Assignment 3: K Means Clustering

In this assignment we will be applying the K-means clustering algorithm we looked at in class. At the following link you can find a description of K-means:

https://www.cs.uic.edu/~wilkinson/Applets/cluster.html

```
library(tidyr)
library(dplyr)
```

```
## Warning: package 'dplyr' was built under R version 3.5.2
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(car)
```

```
## Warning: package 'car' was built under R version 3.5.2
```

```
## Loading required package: carData
```

```
##
## Attaching package: 'car'
```

```
## The following object is masked from 'package:dplyr':
##
##     recode
```

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.2.1 --
```

```
## <U+221A> ggplot2 3.2.1     <U+221A> purrr   0.2.5
## <U+221A> tibble  2.1.3     <U+221A> stringr 1.3.1
## <U+221A> readr   1.1.1     <U+221A> forcats 0.3.0
```

```
## Warning: package 'ggplot2' was built under R version 3.5.2
```

```
## Warning: package 'tibble' was built under R version 3.5.2
```

```
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## x car::recode()   masks dplyr::recode()
## x purrr::some()   masks car::some()
```

Now, upload the file "Class_Motivation.csv" from the Assignment 3 Repository as a data frame called "K1""

```
K1 <- read.csv("~/Desktop/master fall/hudk4050/assignment3/Class_Motivation.csv")
```

This file contains the self-reported motivation scores for a class over five weeks. We are going to look for patterns in motivation over this time and sort people into clusters based on those patterns.

But before we do that, we will need to manipulate the data frame into a structure that can be analyzed by our clustering algorithm.

The algorithm will treat each row as a value belonging to a person, so we need to remove the id variable.

```
K2 <- dplyr::select(K1, 2:6)
```

It is important to think about the meaning of missing values when clustering. We could treat them as having meaning or we could remove those people who have them. Neither option is ideal. What problems do you foresee if we recode or remove these values? Write your answers below:

We will remove people with missing values for this assignment, but keep in mind the issues that you have identified.

```
K3 <- na.omit(K2) #This command create a data frame with only those people with no missing values. It "
```

Another pre-processing step used in K-means is to standardize the values so that they have the same range. We do this because we want to treat each week as equally important - if we do not standardise then the week with the largest range will have the greatest impact on which clusters are formed. We standardise the values by using the "scale()" command.

```
K3 <- scale(K3)
```

Now we will run the K-means clustering algorithm we talked about in class. 1) The algorithm starts by randomly choosing some starting values 2) Associates all observations near to those values with them 3) Calculates the mean of those clusters of values 4) Selects the observation closest to the mean of the cluster 5) Re-associates all observations closest to this observation 6) Continues this process until the clusters are no longer changing

Notice that in this case we have 5 variables and in class we only had 2. It is impossible to vizualise this process with 5 variables.

Also, we need to choose the number of clusters we think are in the data. We will start with 2.

```
fit <- kmeans(K3, 2)
#We have created an object called "fit" that contains all the details of our clustering including which
#We can access the list of clusters by typing "fit$cluster", the top row corresponds to the original or
fit$cluster
```

```
##  1  3  7 10 12 14 15 17 18 19 20 22 25 26 27 28 29 30 31 32 33 34 35
##  2  1  2  2  2  2  1  1  1  1  2  2  1  1  1  1  1  1  1  1  1  1  2
```

```
#We can also attach these clusters to the original dataframe by using the "data.frame" command to creat
K4 <- data.frame(K3, fit$cluster)
#Have a look at the K4 dataframe. Lets change the names of the variables to make it more convenient wit
names(K4) <- c("1","2","3","4","5","cluster")
```

Now we need to visualize the clusters we have created. To do so we want to play with the structure of our data. What would be most useful would be if we could visualize average motivation by cluster, by week. To do this we will need to convert our data from wide to long format. Remember your old friends tidyr and dplyr!

First lets use tidyr to convert from wide to long format.

```
K5 <- gather(K4, "week", "motivation", 1:5)
```

Now lets use dplyr to average our motivation values by week and by cluster.

