Stats 199 Research

Justin Yee

April 18, 2019

Goal of a Question: produce large variability and a large percentage of wrongs; a question that discriminates between students, and figure out why: is the question confusing or something different in the students?

Distractors of question will be different ways of being wrong: different answer choices matter; not just right or wrong

Clusters of types of students: What type of questions are they missing? Are they choosing similar responses to other students? Clustering to see this...

Psychometrics: Don't want to ask questions if they don't give you additional info

Clusters of Students vs Clusters of Questions

Which questions are picking up the same kind of knowledge vs unique knowledge

Unique types of students** MOST Interested here

Always about 10% of the students who fail

HOW TO IDENTIFY AND HELP THE LOWER 10% OF THE CLASS

What do the clicker questions tell about the students in the class?

THINGS TO STUDY:

PLACE TO START: used in advertising/marketing *Correspondence Analysis: Pushing multidimentionsal into 2D Graphs,* Homols: homogonized alternative least squares: Clusters in categorical data: more sophisticated way of measuring distance between answers

Multivariate anlysis in the title for a textbook to look for: Chapter on correspondence analysis: related to PCA, cluster analysis...

Biplots: Greenacre

Given 3 weeks to buy the clickers

5 PAGES FOR THE RESEARCH PAPER

Attached is a sample data set that represents responses from one lecture (from Week 2, Lecture 1—there were two lectures per week). Each lecture has a similar file. Clicker participation is effectively optional, since the points count for very little and so some students choose to not participate.

Total is total number of questions answered. Percentate is the percentage of questions answered

Q2_Key_B: contains responses for question 2. The correct answer is B. NV means "no value" which means no response was recorded. Q2.2_Key_B: if a low percent of students got the question right, we discussed it and then answered the question again (and sometimes a third time). This contains the answers for the second attempt at question 2.
Q10_NOKEY: contains responses for question 10, which had no right or wrong answer. Still, it might be interesting to see if a choice on a "no key" question aligns with students who get more correct than incorrect, or otherwise correlates with a particular type of response on other questions.

Some questions I'd like you to consider (on this and on a merge of all clicker questions)

a) Are there groups of students who consistently get questions wrong?

b) The "best" I can ever get seems to be about 90% correct. This means that even on questions I consider very easy, 10% get it wrong. Are those in the 10% just guessing at an answer? Or are they really getting it wrong?
c) Which questions are hardest?
d) What happens when we ask questions a second (or third) time? Are some responses more likely to switch to the correct answer than others? What percent of correct-answers switch to wrong?
e) I imagine there is a block of students, probably a fairly large block, that get most questions correct. How can we identify this block? What do we learn from their incorrect answers? For example, do they all seem to get the same ones wrong? If so, do they tend to choose similar wrong answers? Same thing for the block of students who get most answers incorrect (if a block exists). What do we learn from their correct answers?
f) Can you identify "types" of student responses? In other words, do the responses cluster in some way, and what defines these clusters?

A good place to start is with this fairly small file. I'm working on anonymizing the files so we can merge them. However, it seems that some entries in each file will be impossible to match because they didn't register their clickers correctly.

Hi, Attached are the complete data for the project. My advice for this week is to focus on descriptive statistics (including graphics) and not dive into any modeling just yet, although we can discuss ideas. Ultimately it would be good to merge the data, but might be best to also study it file by file before going deeper.

Note that the last column contains a unique ID that can be used to merge files.

Finally, note that some students have multiple responses for unknown reasons, but that some of these are clearly not legitimate (mostly if not completely empty.) They should be deleted.

Merging the data based on key 'researchID'

```r
library(dplyr)
```

```
##
## 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
```

```r
library(purrr)
library(reshape)
```

```
##
## Attaching package: 'reshape'

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

```r
library(reshape2)
```

```
##
## Attaching package: 'reshape2'

## The following objects are masked from 'package:reshape':
##
##     colsplit, melt, recast
```

```r
library(tidyverse)
```

```
## -- Attaching packages ---------------------------------------------------------------------

## v ggplot2 3.1.0     v readr   1.3.0
## v tibble  1.4.2     v stringr 1.3.1
## v tidyr   0.8.2     v forcats 0.3.0

## -- Conflicts ------------------------------------------------------------------------------
## x tidyr::expand()   masks reshape::expand()
## x dplyr::filter()   masks stats::filter()
## x dplyr::lag()      masks stats::lag()
## x reshape::rename() masks dplyr::rename()
```

```r
library(tidyr)
#Testing a full merge with the key = one column

sample <- data.frame(one = factor(c("a","b","c","d")), two = factor(c("1","2","3","4")))

sample2 <- data.frame(one = factor(c("b")), two = factor(c("1")), three = factor(c("5")))

sample3 <- data.frame(one = c("a","c"), two = c("1","2"), three = c("1","2"), four = c("1","2"))

#total_data <- merge_all(dfs = list(sample,sample2), by = "one")

total_data2 <- list(sample,sample2,sample3) %>% reduce(full_join, by = "one")
```

```
## Warning: Column `one` joining factors with different levels, coercing to
## character vector

## Warning: Column `one` joining character vector and factor, coercing into
## character vector
```

```r
total_data3 <- list(sample3,sample2,sample) %>% reduce(full_join, by = "two")
```

```
## Warning: Column `two` joining factors with different levels, coercing to
## character vector

## Warning: Column `two` joining character vector and factor, coercing into
## character vector
```

```r
#Real Merge - I figured out the problem - Multiple IDS in each sheet, must filter
total_data <- list(session1.2, session2.1,session2.2,
 session3.1, session6.1, session8.1, session13,
session14, session16, session17) %>% reduce(full_join, by = "researchID")
```

```
## Warning: Column `researchID` joining factors with different levels,
## coercing to character vector

## Warning: Column `researchID` joining character vector and factor, coercing
## into character vector

## Warning: Column `researchID` joining character vector and factor, coercing
## into character vector

## Warning: Column `researchID` joining character vector and factor, coercing
## into character vector
```

```
## Warning: Column `researchID` joining character vector and factor, coercing
## into character vector

## Warning: Column `researchID` joining character vector and factor, coercing
## into character vector

## Warning: Column `researchID` joining character vector and factor, coercing
## into character vector

## Warning: Column `researchID` joining character vector and factor, coercing
## into character vector
```

```
nrow(total_data[total_data$researchID=="ab70",])
```

```
## [1] 5696
```

```
test_data <- full_join(session1.2,session2.1, by = "researchID")


#I want the column with the least amount of NV responses: filter the total data
#Want to look at each unique ID rows, then perform summation of NV count row wise
uniqueID <- unique(total_data$researchID)

uniqueID <- uniqueID[order(uniqueID)]

matrix_total <- as.matrix(total_data)

logical_matrix <- matrix_total == "NV"

count_matrix <- apply(logical_matrix,1,sum,na.rm=TRUE)

#total_count is a dataframe now has the column name count_matrix which counts the number of NV vals
total_count <- as.data.frame(cbind(matrix_total,count_matrix))

total_count$count_matrix <- as.integer(total_count$count_matrix)

#Returns the minimum number of NV's for a unique researchID
min_NV <- total_count %>% group_by(researchID) %>% summarise(min = min(count_matrix,na.rm=TRUE))

clean_total2 <- data.frame()

#Now the uniqueIDs match with the min_NV tibble
for(i in 1:61){
  #For every uniqueID, return the row with the min_NV
  clean_total <- total_count[total_count$researchID==uniqueID[i],] %>% filter(count_matrix ==  min_NV$mi

  clean_total2 <- rbind(clean_total2,clean_total)
}


#clean_total2 is the best result so far: Now down to 88 observations, just need to look at the duplicat
#There is an issue of there being multiple answers for a single type of question
#write.csv(clean_total2,file = "total_data.csv")

#Looking at the non-duplicates only
```

```r
freq_df <- data.frame(table(clean_total2$researchID))

#gives names of the non-duplicates, subsets clean_total2
clean_no_dup <- clean_total2 %>% filter(researchID %in%  freq_df$Var1[which(freq_df$Freq==1)])
```

Now, we will work with clean_no_dup… Check with Others: 97 total questions, 50 no-dup students

```r
#Use regex to get the correct answers
library(stringr)
library(ggplot2)
library(knitr)
library(xtable)
library(stargazer)
```

```
##
## Please cite as:
##  Hlavac, Marek (2018). stargazer: Well-Formatted Regression and Summary Statistics Tables.
##  R package version 5.2.2. https://CRAN.R-project.org/package=stargazer
```

```r
library(gridExtra)
```

```
##
## Attaching package: 'gridExtra'
## The following object is masked from 'package:dplyr':
##
##     combine
```

```r
clean_total2<-read.csv("total_data.csv", header=TRUE)
#Looking at the non-duplicates only
freq_df <- data.frame(table(clean_total2$researchID))

#gives names of the non-duplicates, subsets clean_total2
clean_no_dup <- clean_total2 %>% filter(researchID %in%  freq_df$Var1[which(freq_df$Freq==1)])


#Getting rid of unneccessary columns
clean_no_dup <- clean_no_dup[,c(8,118, which( clean_no_dup %>% names() %>% str_extract(pattern = "Q") ==

#Tidying to Group by Question Difficulty
clean_no_dup_tidy <- clean_no_dup %>% gather("Question","Response",-c(1,2))
```

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

```r
clean_no_dup_tidy$Response <- factor(clean_no_dup_tidy$Response, levels = c("A","B","C","D","E","NV"))

clean_no_dup_tidy$Question <- factor(clean_no_dup_tidy$Question)

#Key = B indices for tidy data

correct_total_vector<- rep(NA,4850)

correct_total_vector[which( clean_no_dup_tidy$Question %>% str_extract(pattern = "A") == "A")] <- "A"
```

