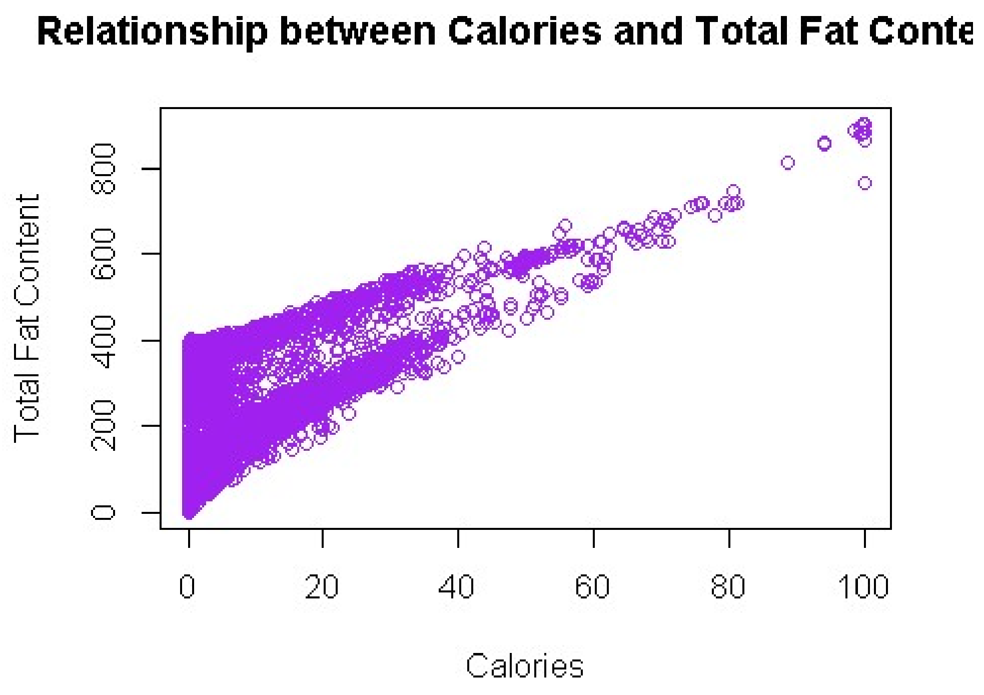
9. Create a boxplot to illustrate the distribution of values for TotalFat, Protein and Carbohydrate. (6 points)

```
boxplot(USDAclean$TotalFat, USDAclean$Protein, USDAclean$Carbohydrate, names
= c("Total Fat", "Protein", "Carbohydrate"), col = c("orange", "blue",
"pink"), xlab = "Amount", main = "Distribution of Values for Total Fat,
Protein, and Carbohydrates", horizontal = FALSE)
```



10. Create a scatterplot to illustrate the relationship between a food's TotalFat content and its Calorie content. (6 points)

```
plot(USDAclean$Calories~USDAclean$TotalFat, main = "Relationship between
Calories and Total Fat Content", xlab = "Calories", ylab = "Total Fat
Content", col = "purple")
```



11. Add a variable to the data frame that takes value 1 if the food has higher sodium than average, 0 otherwise. Call this variable HighSodium. Do the same for High Calories, High Protein, High Sugar, and High Fat. How many foods have both high sodium and high fat? (8 points)

```
HighSodium <- 0 #Initialize
HighCalories <- 0
HighProtein <- 0
HighSugar <- 0
HighFat <- 0

USDAclean <- cbind(USDAclean, HighSodium) #add column
USDAclean <- cbind(USDAclean, HighCalories, HighProtein,
                   HighSugar, HighFat) #add columns

#For loop to calculate the if and else for High Sodium #condition
for(i in 1:length(USDAclean$Description))
{
    if(USDAclean$Sodium[i] > mean(USDAclean$Sodium))
       {
         USDAclean$HighSodium[i] = 1
       }
       else
         {
           USDAclean$HighSodium[i] = 0
         }
}
```