```
K6 <- K5 %>% group_by(week,cluster)
K6 <- summarise(K6, avg = mean(motivation))
```

Now it's time to do some visualization:

https://www.cs.uic.edu/~wilkinson/TheGrammarOfGraphics/GOG.html

And you can see the range of available graphics in ggplot here:

http://ggplot2.tidyverse.org/reference/index.html

We are going to create a line plot similar to the one created in the school dropout paper we looked at in class (Bowers, 2010). It will have motivation on the Y-axis and weeks on the X-axis. To do this we will want our weeks variables to be treated as a number, but because it was created from a variable name it is currently being treated as a character variable. You can see this if you click on the arrow on the left of K6 in the Data pane. Week is designated by "chr". To convert it to numeric, we use the as.numeric command.

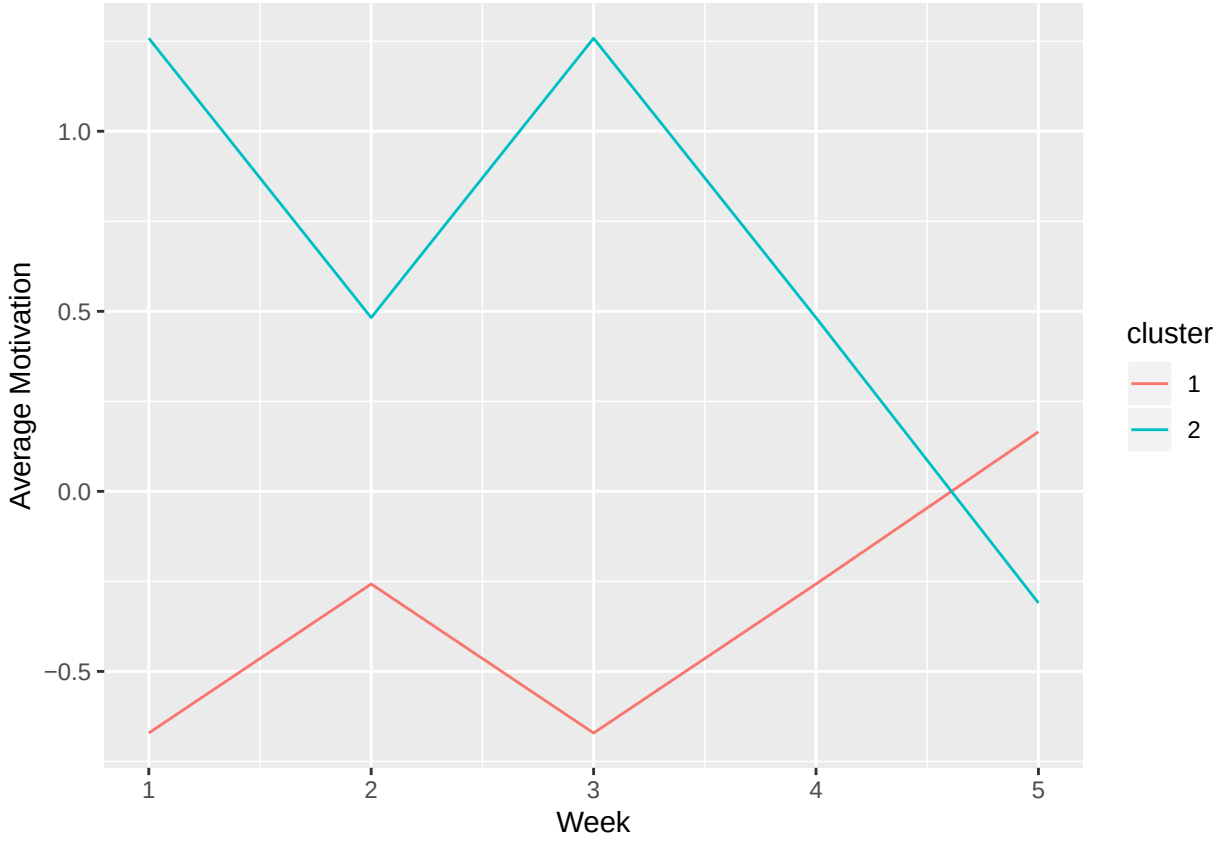
Likewise, since "cluster" is not numeric but rather a categorical label we want to convert it from an "integer" format to a "factor" format so that ggplot does not treat it as a number. We can do this with the as.factor() command.