```r
correct_total_vector[which( clean_no_dup_tidy$Question %>% str_extract(pattern = "B") == "B")] <- "B"

correct_total_vector[which( clean_no_dup_tidy$Question %>% str_extract(pattern = "C") == "C")] <- "C"

correct_total_vector[which( clean_no_dup_tidy$Question %>% str_extract(pattern = "D") == "D")] <- "D"

correct_total_vector[which( clean_no_dup_tidy$Question %>% str_extract(pattern = "_E") == "_E")] <- "E"

correct_total_vector[which( clean_no_dup_tidy$Question %>% str_extract(pattern = "yE") == "yE")] <- "E"

table(clean_no_dup_tidy$Response)
```

A B C D E NV 1052 1197 646 390 76 971

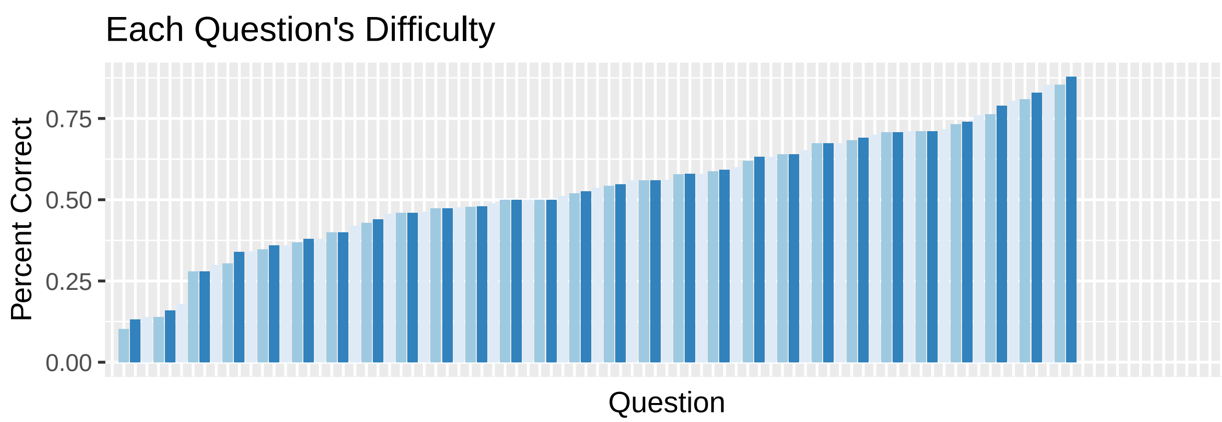
```r
#Total percent correct across All Questions and All Students = 51.87%
mean(correct_total_vector == clean_no_dup_tidy$Response, na.rm = TRUE)
```

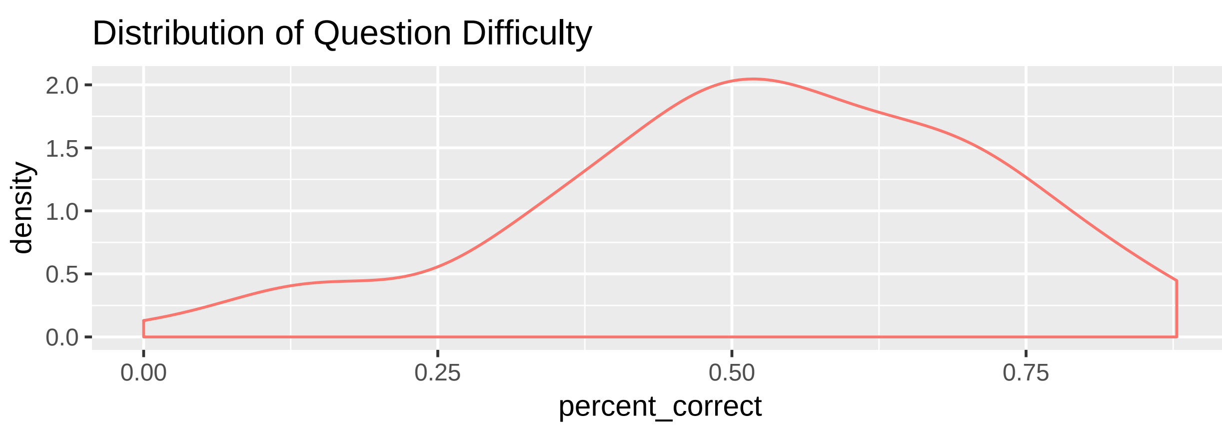
[1] 0.5186744

```r
#Total Percent Correct For Each Question
total_percent_correct <- data.frame(percent_correct = c(rep(NA,97)))

temp_total<- c()
for(i in seq(from = 50, to = 4850,by = 50)){
  temp_total[i] <- mean((correct_total_vector == clean_no_dup_tidy$Response)[(i-49):i],na.rm=TRUE)
}


temp_total<- temp_total[!(is.na(temp_total) & !is.nan(temp_total))]

total_percent_correct$percent_correct <- temp_total

#Now we have a dataframe for the percent correct for each question
total_percent_correct <- total_percent_correct %>% mutate(Question = unique(clean_no_dup_tidy$Question))

#Bar Plot revealing question difficulty
library(RColorBrewer)

colourCount = length(unique(clean_no_dup_tidy$Question))
getPalette = colorRampPalette(brewer.pal(9, "Blues"))

total_percent_correct <- total_percent_correct %>% arrange(percent_correct)


total_percent_correct$Question <- factor(total_percent_correct$Question, levels = total_percent_correct$


plot1 <- ggplot(data = total_percent_correct, aes(x = Question, y = percent_correct, fill = Question)) +
        axis.text.x=element_blank(),
        axis.ticks.x=element_blank()) + ylab(label="Percent Correct") + ggtitle("Each Question's Diffic
  scale_fill_manual(values = rep(c('#deebf7','#9ecae1','#3182bd'), 33, drop = FALSE))


#Density Plot of Question Difficulty
plot2 <- ggplot(data = total_percent_correct, aes(x=percent_correct, color = "darkorange")) + geom_dens
```

```r
grid.arrange(plot1,plot2)
```

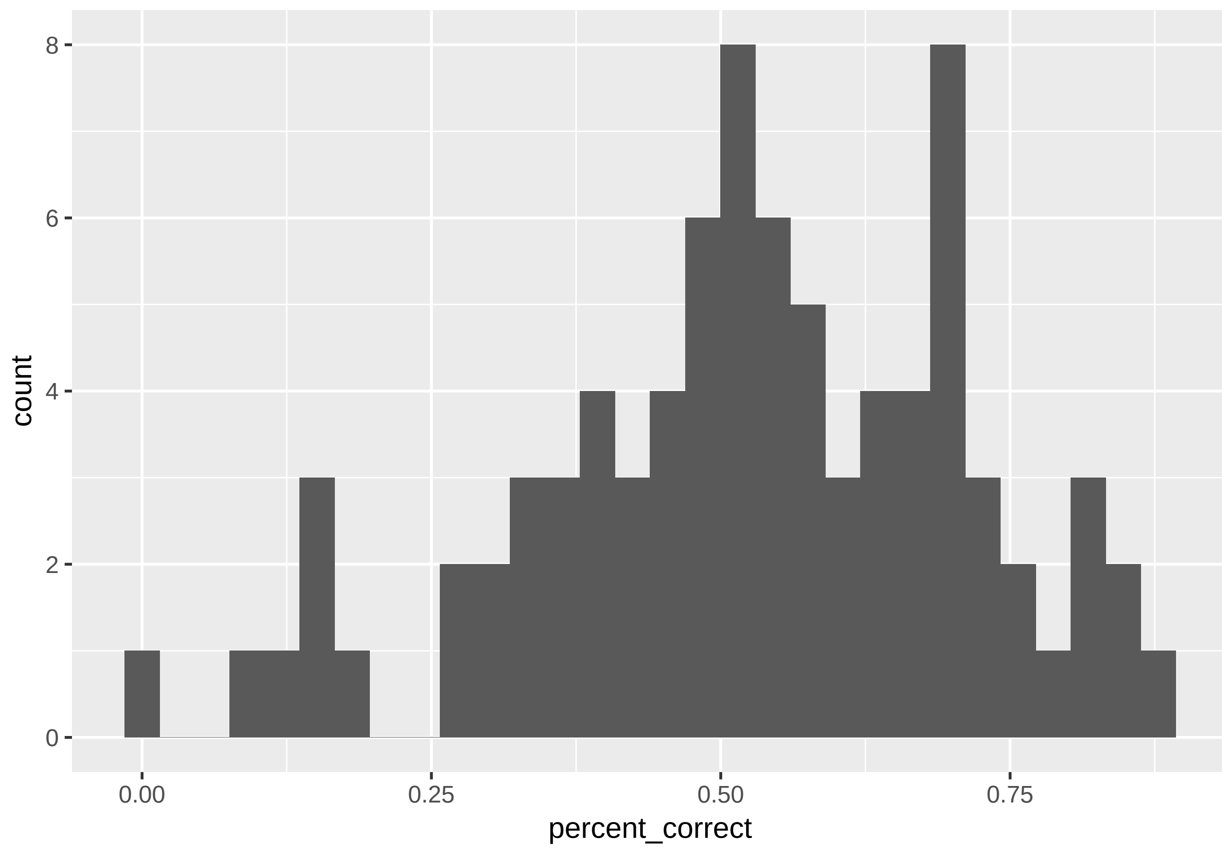
```
## Warning: Removed 13 rows containing missing values (geom_bar).
## Warning: Removed 13 rows containing non-finite values (stat_density).
```






```r
#Histogram of Question Difficulty
ggplot(data = total_percent_correct, aes(x=percent_correct)) + geom_histogram()
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## Warning: Removed 13 rows containing non-finite values (stat_bin).
```




```
##Summary Stats by Question

mean(total_percent_correct$percent_correct,na.rm=TRUE)
```

[1] 0.5253363

```
sd(total_percent_correct$percent_correct,na.rm=TRUE)
```

[1] 0.1905528

```
summary(total_percent_correct$percent_correct,na.rm=TRUE)
```

Min. 1st Qu. Median Mean 3rd Qu. Max. NA's 0.0000 0.4150 0.5315 0.5253 0.6739 0.8780 13

```
stargazer(total_percent_correct[1,])
```

% Table created by stargazer v.5.2.2 by Marek Hlavac, Harvard University. E-mail: hlavac at fas.harvard.edu
% Date and time: Sat, Jun 08, 2019 - 6:51:34 PM

Table 1:

Statistic	N	Mean	St. Dev.	Min	Pctl(25)	Pctl(75)	Max
percent_correct	1	0.000		0	0	0	0

```
stargazer(total_percent_correct$percent_correct,na.rm=TRUE)
```

% Table created by stargazer v.5.2.2 by Marek Hlavac, Harvard University. E-mail: hlavac at fas.harvard.edu
% Date and time: Sat, Jun 08, 2019 - 6:51:34 PM

Table 2:

0	0.102	0.132	0.140	0.140	0.160	0.180	0.280	0.280	0.300	0.304	0.340	0.341	0.347

% Table created by stargazer v.5.2.2 by Marek Hlavac, Harvard University. E-mail: hlavac at fas.harvard.edu
% Date and time: Sat, Jun 08, 2019 - 6:51:34 PM

Table 3:

TRUE

```
stargazer(total_percent_correct)
```

% Table created by stargazer v.5.2.2 by Marek Hlavac, Harvard University. E-mail: hlavac at fas.harvard.edu
% Date and time: Sat, Jun 08, 2019 - 6:51:34 PM

Table 4:

Statistic	N	Mean	St. Dev.	Min	Pctl(25)	Pctl(75)	Max
percent_correct	84	0.525	0.191	0.000	0.415	0.674	0.878

Now, we will look at the total data with respective to grouping by student

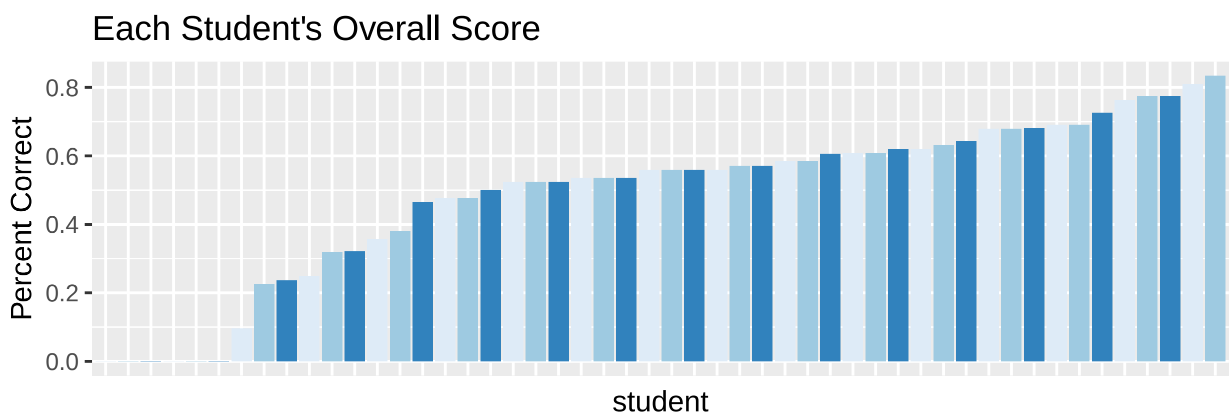
```
#SUBSET EXAMPLE ##########################################################

