Cereals Data Markdown

Mohini Mishra

6/10/2020

setwd("E:/Data Science/R/class 7")  
getwd()

## [1] "E:/Data Science/R/class 7"

library(tidyverse)

## -- Attaching packages ---------------------------------------------------------------------------------- tidyverse 1.3.0 --

## v ggplot2 3.3.0 v purrr 0.3.3  
## v tibble 3.0.0 v dplyr 0.8.5  
## v tidyr 1.0.2 v stringr 1.4.0  
## v readr 1.3.1 v forcats 0.5.0

## -- Conflicts ------------------------------------------------------------------------------------- tidyverse\_conflicts() --  
## x dplyr::filter() masks stats::filter()  
## x dplyr::lag() masks stats::lag()

library(psych)

##   
## Attaching package: 'psych'

## The following objects are masked from 'package:ggplot2':  
##   
## %+%, alpha

#-----------Importing the cereals\_data file-----------------  
cereals\_data<-read.csv("cereals\_data.csv")  
View(cereals\_data)  
dim(cereals\_data)

## [1] 77 16

str(cereals\_data)

## 'data.frame': 77 obs. of 16 variables:  
## $ name : Factor w/ 77 levels "100%\_Bran","100%\_Natural\_Bran",..: 47 45 46 71 50 40 8 52 13 29 ...  
## $ mfr : Factor w/ 7 levels "A","G","K","N",..: 3 7 7 2 3 3 2 2 2 3 ...  
## $ type : Factor w/ 2 levels "C","H": 1 1 1 1 1 1 1 1 1 1 ...  
## $ calories: int 160 150 150 140 140 140 130 130 120 120 ...  
## $ protein : int 3 4 4 3 3 3 3 3 1 3 ...  
## $ fat : int 2 3 3 1 2 1 2 2 3 0 ...  
## $ sodium : int 150 95 150 190 220 170 210 170 210 240 ...  
## $ fiber : num 3 3 3 4 3 2 2 1.5 0 5 ...  
## $ carbo : num 17 16 16 15 21 20 18 13.5 13 14 ...  
## $ sugars : int 13 11 11 14 7 9 8 10 9 12 ...  
## $ potass : int 160 170 170 230 130 95 100 120 45 190 ...  
## $ vitamins: int 25 25 25 100 25 100 25 25 25 25 ...  
## $ shelf : int 3 3 3 3 3 3 3 3 2 3 ...  
## $ weight : num 1.5 1 1 1.5 1.33 1.3 1.33 1.25 1 1.33 ...  
## $ cups : num 0.67 1 1 1 0.67 0.75 0.75 0.5 0.75 0.67 ...  
## $ rating : num 30.3 37.1 34.1 28.6 40.7 ...

cereals\_data1<-cereals\_data  
head(cereals\_data1)

## name mfr type calories protein fat sodium fiber  
## 1 Mueslix\_Crispy\_Blend K C 160 3 2 150 3  
## 2 Muesli\_Raisins,\_Dates,\_&\_Almonds R C 150 4 3 95 3  
## 3 Muesli\_Raisins,\_Peaches,\_&\_Pecans R C 150 4 3 150 3  
## 4 Total\_Raisin\_Bran G C 140 3 1 190 4  
## 5 Nutri-Grain\_Almond-Raisin K C 140 3 2 220 3  
## 6 Just\_Right\_Fruit\_&\_Nut K C 140 3 1 170 2  
## carbo sugars potass vitamins shelf weight cups rating  
## 1 17 13 160 25 3 1.50 0.67 30.31335  
## 2 16 11 170 25 3 1.00 1.00 37.13686  
## 3 16 11 170 25 3 1.00 1.00 34.13976  
## 4 15 14 230 100 3 1.50 1.00 28.59278  
## 5 21 7 130 25 3 1.33 0.67 40.69232  
## 6 20 9 95 100 3 1.30 0.75 36.47151

#=============Data Clening & wrangling===============  
  
#---------Replacing short column names with complete name--------  
colnames(cereals\_data1) <-c("Name", "Manufacturer", "Type", "Calories", "Protein", "Fat",   
 "Sodium", "Fiber", "Carbohydrates", "Sugar", "Potassium",   
 "Vitamins", "Shelf", "Weight", "Cups", "Rating")  
variable.names(cereals\_data1)

## [1] "Name" "Manufacturer" "Type" "Calories"   
## [5] "Protein" "Fat" "Sodium" "Fiber"   
## [9] "Carbohydrates" "Sugar" "Potassium" "Vitamins"   
## [13] "Shelf" "Weight" "Cups" "Rating"

variable.names(cereals\_data)

## [1] "name" "mfr" "type" "calories" "protein" "fat"   
## [7] "sodium" "fiber" "carbo" "sugars" "potass" "vitamins"  
## [13] "shelf" "weight" "cups" "rating"

#-----------creating anothe variable Manufacturer\_Name------  
cereals\_data1$Manufacturer\_Name <- cereals\_data1$Manufacturer  
dim(cereals\_data1)

## [1] 77 17

cereals\_data1$Manufacturer\_Name <- gsub(pattern = "P", replacement = "Post",x = cereals\_data1$Manufacturer\_Name)  
cereals\_data1$Manufacturer\_Name <- gsub(pattern = "A", replacement = "American Home..",x = cereals\_data1$Manufacturer\_Name)  
cereals\_data1$Manufacturer\_Name <- gsub(pattern = "G", replacement = "General Mills",x = cereals\_data1$Manufacturer\_Name)  
cereals\_data1$Manufacturer\_Name <- gsub(pattern = "K", replacement = "Kelloggs",x = cereals\_data1$Manufacturer\_Name)  
cereals\_data1$Manufacturer\_Name <- gsub(pattern = "N", replacement = "Nabisco",x = cereals\_data1$Manufacturer\_Name)  
cereals\_data1$Manufacturer\_Name <- gsub(pattern = "Q", replacement = "Quaker Oats",x = cereals\_data1$Manufacturer\_Name)  
cereals\_data1$Manufacturer\_Name <- gsub(pattern = "R", replacement = "Ralston Purina",x = cereals\_data1$Manufacturer\_Name)  
variable.names(cereals\_data1)

## [1] "Name" "Manufacturer" "Type"   
## [4] "Calories" "Protein" "Fat"   
## [7] "Sodium" "Fiber" "Carbohydrates"   
## [10] "Sugar" "Potassium" "Vitamins"   
## [13] "Shelf" "Weight" "Cups"   
## [16] "Rating" "Manufacturer\_Name"

cereals\_data1$Manufacturer\_Name

## [1] "Kelloggs" "Ralston Purina" "Ralston Purina" "General Mills"   
## [5] "Kelloggs" "Kelloggs" "General Mills" "General Mills"   
## [9] "General Mills" "Kelloggs" "Kelloggs" "Kelloggs"   
## [13] "Post" "Post" "Post" "Quaker Oats"   
## [17] "Quaker Oats" "Quaker Oats" "General Mills" "General Mills"   
## [21] "General Mills" "General Mills" "General Mills" "General Mills"   
## [25] "General Mills" "General Mills" "General Mills" "General Mills"   
## [29] "General Mills" "General Mills" "General Mills" "Kelloggs"   
## [33] "Kelloggs" "Kelloggs" "Kelloggs" "Kelloggs"   
## [37] "Kelloggs" "Kelloggs" "Kelloggs" "Kelloggs"   
## [41] "Kelloggs" "Post" "Post" "Post"   
## [45] "Ralston Purina" "Ralston Purina" "Ralston Purina" "American Home.."  
## [49] "General Mills" "General Mills" "General Mills" "General Mills"   
## [53] "General Mills" "Kelloggs" "Kelloggs" "Kelloggs"   
## [57] "Nabisco" "Post" "Post" "Quaker Oats"   
## [61] "Quaker Oats" "Quaker Oats" "Ralston Purina" "Ralston Purina"   
## [65] "Kelloggs" "Kelloggs" "Nabisco" "Nabisco"   
## [69] "Nabisco" "Post" "Ralston Purina" "Nabisco"   
## [73] "Kelloggs" "Nabisco" "Kelloggs" "Quaker Oats"   
## [77] "Quaker Oats"

dim(cereals\_data1)

## [1] 77 17

#--------Replace H and C in Type with Hot and Cold-------  
cereals\_data1$Type <- gsub("H", "Hot", x = cereals\_data1$Type)  
cereals\_data1$Type <- gsub("C", "Cold", x = cereals\_data1$Type)  
  
#--------Rounding off Rating to two decimal points-------  
cereals\_data1$Rating<-round(cereals\_data1$Rating,2)  
#----Removing 1st variable "Names" and alloting to rows------  
library(dplyr)  
rownames(cereals\_data1)<-cereals\_data1[ ,1] #---OR---rownames(cereals\_data1)= cereals\_data1$Name;rownames(cereals\_data1)  
head(cereals\_data1)

## Name  
## Mueslix\_Crispy\_Blend Mueslix\_Crispy\_Blend  
## Muesli\_Raisins,\_Dates,\_&\_Almonds Muesli\_Raisins,\_Dates,\_&\_Almonds  
## Muesli\_Raisins,\_Peaches,\_&\_Pecans Muesli\_Raisins,\_Peaches,\_&\_Pecans  
## Total\_Raisin\_Bran Total\_Raisin\_Bran  
## Nutri-Grain\_Almond-Raisin Nutri-Grain\_Almond-Raisin  
## Just\_Right\_Fruit\_&\_Nut Just\_Right\_Fruit\_&\_Nut  
## Manufacturer Type Calories Protein Fat Sodium  
## Mueslix\_Crispy\_Blend K Cold 160 3 2 150  
## Muesli\_Raisins,\_Dates,\_&\_Almonds R Cold 150 4 3 95  
## Muesli\_Raisins,\_Peaches,\_&\_Pecans R Cold 150 4 3 150  
## Total\_Raisin\_Bran G Cold 140 3 1 190  
## Nutri-Grain\_Almond-Raisin K Cold 140 3 2 220  
## Just\_Right\_Fruit\_&\_Nut K Cold 140 3 1 170  
## Fiber Carbohydrates Sugar Potassium Vitamins  
## Mueslix\_Crispy\_Blend 3 17 13 160 25  
## Muesli\_Raisins,\_Dates,\_&\_Almonds 3 16 11 170 25  
## Muesli\_Raisins,\_Peaches,\_&\_Pecans 3 16 11 170 25  
## Total\_Raisin\_Bran 4 15 14 230 100  
## Nutri-Grain\_Almond-Raisin 3 21 7 130 25  
## Just\_Right\_Fruit\_&\_Nut 2 20 9 95 100  
## Shelf Weight Cups Rating Manufacturer\_Name  
## Mueslix\_Crispy\_Blend 3 1.50 0.67 30.31 Kelloggs  
## Muesli\_Raisins,\_Dates,\_&\_Almonds 3 1.00 1.00 37.14 Ralston Purina  
## Muesli\_Raisins,\_Peaches,\_&\_Pecans 3 1.00 1.00 34.14 Ralston Purina  
## Total\_Raisin\_Bran 3 1.50 1.00 28.59 General Mills  
## Nutri-Grain\_Almond-Raisin 3 1.33 0.67 40.69 Kelloggs  
## Just\_Right\_Fruit\_&\_Nut 3 1.30 0.75 36.47 Kelloggs

cereals\_data2 <-cereals\_data1[ ,-1]  
rownames(cereals\_data2)

## [1] "Mueslix\_Crispy\_Blend"   
## [2] "Muesli\_Raisins,\_Dates,\_&\_Almonds"   
## [3] "Muesli\_Raisins,\_Peaches,\_&\_Pecans"   
## [4] "Total\_Raisin\_Bran"   
## [5] "Nutri-Grain\_Almond-Raisin"   
## [6] "Just\_Right\_Fruit\_&\_Nut"   
## [7] "Basic\_4"   
## [8] "Oatmeal\_Raisin\_Crisp"   
## [9] "Cinnamon\_Toast\_Crunch"   
## [10] "Fruitful\_Bran"   
## [11] "Raisin\_Bran"   
## [12] "Nut&Honey\_Crunch"   
## [13] "Great\_Grains\_Pecan"   
## [14] "Fruit\_&\_Fibre\_Dates,\_Walnuts,\_and\_Oats"  
## [15] "Post\_Nat.\_Raisin\_Bran"   
## [16] "100%\_Natural\_Bran"   
## [17] "Honey\_Graham\_Ohs"   
## [18] "Cap'n'Crunch"   
## [19] "Cheerios"   
## [20] "Clusters"   
## [21] "Kix"   
## [22] "Triples"   
## [23] "Total\_Corn\_Flakes"   
## [24] "Wheaties\_Honey\_Gold"   
## [25] "Honey\_Nut\_Cheerios"   
## [26] "Apple\_Cinnamon\_Cheerios"   
## [27] "Trix"   
## [28] "Lucky\_Charms"   
## [29] "Golden\_Grahams"   
## [30] "Cocoa\_Puffs"   
## [31] "Count\_Chocula"   
## [32] "Special\_K"   
## [33] "Crispix"   
## [34] "Rice\_Krispies"   
## [35] "Cracklin'\_Oat\_Bran"   
## [36] "Just\_Right\_Crunchy\_\_Nuggets"   
## [37] "Corn\_Pops"   
## [38] "Apple\_Jacks"   
## [39] "Froot\_Loops"   
## [40] "Frosted\_Flakes"   
## [41] "Smacks"   
## [42] "Grape-Nuts"   
## [43] "Honey-comb"   
## [44] "Fruity\_Pebbles"   
## [45] "Rice\_Chex"   
## [46] "Corn\_Chex"   
## [47] "Almond\_Delight"   
## [48] "Maypo"   
## [49] "Wheaties"   
## [50] "Total\_Whole\_Grain"   
## [51] "Multi-Grain\_Cheerios"   
## [52] "Raisin\_Nut\_Bran"   
## [53] "Crispy\_Wheat\_&\_Raisins"   
## [54] "Frosted\_Mini-Wheats"   
## [55] "Corn\_Flakes"   
## [56] "Product\_19"   
## [57] "Cream\_of\_Wheat\_(Quick)"   
## [58] "Grape\_Nuts\_Flakes"   
## [59] "Golden\_Crisp"   
## [60] "Quaker\_Oatmeal"   
## [61] "Quaker\_Oat\_Squares"   
## [62] "Life"   
## [63] "Wheat\_Chex"   
## [64] "Double\_Chex"   
## [65] "Nutri-grain\_Wheat"   
## [66] "Raisin\_Squares"   
## [67] "Shredded\_Wheat\_'n'Bran"   
## [68] "Shredded\_Wheat\_spoon\_size"   
## [69] "Strawberry\_Fruit\_Wheats"   
## [70] "Bran\_Flakes"   
## [71] "Bran\_Chex"   
## [72] "Shredded\_Wheat"   
## [73] "All-Bran"   
## [74] "100%\_Bran"   
## [75] "All-Bran\_with\_Extra\_Fiber"   
## [76] "Puffed\_Wheat"   
## [77] "Puffed\_Rice"

head(cereals\_data2)

## Manufacturer Type Calories Protein Fat Sodium  
## Mueslix\_Crispy\_Blend K Cold 160 3 2 150  
## Muesli\_Raisins,\_Dates,\_&\_Almonds R Cold 150 4 3 95  
## Muesli\_Raisins,\_Peaches,\_&\_Pecans R Cold 150 4 3 150  
## Total\_Raisin\_Bran G Cold 140 3 1 190  
## Nutri-Grain\_Almond-Raisin K Cold 140 3 2 220  
## Just\_Right\_Fruit\_&\_Nut K Cold 140 3 1 170  
## Fiber Carbohydrates Sugar Potassium Vitamins  
## Mueslix\_Crispy\_Blend 3 17 13 160 25  
## Muesli\_Raisins,\_Dates,\_&\_Almonds 3 16 11 170 25  
## Muesli\_Raisins,\_Peaches,\_&\_Pecans 3 16 11 170 25  
## Total\_Raisin\_Bran 4 15 14 230 100  
## Nutri-Grain\_Almond-Raisin 3 21 7 130 25  
## Just\_Right\_Fruit\_&\_Nut 2 20 9 95 100  
## Shelf Weight Cups Rating Manufacturer\_Name  
## Mueslix\_Crispy\_Blend 3 1.50 0.67 30.31 Kelloggs  
## Muesli\_Raisins,\_Dates,\_&\_Almonds 3 1.00 1.00 37.14 Ralston Purina  
## Muesli\_Raisins,\_Peaches,\_&\_Pecans 3 1.00 1.00 34.14 Ralston Purina  
## Total\_Raisin\_Bran 3 1.50 1.00 28.59 General Mills  
## Nutri-Grain\_Almond-Raisin 3 1.33 0.67 40.69 Kelloggs  
## Just\_Right\_Fruit\_&\_Nut 3 1.30 0.75 36.47 Kelloggs

# ------Change cereal type,shelf to factor----  
str(cereals\_data2)

## 'data.frame': 77 obs. of 16 variables:  
## $ Manufacturer : Factor w/ 7 levels "A","G","K","N",..: 3 7 7 2 3 3 2 2 2 3 ...  
## $ Type : chr "Cold" "Cold" "Cold" "Cold" ...  
## $ Calories : int 160 150 150 140 140 140 130 130 120 120 ...  
## $ Protein : int 3 4 4 3 3 3 3 3 1 3 ...  
## $ Fat : int 2 3 3 1 2 1 2 2 3 0 ...  
## $ Sodium : int 150 95 150 190 220 170 210 170 210 240 ...  
## $ Fiber : num 3 3 3 4 3 2 2 1.5 0 5 ...  
## $ Carbohydrates : num 17 16 16 15 21 20 18 13.5 13 14 ...  
## $ Sugar : int 13 11 11 14 7 9 8 10 9 12 ...  
## $ Potassium : int 160 170 170 230 130 95 100 120 45 190 ...  
## $ Vitamins : int 25 25 25 100 25 100 25 25 25 25 ...  
## $ Shelf : int 3 3 3 3 3 3 3 3 2 3 ...  
## $ Weight : num 1.5 1 1 1.5 1.33 1.3 1.33 1.25 1 1.33 ...  
## $ Cups : num 0.67 1 1 1 0.67 0.75 0.75 0.5 0.75 0.67 ...  
## $ Rating : num 30.3 37.1 34.1 28.6 40.7 ...  
## $ Manufacturer\_Name: chr "Kelloggs" "Ralston Purina" "Ralston Purina" "General Mills" ...