```
K6$week <- as.numeric(K6$week)
K6$cluster <- as.factor(K6$cluster)
```

Now we can plot our line plot using the ggplot command, "ggplot()".

- The first argument in a ggplot is the dataframe we are using: K6
- Next is what is called an aesthetic (aes), the aesthetic tells ggplot which variables to use and how to use them. Here we are using the variables "week" and "avg" on the x and y axes and we are going color these variables using the "cluster" variable
- Then we are going to tell ggplot which type of plot we want to use by specifiying a "geom()", in this case a line plot: geom_line()
- Finally we are going to clean up our axes labels: xlab("Week") & ylab("Average Motivation")

```
library(tidyverse)
ggplot(K6, aes(week, avg, colour =cluster))+geom_line() + xlab("Week") + ylab("Average Motivation")
```



What patterns do you see in the plot?

It would be useful to determine how many people are in each cluster. We can do this easily with dplyr.

```
K7 <- count(K4,cluster)
K7
```

```
## # A tibble: 2 x 2
##   cluster     n
##     <int> <int>
## 1       1    15
## 2       2     8
```

Look at the number of people in each cluster, now repeat this process for 3 rather than 2 clusters. Which cluster grouping do you think is more informative? Write your answer below:

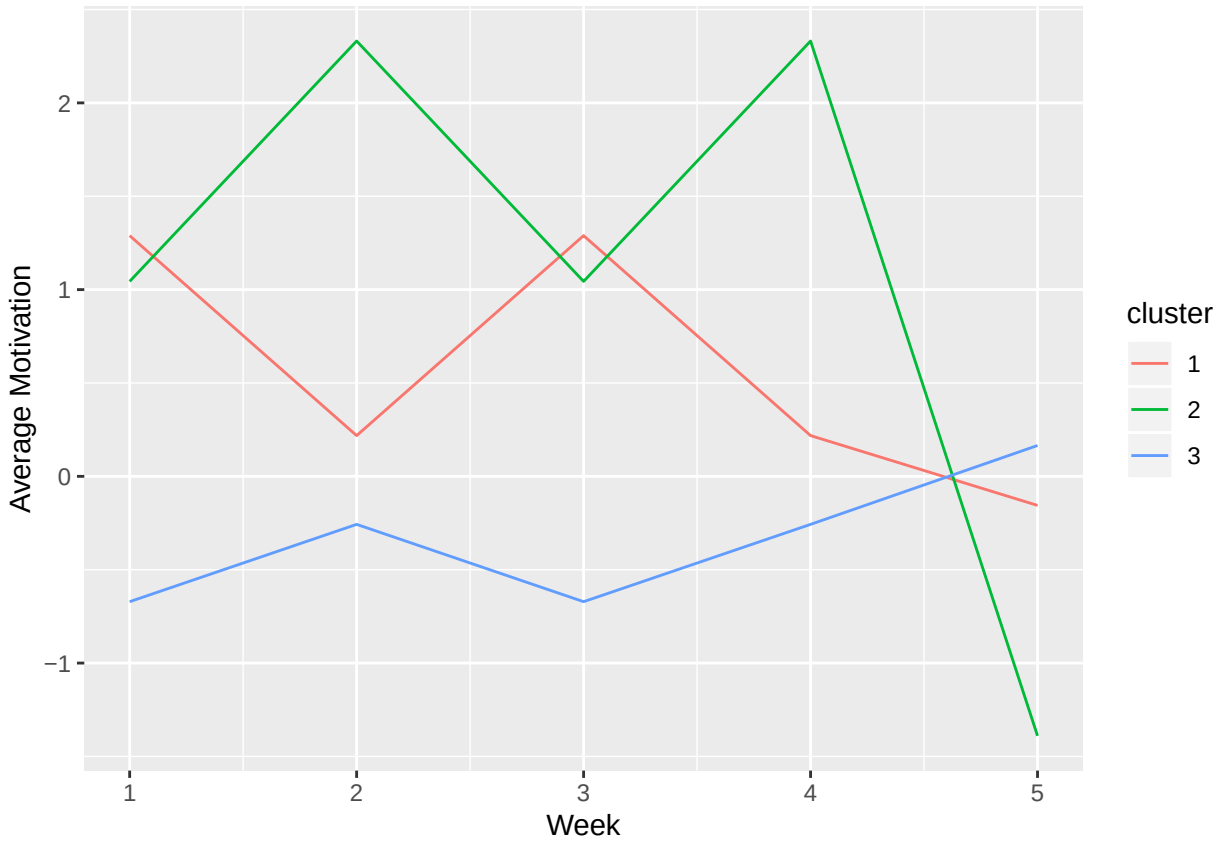
```
fit1 <- kmeans(K3, 3)
K8 <- data.frame(K3, fit1$cluster)
K9 <- count(K8, fit1.cluster)
K9
```