#Percent Correct Overall from Question Arrangement
mean(correct_total_vector == clean_no_dup_tidy$Response, na.rm = TRUE)
```

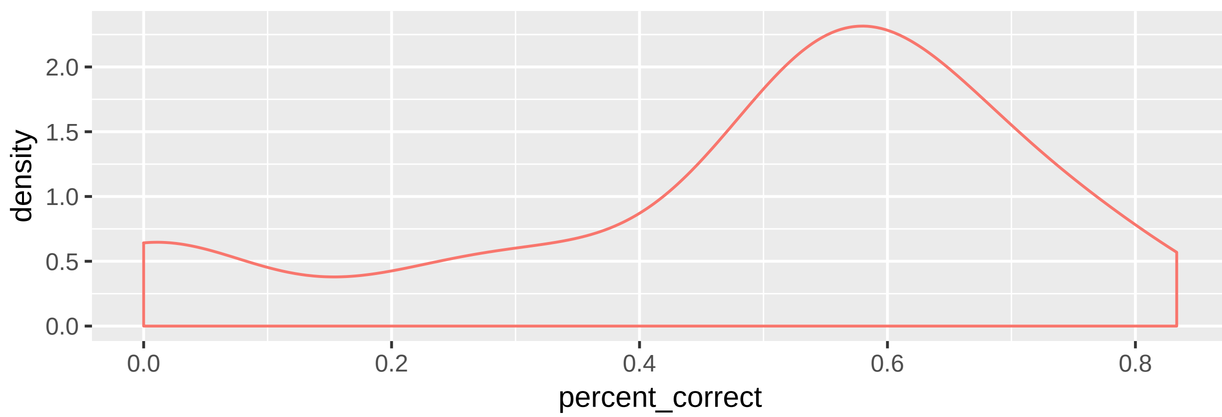
[1] 0.5186744

```
clean_no_dup_student <- clean_no_dup_tidy %>% arrange(researchID)

correct_total_vector2 <- rep(NA,4850)

correct_total_vector2[which( clean_no_dup_student$Question %>% str_extract(pattern = "A") == "A")] <- "

correct_total_vector2[which( clean_no_dup_student$Question %>% str_extract(pattern = "B") == "B")] <- "

correct_total_vector2[which( clean_no_dup_student$Question %>% str_extract(pattern = "C") == "C")] <- "

correct_total_vector2[which( clean_no_dup_student$Question %>% str_extract(pattern = "D") == "D")] <- "

correct_total_vector2[which( clean_no_dup_student$Question %>% str_extract(pattern = "_E") == "_E")] <-

correct_total_vector2[which( clean_no_dup_student$Question %>% str_extract(pattern = "yE") == "yE")] <-



#Percent Correct Overall from Student arrangement check YES
mean(correct_total_vector2 == clean_no_dup_student$Response, na.rm = TRUE)
```

[1] 0.5186744

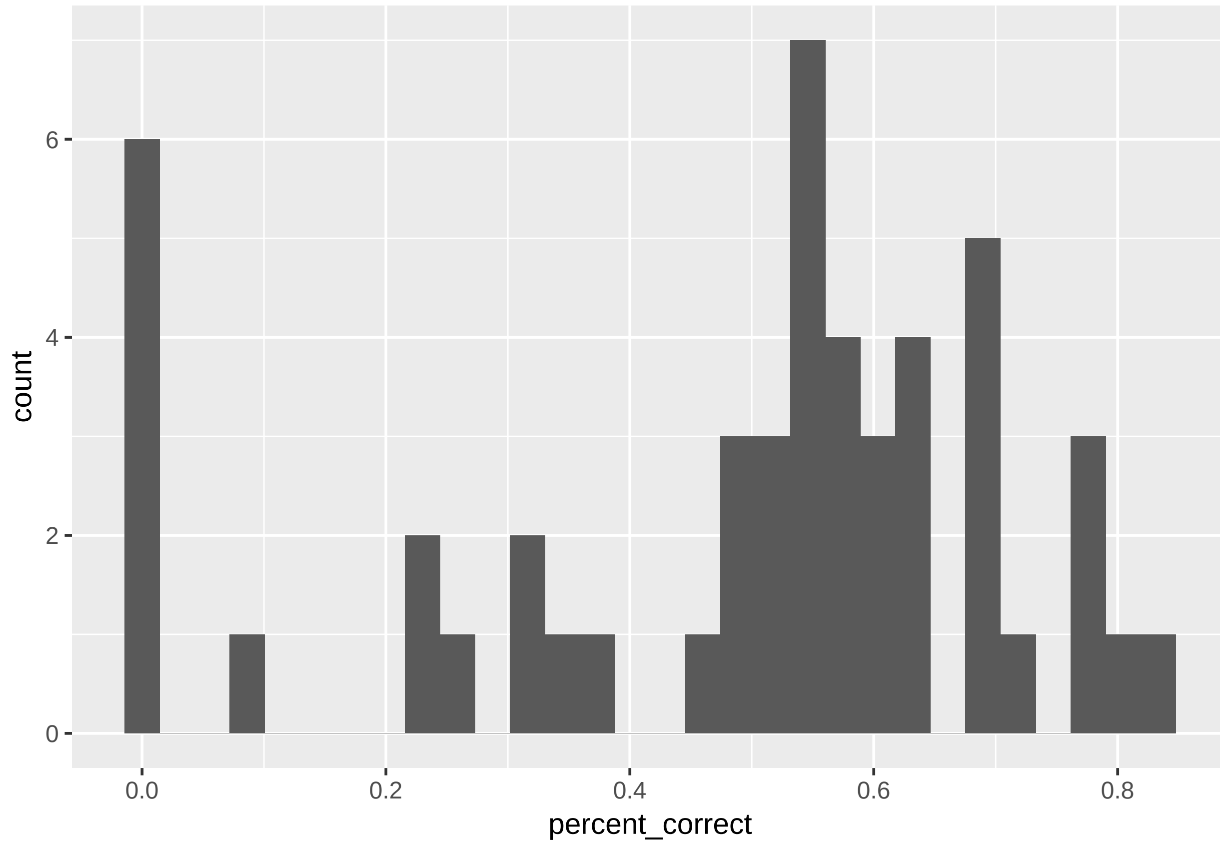
```r
#New Student every 16 rows, 47 students total


temp_total_student<- c()
for(i in seq(from = 97, to = 4850,by = 97)){
  temp_total_student[i] <- mean((correct_total_vector2 == clean_no_dup_student$Response)[(i-96):i],na.r
}

percent_correct_student_total<-data.frame(percent_correct=c(rep(NA,50)),student=unique(clean_no_dup_stu

percent_correct_student_total$percent_correct<- temp_total_student[c(which(!is.na(temp_total_student)),


#percent_correct_student is correct by checking the first student (correct_vector2 == session2.1_tidy_s


percent_correct_student_total$student <- factor(percent_correct_student_total$student, levels = percent_

#Bar plot revealing student level
studentbarplot1 <- ggplot(data = percent_correct_student_total, aes(x = student, y = percent_correct, fi
        axis.text.x=element_blank(),
        axis.ticks.x=element_blank()) + ylab(label="Percent Correct") + ggtitle("Each Student's Overall
  scale_fill_manual(values = rep(c('#deebf7','#9ecae1','#3182bd'), 33, drop = FALSE))


#Density Plot of Student Levels
studentdensityplot <- ggplot(data = percent_correct_student_total, aes(x=percent_correct, col = "darkora

grid.arrange(studentbarplot1,studentdensityplot)
```






```r
#Histogram of Student Levels
ggplot(data = percent_correct_student_total, aes(x=percent_correct)) + geom_histogram()
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```




```
mean(percent_correct_student_total$percent_correct)
```

[1] 0.4851522

```
sd(percent_correct_student_total$percent_correct)
```

[1] 0.2369275

```
summary(percent_correct_student_total$percent_correct)
```

Min. 1st Qu. Median Mean 3rd Qu. Max. 0.0000 0.3629 0.5595 0.4852 0.6280 0.8333

```
stargazer(percent_correct_student_total)
```

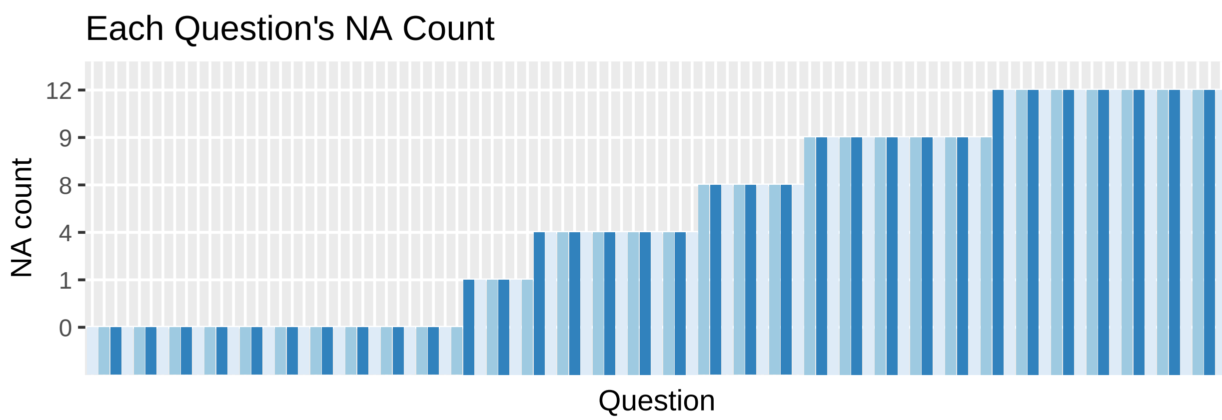
% Table created by stargazer v.5.2.2 by Marek Hlavac, Harvard University. E-mail: hlavac at fas.harvard.edu
% Date and time: Sat, Jun 08, 2019 - 6:51:35 PM