cereals\_data2$Type <- factor(cereals\_data2$Type)  
cereals\_data2$Shelf <- factor(cereals\_data2$Shelf)  
sapply(cereals\_data2, FUN = class)

## Manufacturer Type Calories Protein   
## "factor" "factor" "integer" "integer"   
## Fat Sodium Fiber Carbohydrates   
## "integer" "integer" "numeric" "numeric"   
## Sugar Potassium Vitamins Shelf   
## "integer" "integer" "integer" "factor"   
## Weight Cups Rating Manufacturer\_Name   
## "numeric" "numeric" "numeric" "character"

#=========Evaluating each variable through concepts of sample statistics=====  
  
summary(cereals\_data2)

## Manufacturer Type Calories Protein Fat   
## A: 1 Cold:74 Min. : 50.0 Min. :1.000 Min. :0.000   
## G:22 Hot : 3 1st Qu.:100.0 1st Qu.:2.000 1st Qu.:0.000   
## K:23 Median :110.0 Median :3.000 Median :1.000   
## N: 6 Mean :106.9 Mean :2.545 Mean :1.013   
## P: 9 3rd Qu.:110.0 3rd Qu.:3.000 3rd Qu.:2.000   
## Q: 8 Max. :160.0 Max. :6.000 Max. :5.000   
## R: 8   
## Sodium Fiber Carbohydrates Sugar   
## Min. : 0.0 Min. : 0.000 Min. : 5.0 Min. : 0.000   
## 1st Qu.:130.0 1st Qu.: 1.000 1st Qu.:12.0 1st Qu.: 3.000   
## Median :180.0 Median : 2.000 Median :14.5 Median : 7.000   
## Mean :159.7 Mean : 2.152 Mean :14.8 Mean : 7.026   
## 3rd Qu.:210.0 3rd Qu.: 3.000 3rd Qu.:17.0 3rd Qu.:11.000   
## Max. :320.0 Max. :14.000 Max. :23.0 Max. :15.000   
## NA's :1 NA's :1   
## Potassium Vitamins Shelf Weight Cups   
## Min. : 15.00 Min. : 0.00 1:20 Min. :0.50 Min. :0.250   
## 1st Qu.: 42.50 1st Qu.: 25.00 2:21 1st Qu.:1.00 1st Qu.:0.670   
## Median : 90.00 Median : 25.00 3:36 Median :1.00 Median :0.750   
## Mean : 98.67 Mean : 28.25 Mean :1.03 Mean :0.821   
## 3rd Qu.:120.00 3rd Qu.: 25.00 3rd Qu.:1.00 3rd Qu.:1.000   
## Max. :330.00 Max. :100.00 Max. :1.50 Max. :1.500   
## NA's :2   
## Rating Manufacturer\_Name   
## Min. :18.04 Length:77   
## 1st Qu.:33.17 Class :character   
## Median :40.40 Mode :character   
## Mean :42.67   
## 3rd Qu.:50.83   
## Max. :93.70   
##

describe(cereals\_data2[,4:11])

## vars n mean sd median trimmed mad min max range skew  
## Protein 1 77 2.55 1.09 3.0 2.48 1.48 1 6 5 0.72  
## Fat 2 77 1.01 1.01 1.0 0.89 1.48 0 5 5 1.12  
## Sodium 3 77 159.68 83.83 180.0 163.25 59.30 0 320 320 -0.55  
## Fiber 4 77 2.15 2.38 2.0 1.77 1.48 0 14 14 2.34  
## Carbohydrates 5 76 14.80 3.91 14.5 14.79 3.71 5 23 18 0.11  
## Sugar 6 76 7.03 4.38 7.0 7.03 5.93 0 15 15 0.04  
## Potassium 7 75 98.67 70.41 90.0 88.11 66.72 15 330 315 1.34  
## Vitamins 8 77 28.25 22.34 25.0 24.60 0.00 0 100 100 2.37  
## kurtosis se  
## Protein 0.93 0.12  
## Fat 1.71 0.11  
## Sodium -0.47 9.55  
## Fiber 7.73 0.27  
## Carbohydrates -0.46 0.45  
## Sugar -1.20 0.50  
## Potassium 1.63 8.13  
## Vitamins 5.55 2.55

#===========Data Visualization-I ============  
  
library(ggplot2)  
library(gplots)

##   
## Attaching package: 'gplots'

## The following object is masked from 'package:stats':  
##   
## lowess

library(superheat)  
library(corrplot)

## corrplot 0.84 loaded

library(readr)  
library(plotrix)

##   
## Attaching package: 'plotrix'

## The following object is masked from 'package:gplots':  
##   
## plotCI

## The following object is masked from 'package:psych':  
##   
## rescale

library(ggcorrplot)  
library(ggpubr)

## Loading required package: magrittr

##   
## Attaching package: 'magrittr'

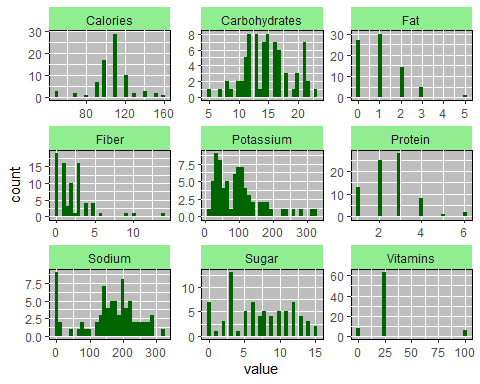
## The following object is masked from 'package:purrr':  
##   
## set\_names

## The following object is masked from 'package:tidyr':  
##   
## extract

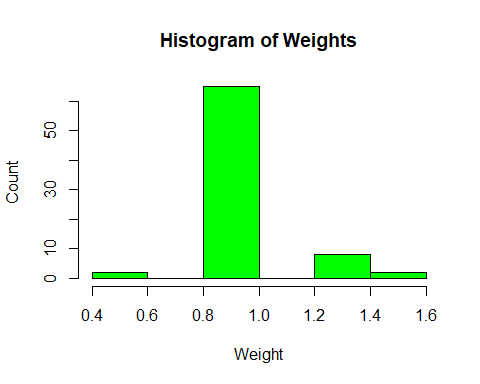
#-----All nutrients in single plot---  
c1<- subset(cereals\_data2, select = c(Calories:Vitamins))  
as.data.frame(c1)%>%  
 gather() %>%   
 ggplot(aes(value)) +   
 facet\_wrap(~ key, scales = "free") +   
 geom\_histogram(fill = "darkgreen") + theme(strip.background = element\_rect(fill="lightgreen"))+  
 theme(panel.background = element\_rect(fill = 'grey', colour = 'black'))

## `stat\_bin()` using `bins = 30`. Pick better value with `binwidth`.

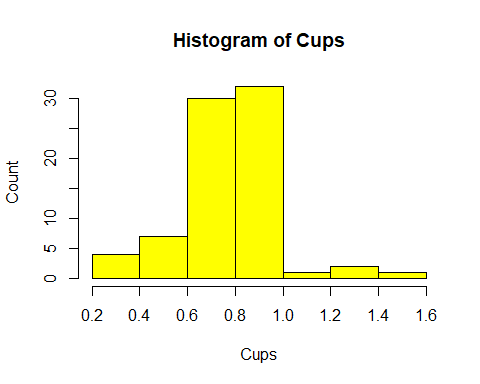
## Warning: Removed 4 rows containing non-finite values (stat\_bin).



#-----Histogram-Distribution of weight and cups-----  
  
p<-cereals\_data2$Weight  
  
hist(p,  
 breaks=7,  
 col="green",  
 xlab = "Weight",  
 ylab = "Count",  
 main = "Histogram of Weights", plot=TRUE)



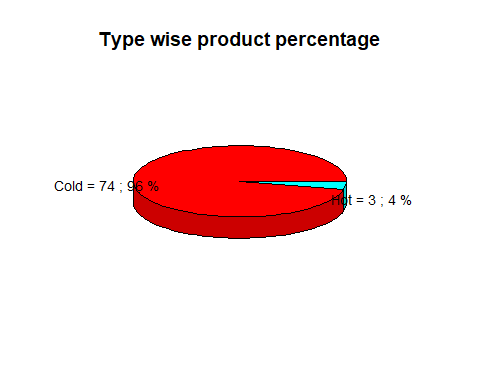
q<-cereals\_data2$Cups  
hist(q,  
 breaks=7,  
 col="yellow",  
 xlab = "Cups",  
 ylab = "Count",  
 main = "Histogram of Cups", plot=TRUE)



#--------3d--Pie Chart- Type wise product percentage------  
  
ctype\_cat<-c("Cold","Hot")  
ctype\_count<-c(74,3)  
ctype\_count\_pct<-round(ctype\_count/sum(ctype\_count)\*100)  
ctype\_count\_pct

## [1] 96 4

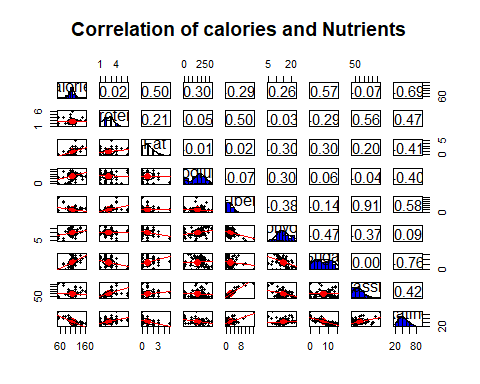
lbls<-paste(ctype\_cat,"= ",ctype\_count, ";", ctype\_count\_pct,"%",sep = " ")  
#pie(ctype\_count,labels=lbls,col = rainbow(length(lbls)),radius=0.85,main = "Cereal's count in each type")  
  
a<-table(cereals\_data2$Type)  
grp<-c("Cold","Hot")  
cnt<-round(a)  
grpd<-paste(grp,"=",cnt)  
lbl<-paste(grpd,";",ctype\_count\_pct,"%")  
pie3D(a,  
 labels = lbl,  
 labelcex=0.9,  
 main = "Type wise product percentage",  
 col = rainbow(length(a)))



#------Co-relation plot------  
variable.names(cereals\_data2)

## [1] "Manufacturer" "Type" "Calories"   
## [4] "Protein" "Fat" "Sodium"   
## [7] "Fiber" "Carbohydrates" "Sugar"   
## [10] "Potassium" "Vitamins" "Shelf"   
## [13] "Weight" "Cups" "Rating"   
## [16] "Manufacturer\_Name"

pairs.panels(cereals\_data2[,-c(1,2,12,13,14,16,11,17,18)],  
 method = "pearson", #coorelation method  
 hist.col = "blue",  
 main="Correlation of calories and Nutrients",  
 density = TRUE, # show density plots  
 ellipses = TRUE, # show correlation ellipses  
 lm=TRUE, #linear regression fits   
 cex.cor = 2,  
 cex.labels=1.5  
)



#==============Data Manipulation===========  
  
#-----------Creating additional variable Calories\_Cat,Rating\_Cat---------  
table(cereals\_data2$Calories)

##   
## 50 70 80 90 100 110 120 130 140 150 160   
## 3 2 1 7 17 29 10 2 3 2 1

cereals\_data2 <- within(cereals\_data2, {  
 Calories\_cat <- NA  
 Calories\_cat[cereals\_data2$Calories <= 100] <-"Low\_Calories"  
 Calories\_cat[cereals\_data2$Calories > 100 & cereals\_data2$Calories < 130] <-"Medium\_Calories"  
 Calories\_cat[cereals\_data2$Calories >= 130] <-"High\_Calories"  
})  
table(cereals\_data2$Calories\_cat)

##   
## High\_Calories Low\_Calories Medium\_Calories   
## 8 30 39

cereals\_data2$Calories\_cat<-factor(cereals\_data2$Calories\_cat)  
str(cereals\_data2$Calories\_cat)

## Factor w/ 3 levels "High\_Calories",..: 1 1 1 1 1 1 1 1 3 3 ...

cereals\_data2<- within(cereals\_data2, {  
 Rating\_Cat <- NA  
 Rating\_Cat[cereals\_data2$Rating <= 40] <- "Poor"  
 Rating\_Cat[cereals\_data2$Rating > 40 & cereals\_data2$Rating < 64] <- "Satisfactory"  
 Rating\_Cat[cereals\_data2$Rating >= 64] <- "Good"  
})  
table(cereals\_data2$Rating\_Cat)

##   
## Good Poor Satisfactory   
## 6 37 34

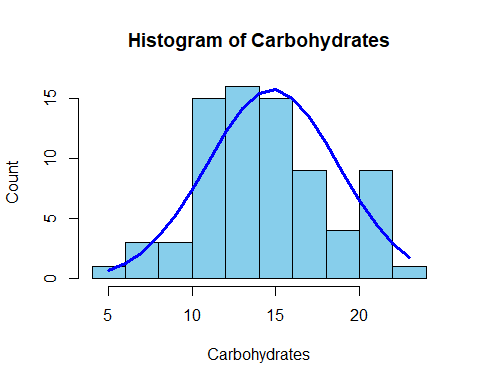
#==============Missing Value Treatment using kNN Imputation==============  
  
#----Calculate NAs------  
sum(is.na(cereals\_data2))

## [1] 4

summary(cereals\_data2)

## Manufacturer Type Calories Protein Fat   
## A: 1 Cold:74 Min. : 50.0 Min. :1.000 Min. :0.000   
## G:22 Hot : 3 1st Qu.:100.0 1st Qu.:2.000 1st Qu.:0.000   
## K:23 Median :110.0 Median :3.000 Median :1.000   
## N: 6 Mean :106.9 Mean :2.545 Mean :1.013   
## P: 9 3rd Qu.:110.0 3rd Qu.:3.000 3rd Qu.:2.000   
## Q: 8 Max. :160.0 Max. :6.000 Max. :5.000   
## R: 8   
## Sodium Fiber Carbohydrates Sugar   
## Min. : 0.0 Min. : 0.000 Min. : 5.0 Min. : 0.000   
## 1st Qu.:130.0 1st Qu.: 1.000 1st Qu.:12.0 1st Qu.: 3.000   
## Median :180.0 Median : 2.000 Median :14.5 Median : 7.000   
## Mean :159.7 Mean : 2.152 Mean :14.8 Mean : 7.026   
## 3rd Qu.:210.0 3rd Qu.: 3.000 3rd Qu.:17.0 3rd Qu.:11.000   
## Max. :320.0 Max. :14.000 Max. :23.0 Max. :15.000   
## NA's :1 NA's :1   
## Potassium Vitamins Shelf Weight Cups   
## Min. : 15.00 Min. : 0.00 1:20 Min. :0.50 Min. :0.250   
## 1st Qu.: 42.50 1st Qu.: 25.00 2:21 1st Qu.:1.00 1st Qu.:0.670   
## Median : 90.00 Median : 25.00 3:36 Median :1.00 Median :0.750   
## Mean : 98.67 Mean : 28.25 Mean :1.03 Mean :0.821   
## 3rd Qu.:120.00 3rd Qu.: 25.00 3rd Qu.:1.00 3rd Qu.:1.000   
## Max. :330.00 Max. :100.00 Max. :1.50 Max. :1.500   
## NA's :2   
## Rating Manufacturer\_Name Calories\_cat Rating\_Cat   
## Min. :18.04 Length:77 High\_Calories : 8 Length:77   
## 1st Qu.:33.17 Class :character Low\_Calories :30 Class :character   
## Median :40.40 Mode :character Medium\_Calories:39 Mode :character   
## Mean :42.67   
## 3rd Qu.:50.83   
## Max. :93.70   
##

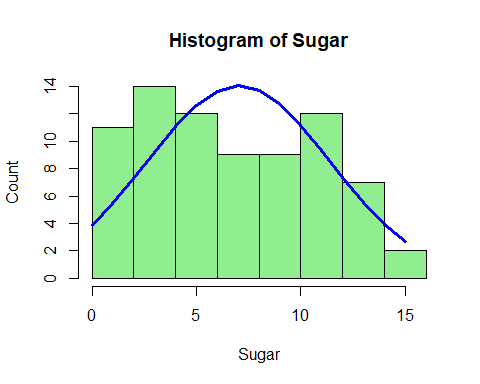
#-------Histograms to check the distribution for imputation-----  
  
#par(mfrow=c(1,3))  
x<-cereals\_data2$Carbohydrates  
h<-hist(x,  
 breaks=10,  
 col="skyblue",  
 xlab = "Carbohydrates",  
 ylab = "Count",  
 main = "Histogram of Carbohydrates")  
xfit<-seq(min(x,na.rm = T),max(x,na.rm = T),length(40))  
yfit<-dnorm(xfit, mean = mean(x,na.rm = T), sd=sd(x,na.rm = T))  
yfit<-yfit \* diff(h$mids[1:2]\*length(x)) #h$mids are mid points of the intervals  
lines(xfit, yfit, col="blue",lwd=3)



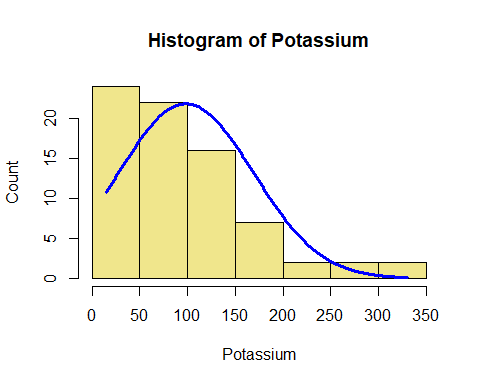
y<-cereals\_data2$Sugar  
h<-hist(y,  
 breaks=10,  
 col="lightgreen",  
 xlab = "Sugar",  
 ylab = "Count",  
 main = "Histogram of Sugar")  
xfit<-seq(min(y,na.rm = T),max(y,na.rm = T),length(40));xfit

## [1] 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15

yfit<-dnorm(xfit, mean = mean(y,na.rm = T), sd=sd(y,na.rm = T))  
yfit<-yfit \* diff(h$mids[1:2]\*length(y))  
lines(xfit, yfit, col="blue",lwd=3)



z<-cereals\_data2$Potassium  
h<-hist(z,  
 breaks=10,  
 col="khaki",  
 xlab = "Potassium",  
 ylab = "Count",  
 main = "Histogram of Potassium")  
xfitt<-seq(min(z,na.rm = T),max(z,na.rm = T),length(40))  
yfitt<-dnorm(xfitt, mean = mean(z,na.rm = T), sd=sd(z,na.rm = T))  
yfitt<-yfitt \* diff(h$mids[1:2]\*length(z))  
lines(xfitt, yfitt, col="blue",lwd=3)



#----kNN Imputation-----  
library(VIM)

## Loading required package: colorspace

## Loading required package: grid

## Loading required package: data.table

##   
## Attaching package: 'data.table'

## The following objects are masked from 'package:dplyr':  
##   
## between, first, last

## The following object is masked from 'package:purrr':  
##   
## transpose

## VIM is ready to use.   
## Since version 4.0.0 the GUI is in its own package VIMGUI.  
##   
## Please use the package to use the new (and old) GUI.

