CMTH 642 Data Analytics: Advanced Methods Assignment 1

1. Read the csv files in the folder. (4 points)

```
USDA_Micro <- read.csv('USDA_Micronutrients2.csv', header=T, sep=",")
USDA_Macro <- read.csv('USDA_Macronutrients2.csv', header=T, sep=",")
```

2. Merge the data frames using the variable "ID". Name the Merged Data Frame "USDA". (4 points)

```
USDA <- merge(USDA_Micro, USDA_Macro, by="ID")
```

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)

```
# Type & Class
sapply(USDA, class)
```

```
##            ID        Sodium   Cholesterol         Sugar       Calcium
##     "integer"      "factor"     "integer"     "numeric"     "integer"
##          Iron     Potassium      VitaminC      VitaminE      VitaminD
##     "numeric"      "factor"     "numeric"     "numeric"     "numeric"
##   Description      Calories       Protein      TotalFat  Carbohydrate
##      "factor"     "integer"     "numeric"     "numeric"     "numeric"
```

```
sapply(USDA, typeof)
```

```
##            ID        Sodium   Cholesterol         Sugar       Calcium
##     "integer"     "integer"     "integer"      "double"     "integer"
##          Iron     Potassium      VitaminC      VitaminE      VitaminD
##      "double"     "integer"      "double"      "double"      "double"
##   Description      Calories       Protein      TotalFat  Carbohydrate
##     "integer"     "integer"      "double"      "double"      "double"
```

```
# str(USDAClean)

# Basic Cleaning & Type Coercion
USDA$Sodium <- gsub(",","", USDA$Sodium)
USDA$Potassium <- gsub(",","", USDA$Potassium)
USDA$Sodium <- as.numeric(USDA$Sodium)
USDA$Potassium <- as.numeric(USDA$Potassium)
```

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

```
USDA <- USDA[apply(USDA, 1 ,function(x) sum(is.na(x)) <= 4), ] #6887 Records remain
```

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)

```
# The wording of this question, "Sugar, vitamin E and Vitamin D" implies the following:

# USDA[is.na(USDA$Sugar) & is.na(USDA$VitaminE) & is.na(USDA$VitaminD), 'Sugar'] <- mean(USDA$Sugar, na
# USDA[is.na(USDA$Sugar) & is.na(USDA$VitaminE) & is.na(USDA$VitaminD), 'VitaminD'] <- mean(USDA$Vitami
# USDA[is.na(USDA$Sugar) & is.na(USDA$VitaminE) & is.na(USDA$VitaminD), 'VitaminE'] <- mean(USDA$Vitami

# However, after discussing with Riyad, I have decided to do the following (which makes a bit more sens

# USDA[is.na(USDA$Sugar), 'Sugar'] <- mean(USDA$Sugar, na.rm=T)
# USDA[is.na(USDA$VitaminE), 'VitaVitaminE'] <- mean(USDA$VitaVitaminE, na.rm=T)
# USDA[is.na(USDA$VitaminD), 'VitaminD'] <- mean(USDA$VitaminD, na.rm=T)

# If you want to do this on many attributes the following for loop works pretty slick:

for(att in c('Sugar','VitaminE','VitaminD')){
  USDA[is.na(USDA[,att]),att] <- mean(USDA[,att], na.rm=T)
}
```

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 <- USDA[complete.cases(USDA),]
nrow(USDAClean) #6310
```