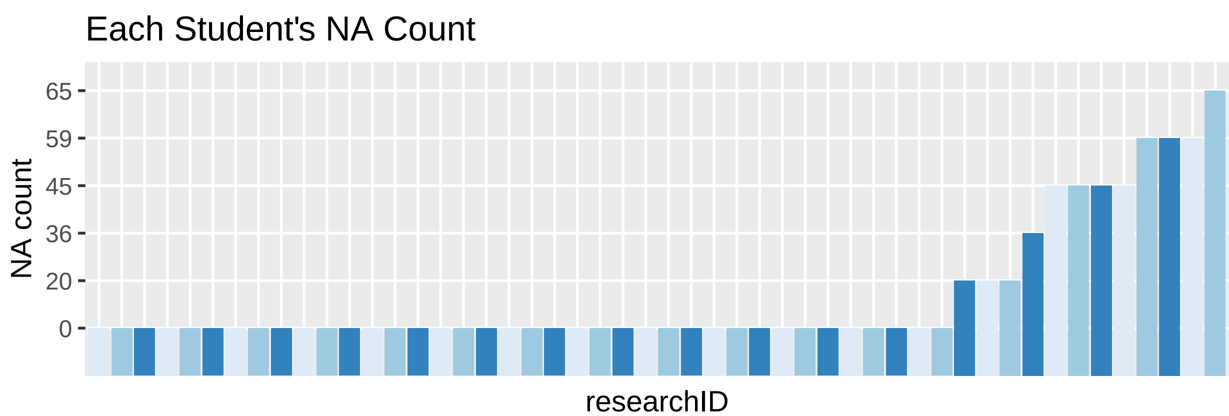
Table 5:

Statistic	N	Mean	St. Dev.	Min	Pctl(25)	Pctl(75)	Max
percent_correct	50	0.485	0.237	0.000	0.363	0.628	0.833

```
#Looking at the count of NA responses in students and in questions

#By Question
#Creating new logical column if NA response or not
question_na <- clean_no_dup_tidy %>% group_by(Question) %>% mutate(N_A = is.na(Response))
```

```r
question_na <- question_na %>% group_by(Question) %>% mutate(N_A_count = sum(N_A))

question_na_count <- distinct(question_na, Question, N_A_count)

question_na_count$Question <- factor(question_na_count$Question, levels = question_na_count$Question[ord

#By Student
student_na <- ungroup(question_na) %>% select(-N_A_count)

student_na <- student_na %>% group_by(researchID) %>% mutate(N_A_count = sum(N_A))

student_na_count <- distinct(student_na, researchID, N_A_count) %>% arrange(N_A_count)


student_na_count$researchID <- factor(student_na_count$researchID, levels = student_na_count$researchID


#Bar Plots
question_NA_plot <- ggplot(data = question_na_count, aes(x = Question, y = factor(N_A_count), fill = Que
        axis.text.x=element_blank(),
        axis.ticks.x=element_blank()) + ylab(label="NA count") + ggtitle("Each Question's NA Count")



student_NA_plot <- ggplot(data = student_na_count, aes(x = researchID, y = factor(N_A_count), fill = res
        axis.text.x=element_blank(),
        axis.ticks.x=element_blank()) + ylab(label="NA count") + ggtitle("Each Student's NA Count")


grid.arrange(question_NA_plot, student_NA_plot)
```






```
#Looking at students with 20 or more NA_count
#12 students
most_na_student <- student_na_count %>% filter(N_A_count >= 20)
```

AGNES cluster analysis - Aglomerrative Nesting - similar to K-means

Knot testing: Indicator variables: Chapter on splines

spaghetti plots in r

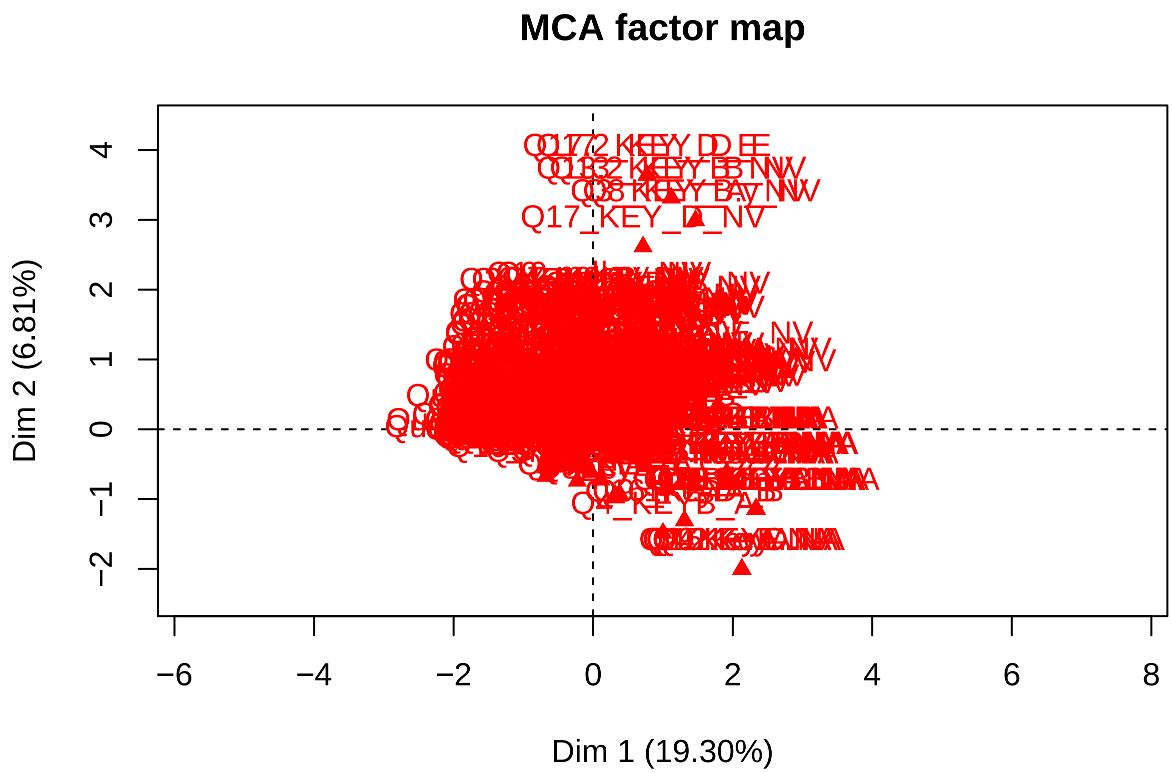
Transition Matrixes on repeated questions

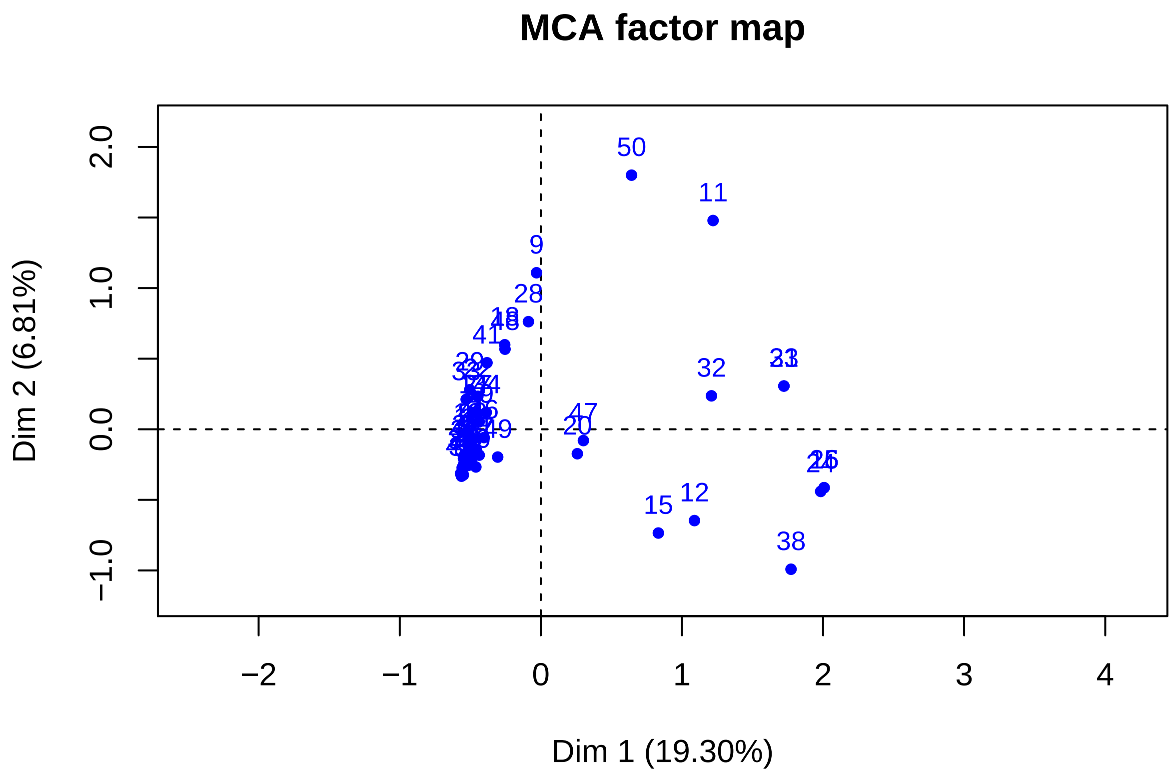
Correspondence Analysis Exploration

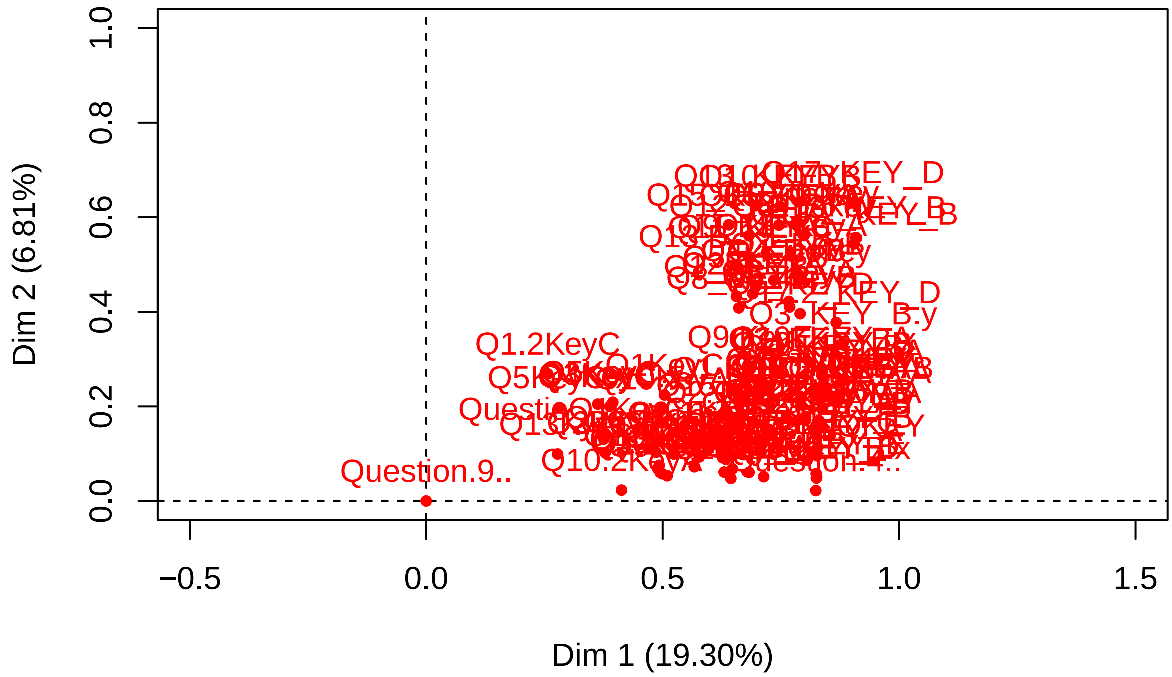
```
#Package FactoMineR
#clean_no_dup2 is the dataset with all questions as variables and count of NVs column


library(FactoMineR)
library(factoextra)
```

```
## Welcome! Related Books: `Practical Guide To Cluster Analysis in R` at https://goo.gl/13EFCZ
```