## Suggestions and bug-reports can be submitted at: https://github.com/alexkowa/VIM/issues

##   
## Attaching package: 'VIM'

## The following object is masked from 'package:datasets':  
##   
## sleep

cereals\_data2<-kNN(cereals\_data2, k=5)

## Warning in gowerD(don\_dist\_var, imp\_dist\_var, weights = weightsx, numericalX, :  
## NAs introduced by coercion  
  
## Warning in gowerD(don\_dist\_var, imp\_dist\_var, weights = weightsx, numericalX, :  
## NAs introduced by coercion  
  
## Warning in gowerD(don\_dist\_var, imp\_dist\_var, weights = weightsx, numericalX, :  
## NAs introduced by coercion  
  
## Warning in gowerD(don\_dist\_var, imp\_dist\_var, weights = weightsx, numericalX, :  
## NAs introduced by coercion  
  
## Warning in gowerD(don\_dist\_var, imp\_dist\_var, weights = weightsx, numericalX, :  
## NAs introduced by coercion  
  
## Warning in gowerD(don\_dist\_var, imp\_dist\_var, weights = weightsx, numericalX, :  
## NAs introduced by coercion  
  
## Warning in gowerD(don\_dist\_var, imp\_dist\_var, weights = weightsx, numericalX, :  
## NAs introduced by coercion  
  
## Warning in gowerD(don\_dist\_var, imp\_dist\_var, weights = weightsx, numericalX, :  
## NAs introduced by coercion  
  
## Warning in gowerD(don\_dist\_var, imp\_dist\_var, weights = weightsx, numericalX, :  
## NAs introduced by coercion  
  
## Warning in gowerD(don\_dist\_var, imp\_dist\_var, weights = weightsx, numericalX, :  
## NAs introduced by coercion  
  
## Warning in gowerD(don\_dist\_var, imp\_dist\_var, weights = weightsx, numericalX, :  
## NAs introduced by coercion  
  
## Warning in gowerD(don\_dist\_var, imp\_dist\_var, weights = weightsx, numericalX, :  
## NAs introduced by coercion

sum(is.na(cereals\_data2))

## [1] 0

variable.names(cereals\_data2)

## [1] "Manufacturer" "Type" "Calories"   
## [4] "Protein" "Fat" "Sodium"   
## [7] "Fiber" "Carbohydrates" "Sugar"   
## [10] "Potassium" "Vitamins" "Shelf"   
## [13] "Weight" "Cups" "Rating"   
## [16] "Manufacturer\_Name" "Calories\_cat" "Rating\_Cat"   
## [19] "Manufacturer\_imp" "Type\_imp" "Calories\_imp"   
## [22] "Protein\_imp" "Fat\_imp" "Sodium\_imp"   
## [25] "Fiber\_imp" "Carbohydrates\_imp" "Sugar\_imp"   
## [28] "Potassium\_imp" "Vitamins\_imp" "Shelf\_imp"   
## [31] "Weight\_imp" "Cups\_imp" "Rating\_imp"   
## [34] "Manufacturer\_Name\_imp" "Calories\_cat\_imp" "Rating\_Cat\_imp"

#--------Removing additional rows form 18 to 34 after kNN-------  
cereals\_data2<-select(cereals\_data2,-19:-36)  
dim(cereals\_data2)

## [1] 77 18

sum(is.na(cereals\_data2))

## [1] 0

variable.names(cereals\_data2)

## [1] "Manufacturer" "Type" "Calories"   
## [4] "Protein" "Fat" "Sodium"   
## [7] "Fiber" "Carbohydrates" "Sugar"   
## [10] "Potassium" "Vitamins" "Shelf"   
## [13] "Weight" "Cups" "Rating"   
## [16] "Manufacturer\_Name" "Calories\_cat" "Rating\_Cat"

#============Data Visualisation-II ===========  
  
#---Box plot of all nutrients----  
  
cereals\_data3<-cereals\_data2  
names(cereals\_data3)

## [1] "Manufacturer" "Type" "Calories"   
## [4] "Protein" "Fat" "Sodium"   
## [7] "Fiber" "Carbohydrates" "Sugar"   
## [10] "Potassium" "Vitamins" "Shelf"   
## [13] "Weight" "Cups" "Rating"   
## [16] "Manufacturer\_Name" "Calories\_cat" "Rating\_Cat"

cereals\_data3<- subset(cereals\_data3, select = c(Calories:Potassium))  
str(cereals\_data3)

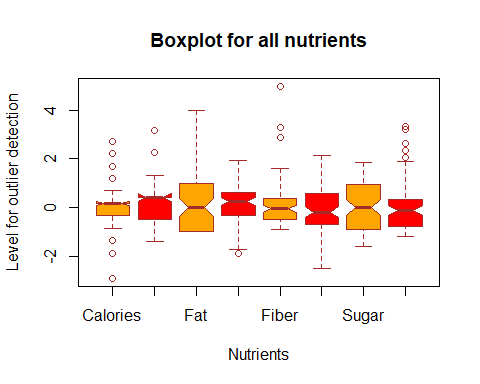
## 'data.frame': 77 obs. of 8 variables:  
## $ Calories : int 160 150 150 140 140 140 130 130 120 120 ...  
## $ Protein : int 3 4 4 3 3 3 3 3 1 3 ...  
## $ Fat : int 2 3 3 1 2 1 2 2 3 0 ...  
## $ Sodium : int 150 95 150 190 220 170 210 170 210 240 ...  
## $ Fiber : num 3 3 3 4 3 2 2 1.5 0 5 ...  
## $ Carbohydrates: num 17 16 16 15 21 20 18 13.5 13 14 ...  
## $ Sugar : int 13 11 11 14 7 9 8 10 9 12 ...  
## $ Potassium : int 160 170 170 230 130 95 100 120 45 190 ...

cereals\_data3<- scale(cereals\_data3)  
head(cereals\_data3,5)

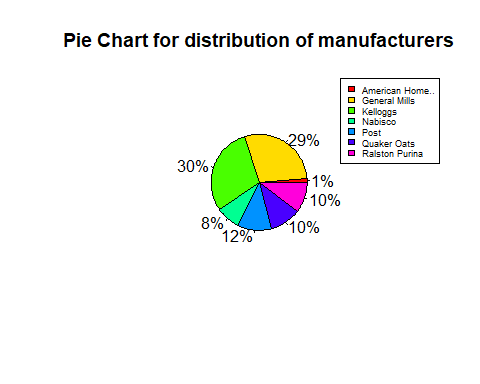
## Calories Protein Fat Sodium Fiber Carbohydrates Sugar  
## 1 2.726163 0.4151897 0.98066557 -0.1154129 0.3558214 0.56863582 1.37771570  
## 2 2.212924 1.3286071 1.97423464 -0.7714846 0.3558214 0.31107724 0.92045661  
## 3 2.212924 1.3286071 1.97423464 -0.1154129 0.3558214 0.31107724 0.92045661  
## 4 1.699686 0.4151897 -0.01290349 0.3617302 0.7753964 0.05351867 1.60634524  
## 5 1.699686 0.4151897 0.98066557 0.7195875 0.3558214 1.59887013 0.00593843  
## Potassium  
## 1 0.8865187  
## 2 1.0303795  
## 3 1.0303795  
## 4 1.8935441  
## 5 0.4549364

boxplot(cereals\_data3,  
 main = "Boxplot for all nutrients",  
 xlab = "Nutrients",  
 ylab = "Level for outlier detection",  
 col = c("orange","red","orange","red","orange","red","orange","red"),  
 border = "brown",  
 horizontal = F,  
 notch = TRUE)

## Warning in bxp(list(stats = structure(c(-0.866506555101683,  
## -0.353268057079917, : some notches went outside hinges ('box'): maybe set  
## notch=FALSE



#----Pie Chart for Manufacturer wise distribution-------  
  
cereals\_data4<- cereals\_data2%>%  
 count(Manufacturer\_Name) %>%  
 arrange(Manufacturer\_Name) %>%  
 mutate(prop = round(n \* 100 / sum(n), 0),  
 lab.ypos = (cumsum(prop) - (0.8\*prop)),   
 labl = paste0(prop, "", "%"))  
pie(cereals\_data4$n,   
 radius = 0.5,  
 labels = cereals\_data4$labl,   
 main = "Pie Chart for distribution of manufacturers",  
 col = rainbow(length(cereals\_data4$n)))  
legend("topright",as.vector(cereals\_data4$Manufacturer\_Name), cex = 0.55,  
 fill = rainbow(length(cereals\_data4$n)))



#-----------Manufacturer wise average content of nutrient----------  
  
head(cereals\_data2)

## Manufacturer Type Calories Protein Fat Sodium Fiber Carbohydrates Sugar  
## 1 K Cold 160 3 2 150 3 17 13  
## 2 R Cold 150 4 3 95 3 16 11  
## 3 R Cold 150 4 3 150 3 16 11  
## 4 G Cold 140 3 1 190 4 15 14  
## 5 K Cold 140 3 2 220 3 21 7  
## 6 K Cold 140 3 1 170 2 20 9  
## Potassium Vitamins Shelf Weight Cups Rating Manufacturer\_Name Calories\_cat  
## 1 160 25 3 1.50 0.67 30.31 Kelloggs High\_Calories  
## 2 170 25 3 1.00 1.00 37.14 Ralston Purina High\_Calories  
## 3 170 25 3 1.00 1.00 34.14 Ralston Purina High\_Calories  
## 4 230 100 3 1.50 1.00 28.59 General Mills High\_Calories  
## 5 130 25 3 1.33 0.67 40.69 Kelloggs High\_Calories  
## 6 95 100 3 1.30 0.75 36.47 Kelloggs High\_Calories  
## Rating\_Cat  
## 1 Poor  
## 2 Poor  
## 3 Poor  
## 4 Poor  
## 5 Satisfactory  
## 6 Poor

str(cereals\_data2)

## 'data.frame': 77 obs. of 18 variables:  
## $ Manufacturer : Factor w/ 7 levels "A","G","K","N",..: 3 7 7 2 3 3 2 2 2 3 ...  
## $ Type : Factor w/ 2 levels "Cold","Hot": 1 1 1 1 1 1 1 1 1 1 ...  
## $ Calories : int 160 150 150 140 140 140 130 130 120 120 ...  
## $ Protein : int 3 4 4 3 3 3 3 3 1 3 ...  
## $ Fat : int 2 3 3 1 2 1 2 2 3 0 ...  
## $ Sodium : int 150 95 150 190 220 170 210 170 210 240 ...  
## $ Fiber : num 3 3 3 4 3 2 2 1.5 0 5 ...  
## $ Carbohydrates : num 17 16 16 15 21 20 18 13.5 13 14 ...  
## $ Sugar : int 13 11 11 14 7 9 8 10 9 12 ...  
## $ Potassium : int 160 170 170 230 130 95 100 120 45 190 ...  
## $ Vitamins : int 25 25 25 100 25 100 25 25 25 25 ...  
## $ Shelf : Factor w/ 3 levels "1","2","3": 3 3 3 3 3 3 3 3 2 3 ...  
## $ Weight : num 1.5 1 1 1.5 1.33 1.3 1.33 1.25 1 1.33 ...  
## $ Cups : num 0.67 1 1 1 0.67 0.75 0.75 0.5 0.75 0.67 ...  
## $ Rating : num 30.3 37.1 34.1 28.6 40.7 ...  
## $ Manufacturer\_Name: chr "Kelloggs" "Ralston Purina" "Ralston Purina" "General Mills" ...  
## $ Calories\_cat : Factor w/ 3 levels "High\_Calories",..: 1 1 1 1 1 1 1 1 3 3 ...  
## $ Rating\_Cat : chr "Poor" "Poor" "Poor" "Poor" ...

hist\_data <- cereals\_data2  
hist\_data <- cereals\_data2[, c(3:11,16)]  
str(hist\_data)

## 'data.frame': 77 obs. of 10 variables:  
## $ Calories : int 160 150 150 140 140 140 130 130 120 120 ...  
## $ Protein : int 3 4 4 3 3 3 3 3 1 3 ...  
## $ Fat : int 2 3 3 1 2 1 2 2 3 0 ...  
## $ Sodium : int 150 95 150 190 220 170 210 170 210 240 ...  
## $ Fiber : num 3 3 3 4 3 2 2 1.5 0 5 ...  
## $ Carbohydrates : num 17 16 16 15 21 20 18 13.5 13 14 ...  
## $ Sugar : int 13 11 11 14 7 9 8 10 9 12 ...  
## $ Potassium : int 160 170 170 230 130 95 100 120 45 190 ...  
## $ Vitamins : int 25 25 25 100 25 100 25 25 25 25 ...  
## $ Manufacturer\_Name: chr "Kelloggs" "Ralston Purina" "Ralston Purina" "General Mills" ...

rownames(hist\_data) <- c()  
head(hist\_data)

## Calories Protein Fat Sodium Fiber Carbohydrates Sugar Potassium Vitamins  
## 1 160 3 2 150 3 17 13 160 25  
## 2 150 4 3 95 3 16 11 170 25  
## 3 150 4 3 150 3 16 11 170 25  
## 4 140 3 1 190 4 15 14 230 100  
## 5 140 3 2 220 3 21 7 130 25  
## 6 140 3 1 170 2 20 9 95 100  
## Manufacturer\_Name  
## 1 Kelloggs  
## 2 Ralston Purina  
## 3 Ralston Purina  
## 4 General Mills  
## 5 Kelloggs  
## 6 Kelloggs

hist\_data1 <- hist\_data %>%  
 group\_by(Manufacturer\_Name) %>%   
 summarise(Calories = round(mean(Calories),2),   
 Protein = round(mean(Protein),2),  
 Fat = round(mean(Fat),2),  
 Fiber = round(mean(Fiber),2),  
 Sugar = round(mean(Sugar),2),  
 Potassium = round(mean(Potassium),2),  
 )

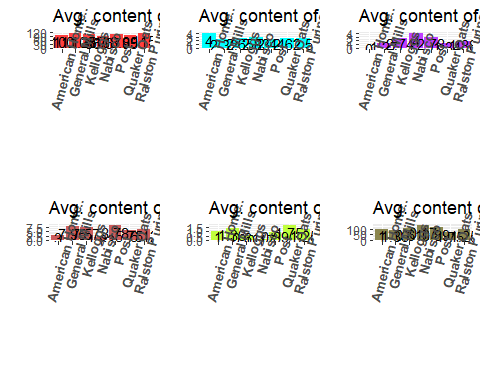
head(hist\_data1,20)

## # A tibble: 7 x 7  
## Manufacturer\_Name Calories Protein Fat Fiber Sugar Potassium  
## <chr> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>  
## 1 American Home.. 100 4 1 0 3 95   
## 2 General Mills 111. 2.32 1.36 1.27 7.95 85.2  
## 3 Kelloggs 109. 2.65 0.61 2.74 7.57 103.   
## 4 Nabisco 86.7 2.83 0.17 4 1.83 137.   
## 5 Post 109. 2.44 0.89 2.78 8.78 114.   
## 6 Quaker Oats 95 2.62 1.75 1.34 5.75 74.4  
## 7 Ralston Purina 115 2.5 1.25 1.88 6.12 99.4

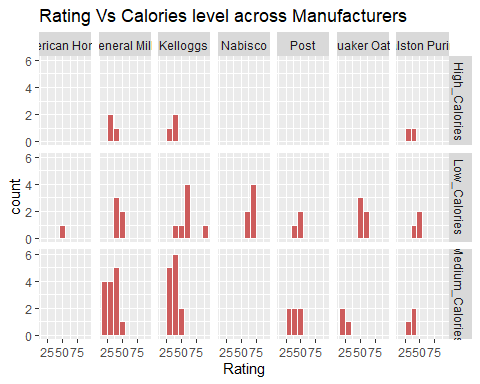
str(hist\_data1)

## tibble [7 x 7] (S3: tbl\_df/tbl/data.frame)  
## $ Manufacturer\_Name: chr [1:7] "American Home.." "General Mills" "Kelloggs" "Nabisco" ...  
## $ Calories : num [1:7] 100 111.4 108.7 86.7 108.9 ...  
## $ Protein : num [1:7] 4 2.32 2.65 2.83 2.44 2.62 2.5  
## $ Fat : num [1:7] 1 1.36 0.61 0.17 0.89 1.75 1.25  
## $ Fiber : num [1:7] 0 1.27 2.74 4 2.78 1.34 1.88  
## $ Sugar : num [1:7] 3 7.95 7.57 1.83 8.78 5.75 6.12  
## $ Potassium : num [1:7] 95 85.2 103 136.7 113.9 ...