```
## [1] 6310
```

```
# sum(is.na(USDA)) # 690
# sum(is.na(USDAClean)) # 0
```

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

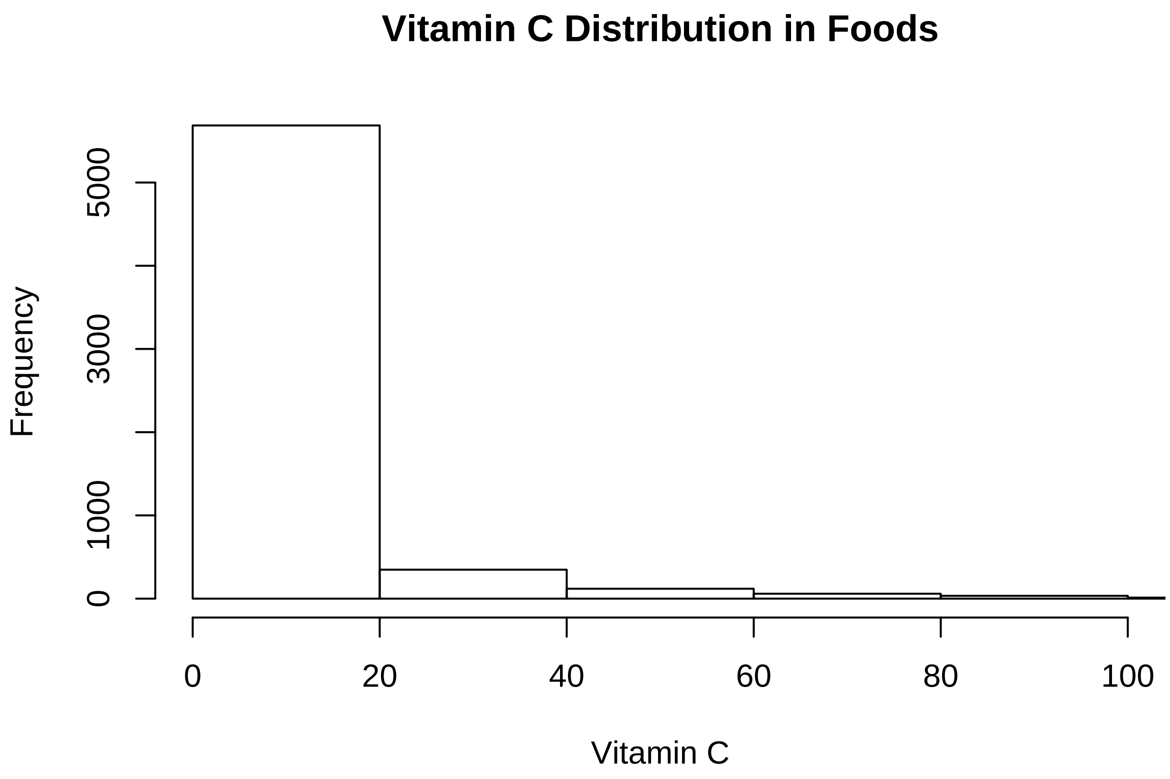
```
USDAClean[USDAClean$Sodium == max(USDAClean$Sodium),]
```

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

```
# SALT, TABLE (Table Salt)
```

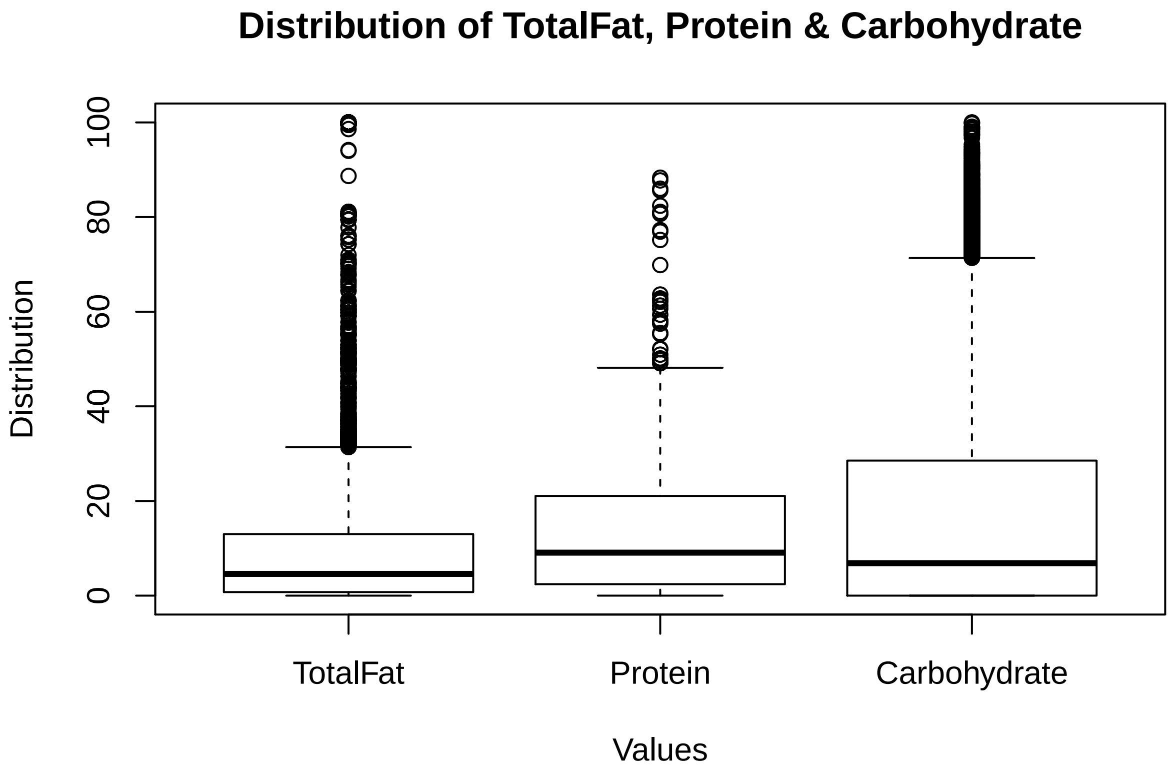
8. Create a histogram of Vitamin C distribution in foods, with a limit of 0 to 100 on the x-axis and breaks of 100. (6 points)