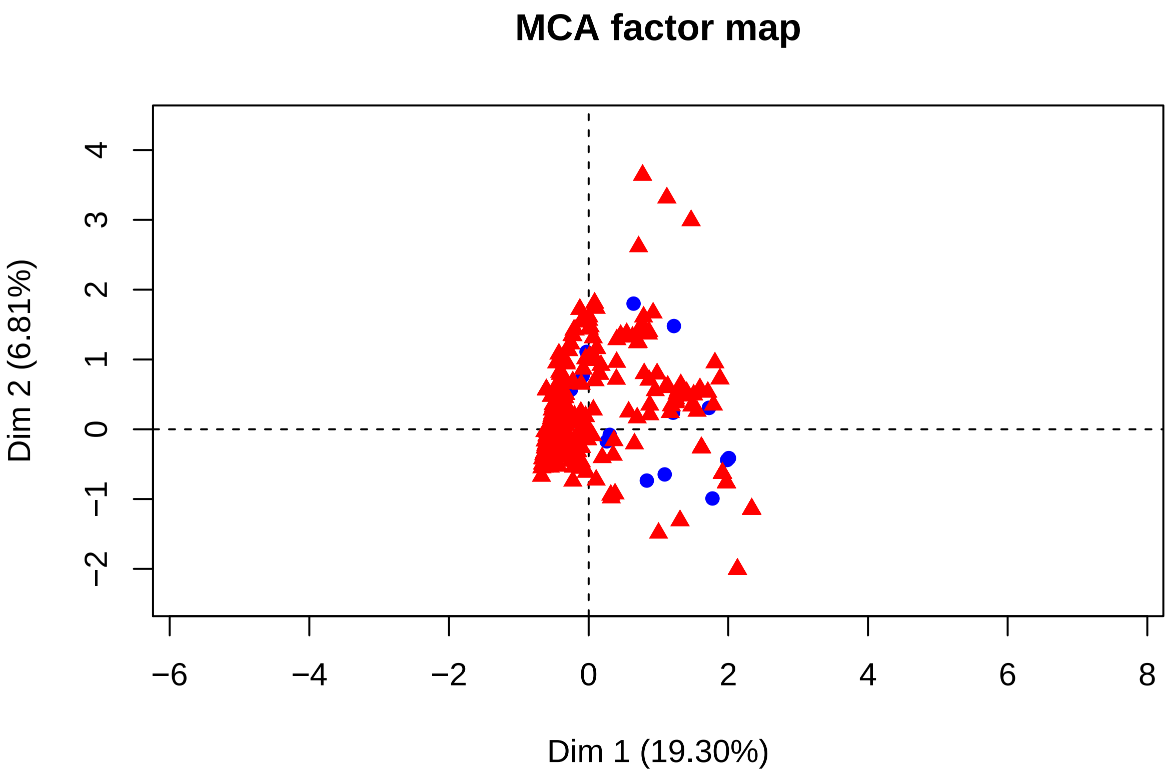
```
res<- MCA(clean_no_dup[,c(-1,-2)])
```


MCA factor map
Dim 2 (6.81%)
Dim 1 (19.30%)
−6
−4
−2
0
2
4
6
8
4
3
2
1
0
−1
−2
Q17_KEY_D_NV

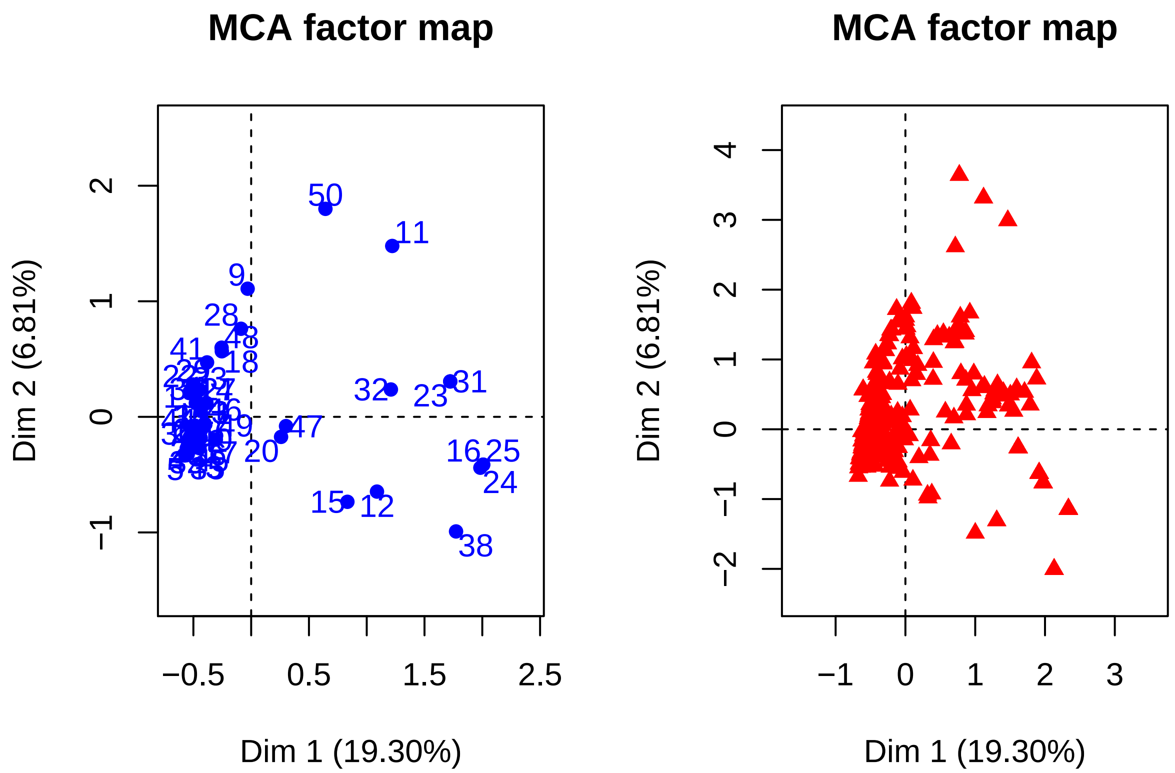

MCA factor map
Dim 1 (19.30%)
Dim 2 (6.81%)
50
11
9
28
32
15
12
38



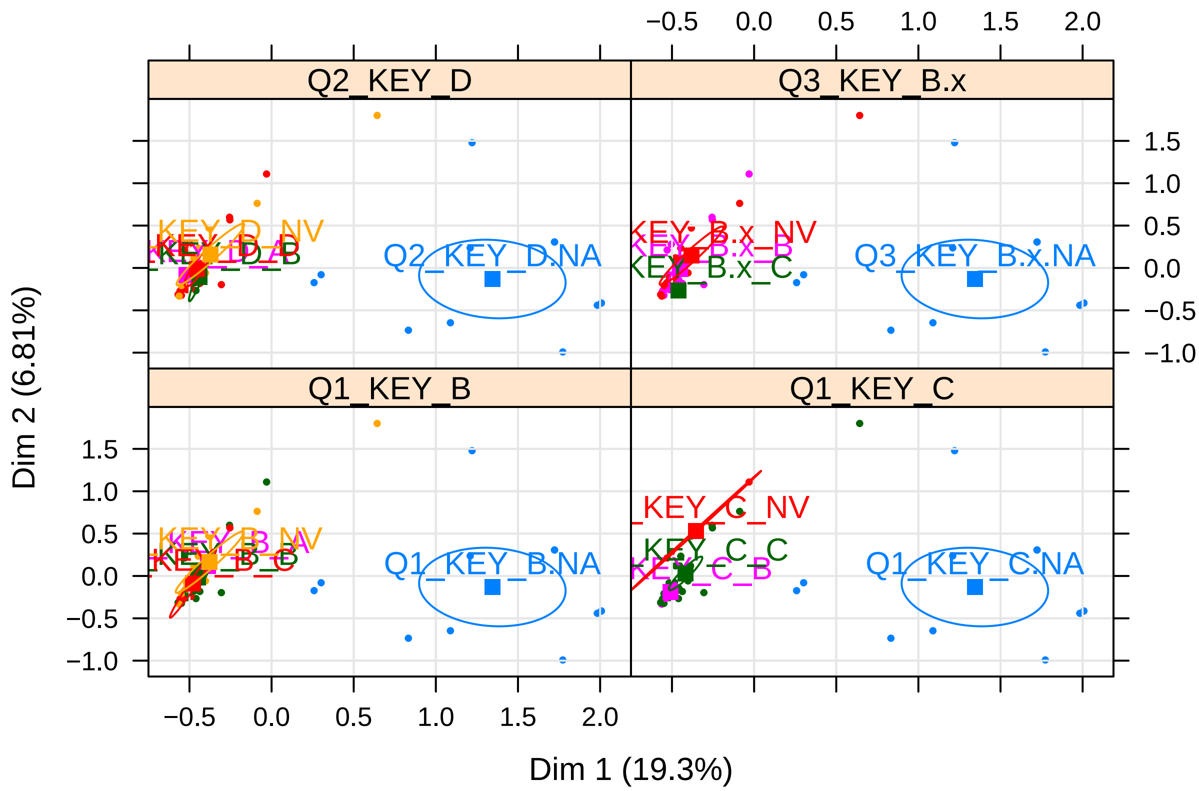

```r
#summary(res)

plot(res, label = "none")
```