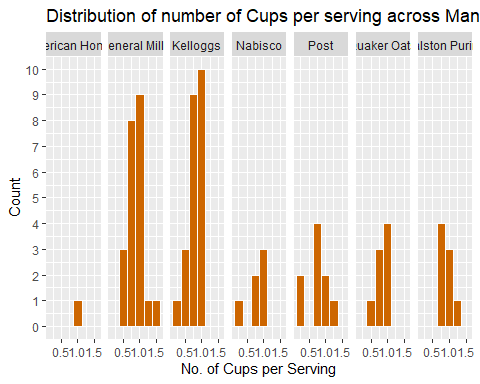
n1 <- ggplot(hist\_data1, aes(x = Manufacturer\_Name, y = Calories)) +  
 geom\_bar(stat = "identity", fill = "brown1") +  
 geom\_text(aes(label = Calories), position = position\_stack(vjust = 0.5)) +  
 theme(axis.text.x = element\_text(angle = 75, size = 10, face = "bold")) +  
 labs(title = "Avg. content of Calories (per serving)",  
 x = "",  
 y = "")  
n2 <- ggplot(hist\_data1, aes(x = Manufacturer\_Name, y = Protein)) +  
 geom\_bar(stat = "identity", fill = "cyan1") +  
 geom\_text(aes(label = Protein), position = position\_stack(vjust = 0.5)) +  
 theme(axis.text.x = element\_text(angle = 75, size = 10, face = "bold")) +  
 labs(title = "Avg. content of Protien (grams)",  
 x = "",  
 y = "")  
n3 <- ggplot(hist\_data1, aes(x = Manufacturer\_Name, y = Fiber)) +  
 geom\_bar(stat = "identity", fill = "darkorchid1") +  
 geom\_text(aes(label = Fiber), position = position\_stack(vjust = 0.5)) +  
 theme(axis.text.x = element\_text(angle = 75, size = 10, face = "bold")) +  
 labs(title = "Avg. content of Fiber (grams)",  
 x = "",  
 y = "")  
n4 <- ggplot(hist\_data1, aes(x = Manufacturer\_Name, y = Sugar)) +  
 geom\_bar(stat = "identity", fill = "indianred") +  
 geom\_text(aes(label = Sugar), position = position\_stack(vjust = 0.5)) +  
 theme(axis.text.x = element\_text(angle = 75, size = 10, face = "bold")) +  
 labs(title = "Avg. content of Sugar (grams)",  
 x = "",  
 y = "")  
n5 <- ggplot(hist\_data1, aes(x = Manufacturer\_Name, y = Fat)) +  
 geom\_bar(stat = "identity", fill = "olivedrab1") +  
 geom\_text(aes(label = Fat), position = position\_stack(vjust = 0.5)) +  
 theme(axis.text.x = element\_text(angle = 75, size = 10, face = "bold")) +  
 labs(title = "Avg. content of Fat (grams)",  
 x = "",  
 y = "")  
n6 <- ggplot(hist\_data1, aes(x = Manufacturer\_Name, y = Potassium )) +  
 geom\_bar(stat = "identity", fill = "khaki4") +  
 geom\_text(aes(label = Fat), position = position\_stack(vjust = 0.5)) +  
 theme(axis.text.x = element\_text(angle = 75, size = 10, face = "bold")) +  
 labs(title = "Avg. content of Potassium (milligrams)",  
 x = "",  
 y = "")  
  
figure <- ggarrange(n1,n2,n3,n4,n5,n6, ncol = 3, nrow = 2)  
figure



#-----------Histogram of Rating Vs Calories level across manufacturers---  
  
ggplot(cereals\_data2, aes(x = Rating)) +  
 geom\_histogram(color = "white",  
 fill = "indianred",  
 binwidth = 10) +  
 facet\_grid(Calories\_cat ~ Manufacturer\_Name) +  
 labs(title = "Rating Vs Calories level across Manufacturers",  
 x = "Rating")

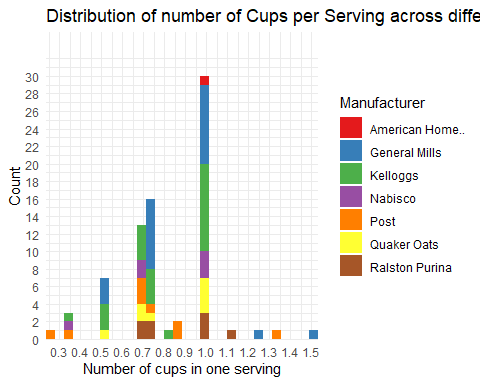


#------Histogram of Distribution of Cups per serving across manufacturers-I----  
ggplot(cereals\_data2, aes(x = Cups)) +  
 geom\_histogram(color = "white",  
 fill = "darkorange3",  
 binwidth = 0.25) +  
 scale\_x\_continuous(breaks = seq(0,1.5,0.5)) +  
 scale\_y\_continuous(breaks = seq(0,25,1)) +  
 facet\_grid(~ Manufacturer\_Name) +  
 labs(title = "Distribution of number of Cups per serving across Manufacturers",  
 x = "No. of Cups per Serving", y = "Count")

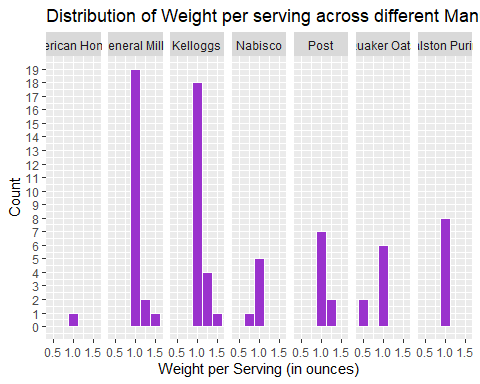


#------Histogram of Distribution of Cups per serving across manufacturers-II----  
cereals\_data2 %>%   
 ggplot(aes(x = Cups, fill = Manufacturer\_Name)) +  
 geom\_histogram() +  
 scale\_fill\_brewer(palette = "Set1") +  
 scale\_x\_continuous(name = "Number of cups in one serving", expand = c(0,0),breaks = seq(0,1.5,0.1)) +  
 scale\_y\_continuous(name = "Count", expand = c(0,0), limits = c(0, 35),breaks = seq(0,30,2)) +  
 labs(fill = "Manufacturer", title = "Distribution of number of Cups per Serving across different manufacturers") +  
 theme\_minimal()

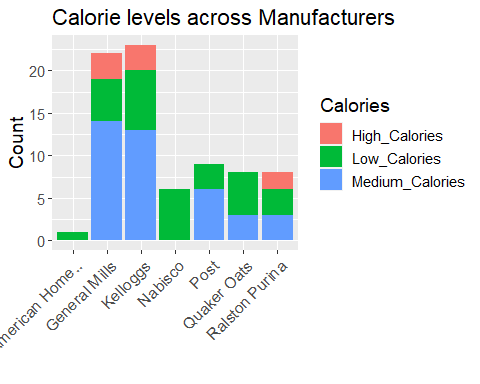
## `stat\_bin()` using `bins = 30`. Pick better value with `binwidth`.



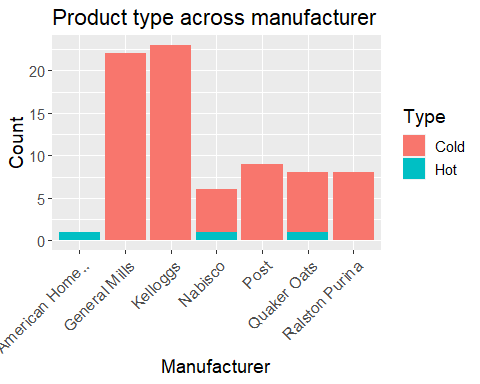
#------Histogram of Distribution of Weight per serving across manufacturers----  
ggplot(cereals\_data2, aes(x = Weight)) +  
 geom\_histogram(color = "white",  
 fill = "darkorchid3",  
 binwidth = 0.25) +  
 scale\_x\_continuous(breaks = seq(0,2,0.5)) +  
 scale\_y\_continuous(breaks = seq(0,25,1)) +  
 facet\_grid(~ Manufacturer\_Name) +  
 labs(title = "Distribution of Weight per serving across different Manufacturers",  
 x = "Weight per Serving (in ounces)", y = "Count")



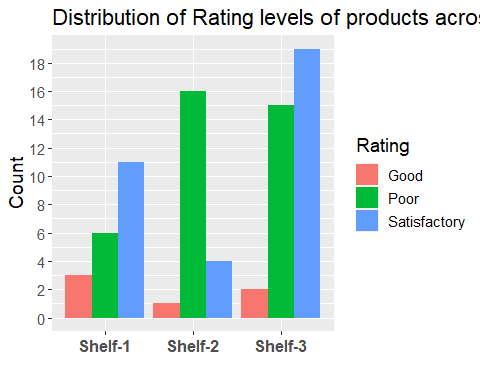
#---------Calorie level across Manufacturers-----  
ggplot(cereals\_data2,   
 aes(x = Manufacturer\_Name,   
 fill = Calories\_cat)) +   
 geom\_bar(position = "stack") +  
 labs(y = "Count",   
 fill = "Calories",  
 x = "",  
 title = "Calorie levels across Manufacturers") +   
 theme(axis.text.x= element\_text(angle = 45, hjust = 1, size = 12),  
 text = element\_text(size = 14))



#---------Product Type across Manufacturers-----  
ggplot(cereals\_data2,   
 aes(x = Manufacturer\_Name,   
 fill = Type)) +   
 geom\_bar(position = "stack") +  
 labs(y = "Count",   
 fill = "Type",  
 x = "Manufacturer",  
 title = "Product type across manufacturer") +   
 theme(axis.text.x= element\_text(angle = 45, hjust = 1,size = 12),  
 text = element\_text(size = 14))



#----------Rating category across Shelves----------  
ggplot(cereals\_data2,   
 aes(x = factor(Shelf, labels = c("Shelf-1","Shelf-2","Shelf-3")),  
 fill = Rating\_Cat)) +   
 geom\_bar(position = "dodge") +  
 labs(y = "Count",   
 fill = "Rating",  
 x = "",  
 title = "Distribution of Rating levels of products across Shelf numbers"  
 ) +   
 scale\_y\_continuous(breaks = seq(0, 25, 2)) +  
 theme(axis.text.x = element\_text( size = 12, face = "bold"),  
 text = element\_text(size = 14))

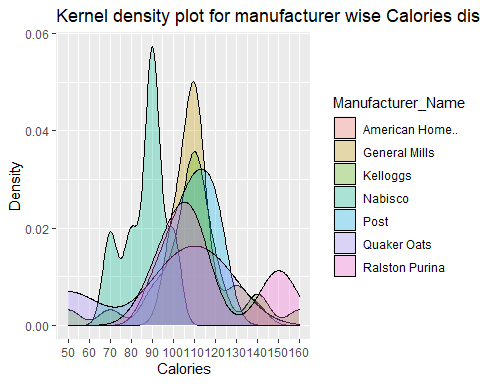


#-----Grouped kernel density plots of calorie distribution across manufacturers----  
table(cereals\_data2$Manufacturer)

##   
## A G K N P Q R   
## 1 22 23 6 9 8 8

ggplot(cereals\_data2,  
 aes(x = Calories,  
 fill = Manufacturer\_Name)) +  
 geom\_density(alpha = 0.3) +  
 scale\_x\_continuous(breaks = seq(50,180,10)) +  
 labs(title = "Kernel density plot for manufacturer wise Calories distribution",  
 x = "Calories",  
 y = "Density")

## Warning: Groups with fewer than two data points have been dropped.



#----Heatmap------  
  
names(cereals\_data2)

## [1] "Manufacturer" "Type" "Calories"   
## [4] "Protein" "Fat" "Sodium"   
## [7] "Fiber" "Carbohydrates" "Sugar"   
## [10] "Potassium" "Vitamins" "Shelf"   
## [13] "Weight" "Cups" "Rating"   
## [16] "Manufacturer\_Name" "Calories\_cat" "Rating\_Cat"

cereals\_datahm<-cereals\_data2[ ,3:11]  
rownames(cereals\_datahm)<-cereals\_data[,1]  
variable.names(cereals\_datahm)

## [1] "Calories" "Protein" "Fat" "Sodium"   
## [5] "Fiber" "Carbohydrates" "Sugar" "Potassium"   
## [9] "Vitamins"

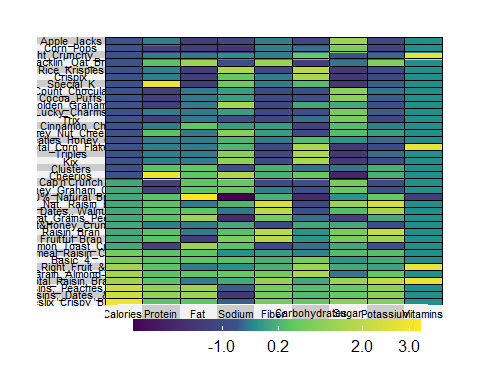
str(cereals\_datahm)

## 'data.frame': 77 obs. of 9 variables:  
## $ Calories : int 160 150 150 140 140 140 130 130 120 120 ...  
## $ Protein : int 3 4 4 3 3 3 3 3 1 3 ...  
## $ Fat : int 2 3 3 1 2 1 2 2 3 0 ...  
## $ Sodium : int 150 95 150 190 220 170 210 170 210 240 ...  
## $ Fiber : num 3 3 3 4 3 2 2 1.5 0 5 ...  
## $ Carbohydrates: num 17 16 16 15 21 20 18 13.5 13 14 ...  
## $ Sugar : int 13 11 11 14 7 9 8 10 9 12 ...  
## $ Potassium : int 160 170 170 230 130 95 100 120 45 190 ...  
## $ Vitamins : int 25 25 25 100 25 100 25 25 25 25 ...

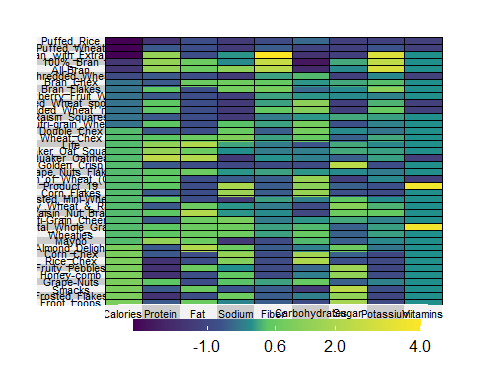
head(cereals\_datahm)

## Calories Protein Fat Sodium Fiber  
## Mueslix\_Crispy\_Blend 160 3 2 150 3  
## Muesli\_Raisins,\_Dates,\_&\_Almonds 150 4 3 95 3  
## Muesli\_Raisins,\_Peaches,\_&\_Pecans 150 4 3 150 3  
## Total\_Raisin\_Bran 140 3 1 190 4  
## Nutri-Grain\_Almond-Raisin 140 3 2 220 3  
## Just\_Right\_Fruit\_&\_Nut 140 3 1 170 2  
## Carbohydrates Sugar Potassium Vitamins  
## Mueslix\_Crispy\_Blend 17 13 160 25  
## Muesli\_Raisins,\_Dates,\_&\_Almonds 16 11 170 25  
## Muesli\_Raisins,\_Peaches,\_&\_Pecans 16 11 170 25  
## Total\_Raisin\_Bran 15 14 230 100  
## Nutri-Grain\_Almond-Raisin 21 7 130 25  
## Just\_Right\_Fruit\_&\_Nut 20 9 95 100

cereals\_datahm2<-cereals\_datahm[1:38, ]  
superheat(cereals\_datahm2,scale = TRUE, row.dendrogram = FALSE,  
 left.label.text.size=3,  
 bottom.label.text.size=3,  
 bottom.label.size = .05,)



cereals\_datahm3<-cereals\_datahm[39:77, ]  
superheat(cereals\_datahm3, scale = TRUE, row.dendrogram = FALSE,  
 left.label.text.size=3,  
 bottom.label.text.size=3,  
 bottom.label.size = .05,)



#===============cluster Analysis=============  
  
#--------Select required Variables from main data-------  
cereals\_cluster1<- select(cereals\_data2,Calories,Protein,Fat,Sugar,Rating)  
head(cereals\_cluster1)

## Calories Protein Fat Sugar Rating  
## 1 160 3 2 13 30.31  
## 2 150 4 3 11 37.14  
## 3 150 4 3 11 34.14  
## 4 140 3 1 14 28.59  
## 5 140 3 2 7 40.69  
## 6 140 3 1 9 36.47

cereals\_cluster2<-cereals\_cluster1  
dim(cereals\_cluster2)

## [1] 77 5

summary(cereals\_cluster2)

## Calories Protein Fat Sugar   
## Min. : 50.0 Min. :1.000 Min. :0.000 Min. : 0.000   
## 1st Qu.:100.0 1st Qu.:2.000 1st Qu.:0.000 1st Qu.: 3.000   
## Median :110.0 Median :3.000 Median :1.000 Median : 7.000   
## Mean :106.9 Mean :2.545 Mean :1.013 Mean : 6.974   
## 3rd Qu.:110.0 3rd Qu.:3.000 3rd Qu.:2.000 3rd Qu.:11.000   
## Max. :160.0 Max. :6.000 Max. :5.000 Max. :15.000   
## Rating   
## Min. :18.04   
## 1st Qu.:33.17   
## Median :40.40   
## Mean :42.67   
## 3rd Qu.:50.83   
## Max. :93.70

sum(is.na(cereals\_cluster2))

## [1] 0

cereals\_cluster2<-na.omit(cereals\_cluster2)  
#-------Scale the data-----  
cereals\_cluster2.scaled<-scale(cereals\_cluster2)  
head(cereals\_cluster2.scaled)

## Calories Protein Fat Sugar Rating  
## 1 2.726163 0.4151897 0.98066557 1.37771570 -0.8795515  
## 2 2.212924 1.3286071 1.97423464 0.92045661 -0.3933371  
## 3 2.212924 1.3286071 1.97423464 0.92045661 -0.6069012  
## 4 1.699686 0.4151897 -0.01290349 1.60634524 -1.0019949  
## 5 1.699686 0.4151897 0.98066557 0.00593843 -0.1406195  
## 6 1.699686 0.4151897 -0.01290349 0.46319752 -0.4410331

head(cereals\_cluster2.scaled,3)

## Calories Protein Fat Sugar Rating  
## 1 2.726163 0.4151897 0.9806656 1.3777157 -0.8795515  
## 2 2.212924 1.3286071 1.9742346 0.9204566 -0.3933371  
## 3 2.212924 1.3286071 1.9742346 0.9204566 -0.6069012

str(cereals\_cluster2.scaled)

## num [1:77, 1:5] 2.73 2.21 2.21 1.7 1.7 ...  
## - attr(\*, "dimnames")=List of 2  
## ..$ : chr [1:77] "1" "2" "3" "4" ...  
## ..$ : chr [1:5] "Calories" "Protein" "Fat" "Sugar" ...  
## - attr(\*, "scaled:center")= Named num [1:5] 106.88 2.55 1.01 6.97 42.67  
## ..- attr(\*, "names")= chr [1:5] "Calories" "Protein" "Fat" "Sugar" ...  
## - attr(\*, "scaled:scale")= Named num [1:5] 19.48 1.09 1.01 4.37 14.05  
## ..- attr(\*, "names")= chr [1:5] "Calories" "Protein" "Fat" "Sugar" ...