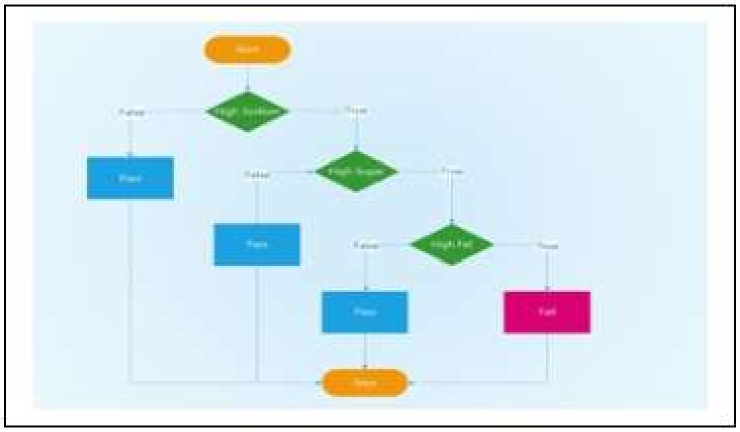
```
#High Calories, using extraction
USDAclean$HighCalories[USDAclean$Calories > mean(USDAclean$Calories)] <- 1
USDAclean$HighCalories[USDAclean$Calories <= mean(USDAclean$Calories)] <- 0

#HighProtein, using if else statement instead of for loop
USDAclean$HighProtein <- ifelse(USDAclean$Protein >
mean(USDAclean$Protein),1,0)

#HighSugar
USDAclean$HighSugar <- ifelse(USDAclean$Sugar > mean(USDAclean$Sugar),1,0)

#HighFat
USDAclean$HighFat <- ifelse(USDAclean$TotalFat >
mean(USDAclean$TotalFat),1,0)

index <- USDAclean[which(USDAclean$HighFat == 1 & USDAclean$HighSodium ==
1),]
nrow(index) #High Sodium & High Fat foods

## [1] 644

#Therefore, there are 644 foods that have high sodium and high
#fat content.

head(USDAclean) #take a look at the data

##     ID              Description Calories Protein TotalFat Carbohydrate
Sodium
## 1 1001          BUTTER,WITH SALT      717    0.85    81.11         0.06
714
## 2 1002 BUTTER,WHIPPED,WITH SALT      717    0.85    81.11         0.06
827
## 3 1003     BUTTER OIL,ANHYDROUS      876    0.28    99.48         0.00
2
## 4 1004              CHEESE,BLUE      353   21.40    28.74         2.34
1395
## 5 1005             CHEESE,BRICK      371   23.24    29.68         2.79
560
## 6 1006              CHEESE,BRIE      334   20.75    27.68         0.45
629
##   Cholesterol Sugar Calcium Iron Potassium VitaminC VitaminE VitaminD
## 1         215  0.06      24 0.02        24        0     2.32      1.5
## 2         219  0.06      24 0.16        26        0     2.32      1.5
## 3         256  0.00       4 0.00         5        0     2.80      1.8
## 4          75  0.50     528 0.31       256        0     0.25      0.5
## 5          94  0.51     674 0.43       136        0     0.26      0.5
## 6         100  0.45     184 0.50       152        0     0.24      0.5
##   HighSodium HighCalories HighProtein HighSugar HighFat
## 1          1            1           0         0       1
## 2          1            1          0         0        1
## 3          0            1          0         0        1
## 4          1            1          1         0        1
## 5          1            1          1         0        1
## 6          1            1          1         0        1
```

12. Calculate the average amount of iron, for high and low protein foods. (8 points)

```
highavg <- mean(USDAclean$Iron[USDAclean$HighProtein == 1]) #high #protein
foods average iron
highavg

## [1] 3.069541

lowavg <- mean(USDAclean$Iron[USDAclean$HighProtein == 0]) #low
#protein foods average iron
lowavg

## [1] 2.696634

ironmean <- cbind(highavg,lowavg)
colnames(ironmean) <- c("High Protein", "Low Protein")
rownames(ironmean) <- c("Average Iron")
ironmean

##              High Protein Low Protein
## Average Iron     3.069541    2.696634

#The average amount of iron for high protein foods is 3.069541
#for High Protein Levels and 2.696634 for low protein foods.

#You can also do this using the aggregate function,
#aggregate(USDAclean$Iron), list(USDAclean$HighProtein) where #the FUN =
mean.
```

13. Create a script for a "HealthCheck" program to detect unhealthy foods. Use the algorithm flowchart below as a basis for this script. (8 points)

```
require(jpeg)

## Loading required package: jpeg

img<-readJPEG("HealthCheck.jpg")
plot(1:4, ty = 'n', ann = F, xaxt = 'n', yaxt = 'n')
rasterImage(img,1,1,4,4)
```



```r
HC <- function(x,y,z)
{
  ifelse(x == 1 & y == 1 & z == 1, "Fail", "Pass")
        }
HC(0,1,1) #Testing

## [1] "Pass"
```

14. Add a new variable called HealthCheck to the data frame using the output of the function. (8 points)

```r
HealthCheck <- 0 #Initialize
USDAclean <- cbind(USDAclean, HealthCheck) #add column

USDAclean$HealthCheck <-
HC(USDAclean$HighSodium,USDAclean$HighSugar,USDAclean$HighFat)

head(USDAclean)

##     ID              Description Calories Protein TotalFat Carbohydrate
Sodium
## 1 1001         BUTTER,WITH SALT      717    0.85    81.11         0.06
714
## 2 1002 BUTTER,WHIPPED,WITH SALT      717    0.85    81.11         0.06
827
## 3 1003     BUTTER OIL,ANHYDROUS      876    0.28    99.48         0.00
2
## 4 1004              CHEESE,BLUE      353   21.40    28.74         2.34
```

```
1395
## 5 1005              CHEESE,BRICK      371   23.24    29.68          2.79
560
## 6 1006               CHEESE,BRIE      334   20.75    27.68          0.45
629
##   Cholesterol Sugar Calcium Iron Potassium VitaminC VitaminE VitaminD
## 1         215  0.06      24 0.02        24        0     2.32      1.5
## 2         219  0.06      24 0.16        26        0     2.32      1.5
## 3         256  0.00       4 0.00         5        0     2.80      1.8
## 4          75  0.50     528 0.31       256        0     0.25      0.5
## 5          94  0.51     674 0.43       136        0     0.26      0.5
## 6         100  0.45     184 0.50       152        0     0.24      0.5
##   HighSodium HighCalories HighProtein HighSugar HighFat HealthCheck
## 1          1            1           0         0       1        Pass
## 2          1            1           0         0       1        Pass
## 3          0            1           0         0       1        Pass
## 4          1            1           1         0       1        Pass
## 5          1            1           1         0       1        Pass
## 6          1            1           1         0       1        Pass
```

15. How many foods in the USDAclean data frame fail the HealthCheck? (8 points)

```
sum(USDAclean$HealthCheck == "Fail")

## [1] 237
```

16. Save your final data frame as “USDAclean_ [your last name].” (3 points)

```
USDAclean_Yip <- USDAclean
```

This is the end of Assignment 1

Ceni Babaoglu, PhD