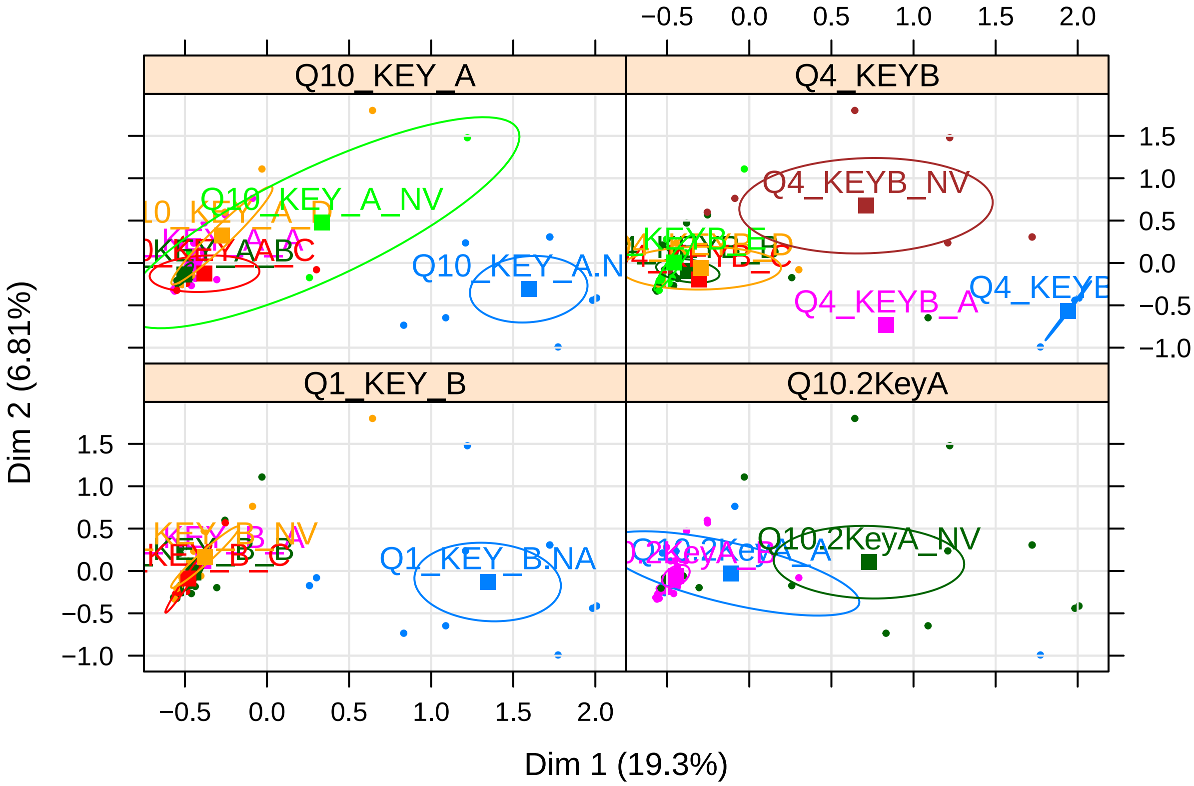

```r
par(mfrow=c(1,2))
plot(res, invisible = c("var"),autoLab = "y")
plot(res, invisible = c("ind"),label="none")
```



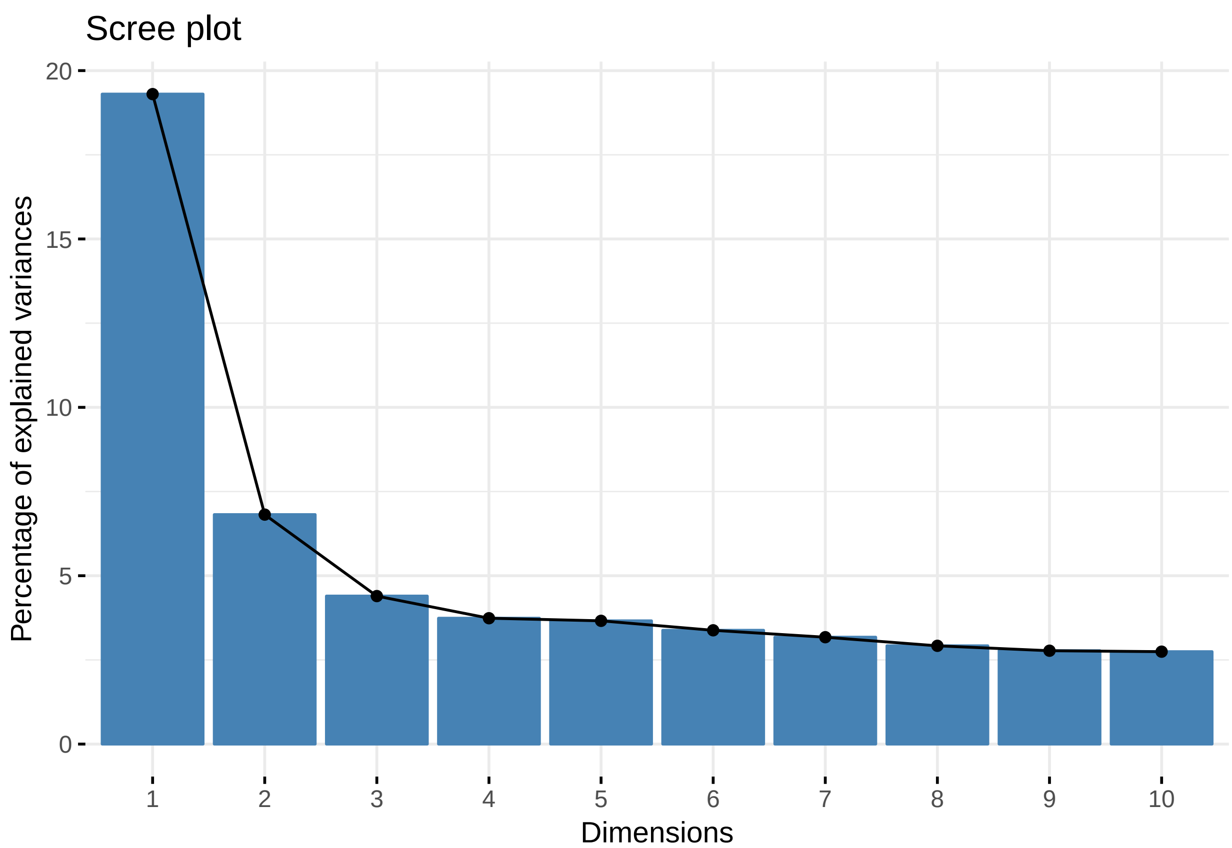

```r
plotellipses(res, keepvar= c(1:3,5))
```



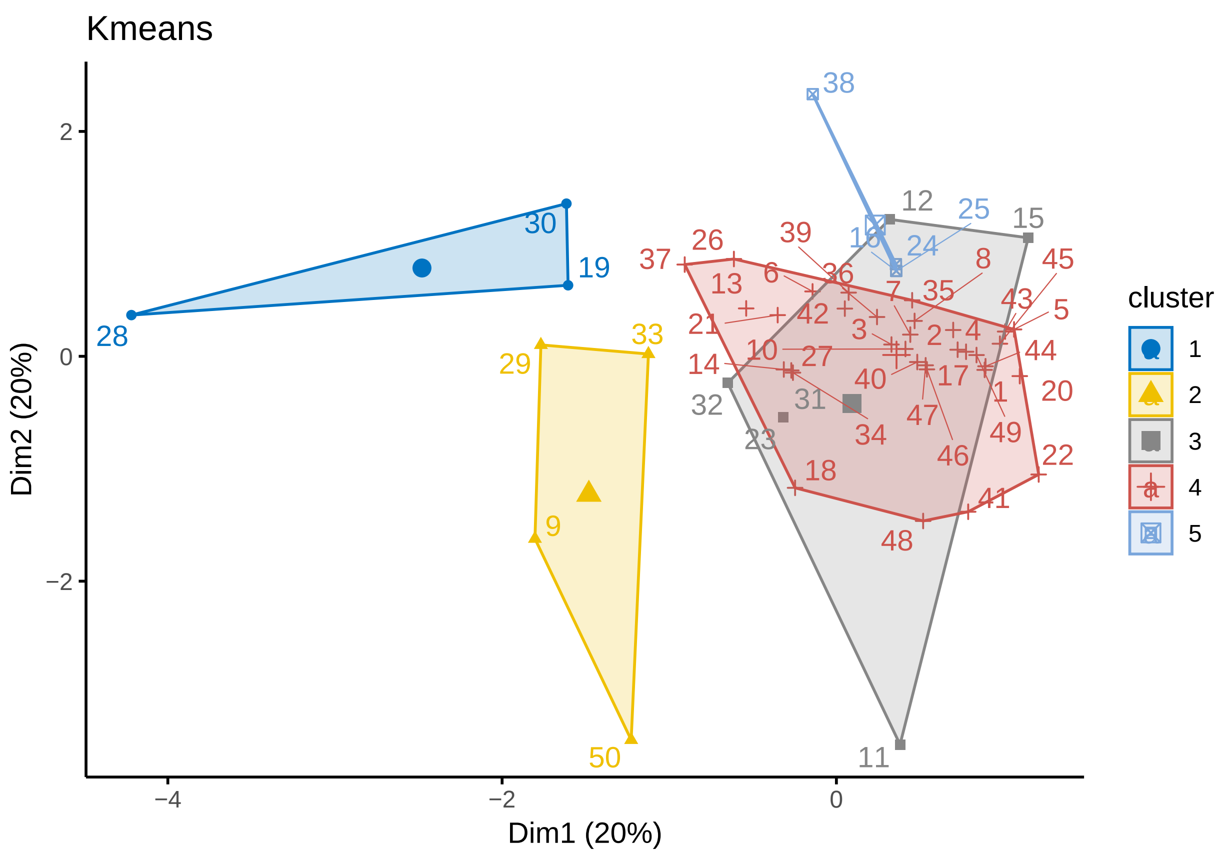

```
plotellipses(res, keepvar= c(1,30,48,90))
```