```
# hist(USDAClean[USDAClean$VitaminC < 100.1, 'VitaminC'], breaks=seq(-0.1,100.1,by=0.1), xlim=c(0,100),
hist(USDAClean$VitaminC, breaks=100, xlim=c(0,100), main="Vitamin C Distribution in Foods", xlab="Vitami
```



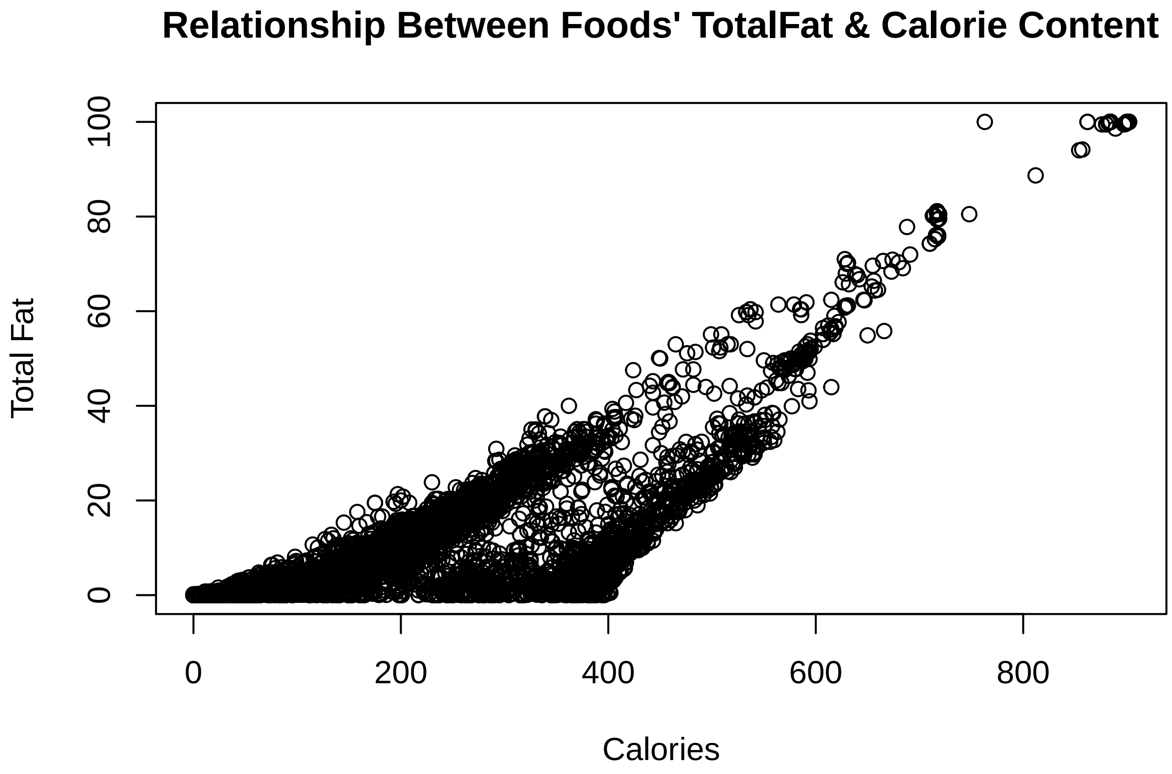
9. Create a boxplot to illustrate the distribution of values for TotalFat, Protein and Carbohydrate. (6 points)

```
boxplot(USDAClean[,c('TotalFat','Protein','Carbohydrate')], main="Distribution of TotalFat, Protein & Ca
```



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 Foods' TotalFat & Calorie Conte
```