#---------Calculate the euclidean distance----  
library(cluster)  
library(factoextra)

## Welcome! Want to learn more? See two factoextra-related books at https://goo.gl/ve3WBa

dist.ecul\_cereals\_cluster2.scaled<-dist(cereals\_cluster2.scaled,method = "euclidean")  
(dist.ecul\_cereals\_cluster2.scaled)

## 1 2 3 4 5 6  
## 2 1.590727783   
## 3 1.538943990 0.213564151   
## 4 1.451929191 2.426390449 2.381807145   
## 5 1.865863965 1.727754107 1.771633915 2.071339915   
## 6 1.751992119 2.292994510 2.298491062 1.273367542 1.134245862   
## 7 1.976624165 1.829114470 1.840714195 1.869282031 0.619031639 1.142152013  
## 8 1.685605632 1.776020700 1.731022863 1.450678837 1.124832374 1.219229913  
## 9 3.151685720 3.407201794 3.335864855 3.168005603 2.791869302 3.121632498  
## 10 2.965956826 3.495459019 3.518781089 1.741525427 2.511913925 1.617464863  
## 11 2.379077462 2.688646686 2.709038996 1.356354434 1.832480353 1.250418340  
## 12 2.621687476 3.182770189 3.155329030 1.789898712 1.916265484 1.451001382  
## 13 3.263941285 2.479373596 2.540969664 3.425236958 1.626076705 2.598316165  
## 14 2.292498479 2.077718934 2.116001400 1.909874529 1.234652309 1.481033379  
## 15 2.354021322 2.761647968 2.773732800 1.219534090 2.154831590 1.539465290  
## 16 3.804507729 2.770346678 2.761221838 4.344827745 3.196672057 4.114877735  
## 17 2.849916591 3.471101353 3.410256303 2.465249285 2.649958175 2.582147337  
## 18 2.892607757 3.573222421 3.497537032 2.480180981 2.880560875 2.751351661  
## 19 4.872448904 3.834887167 3.894368346 4.710796333 3.503622729 3.904755088  
## 20 2.997174268 2.631792605 2.659147782 2.574108322 1.539853889 1.909256368  
## 21 3.746748975 3.855954630 3.870122642 3.178788828 2.244823611 2.264003591  
## 22 3.745189882 3.855606926 3.869265445 3.176594323 2.245268186 2.263216322  
## 23 3.742022822 3.854955661 3.867555265 3.172117923 2.246313082 2.261701349  
## 24 3.144655061 3.460589580 3.463005022 2.319383665 2.084976691 1.804916596  
## 25 2.836528464 3.039187583 3.016240006 1.799509911 2.073002008 1.603362257  
## 26 2.809512734 2.981010638 2.949581835 2.243405327 2.075795547 2.118960938  
## 27 3.315921008 4.021370997 3.991430565 2.433251081 2.974991720 2.561998262  
## 28 2.919602088 3.478809574 3.439650382 1.852477628 2.546886399 2.038691381  
## 29 3.458462591 4.096735090 4.052557989 2.670413932 2.889636468 2.553736016  
## 30 3.346678201 4.114881797 4.066941018 2.435929248 3.195283950 2.738562895  
## 31 3.350660756 4.120978201 4.071839902 2.440183722 3.205040119 2.747294486  
## 32 5.090110777 4.591040699 4.648597980 4.499287047 3.930523891 3.762344667  
## 33 4.241500216 4.501630308 4.539499074 3.495149864 2.861039909 2.573965991  
## 34 4.138595090 4.454347216 4.471108225 3.353087005 2.826694577 2.481689170  
## 35 3.158379189 2.437379244 2.467199325 3.097348831 1.832537910 2.570759155  
## 36 3.341195813 3.579021954 3.582758128 2.620899342 2.081496696 1.917162233  
## 37 3.751704940 4.546412022 4.546880158 2.676993991 3.329523594 2.677327886  
## 38 3.385574800 4.121513820 4.112392525 2.073291474 3.162539753 2.356736942  
## 39 2.902610766 3.439901182 3.424708175 1.823111703 2.537406274 2.033068107  
## 40 3.753257793 4.557726225 4.543694678 2.684554816 3.305558799 2.651892317  
## 41 2.936022808 3.537515140 3.518511192 1.814565160 2.826858441 2.286039358  
## 42 4.296050229 4.314376954 4.376408861 3.576952293 2.823223053 2.585937538  
## 43 3.754059741 4.578842336 4.555874534 2.676898556 3.349148597 2.684609205  
## 44 3.314885267 4.018105749 3.989208050 2.432842837 2.968880291 2.557241800  
## 45 4.570423643 4.996188991 5.015504893 3.890198183 3.312438764 3.067817890  
## 46 4.150234254 4.458259037 4.478028026 3.369745227 2.827197148 2.489917557  
## 47 2.968233234 3.008017977 3.001642892 2.498779476 1.859866713 2.065585264  
## 48 4.425313978 3.933052519 4.006620727 3.855962754 2.808662380 2.939852040  
## 49 4.241708164 3.971388125 4.031985394 3.635973248 2.576866525 2.693501617  
## 50 4.129399724 3.895261011 3.938037228 3.492020991 2.493744542 2.573442668  
## 51 3.788457945 3.901650410 3.919029557 3.011123881 2.467814250 2.363586291  
## 52 3.352091149 2.985047716 3.005654667 2.776525537 2.066847467 2.303680155  
## 53 3.456800227 3.732067670 3.734266804 2.485404940 2.570924434 2.258683045  
## 54 4.392932147 4.406861438 4.484524253 3.500197516 3.121512778 2.799461770  
## 55 4.670779810 4.840104990 4.872101194 3.882613824 3.231085138 3.007368622  
## 56 4.392416473 4.443647184 4.463651194 3.517282706 3.000501039 2.685477759  
## 57 5.310353632 5.141110489 5.225845276 4.689661195 3.688477583 3.664142650  
## 58 4.027060552 3.792268916 3.857613932 3.353339575 2.463398833 2.507155773  
## 59 3.820824706 4.434155004 4.432817624 2.512605215 3.534546081 2.815216974  
## 60 4.492353737 3.563987163 3.628205918 4.167801013 2.984853729 3.386135574  
## 61 3.966595406 3.551942876 3.610821983 3.257637630 2.546101712 2.526088084  
## 62 3.744571049 3.036319192 3.084456605 3.286582588 2.282614572 2.627620517  
## 63 4.197651523 3.940142048 3.994367328 3.580099466 2.541220672 2.644905157  
## 64 4.313633642 4.577230584 4.606003876 3.394943288 3.045307991 2.680588979  
## 65 5.247890638 5.099154195 5.170215454 4.470507154 3.696036542 3.585308240  
## 66 4.839837973 4.968743927 5.028629916 3.921199001 3.536499825 3.268010173  
## 67 5.964268219 5.708653929 5.811145446 5.337151958 4.344893717 4.373136943  
## 68 5.902471157 5.654290964 5.753338968 5.265239619 4.280241374 4.300611906  
## 69 5.031140477 5.107709392 5.177831184 4.175995989 3.652935133 3.449413058  
## 70 4.783534220 4.733779760 4.790193362 3.860478969 3.398567058 3.137834371  
## 71 4.368424692 4.336248272 4.383255385 3.591796793 2.969724654 2.949757725  
## 72 6.145428114 6.029562987 6.111209329 5.432929190 4.546831055 4.544604513  
## 73 5.549838876 5.020642107 5.092172047 4.774468430 4.088715893 4.153247645  
## 74 5.750812618 5.202510572 5.297379628 5.011837414 4.321144458 4.402128368  
## 75 8.115025907 7.600177917 7.715445661 7.408573276 6.551451381 6.632513893  
## 76 7.135045482 6.948365489 7.007995932 6.277512816 5.586190437 5.564313531  
## 77 7.258912160 7.203272472 7.256097013 6.414894744 5.764127807 5.734543967  
## 7 8 9 10 11 12  
## 2   
## 3   
## 4   
## 5   
## 6   
## 7   
## 8 0.655109249   
## 9 2.478491163 2.283162018   
## 10 2.264674415 2.233251028 3.869112086   
## 11 1.453242094 1.361255749 3.109928265 1.001437624   
## 12 1.547288600 1.462401989 2.302174904 1.707732444 1.321688981   
## 13 1.573757035 2.080383529 2.840293746 3.513725100 2.740713519 2.714663871  
## 14 0.740804558 0.904955142 2.575458048 2.039081779 1.100104550 1.577015742  
## 15 1.770765221 1.537431814 3.199754379 1.116920928 0.468299543 1.568119589  
## 16 3.032405220 3.069244793 2.890402171 5.076120163 4.095423755 4.094509638  
## 17 2.288540016 2.006509217 1.103432160 3.032617433 2.430909653 1.535903259  
## 18 2.503305055 2.142498018 1.213951946 3.156297937 2.570301284 1.734124065  
## 19 3.475324734 3.858373800 5.505044379 4.304268304 3.969210425 4.488453613  
## 20 1.078489329 1.423311990 2.635028687 2.349652412 1.601238893 1.688372648  
## 21 2.050970604 2.414129670 2.973123705 2.516936662 2.309055235 1.607912483  
## 22 2.050284692 2.411747478 2.968831787 2.517419543 2.309079487 1.604115973  
## 23 2.048992932 2.406906015 2.959990374 2.518530814 2.309248363 1.596373495  
## 24 1.696710726 1.803114401 2.541017780 1.743414134 1.407782483 0.717574192  
## 25 1.559018648 1.429259441 2.871082405 1.400499160 0.901345530 1.075508552  
## 26 1.544155940 1.375668643 1.616489075 2.434527350 1.666061927 1.141804098  
## 27 2.579121937 2.371537381 2.236349514 2.341006484 2.066909192 1.261765345  
## 28 2.061598938 1.776057745 2.399766894 1.766289205 1.376003124 0.886243075  
## 29 2.513721091 2.377927542 2.071812421 2.561462511 2.298353891 1.134704454  
## 30 2.778713447 2.479889157 2.256476184 2.516667925 2.244114327 1.481668881  
## 31 2.787671598 2.485358164 2.254375323 2.529268134 2.256892529 1.490191676  
## 32 3.889745353 4.204689341 6.124624527 3.570655296 3.737442984 4.383112936  
## 33 2.763362903 3.125595780 3.947206578 2.346688746 2.571908904 2.143244853  
## 34 2.684451961 2.985965583 3.747533709 2.309286987 2.515447876 1.925118895  
## 35 1.466981305 1.737254034 2.442670418 3.233645368 2.350764736 2.411942023  
## 36 1.756593874 1.974393493 2.632536390 2.017255352 1.737113847 0.977040256  
## 37 3.030525922 2.948347515 3.302928218 1.933878756 2.156221448 1.652087138  
## 38 2.791841004 2.590471186 3.491752027 1.272445552 1.575424783 1.615830420  
## 39 2.073686893 1.833386727 2.580845931 1.590765271 1.184016822 1.061288359  
## 40 2.994839865 2.897746828 3.168814189 2.029312536 2.224880768 1.518459403  
## 41 2.368023416 2.045735999 2.754631583 1.743869443 1.376575697 1.467611452  
## 42 2.767773101 3.197794826 4.480110096 2.295723603 2.548240354 2.598741910  
## 43 3.026426613 2.899446217 3.124153761 2.101717733 2.280578590 1.516926753  
## 44 2.574082919 2.370005156 2.241464092 2.333034353 2.059088707 1.259480361  
## 45 3.216542502 3.515838634 3.768609454 2.971994789 3.138966711 2.320720444  
## 46 2.691105518 3.001866524 3.772944443 2.309257690 2.518574780 1.950916664  
## 47 1.387026911 1.474904387 1.792072592 2.471080958 1.744119615 1.184765858  
## 48 2.665799735 3.125806356 4.540016998 2.842262404 2.711766865 3.069589149  
## 49 2.395316871 2.860764855 3.916185182 2.615533606 2.461318080 2.479821158  
## 50 2.265762747 2.692735597 3.724539841 2.536932440 2.359057723 2.278130687  
## 51 2.109285052 2.345522795 2.897145990 2.182002592 1.942529452 1.431891211  
## 52 1.551315923 1.735919944 2.726403227 2.418170072 1.696513815 1.847202857  
## 53 2.098825529 2.087731587 2.691719517 1.789683362 1.464631978 1.142155419  
## 54 2.945005069 3.276379600 4.583453724 1.970388303 2.279133067 2.679613193  
## 55 3.070794278 3.420902412 4.095148829 2.689581387 2.884971993 2.426883951  
## 56 2.779754287 3.082130311 4.187997043 2.299741636 2.510030150 2.331631233  
## 57 3.673486930 4.175223975 5.257367533 3.373690476 3.578274229 3.630365821  
## 58 2.230398409 2.652483068 3.801325529 2.285178437 2.108993118 2.283182406  
## 59 3.119505184 2.928706284 3.723026541 1.589701006 1.851242504 2.016575379  
## 60 2.824630559 3.220901724 4.705863148 3.614054049 3.207887331 3.694231596  
## 61 2.278016330 2.620993019 4.177198815 2.263232603 2.074171764 2.608880979  
## 62 1.939695023 2.272326762 3.649402933 2.795105722 2.223434064 2.655565723  
## 63 2.342757829 2.795482044 3.843607480 2.581588258 2.418586237 2.402204827  
## 64 2.809543861 3.071950833 3.827967995 2.122453998 2.359392617 1.982292682  
## 65 3.554364785 3.978279720 5.018917895 3.058574488 3.269542316 3.351083486  
## 66 3.301823119 3.601529672 4.353197649 2.474788826 2.717117060 2.664662003  
## 67 4.313773115 4.817732648 5.827698970 3.945645771 4.143391565 4.299322834  
## 68 4.241370356 4.741264379 5.749016379 3.875057550 4.072573380 4.212391953  
## 69 3.463058001 3.813136716 4.566067801 2.733298250 2.966640123 2.930277232  
## 70 3.158366193 3.481152325 4.594875697 2.386938995 2.630511262 2.793044543  
## 71 2.642864663 2.939205751 3.460465160 2.531106837 2.362098965 2.170100951  
## 72 4.432483111 4.881874313 5.481683174 4.041147644 4.221292921 4.108207547  
## 73 3.783565498 4.106979945 5.179809487 3.561706847 3.470256427 3.895151170  
## 74 4.061693895 4.409004329 5.520274725 3.753362589 3.688450289 4.230480534  
## 75 6.423094759 6.866038237 7.822778682 5.944122771 6.105342073 6.486968663  
## 76 5.329614785 5.678138579 6.026064053 4.870241574 5.011426702 4.865938051  
## 77 5.508284846 5.833299420 5.875358010 5.074093452 5.205903617 4.876753867  
## 13 14 15 16 17 18  
## 2   
## 3   
## 4   
## 5   
## 6   
## 7   
## 8   
## 9   
## 10   
## 11   
## 12   
## 13   
## 14 1.729199996   
## 15 3.081845571 1.368063441   
## 16 2.343989308 3.055778426 4.213330267   
## 17 3.128937092 2.286633286 2.467271497 3.665457467   
## 18 3.402643298 2.489848177 2.553496996 3.787599962 0.355822453   
## 19 3.058462156 3.535119739 4.294111412 4.543702032 5.529786729 5.733415876  
## 20 1.367245237 0.857453958 1.960894474 3.067437782 2.485386459 2.727863817  
## 21 2.305966811 2.158826007 2.726265865 4.282477760 2.638052486 2.931988732  
## 22 2.307861627 2.159358461 2.725943241 4.281678499 2.633727329 2.927235931  
## 23 2.311910424 2.160590569 2.725373467 4.280081957 2.624825797 2.917433790  
## 24 2.520277238 1.551576766 1.730119904 4.113072845 1.895935235 2.142590759  
## 25 2.682342252 1.319953749 1.154134483 4.038599453 2.251471396 2.378530990  
## 26 2.305026006 1.325708995 1.809119365 3.208224482 1.202421902 1.404829956  
## 27 3.542279333 2.382435337 2.079846582 4.519806193 1.215762612 1.314686429  
## 28 3.199488863 1.820557748 1.390090850 4.242090686 1.502296244 1.570866139  
## 29 3.363221597 2.474977946 2.430326427 4.469123434 1.215958855 1.374471484  
## 30 3.805341328 2.594862659 2.192830506 4.619809902 1.209758194 1.189458303  
## 31 3.815849723 2.607018884 2.204796362 4.624063484 1.208760789 1.182878013  
## 32 4.120779567 3.877877633 4.033507117 5.968008329 5.776497238 5.961908381  
## 33 3.168698933 2.790876092 2.970816471 5.284880162 3.400996877 3.674420956  
## 34 3.189719657 2.758336779 2.906417974 5.225268780 3.187801700 3.442463885  
## 35 0.937790527 1.312309386 2.609205935 2.115784315 2.678381699 2.905127299  
## 36 2.385983136 1.737636180 2.109964245 4.139371665 2.116450330 2.386919112  
## 37 4.042344312 2.809346997 2.195110458 5.397498426 2.290185640 2.409769184  
## 38 4.002396519 2.487405794 1.481703878 5.259499034 2.482741587 2.532911349  
## 39 3.196526382 1.714592435 1.144835430 4.267936307 1.683992505 1.776159251  
## 40 4.011624202 2.838515141 2.294762815 5.365013003 2.162466191 2.274723345  
## 41 3.528256276 1.966609309 1.171083557 4.415002109 1.834461943 1.853923724  
## 42 3.080574639 2.749353706 2.966088785 5.306130034 3.990400301 4.255472042  
## 43 4.064888550 2.890249071 2.340517301 5.374924877 2.109813558 2.201047149  
## 44 3.535093649 2.374662052 2.073047121 4.517893869 1.222768616 1.325275085  
## 45 3.573234412 3.301627183 3.493237801 5.521590181 3.240316951 3.514053783  
## 46 3.182918099 2.758475763 2.911325396 5.231329024 3.214761013 3.472422079  
## 47 1.892971808 1.234336361 2.006829935 3.159615691 1.536648916 1.813050598  
## 48 2.510601417 2.533739651 3.111130667 4.604012743 4.290369490 4.544278825  
## 49 2.285596088 2.275762054 2.887320910 4.441495307 3.633147093 3.913618266  
## 50 2.249067781 2.183828357 2.787962386 4.355469744 3.440622421 3.709872735  
## 51 2.492075642 1.927390539 2.293354291 4.252337467 