```
## # A tibble: 3 x 2
##   fit1.cluster     n
##          <int> <int>
## 1            1     7
## 2            2     1
## 3            3    15
```

```
fit1$cluster
```

```
##  1  3  7 10 12 14 15 17 18 19 20 22 25 26 27 28 29 30 31 32 33 34 35
##  1  3  1  1  1  2  3  3  3  3  1  1  3  3  3  3  3  3  3  3  3  3  1
```

```
K10 <- data.frame(K3, fit1$cluster)
names(K10) <- c("1","2","3","4","5","cluster")
K11 <- gather(K10, "week", "motivation", 1:5)
K12 <- K11 %>% group_by(week,cluster)
K12 <- summarise(K12, avg = mean(motivation))
K12$week <- as.numeric(K12$week)
K12$cluster <- as.factor(K12$cluster)
ggplot(K12, aes(week, avg, colour =cluster))+geom_line() + xlab("Week") + ylab("Average Motivation")
```



for 3 clusters, cluster group 1 is more informative.

Part II

Using the data collected for Assignment 2 (which classes students were in), cluster the students, then redraw the graph of the class but color the students according the cluster they are in.

```
library(igraph)
```

```
## Warning: package 'igraph' was built under R version 3.5.2
```

```
##
## Attaching package: 'igraph'
```

```
## The following objects are masked from 'package:purrr':
##
##     compose, simplify
```

```
## The following object is masked from 'package:tibble':
##
##     as_data_frame
```

```
## The following objects are masked from 'package:dplyr':
##
##     as_data_frame, groups, union
```

```
## The following object is masked from 'package:tidyr':
##
##     crossing
```

```
## The following objects are masked from 'package:stats':
##
##     decompose, spectrum
```

```
## The following object is masked from 'package:base':
##
##     union
```

```
D1 <- read.csv("~/Desktop/master fall/hudk4050/assignment3/hudk4050.classes.csv")
D3 <- data.frame(D1)
names(D3) <- c("FirstName", "LastName", "Class1", "Class2","Class3","Class4","Class5","Class6")
D4 <- dplyr::select(D3,FirstName,Class1,Class2,Class3,Class4,Class5,Class6)
D6<-  gather(D4,coursenum,course,Class1,Class2,Class3,Class4,Class5,Class6)
```

```
## Warning: attributes are not identical across measure variables;
## they will be dropped
```

```
D7 <- dplyr::select(D6,FirstName,course)
#manage the course,that each student may type same course with diffterent way
D7 <- D7%>%filter(!course == "") %>% filter(course != "4050")
names(D7)<- c("student","course")
D7$course <- gsub(" ","",D7$course)
D7$course <- gsub("5026","HUDM5026",D7$course)
D7$course <- gsub("5126","HUDM5126",D7$course)
D7$course <- gsub("QMSS","G",D7$course)
D7$course <- gsub("GG","G",D7$course)
D7$course <- gsub("GGR","G",D7$course)
#drop hudk4050, since everyone takes hudk4050
D7 <- D7%>% filter(course != "HUDK4050")
#generate person-class matrix
D8<- mutate(D7,enrolled = 1)
D9 <- spread(D8,course,enrolled,0)
#generate person-person matrix
D10<-dplyr::select(D9,-student)
D10 <- as.matrix(D10)
D11 <- t(D10)
D12<- D10 %*% D11
diag(D12)<-0
colnames(D12)<-D9$student
rownames(D12)<-D9$student
```