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)

```
# Long Way
# USDAClean$HighSodium <- as.numeric(USDAClean[,'Sodium'] > mean(USDAClean$Sodium))
# USDAClean$HighCalories <- as.numeric(USDAClean[,'Calories'] > mean(USDAClean$Calories))
# USDAClean$HighProtein <- as.numeric(USDAClean[,'Protein'] > mean(USDAClean$Protein))
# USDAClean$HighSugar <- as.numeric(USDAClean[,'Sugar'] > mean(USDAClean$Sugar))
# USDAClean$HighFat <- as.numeric(USDAClean[,'TotalFat'] > mean(USDAClean$TotalFat))

# Short Way
for(att in c('Sodium', 'Calories','Protein','Sugar','TotalFat')){
  high.att <- paste('High',att,sep='')
  USDAClean[,high.att] <- as.numeric(USDAClean[,att] > mean(USDAClean[,att]))
}

# How many foods have both high sodium and high fat?
nrow(USDAClean[USDAClean$HighSodium & USDAClean$HighTotalFat, ]) #644
```

```
## [1] 644
```

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

```
#0: Low Protein, 1: High Protein
sapply(split(USDAClean$Iron, USDAClean$HighProtein), mean)
```

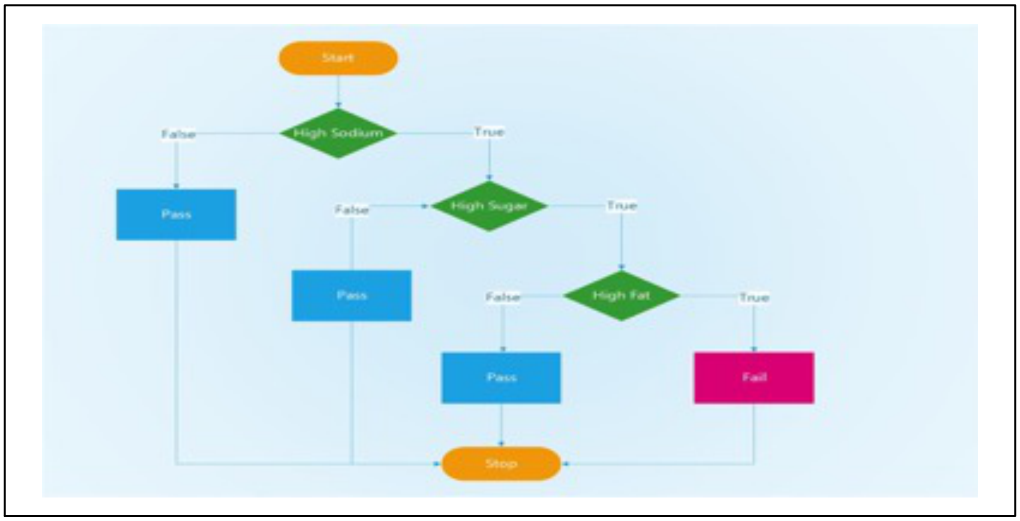
```
##        0        1
## 2.696634 3.069541
```

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)

```
# install.packages(jpeg)
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)
```



```
# As we already inserted the High<Ingredient> columns you can subset the dataframe without abstracting
# the process into its own function:
# bad.foods <- USDAClean[USDAClean$HighSodium & USDAClean$HighSugar & USDAClean$HighFat,]
#
# However, as it seems you want an if/else ladder (in a function?):
# This function will return a false (health check fails) for foods with higher than average sodium, sug
# Otherwise returns true (health check pass)

health.check <- function(obs, data=USDAClean){
  meanSodium <- mean(data$Sodium)
  meanSugar <- mean(data$Sugar)
  meanFat <- mean(data$TotalFat)
  if(as.numeric(obs['Sodium']) > meanSodium & as.numeric(obs['Sugar']) > meanSugar & as.numeric(obs['To
  else return(TRUE)
}
```

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

```
USDAClean$HealthCheck <- apply(USDAClean,1,health.check)
```

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

```
# Total which fail test: Total - Pass
(nrow(USDAClean) - sum(USDAClean$HealthCheck)) #237
```

```
## [1] 237
```

```
# Same as:
# nrow(USDAClean[USDAClean$HighSodium & USDAClean$HighSugar & USDAClean$HighTotalFat,]) #237
```

16. Save your final data frame as “USDAclean_ [your last name]” (4 points)

```r
write.csv(x=USDAClean, file="USDAclean_Clark.csv")
```