2.422390760 2.688002532  
## 52 1.751104602 1.127068640 1.984978521 3.178691351 2.544111389 2.757695009  
## 53 2.861545655 1.728879918 1.654779296 4.232785358 1.991278243 2.179871492  
## 54 3.347480188 2.648097425 2.594982053 5.366109191 3.990004956 4.228597113  
## 55 3.313864857 3.050532636 3.275395594 5.400414673 3.603786707 3.871180639  
## 56 3.175638628 2.750523536 2.904060539 5.227476596 3.692947191 3.919544185  
## 57 3.542674706 3.613092850 3.986988436 5.814486151 4.887453289 5.178783145  
## 58 2.292132618 1.994689389 2.513019664 4.356513726 3.447459117 3.716390705  
## 59 4.202686190 2.703037781 1.720881232 5.397865250 2.753233853 2.794303328  
## 60 2.357570861 2.729464289 3.543746360 4.002652552 4.690261127 4.907718128  
## 61 2.472886476 2.021249153 2.433830332 4.371977328 3.837733890 4.055919066  
## 62 1.756536174 1.680145156 2.550458876 3.410942152 3.557887451 3.770753011  
## 63 2.266036064 2.236258131 2.846434340 4.407042737 3.560051937 3.836762431  
## 64 3.291804002 2.683732326 2.706031866 5.251741195 3.205449418 3.449731176  
## 65 3.526148866 3.382417170 3.646009711 5.680560121 4.595677801 4.863531585  
## 66 3.571840921 3.007062164 3.014602848 5.513947535 3.759555278 4.010294230  
## 67 4.031662285 4.153413984 4.516944125 6.221240927 5.477483998 5.772459250  
## 68 3.972823677 4.086209193 4.449371779 6.167059945 5.396910578 5.690366519  
## 69 3.615576226 3.191200584 3.282247094 5.623107762 4.019732940 4.285725125  
## 70 3.404801161 2.898997697 2.967272820 5.423486804 4.067906793 4.303742372  
## 71 2.723670614 2.317182095 2.682394466 4.513384295 3.043375413 3.311989221  
## 72 4.161672065 4.243499388 4.578726894 6.246763841 5.120454210 5.412546957  
## 73 3.515786176 3.383777549 3.743778774 5.193012337 4.910459103 5.130741941  
## 74 3.763454139 3.615259737 3.936734576 5.424718164 5.230693380 5.460377265  
## 75 5.923274410 6.085623486 6.386474599 7.735660696 7.539036163 7.803973867  
## 76 4.996178941 5.038885284 5.308912460 6.785076053 5.708968067 5.955325928  
## 77 5.205594153 5.235957964 5.489918355 6.921263068 5.553868026 5.799284275  
## 19 20 21 22 23 24  
## 2   
## 3   
## 4   
## 5   
## 6   
## 7   
## 8   
## 9   
## 10   
## 11   
## 12   
## 13   
## 14   
## 15   
## 16   
## 17   
## 18   
## 19   
## 20 3.151931670   
## 21 3.901039395 1.632382759   
## 22 3.902995375 1.632877076 0.009254447   
## 23 3.907124707 1.634070736 0.028475220 0.019220774   
## 24 4.239526767 1.401286337 1.163584700 1.161893404 1.158609041   
## 25 3.833391831 1.377956373 1.932330011 1.929564724 1.923950979 1.084557484  
## 26 4.457767493 1.380495674 2.007052304 2.003877299 1.997403943 1.192644128  
## 27 5.554590530 2.538157547 2.395279449 2.392135013 2.385705729 1.425365950  
## 28 4.856711651 2.018737293 2.242117423 2.238457696 2.230960208 1.135721587  
## 29 5.373479326 2.435170780 1.980964571 1.975844717 1.965307804 1.290179051  
## 30 5.775061841 2.790480548 2.727850803 2.723878678 2.715711085 1.748680821  
## 31 5.783466414 2.801468361 2.738360068 2.734321290 2.726014330 1.762049937  
## 32 2.046037342 3.621503169 3.913337490 3.915686094 3.920629269 4.134915245  
## 33 4.193162589 2.415264941 1.133371997 1.137853521 1.147343899 1.695658157  
## 34 4.246687266 2.370553209 0.998002763 0.998916617 1.001085353 1.546203188  
## 35 3.304030150 0.993575441 2.372090315 2.372444397 2.373294939 2.219749546  
## 36 4.083000300 1.396450520 0.712696551 0.710238069 0.705492603 0.457862153  
## 37 5.680585501 2.949745388 2.473088903 2.472184344 2.470415362 1.630553958  
## 38 5.263069150 2.758488107 2.738384106 2.736939029 2.734035371 1.707387525  
## 39 4.858744061 2.010772943 2.340427111 2.338465714 2.334504003 1.177735845  
## 40 5.650287114 2.920485054 2.339915816 2.337737004 2.333322562 1.551223676  
## 41 5.149235877 2.364957795 2.802184521 2.800315994 2.796529074 1.638894883  
## 42 3.420717439 2.374354855 1.683246797 1.688793440 1.700416518 2.150361067  
## 43 5.700122394 2.968395901 2.392823225 2.389948473 2.384081547 1.604127862  
## 44 5.548745120 2.531154121 2.388546230 2.385470249 2.379183989 1.417079124  
## 45 5.024754615 2.933564969 1.382890559 1.384235728 1.387222658 1.968337060  
## 46 4.236314085 2.371704022 1.005947727 1.007436547 1.010792476 1.560185362  
## 47 4.155755033 1.034533555 1.553597168 1.551562506 1.547504969 1.001889168  
## 48 2.209474622 1.994849504 2.199001042 2.203692164 2.213527127 2.583038920  
## 49 2.995355124 1.649696697 1.367731420 1.373698435 1.386206814 1.899051518  
## 50 3.008961588 1.511796696 1.173352688 1.177547815 1.186444042 1.720483032  
## 51 4.059738529 1.462060958 0.858890415 0.859607301 0.861412055 0.741870964  
## 52 3.309622602 0.564064195 1.841951039 1.842138805 1.842677236 1.465387193  
## 53 4.462192869 1.626535002 1.694747177 1.693582519 1.691322858 0.687386454  
## 54 3.727576880 2.417637217 2.184483037 2.190258284 2.202328895 2.150764536  
## 55 4.211339570 2.550358240 1.234890809 1.238452077 1.246035670 1.899011523  
## 56 3.516354082 2.248245353 1.452862196 1.453916094 1.456290449 1.880053658  
## 57 3.568514229 3.118358442 2.407615575 2.414543603 2.428982108 3.082285048  
## 58 3.099152917 1.466639753 1.463668922 1.469466059 1.481618494 1.687528442  
## 59 5.387553864 2.919973652 2.976300326 2.975431398 2.973717917 1.953553956  
## 60 1.143177939 2.233467266 3.072509554 3.075007598 3.080278213 3.339540142  
## 61 2.429531570 1.599305571 2.146087463 2.149257760 2.155954261 2.170012743  
## 62 2.248765791 1.128356078 2.290483486 2.292253150 2.296043436 2.284811002  
## 63 2.995568291 1.591781984 1.289107077 1.294520571 1.305901837 1.828075688  
## 64 4.313618173 2.309506361 1.261338413 1.264028039 1.269811507 1.434162747  
## 65 3.600449966 2.860965389 2.244198602 2.250198279 2.262729258 2.746579834  
## 66 4.445705827 2.649263958 1.955314531 1.960750096 1.972130240 2.026433733  
## 67 3.925879153 3.666896221 3.104101704 3.111583600 3.127153237 3.694129532  
## 68 3.876253247 3.589370988 3.008865091 3.016218467 3.031524141 3.607324452  
## 69 4.422988560 2.804915137 2.073985029 2.080386959 2.093751432 2.287331974  
## 70 3.657957282 2.460912268 2.021753216 2.026353995 2.036010650 2.197938770  
## 71 4.087854304 1.820104404 1.420838658 1.425442438 1.435147636 1.452580549  
## 72 4.611882928 3.693630304 2.844347830 2.851071898 2.865082363 3.450923199  
## 73 3.123524413 2.842613616 3.134776622 3.139030535 3.147934151 3.280161895  
## 74 3.379768898 3.171988235 3.511625703 3.517104247 3.528533140 3.608147644  
## 75 5.115205485 5.587855944 5.413705411 5.420336620 5.434133633 5.823556716  
## 76 5.252967117 4.406578684 3.715354026 3.719578043 3.728409110 4.178604531  
## 77 5.903299239 4.629359360 3.757894577 3.761676794 3.769592640 4.208217821  
## 25 26 27 28 29 30  
## 2   
## 3   
## 4   
## 5   
## 6   
## 7   
## 8   
## 9   
## 10   
## 11   
## 12   
## 13   
## 14   
## 15   
## 16   
## 17   
## 18   
## 19   
## 20   
## 21   
## 22   
## 23   
## 24   
## 25   
## 26 1.354194843   
## 27 1.897964664 1.430487513   
## 28 1.067179042 1.111499380 0.916298975   
## 29 1.912443854 1.427939545 0.741291176 1.161154849   
## 30 2.039463990 1.588780411 0.423641607 0.983504809 0.917626073   
## 31 2.046632574 1.596289092 0.444210326 0.990766031 0.919932751 0.024203937  
## 32 3.677451018 4.763048909 5.416992454 4.701327564 5.299693600 5.634973851  
## 33 2.377547434 2.835941704 2.813174988 2.698679477 2.531299988 3.163340801  
## 34 2.199821316 2.669981920 2.624327706 2.488065355 2.264243643 2.942436101  
## 35 2.205684269 1.702492064 3.067605834 2.654011866 2.983290181 3.279829637  
## 36 1.349518187 1.439636121 1.762350326 1.538663610 1.457644173 2.087567497  
## 37 2.155461419 2.278456200 1.146277377 1.495517060 1.478156473 1.378831867  
## 38 1.637132569 2.202940500 1.476302344 1.185935378 1.890298725 1.557265462  
## 39 1.145146536 1.222524649 0.993684172 0.452169971 1.424467041 1.135257982  
## 40 2.092240445 2.203224358 1.052831321 1.409328035 1.221502039 1.256917852  
## 41 1.463300307 1.519525782 1.168822867 0.757010517 1.730857650 1.186886814  
## 42 2.463456337 3.198345313 3.447445182 3.106770201 3.261507584 3.782352210  
## 43 2.098639691 2.199619380 1.021967635 1.376319899 1.148884861 1.174182246  
## 44 1.895585724 1.428879657 0.019932654 0.918093577 0.749079272 0.440553708  
## 45 2.876681369 2.986467979 2.691355420 2.868852901 2.286289384 3.031813421  
## 46 2.220097572 2.689329834 2.647010318 2.513809443 2.298259833 2.970301500  
## 47 1.444340455 0.573826560 1.697237873 1.456056218 1.562384938 1.953182890  
## 48 2.554367708 3.225413849 3.965877277 3.441215369 3.813133621 4.269033271  
## 49 2.226789256 2.667748647 3.273370577 2.909243160 3.065329645 3.611907908  
## 50 2.014055593 2.477248970 3.105944469 2.710104170 2.851457299 3.424540211  
## 51 1.532394299 1.629830616 1.937450476 1.747122053 1.707714361 2.277733392  
## 52 1.355399782 1.353899985 2.479505559 1.942618750 2.433385507 2.711530512  
## 53 1.109087187 1.214928539 1.291111651 0.961801095 1.388063565 1.575941673  
## 54 2.343595386 3.119611426 3.261891003 2.908360344 3.293383757 3.591093044  
## 55 2.557114872 2.985335194 2.995688281 2.886574782 2.666910018 3.334464400  
## 56 2.088824701 2.887265765 3.127303449 2.724785384 2.838549498 3.405670738  
## 57 3.485891019 4.060626357 4.355512204 4.105240983 4.150776513 4.719334282  
## 58 1.951205295 2.444021709 3.026860303 2.629769920 2.913768870 3.362826609  
## 59 1.865544652 2.384510063 1.685356962 1.445304682 2.151996895 1.757017938  
## 60 3.021250993 3.554883932 4.640409354 3.992111901 4.492689970 4.881160818  
## 61 1.912404234 2.729717474 3.471848308 2.848210912 3.404795872 3.737031694  
## 62 1.987908584 2.388597429 3.493932621 2.867564519 3.402810979 3.729207053  
## 63 2.144911627 2.594316465 3.208806760 2.833120703 2.984236666 3.540616174  
## 64 2.069482929 2.575258819 2.457504753 2.319844366 2.248827040 2.791507458  
## 65 3.085891151 3.712500094 3.969767673 3.686599871 3.801184064 4.313153251  
## 66 2.587298721 3.035990423 2.934577727 2.840389895 2.895454225 3.289217826  
## 67 4.100411062 4.616035730 4.895600075 4.685315212 4.756396580 5.270035130  
## 68 4.011596340 4.534414541 4.815624495 4.599810109 4.666882142 5.187659294  
## 69 2.870154318 3.290080349 3.240387642 3.161936635 3.181152870 3.607838800  
## 70 2.419548279 3.164500089 3.354144347 3.002799828 3.260014302 3.668713897  
## 71 2.091748486 2.196813029 2.466592454 2.340074041 2.367988363 2.824172416  
## 72 4.052285045 4.375872681 4.475218409 4.429121145 4.293549726 4.849494657  
## 73 3.229809206 3.792437682 4.401249818 3.941082274 4.358144551 4.678775823  
## 74 3.598240538 4.127963032 4.688221422 4.270044868 4.719596173 4.985008095  
## 75 6.034056907 6.548051051 6.895490024 6.638011890 6.853762953 7.233991153  
## 76 4.658484221 4.934169590 5.013187867 4.966640533 4.830078496 5.325082009  
## 77 4.847859179 4.943890647 4.849667886 4.969924140 4.650462907 5.160375206  
## 31 32 33 34 35 36  
## 2   
## 3   
## 4   
## 5   
## 6   
## 7   
## 8   
## 9   
## 10   
## 11   
## 12   
## 13   
## 14   
## 15   
## 16   
## 17   
## 18   
## 19   
## 20   
## 21   
## 22   
## 23   
## 24   
## 25   
## 26   
## 27   
## 28   
## 29   
## 30   
## 31   
## 32 5.644310584   
## 33 3.176565396 3.680488422   
## 34 2.952951890 3.761652418 0.451332238   
## 35 3.289209333 4.247910554 3.281176742 3.248900051   
## 36 2.099049933 4.025552523 1.415501024 1.241102859 2.216662236   
## 37 1.395241842 5.159241776 2.386413304 2.276864074 3.693135423 1.925112669  
## 38 1.568949960 4.657589959 2.698181199 2.569359365 3.542432881 2.095095744  
## 39 1.149795949 4.667066022 2.703317324 2.562744963 2.647434501 1.629552325  
## 40 1.269018936 5.156336482 2.321842720 2.145043458 3.670106776 1.805291563  
## 41 1.199392849 4.928905601 3.123883574 2.992559445 2.925615084 2.091842504  
## 42 3.796357064 2.740305446 1.022971862 1.290708123 3.250676580 1.931527640  
## 43 1.183201653 5.217117371 2.418878357 2.210821394 3.708529805 1.853385994  
## 44 0.461414205 5.410376863 2.803569938 2.617468106 3.061790507 1.755388078  
## 45 3.042835784 4.640940706 1.004131071 0.947159579 3.679796556 1.676317552  
## 46 2.981233412 3.747086127 0.387974873 0.063357365 3.249670438 1.257296139  
## 47 1.963573629 4.515133760 2.459649304 2.334320835 1.435445092 1.104297775  
## 48 4.282042031 2.145439580 2.215448563 2.371238404 2.633183015 2.402891217  
## 49 3.625725232 2.961687682 1.482023975 1.643614981 2.382725230 1.648968330  
## 50 3.436639535 2.995282186 1.444027867 1.507811371 2.289958073 1.445424213  
## 51 2.290963407 3.991168114 1.398098703 1.312273858 2.258167474 0.573347431  
## 52 2.722393939 3.733891081 2.572151108 2.521308759 1.142679008 1.531423342  
## 53 1.590752358 4.314761933 2.096251799 1.977148636 2.368397312 1.048972395  
## 54 3.608219693 2.957503517 1.611980700 1.880943180 3.282039492 2.133593506  
## 55 3.346477495 3.732670799 0.566715529 0.676785046 3.381932919 1.590290523  
## 56 3.415234565 2.908236990 1.116028829 1.049868062 3.160689770 1.637391138  
## 57 4.734629113 2.983528499 1.772943055 2.116580425 3.827249799 2.818269481  
## 58 3.377912979 2.995704981 1.558842532 1.722368154 2.259792950 1.541747376  
## 59 1.769408970 4.770737286 2.911754003 2.816628428 3.669436758 2.343662936  
## 60 4.891126317 2.252392368 3.435023210 3.500803167 2.443416160 3.204336159  
## 61 3.749432330 2.263795813 2.256741844 2.337584294 2.348342549 2.110893130  
## 62 3.739708226 2.885865811 2.834147445 2.852228255 1.502624770 2.231872911  
## 63 3.553838024 2.969193315 1.458509801 1.586398411 2.343194915 1.568623944  
## 64 2.804906451 3.770178603 0.711316762 0.737920653 3.204331149 1.269636560  
## 65 4.327936748 2.971477712 1.662164342 1.945546140 3.620950529 2.533848831  
## 66 3.306334361 3.859783348 1.372674969 1.621616505 3.456435084 1.958030127  
## 67 5.286976537 3.367626180 2.492075929 2.861077028 4.285413891 3.472104313  
## 68 5.204315964 3.315695010 2.399566468 2.761507949 4.219368996 3.380427594  
## 69 3.625366386 3.848212042 1.431612869 1.747536091 3.576849256 2.176412373  
## 70 3.683122806 2.961737181 1.518320118 1.709063040 3.314431602 2.087116090  
## 71 2.840324416 3.992752042 1.592561952 1.697824835 2.503979254 1.363161467  
## 72 4.865694204 4.165032345 2.269139227 2.593063643 4.308678028 3.215373044  
## 73 4.692330450 2.991546313 3.089312271 3.248494331 3.321499570 3.203765157  
## 74 5.000823805 3.192370190 3.369292250 3.596859452 3.606781228 3.564051402  
## 75 7.250913488 4.650997822 4.938646305 5.253688776 6.011075870 5.679648709  
## 76 5.338151099 4.878250104 3.356870453 3.536593176 4.933800527 3.988481105  
## 77 