```
library(klaR)
```

```
## Loading required package: MASS
```

```
##
## Attaching package: 'MASS'
```

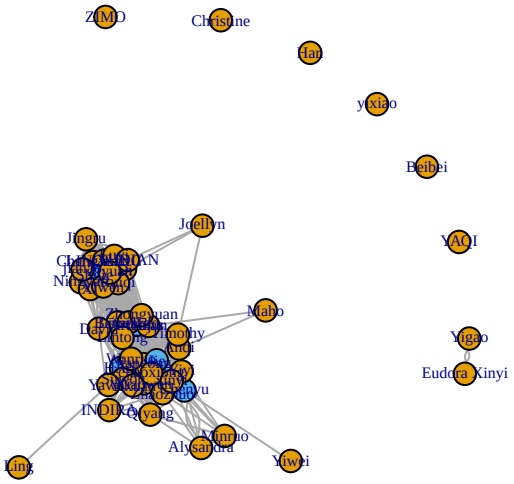
```
## The following object is masked from 'package:dplyr':
##
##     select
```

```
course_get <- dplyr::select(D4,-FirstName)
course_get <- na.omit(course_get)
course_get <- as.data.frame(course_get)
set.seed(4050)
fit2 <- kmodes(course_get,2)
```

```
#plot igraph
g1 <-graph_from_adjacency_matrix(D12,mode="undirected")
V(g1)$label.cex <- seq(0.5,0.5,length.out=6)
```

```
## Warning in vattrs[[name]][index] <- value: number of items to replace is
## not a multiple of replacement length
```

```
plot(g1,layout=layout.fruchterman.reingold,,vertex.size = 10,edge.arrow.size=0.3,vertex.label = D9$stud
```



Part III

In class activity 6 you clustered students in the class by the answers to a questionaire. Create a visualization that shows the overlap between these clusters and the clusters generated in part II.

Please render your code as an .html file using knitr and Pull Resquest both your .Rmd file and .html files to the Assignment 3 repository.

```
library(vcd)
```

```
## Loading required package: grid
```

```
library(grid)
library(ggmosaic)
```

```
##
## Attaching package: 'ggmosaic'
```

```
## The following objects are masked from 'package:vcd':
##
##     mosaic, spine
```

```
D4$FirstName <- str_to_lower(D4$FirstName) %>% str_remove_all("[[:space:]]")
A2DF <- data.frame(D4,fit2$cluster)
DF1 <- read.csv("~/Desktop/master fall/hudk4050/assignment3/HUDK405019-clustering.csv")
DF1 <- data.frame(DF1)
DF1$First.Name <- str_to_lower(DF1$First.Name) %>% str_remove_all("[[:space:]]")
#Wrangle data using dplyr to include only the numerical values.
#Remove location variables
```

```
DF2 <- dplyr::select(DF1, 3:13)
#Remove any characters
DF2 <- DF2 %>% mutate_all(funs(gsub("[a-zA-Z]", "", .)))
```

```
## Warning: funs() is soft deprecated as of dplyr 0.8.0
## Please use a list of either functions or lambdas:
##
##   # Simple named list:
##   list(mean = mean, median = median)
##
##   # Auto named with `tibble::lst()`:
##   tibble::lst(mean, median)
##
##   # Using lambdas
##   list(~ mean(., trim = .2), ~ median(., na.rm = TRUE))
## This warning is displayed once per session.
```

```
#Convert all variables to numeric
DF2 <- DF2 %>% mutate_all(funs(as.numeric(.)))
```

```
## Warning: NAs introduced by coercion
```

```
#Scale the data so that no variable has undue influence
DF2 <- as.data.frame(scale(DF2))

#Replace missing values with average score EG - zero
DF2 <- DF2 %>% mutate_all(funs(ifelse(is.na(.) == TRUE, 0, .)))
set.seed(1)
fit3 <- kmeans(DF2, 2)
fit3$cluster
```