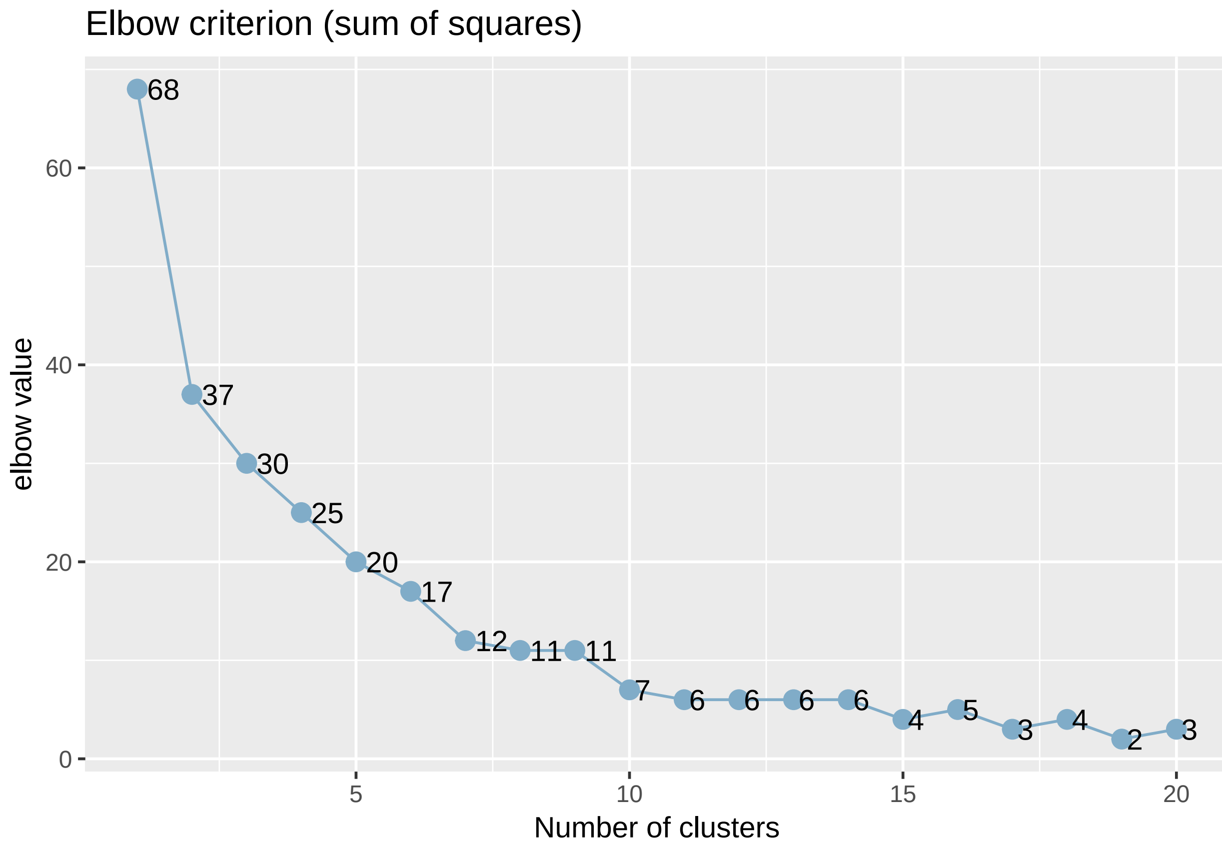

```r
fviz_screeplot(res)
```



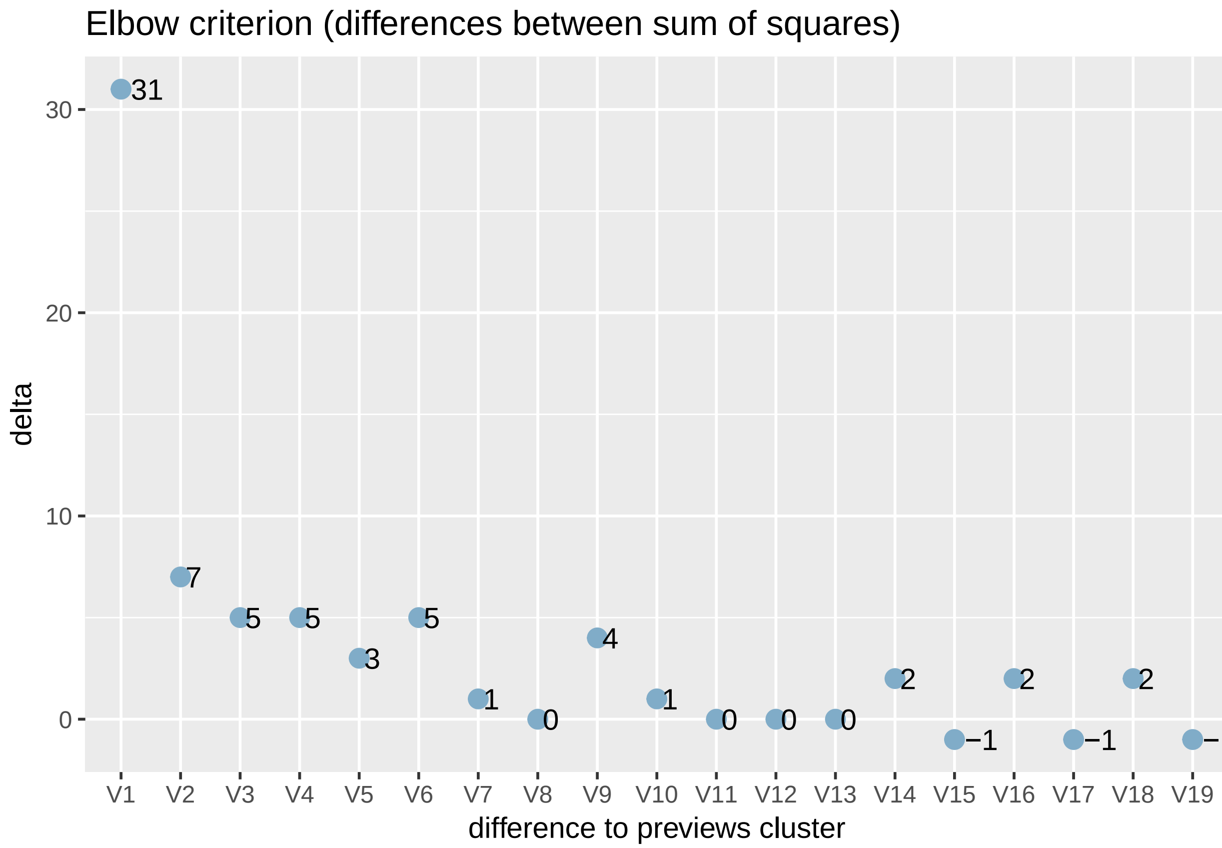

```r
#Gives coordinates for the individual students of the first 5 dimensions from MCA
km<- data.frame(res$ind$coord)

groupes.kmeans<- kmeans(km,centers = 5,nstart =5)

fviz_cluster(groupes.kmeans, data = km, palette = "jco", repel= TRUE, main = "Kmeans", ggtheme = theme_
```



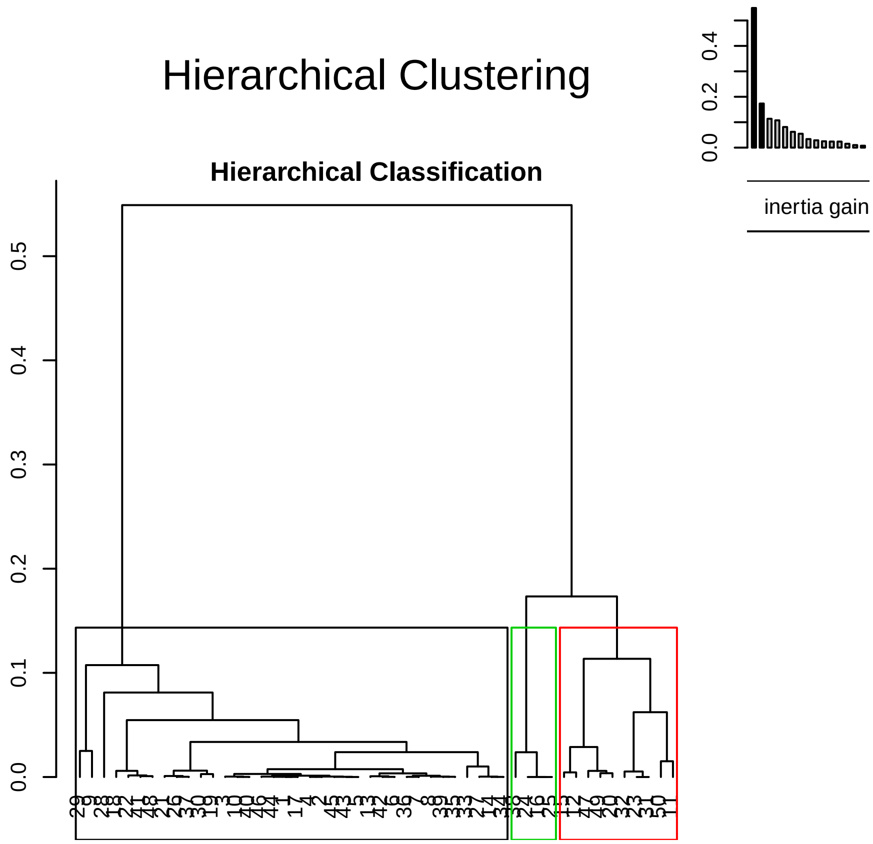

```
library(sjPlot)

sjc.elbow(km,steps=20,show.diff = T)
```

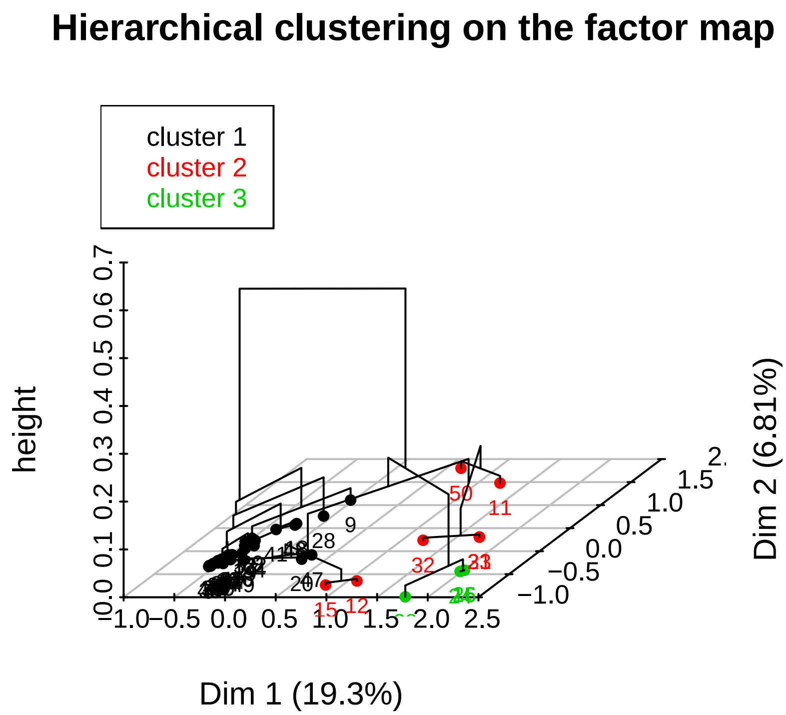


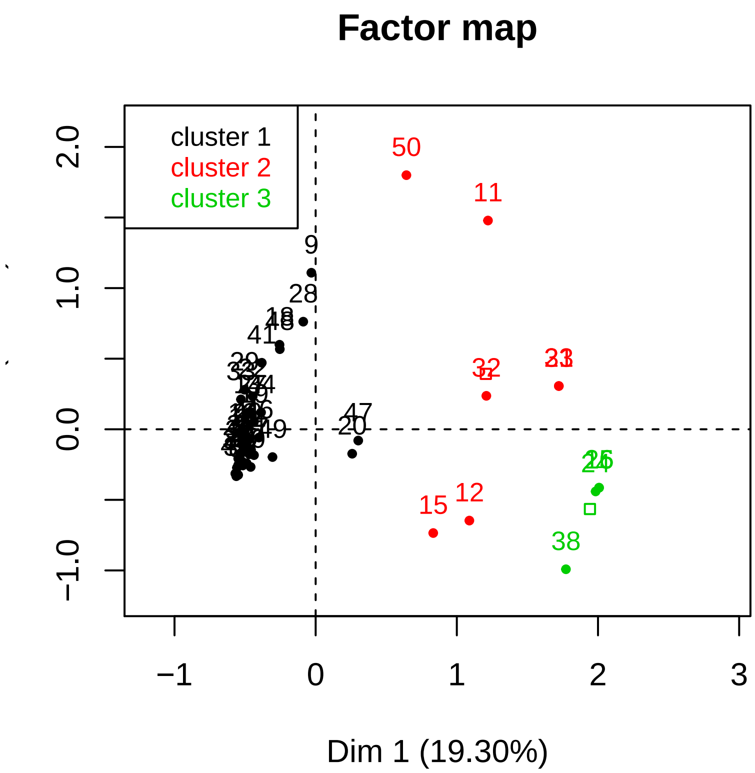

```
## geom_path: Each group consists of only one observation. Do you need to
## adjust the group aesthetic?
```