5.173110996 5.577333582 3.429457727 3.585458099 5.133864068 4.020431385  
## 37 38 39 40 41 42  
## 2   
## 3   
## 4   
## 5   
## 6   
## 7   
## 8   
## 9   
## 10   
## 11   
## 12   
## 13   
## 14   
## 15   
## 16   
## 17   
## 18   
## 19   
## 20   
## 21   
## 22   
## 23   
## 24   
## 25   
## 26   
## 27   
## 28   
## 29   
## 30   
## 31   
## 32   
## 33   
## 34   
## 35   
## 36   
## 37   
## 38 1.038238486   
## 39 1.392253610 1.021822579   
## 40 0.384350574 1.148887196 1.426043955   
## 41 1.548182450 1.028845905 0.462550463 1.630361251   
## 42 3.023129777 3.037600872 3.052493292 3.019917378 3.439869813   
## 43 0.550850884 1.185001441 1.446242314 0.192207735 1.639900853 3.123615935  
## 44 1.136468447 1.471218652 0.987496005 1.048035842 1.164761004 3.436941625  
## 45 2.328778555 2.959146637 2.938038693 2.190687272 3.353089726 2.011153485  
## 46 2.287190966 2.583076411 2.578177205 2.165061947 3.007255917 1.246751000  
## 47 2.372619976 2.416176835 1.522442011 2.301585542 1.897041403 2.813781150  
## 48 3.851820719 3.646192110 3.380156011 3.857747058 3.735658468 1.447765504  
## 49 3.176258758 3.182646338 2.868493590 3.160839831 3.274990715 1.125455404  
## 50 3.069494756 3.054832212 2.717176553 3.017818102 3.135504519 1.216044246  
## 51 2.015368942 2.200010856 1.772284762 1.942330462 2.212920286 1.856452556  
## 52 2.909239325 2.672887735 1.917292866 2.892345053 2.238256166 2.542817559  
## 53 1.514866758 1.460428348 0.902069339 1.500351949 1.301680734 2.478653566  
## 54 2.736614083 2.621480012 2.725707455 2.847257593 3.026151327 1.106995572  
## 55 2.615302396 2.933699726 2.918848172 2.526954956 3.342028498 1.198264378  
## 56 2.828521939 2.788231984 2.783781251 2.731114512 3.185361069 0.988656866  
## 57 3.913647044 4.040645782 4.026471559 3.933763775 4.411980231 1.168341347  
## 58 2.914732920 2.851504255 2.538493231 2.937447312 2.920265725 1.211656858  
## 59 1.252841506 0.581042415 1.227400965 1.416916508 1.154335274 3.207608058  
## 60 4.789914143 4.430176681 3.967836863 4.776205123 4.270068990 2.753570786  
## 61 3.405390173 3.047352369 2.771058376 3.423609907 3.086771039 1.621997249  
## 62 3.750343588 3.412356259 2.832234574 3.742409636 3.135191324 2.417538621  
## 63 3.133145986 3.132028944 2.809111776 3.104788907 3.220332079 1.146971406  
## 64 2.007364294 2.264634020 2.310928307 1.954874256 2.710611501 1.311859240  
## 65 3.535984832 3.600811583 3.607354269 3.557391978 3.974840649 1.142433363  
## 66 2.388861978 2.624427758 2.703889241 2.467203702 3.036151620 1.542043385  
## 67 4.416339116 4.558199763 4.556265656 4.483208212 4.910334488 1.944300927  
## 68 4.343191007 4.482439571 4.478659059 4.402834643 4.836695907 1.853994527  
## 69 2.695743688 2.960352581 3.020223927 2.778523919 3.358283361 1.509603656  
## 70 2.917122372 2.859641477 2.911485479 2.949006079 3.251610218 1.123725563  
## 71 2.378817177 2.583689967 2.250348235 2.401257675 2.628608820 1.853953908  
## 72 4.008962134 4.341516792 4.332495983 4.048806863 4.696857285 2.189991092  
## 73 4.255183162 4.033331565 3.827812797 4.308459590 4.099897640 2.536005204  
## 74 4.470453512 4.263137019 4.092743101 4.576058339 4.334462712 2.766112944  
## 75 6.443589436 6.452365354 6.465485236 6.554451782 6.733894453 4.362383697  
## 76 4.647785363 4.923469016 4.910337294 4.657564957 5.229666421 3.355385623  
## 77 4.491353642 4.941663484 4.926358751 4.490466250 5.242581459 3.683398379  
## 43 44 45 46 47 48  
## 2   
## 3   
## 4   
## 5   
## 6   
## 7   
## 8   
## 9   
## 10   
## 11   
## 12   
## 13   
## 14   
## 15   
## 16   
## 17   
## 18   
## 19   
## 20   
## 21   
## 22   
## 23   
## 24   
## 25   
## 26   
## 27   
## 28   
## 29   
## 30   
## 31   
## 32   
## 33   
## 34   
## 35   
## 36   
## 37   
## 38   
## 39   
## 40   
## 41   
## 42   
## 43   
## 44 1.020786757   
## 45 2.263854513 2.683905877   
## 46 2.235703152 2.639731282 0.942409570   
## 47 2.326949771 1.691803241 2.638006321 2.347086506   
## 48 3.944589874 3.956219393 3.106226714 2.344756217 2.831641512   
## 49 3.252579419 3.263080739 2.259703596 1.614307104 2.211943684 0.942437921  
## 50 3.092034476 3.097357427 2.179504311 1.490799712 2.038606915 1.083630689  
## 51 2.011679104 1.928479919 1.714459258 1.315346377 1.275173584 2.215586592  
## 52 2.937455553 2.472737718 3.075392375 2.523642151 1.114080004 2.071583343  
## 53 1.554899874 1.281968341 2.366844954 1.988124996 1.214947978 2.768588692  
## 54 2.979971974 3.248613468 2.502448070 1.838881557 2.830275410 1.649222277  
## 55 2.610949985 2.987164310 1.083167748 0.643617207 2.600343027 2.187773168  
## 56 2.787692615 3.121122007 1.911616465 1.047739358 2.571022310 1.650665274  
## 57 4.051786664 4.343558987 2.526304336 2.065842124 3.605346772 1.663374071  
## 58 3.038152462 3.015498910 2.360778785 1.693118187 2.035445796 1.040337187  
## 59 1.465898992 1.679148861 3.187871097 2.825828924 2.603318928 3.714622287  
## 60 4.835256127 4.633389340 4.243681252 3.488122084 3.232727210 1.379640354  
## 61 3.500364038 3.463000801 3.143523459 2.321116185 2.440716176 0.784189276  
## 62 3.797720960 3.486842641 3.551570481 2.845380824 2.101775320 1.384526587  
## 63 3.190426332 3.199107700 2.224347367 1.561222012 2.143627189 0.981877633  
## 64 2.052126257 2.447993869 1.263210417 0.717309892 2.276904412 2.257086475  
## 65 3.669291037 3.958402580 2.442929907 1.901854720 3.309386184 1.501156470  
## 66 2.603422092 2.921279435 1.903893573 1.581271754 2.726848353 2.249339774  
## 67 4.616670828 4.882080412 3.153250307 2.807826247 4.162569957 2.122773033  
## 68 4.533623903 4.802373130 3.067169002 2.709080469 4.081978545 2.046510642  
## 69 2.919108045 3.226577491 1.971215843 1.699502793 2.938551860 2.213613059  
## 70 3.054843768 3.343373151 2.347268398 1.676275128 2.850113291 1.518561524  
## 71 2.507351065 2.454348998 1.992088178 1.676128637 1.830567061 2.058536100  
## 72 4.175744463 4.462406210 2.627445434 2.545251208 3.933824589 2.599463128  
## 73 4.400651798 4.391069520 3.836773204 3.222811091 3.491358864 1.638936419  
## 74 4.689210302 4.675944334 4.132691149 3.562341825 3.822749096 1.942065003  
## 75 6.685923716 6.881934195 5.544950013 5.208253698 6.160112574 3.961289982  
## 76 4.753292865 5.003237388 3.572578068 3.508417528 4.575029544 3.423117053  
## 77 4.582968019 4.840040881 3.387552037 3.560520795 4.597605899 3.965974674  
## 49 50 51 52 53 54  
## 2   
## 3   
## 4   
## 5   
## 6   
## 7   
## 8   
## 9   
## 10   
## 11   
## 12   
## 13   
## 14   
## 15   
## 16   
## 17   
## 18   
## 19   
## 20   
## 21   
## 22   
## 23   
## 24   
## 25   
## 26   
## 27   
## 28   
## 29   
## 30   
## 31   
## 32   
## 33   
## 34   
## 35   
## 36   
## 37   
## 38   
## 39   
## 40   
## 41   
## 42   
## 43   
## 44   
## 45   
## 46   
## 47   
## 48   
## 49   
## 50 0.350957087   
## 51 1.404511957 1.233771942   
## 52 1.735051306 1.593566931 1.425288948   
## 53 2.144541800 1.988020384 0.956354648 1.446854521   
## 54 1.433564254 1.586209432 1.886745008 2.400762980 2.186959673   
## 55 1.428345492 1.370045855 1.411054602 2.618610421 2.192581296 1.712216623  
## 56 1.226016441 1.059297439 1.517150428 2.296067102 2.127496639 1.508372792  
## 57 1.518612222 1.753835398 2.593330026 3.227770001 3.334923258 1.659774400  
## 58 0.458587656 0.598295736 1.269926351 1.494766404 1.849944059 1.181309806  
## 59 3.271340121 3.163599462 2.311031459 2.728498697 1.516030768 2.623718976  
## 60 2.080248053 2.100625439 3.090142576 2.296059189 3.484984102 2.899822311  
## 61 1.151824474 1.160145362 1.945536101 1.586914014 2.252587340 1.506586791  
## 62 1.578146556 1.516877835 2.112484935 1.097291580 2.363288402 2.386295416  
## 63 0.128138490 0.222818597 1.334029165 1.676276537 2.081908053 1.481504110  
## 64 1.515818271 1.434611078 1.062873120 2.315627152 1.621905689 1.428122514  
## 65 1.277210568 1.468563693 2.203227054 2.847605247 2.866956211 1.256436905  
## 66 1.620571904 1.713564742 1.557089193 2.548250717 1.986300699 1.093725909  
## 67 2.091403639 2.374953819 3.154317384 3.699211110 3.839531869 2.035089645  
## 68 2.000207508 2.276800418 3.063048440 3.620741728 3.756066082 1.970500491  
## 69 1.612440549 1.763912128 1.783473733 2.734166462 2.297898755 1.145425933  
## 70 1.214360538 1.297608255 1.737871918 2.370924454 2.208794415 0.775389126  
## 71 1.264557660 1.264458871 0.821470190 1.656412788 1.395821489 1.602788628  
## 72 2.179564222 2.388647449 2.816503708 3.648275132 3.532254796 2.223729855  
## 73 1.930189784 2.059260019 2.766209835 2.575929173 3.123181912 2.099340733  
## 74 2.259982331 2.463869200 3.124721171 2.928406737 3.435892300 2.180908149  
## 75 4.226544198 4.482297967 5.229302005 5.430588606 5.736188961 4.039026768  
## 76 3.088399933 3.198752641 3.480211056 4.179396343 4.055573476 3.176637559  
## 77 3.436048967 3.519631947 3.528402657 4.411850270 4.086050609 3.537429171  
## 55 56 57 58 59 60  
## 2   
## 3   
## 4   
## 5   
## 6   
## 7   
## 8   
## 9   
## 10   
## 11   
## 12   
## 13   
## 14   
## 15   
## 16   
## 17   
## 18   
## 19   
## 20   
## 21   
## 22   
## 23   
## 24   
## 25   
## 26   
## 27   
## 28   
## 29   
## 30   
## 31   
## 32   
## 33   
## 34   
## 35   
## 36   
## 37   
## 38   
## 39   
## 40   
## 41   
## 42   
## 43   
## 44   
## 45   
## 46   
## 47   
## 48   
## 49   
## 50   
## 51   
## 52   
## 53   
## 54   
## 55   
## 56 0.991432543   
## 57 1.676267652 1.777153600   
## 58 1.577344837 1.327979580 1.754844249   
## 59 3.066653577 2.925642042 4.115832508 2.912745962   
## 60 3.411033444 2.779781351 2.950872812 2.131081663 4.496100095   
## 61 2.286561454 1.617745546 2.201492113 0.959205250 3.096612061 1.516833567  
## 62 2.850229872 2.308209404 2.921128767 1.450751918 3.469142102 1.207507203  
## 63 1.397158906 1.155618992 1.599586848 0.485449896 3.227999537 2.080861865  
## 64 0.694482748 1.041155235 2.051574533 1.458043850 2.375913226 3.446867826  
## 65 1.435287374 1.408287337 0.770506262 1.417983060 3.598093874 2.827554884  
## 66 1.246690259 1.592949804 1.846199163 1.480109468 2.557488476 3.506305558  
## 67 2.335580783 2.498518486 0.874142187 2.257852649 4.544674185 3.294237984  
## 68 2.232669052 2.387188659 0.781032066 2.175558167 4.472622215 3.235119912  
## 69 1.287420799 1.709776641 1.593733630 1.534114276 2.904547636 3.506983437  
## 70 1.359948700 1.085968706 1.485991239 1.121721991 2.824500626 2.791005472  
## 71 1.463145461 1.688082638 2.273790598 1.092894350 2.541554253 3.040532366  
## 72 1.949612960 2.445813253 1.399192346 2.346352525 4.281373434 3.810232383  
## 73 2.845851376 2.455712097 2.372937677 1.865161398 3.858152709 2.185435984  
## 74 3.179032051 2.886129604 2.479900504 2.146411014 4.061806180 2.495418512  
## 75 4.661557022 4.658207272 3.425165348 4.300244377 6.244772544 4.599186145  
## 76 2.878361632 3.199196806 2.726056444 3.212329110 4.717172120 4.389225247  
## 77 2.958690543 3.503281018 3.161438993 3.549429677 4.741164170 4.985329508  
## 61 62 63 64 65 66  
## 2   
## 3   
## 4   
## 5   
## 6   
## 7   
## 8   
## 9   
## 10   
## 11   
## 12   
## 13   
## 14   
## 15   
## 16   
## 17   
## 18   
## 19   
## 20   
## 21   
## 22   
## 23   
## 24   
## 25   
## 26   
## 27   
## 28   
## 29   
## 30   
## 31   
## 32   
## 33   
## 34   
## 35   
## 36   
## 37   
## 38   
## 39   
## 40   
## 41   
## 42   
## 43   
## 44   
## 45   
## 46   
## 47   
## 48   
## 49   
## 50   
## 51   
## 52   
## 53   
## 54   
## 55   
## 56   
## 57   
## 58   
## 59   
## 60   
## 61   
## 62 1.037171655   
## 63 1.142441187 1.546854702   
## 64 2.124324808 2.709867618 1.477049116   
## 65 1.855075118 2.630695395 1.339607916 1.660134467   
## 66 2.181645035 2.838350032 1.646482554 0.963785029 1.328462891   
## 67 2.669064750 3.351336054 2.192671461 2.647293860 1.150490341 2.138367952  
## 68 2.591439936 3.279078519 2.098619019 2.551889716 1.042470721 2.064650710  
## 69 2.265364147 2.932450635 1.660764516 1.186684643 1.142441187 0.366846157  
## 70 1.486732724 2.328400765 1.233884283 1.227358771 0.820671630 0.952513100  
## 71 1.897764450 2.158868244 1.253181239 1.191276379 1.794074404 1.087479775  
## 72 3.006286874 3.589061920 2.251817025 2.292947345 1.296780979 1.729108722  
## 73 1.709296104 2.101830594 1.971061624 2.801914291 1.829161708 2.348535717  
## 74 2.044273849 2.460707195 2.330368827 3.111843491 2.024943317 2.498770854  
## 75 4.398903670 4.926723458 4.318372866 4.856083406 3.337220043 4.110320093  
## 76 3.703406127 4.163801859 3.124576529 3.108131182 2.305556572 2.528893106  
## 77 4.195791218 4.596886856 3.462679333 3.177179899 2.787003460 2.660854386  
## 67 68 69 70 71 72  
## 2   
## 3   
## 4   
## 5   
## 6   
## 7   
## 8   
## 9   
## 10   
## 11   
## 12   
## 13   
## 14   
## 15   
## 16   
## 17   
## 18   
## 19   
## 20   
## 21   
## 22   
## 23   
## 24   
## 25   
## 26   
## 27   
## 28   
## 29   
## 30   
## 31   
## 32   
## 33   
## 34   
## 35   
## 36   
## 37   
## 38   
## 39   
## 40   
## 41   
## 42   
## 43   
## 44   
## 45   
## 46   
## 47   
## 48   
## 49   
## 50   
## 51   
## 52   
## 53   
## 54   
## 55   
## 56   
## 57   
## 58   
## 59   
## 60   
## 61   
## 62   
## 63   
## 64   
## 65   
## 66   
## 67   
## 68 0.118884044   
## 69 1.816079313 1.748291137   
## 70 1.890990303 1.797725887 1.009862978   
## 71 2.638167256 2.558315762 1.253332299 1.400982518   
## 72 1.137734029 1.096869008 1.403500605 1.879934986 2.232711708   
## 73 2.308307414 2.255625365 2.319091350 1.750706986 2.232024345 2.507597138  
## 74 2.223434064 2.203417831 2.417555589 2.020248108 2.504945511 2.543584321  
## 75 2.631152816 2.694919346 3.851595747 3.824003657 4.526853396 2.998835778  
## 76 2.390503218 2.352586718 2.364090048 2.613916919 2.839243188 1.584090006  
## 77 2.916247305 2.878640812 2.523028124 3.023244005 2.933347857 1.867778663  
## 73 74 75 76  
## 2   
## 3   
## 4   
## 5   
## 6   
## 7   
## 8   
## 9   
## 10   
## 11   
## 12   
## 13   
## 14   
## 15   
## 16   
## 17   
## 18   
## 19   
## 20   
## 21   
## 22   
## 23   
## 24   
## 25   
## 26   
## 27   
## 28   
## 29   
## 30   
## 31   
## 32   
## 33   
## 34   
## 35   
## 36   
## 37   
## 38   
## 39   
## 40   
## 41   
## 42   
## 43   
## 44   
## 45   
## 46   
## 47   
## 48   
## 49   
## 50   
## 51   
## 52   
## 53   
## 54   
## 55   
## 56   
## 57   
## 58   
## 59   
## 60   
## 61   
## 62   
## 63   
## 64   
## 65   
## 66   
## 67   
## 68   
## 69   
## 70   
## 71   
## 72   
## 73   
## 74 0.678252363   
## 75 3.049481929 2.677016270   
## 76 2.598056339 2.721610034 2.847895207   
## 77 3.296296040 3.424527892 3.606618811 0.927354713