```
##  [1] 1 1 1 1 1 1 1 1 1 1 1 1 2 2 1 1 1 1 2 1 1 1 1 1 1 1 1 1 2 1 1 2 1 1 1
## [36] 1 1 1 1 1 1 1 2 1 2 1 2 1 1 1
```

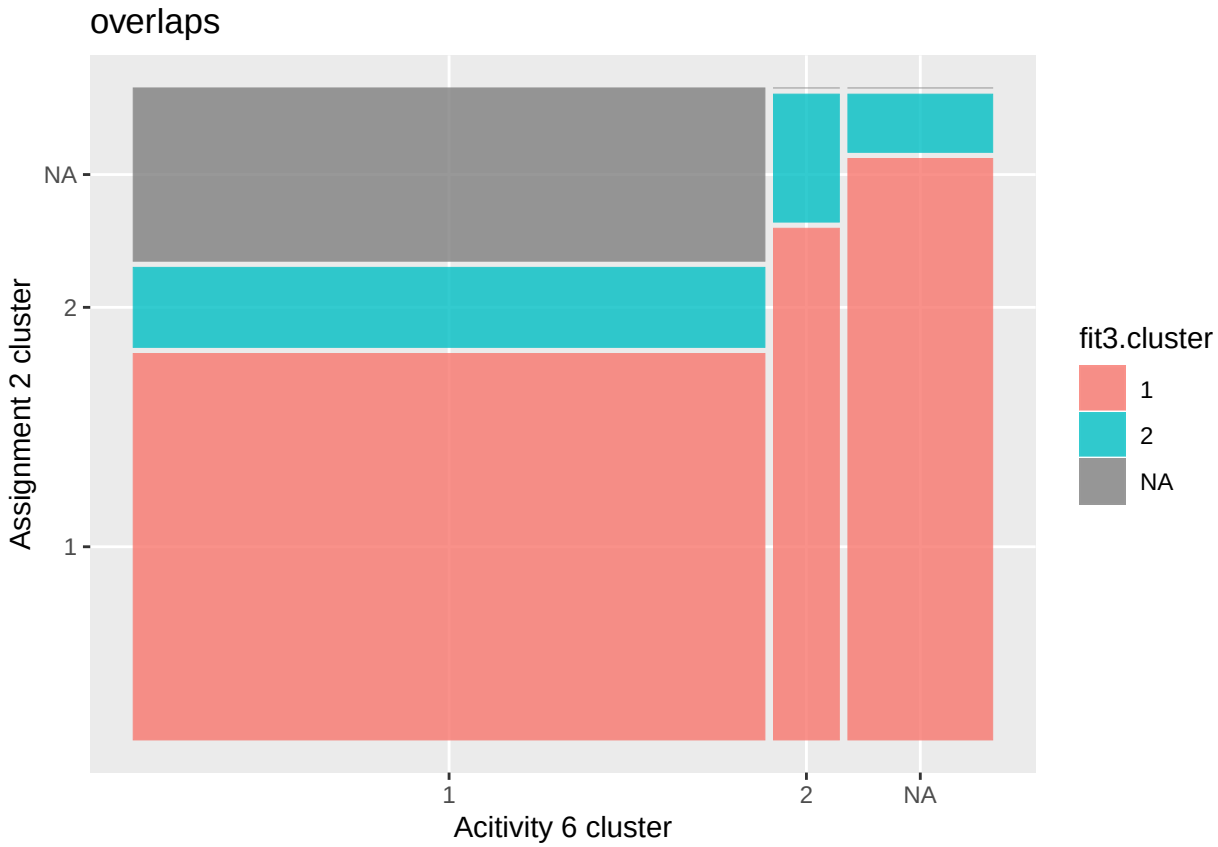
```
P6DF <- data.frame(DF1,fit3$cluster)
P6DF <- dplyr::select(P6DF,-Last.Name)
names(P6DF) <- c("FirstName","Q1","Q2","Q3","Q4","Q5","Q6","Q7","Q8","Q9","Q10","Q11","Q12","Q13","Q14"
combineDF <- full_join(A2DF,P6DF,by = "FirstName")
combineDF <- combineDF %>% mutate(fit2.cluster = factor(fit2.cluster, exclude = NULL),fit3.cluster = fa
combineSummary <- combineDF %>% group_by(fit2.cluster,fit3.cluster) %>% summarise("Freq" = n())
combineSummary
```

```
## # A tibble: 7 x 3
## # Groups:   fit2.cluster [3]
##   fit2.cluster fit3.cluster  Freq
##   <fct>        <fct>        <int>
## 1 1            1               29
## 2 1            2                6
## 3 1            <NA>            13
## 4 2            1                4
## 5 2            2                1
## 6 <NA>         1               10
## 7 <NA>         2                1
```

```
ggplot(combineSummary)+geom_mosaic(aes(weight = Freq, x = product(fit2.cluster),fill=fit3.cluster))+ lab
```

```
## Warning: `parse_quosure()` is deprecated as of rlang 0.2.0.
```

```
## Please use `parse_quo()` instead.
## This warning is displayed once per session.
```




```
P1 <- structable(combineSummary$fit3.cluster ~ combineSummary$fit2.cluster)
```