head(dist.ecul\_cereals\_cluster2.scaled)

## [1] 1.590728 1.538944 1.451929 1.865864 1.751992 1.976624

str(dist.ecul\_cereals\_cluster2.scaled)

## 'dist' num [1:2926] 1.59 1.54 1.45 1.87 1.75 ...  
## - attr(\*, "Size")= int 77  
## - attr(\*, "Labels")= chr [1:77] "1" "2" "3" "4" ...  
## - attr(\*, "Diag")= logi FALSE  
## - attr(\*, "Upper")= logi FALSE  
## - attr(\*, "method")= chr "euclidean"  
## - attr(\*, "call")= language dist(x = cereals\_cluster2.scaled, method = "euclidean")

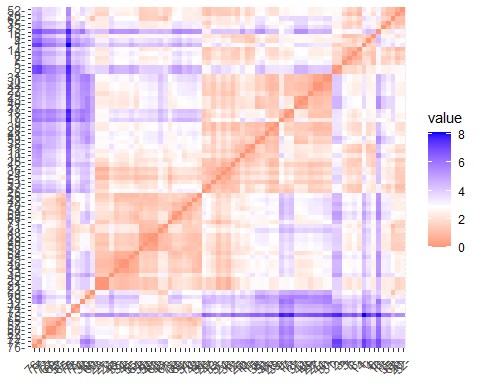
class(dist.ecul\_cereals\_cluster2.scaled)

## [1] "dist"

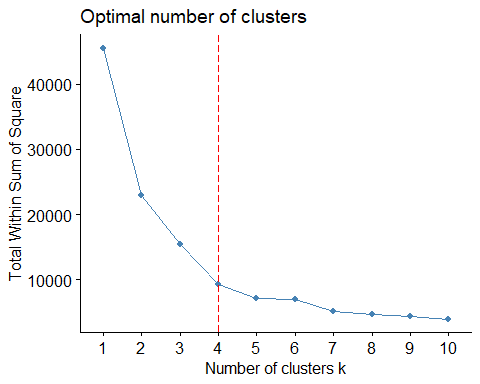
round(as.matrix(dist.ecul\_cereals\_cluster2.scaled)[1:3,1:3],1) #Show only three products

## 1 2 3  
## 1 0.0 1.6 1.5  
## 2 1.6 0.0 0.2  
## 3 1.5 0.2 0.0

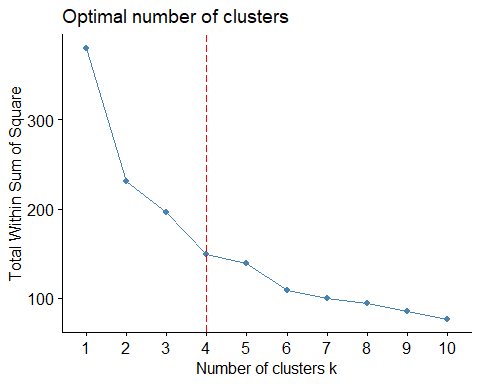
#-----Dissimilarity matrix----   
#Darker color: Higher Distance ; Light Color:Lesser Distance  
fviz\_dist(dist.ecul\_cereals\_cluster2.scaled)



#----Optimum number of clusters----  
  
fviz\_nbclust(cereals\_cluster2, kmeans ,method = 'wss')+  
 geom\_vline(xintercept = 4 , linetype = 5 , col="red" )



fviz\_nbclust(cereals\_cluster2.scaled, kmeans ,method = 'wss')+  
 geom\_vline(xintercept = 4, linetype = 5 , col="red" )



#-----Apply Kmeans----  
set.seed(123)  
km.cereal<-kmeans(cereals\_cluster2.scaled,4,nstart = 20)  
km.cereal

## K-means clustering with 4 clusters of sizes 26, 25, 16, 10  
##   
## Cluster means:  
## Calories Protein Fat Sugar Rating  
## 1 0.3178900 -0.70901630 -0.01290349 1.0523583 -0.8935153  
## 2 -0.2300908 0.04982277 -0.52955941 -0.6799502 0.3309871  
## 3 0.8977508 0.87189842 1.35325397 0.1631212 -0.2145661  
## 4 -1.6876882 0.32384799 -0.80775875 -1.2972500 1.8389778  
##   
## Clustering vector:  
## 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26   
## 3 3 3 1 3 3 3 3 1 1 1 1 3 3 1 3 1 1 3 3 2 2 2 1 1 1   
## 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52   
## 1 1 1 1 1 2 2 2 3 2 1 1 1 1 1 2 1 1 2 2 1 2 2 2 2 3   
## 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77   
## 1 2 2 2 4 2 1 3 2 3 2 2 4 2 4 4 2 2 2 4 4 4 4 4 4   
##   
## Within cluster sum of squares by cluster:  
## [1] 40.16931 37.76791 46.99282 24.89775  
## (between\_SS / total\_SS = 60.6 %)  
##   
## Available components:  
##   
## [1] "cluster" "centers" "totss" "withinss" "tot.withinss"  
## [6] "betweenss" "size" "iter" "ifault"

km.cereal$cluster

## 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26   
## 3 3 3 1 3 3 3 3 1 1 1 1 3 3 1 3 1 1 3 3 2 2 2 1 1 1   
## 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52   
## 1 1 1 1 1 2 2 2 3 2 1 1 1 1 1 2 1 1 2 2 1 2 2 2 2 3   
## 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77   
## 1 2 2 2 4 2 1 3 2 3 2 2 4 2 4 4 2 2 2 4 4 4 4 4 4

km.cereal$totss

## [1] 380

km.cereal$centers

## Calories Protein Fat Sugar Rating  
## 1 0.3178900 -0.70901630 -0.01290349 1.0523583 -0.8935153  
## 2 -0.2300908 0.04982277 -0.52955941 -0.6799502 0.3309871  
## 3 0.8977508 0.87189842 1.35325397 0.1631212 -0.2145661  
## 4 -1.6876882 0.32384799 -0.80775875 -1.2972500 1.8389778

km.cereal$size

## [1] 26 25 16 10

km.cereal$betweenss

## [1] 230.1722

aggregate(cereals\_cluster1, by = list(km.cereal$cluster), mean)

## Group.1 Calories Protein Fat Sugar Rating  
## 1 1 113.0769 1.769231 1.000 11.57692 30.11385  
## 2 2 102.4000 2.600000 0.480 4.00000 47.31480  
## 3 3 124.3750 3.500000 2.375 7.68750 39.65125  
## 4 4 74.0000 2.900000 0.200 1.30000 68.49800

km.cereal$cluster

## 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26   
## 3 3 3 1 3 3 3 3 1 1 1 1 3 3 1 3 1 1 3 3 2 2 2 1 1 1   
## 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52   
## 1 1 1 1 1 2 2 2 3 2 1 1 1 1 1 2 1 1 2 2 1 2 2 2 2 3   
## 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77   
## 1 2 2 2 4 2 1 3 2 3 2 2 4 2 4 4 2 2 2 4 4 4 4 4 4

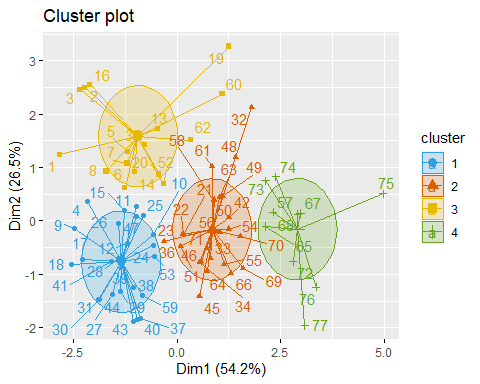
cereals\_cluster<-cbind(cereals\_cluster1,cluster = km.cereal$cluster)  
head(cereals\_cluster)

## Calories Protein Fat Sugar Rating cluster  
## 1 160 3 2 13 30.31 3  
## 2 150 4 3 11 37.14 3  
## 3 150 4 3 11 34.14 3  
## 4 140 3 1 14 28.59 1  
## 5 140 3 2 7 40.69 3  
## 6 140 3 1 9 36.47 3

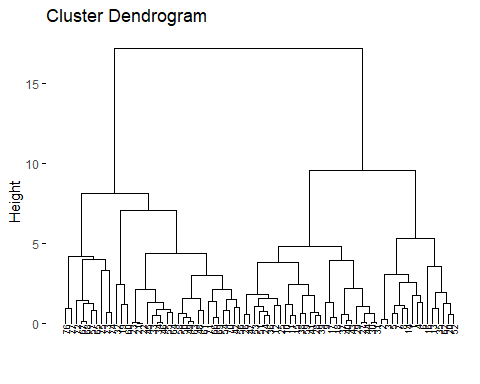
head(cereals\_cluster)

## Calories Protein Fat Sugar Rating cluster  
## 1 160 3 2 13 30.31 3  
## 2 150 4 3 11 37.14 3  
## 3 150 4 3 11 34.14 3  
## 4 140 3 1 14 28.59 1  
## 5 140 3 2 7 40.69 3  
## 6 140 3 1 9 36.47 3

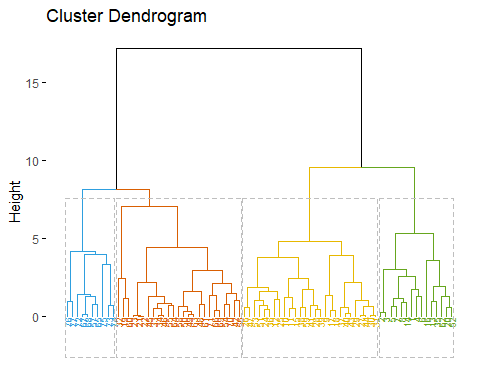
#write.csv(km.cereal,"km.cereal.csv")  
#----Kmeans cluster plot---  
fviz\_cluster(km.cereal, data = cereals\_cluster2.scaled, palette = c("#2E9FDF","#D95F02","#E7B800","#66A61E"),  
 ellipse.type = "euclid",  
 star.plot = TRUE,  
 repel = TRUE,  
 ggtheme = theme())



#-----Hierarchical Clustering: (Agglomeration) Linkage Methods----  
cereals\_cluster3<-hclust(d =dist.ecul\_cereals\_cluster2.scaled, method = "ward.D2")  
#-----Cluster Dendrogram--Black and White---  
fviz\_dend(cereals\_cluster3, cex = 0.5)



#-----Cluster Dendrogram--coloured---  
fviz\_dend(cereals\_cluster3, k=4 ,cex=0.5 , k\_colors = c("#2E9FDF","#D95F02","#E7B800","#66A61E"),  
 color\_labels\_by\_k = TRUE,  
 rect = TRUE)



#------Cut tree-----another way to represent cluster plot---  
cereals\_cluster4<- cutree(cereals\_cluster3, k=4)  
head(cereals\_cluster4, n=4)

## 1 2 3 4   
## 1 1 1 1

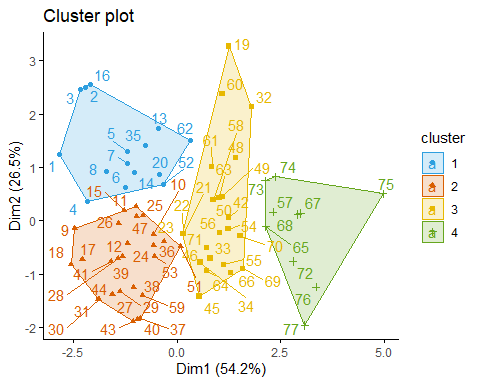
table(cereals\_cluster4)

## cereals\_cluster4  
## 1 2 3 4   
## 15 27 25 10

rownames(cereals\_cluster2.scaled)[cereals\_cluster4==1]

## [1] "1" "2" "3" "4" "5" "6" "7" "8" "13" "14" "16" "20" "35" "52" "62"

fviz\_cluster(list(data = cereals\_cluster2.scaled, cluster = cereals\_cluster4),  
 palette = c("#2E9FDF","#D95F02","#E7B800","#66A61E"),  
 ellipse = TRUE,  
 ellipse.type = "convex",  
 repel = TRUE,  
 show.clust.cent = FALSE,  
 ggtheme = theme\_classic())



#-----Principal component analysis-PCA Plot----  
PCA\_data<-cereals\_cluster1  
  
PCA\_cereals <- prcomp(PCA\_data[], scale. = T)  
summary(PCA\_cereals)

## Importance of components:  
## PC1 PC2 PC3 PC4 PC5  
## Standard deviation 1.6458 1.1517 0.7078 0.5886 0.3428  
## Proportion of Variance 0.5417 0.2653 0.1002 0.0693 0.0235  
## Cumulative Proportion 0.5417 0.8070 0.9072 0.9765 1.0000

fviz\_pca\_var(PCA\_cereals,   
 col.var = "contrib",   
 repel = T,   
 gradient.cols = c("#00AFBB", "#E7B800", "#FC4E07"),  
 title = "Principal Component Analysis: Variable Contribution",  
 legend.title = "Contribution"  
)