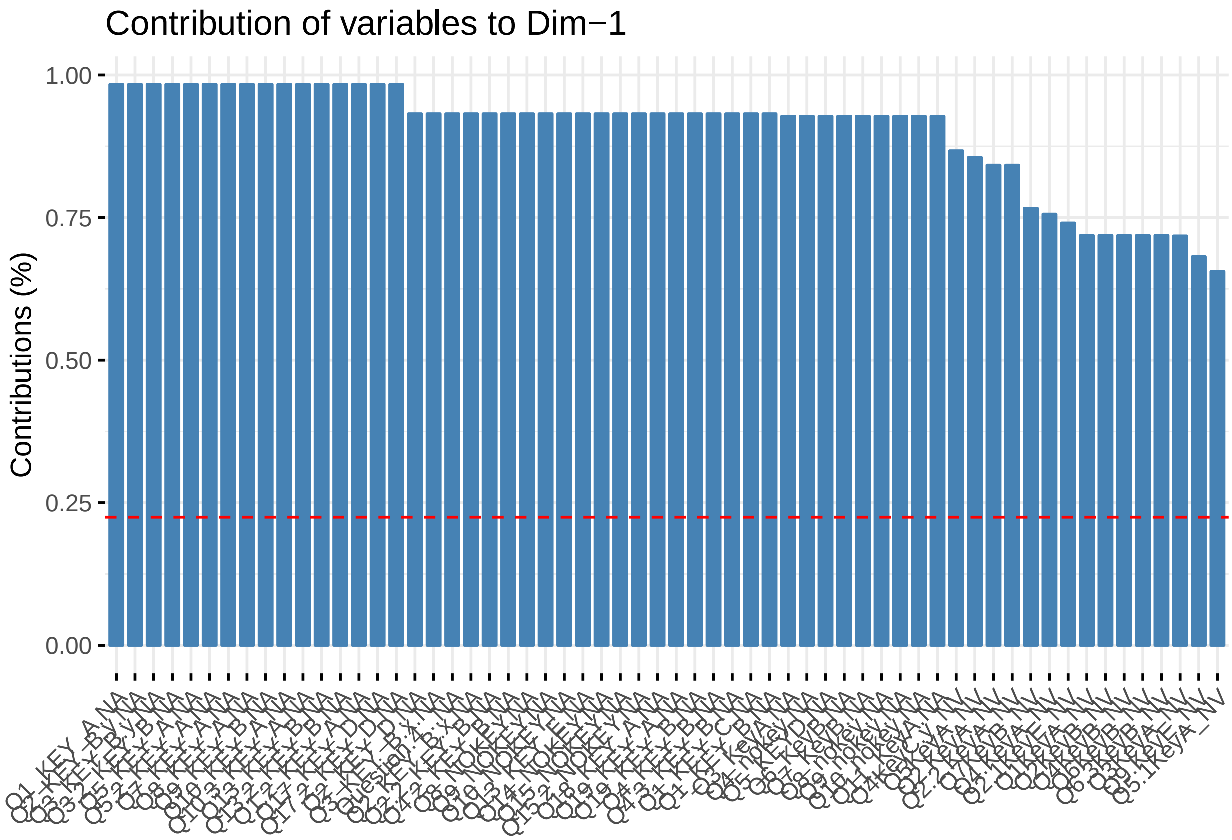

```
hier<- HCPC(res,nb.clust=-1)
```


Hierarchical Clustering
Hierarchical Classification
inertia gain
0.0
0.1
0.2
0.3
0.4
0.5

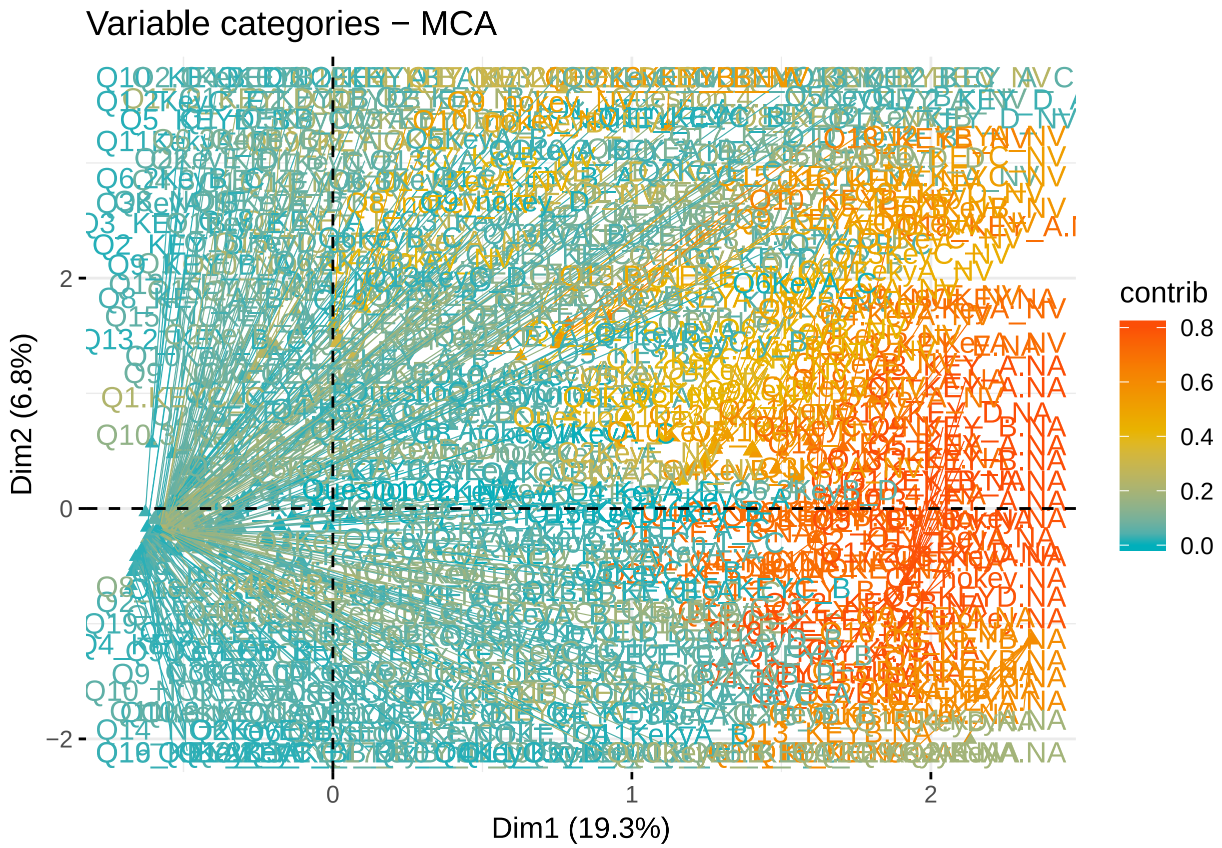

Hierarchical clustering on the factor map
cluster 1
cluster 2
cluster 3
height
Dim 1 (19.3%)
Dim 2 (6.81%)




```r
par(mfrow=c(2,1))
plot(hier, choice=c("3D.map, tree, bar"))


# Total contribution to dimension 1 and 2
fviz_contrib(res, choice = "var", axes = 1, top = 60)
```




```r
fviz_mca_var(res, col.var = "contrib",
             gradient.cols = c("#00AFBB", "#E7B800", "#FC4E07"),
             repel = TRUE, # avoid text overlapping (slow)
             ggtheme = theme_minimal()
             )
```

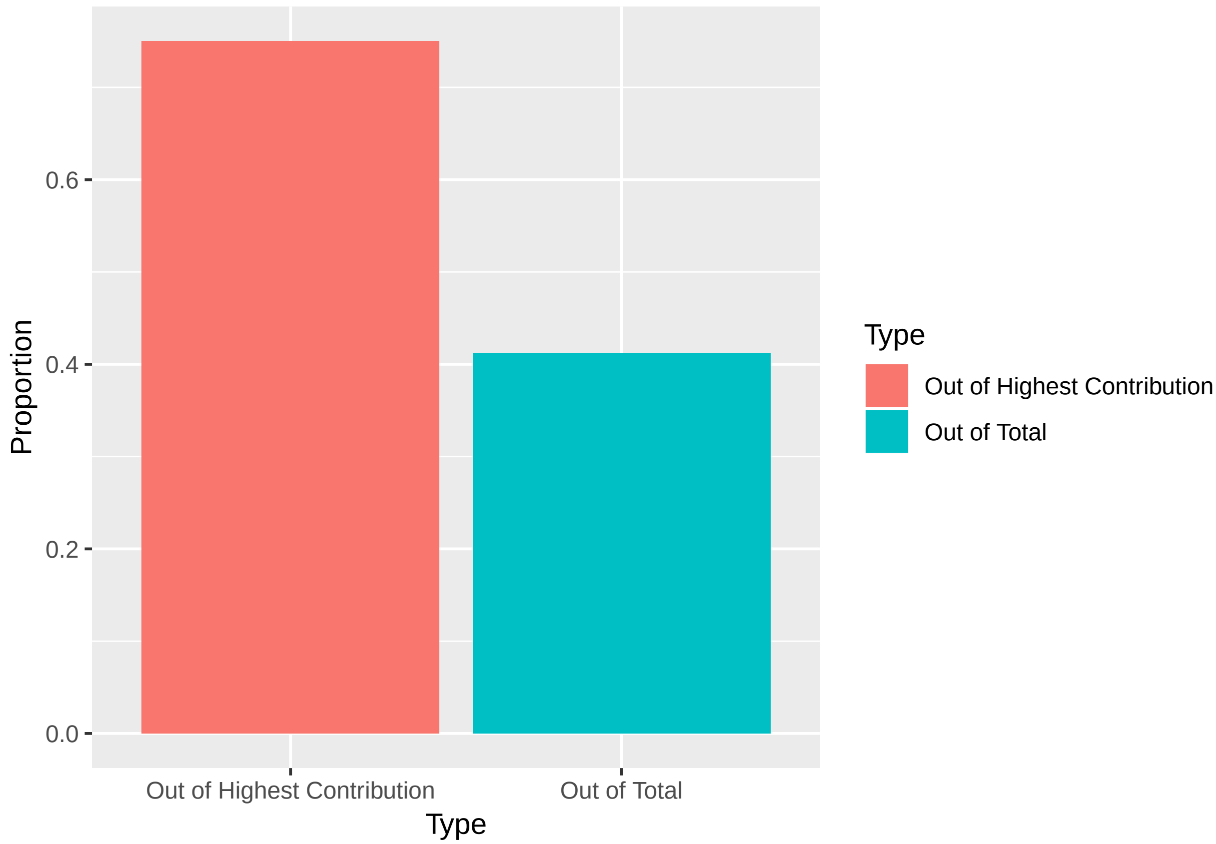



```r
var <- get_mca_var(res)

#The 16 Questions with the highest contribution to dimension 1
#They all have the same contribution -- why?

library(stringr)

most_contribution <- head(sort(var$contrib[,1], decreasing = TRUE),16)

most_contribution<- unlist(strsplit(names(most_contribution), split = ".NA"))
```

```r
#All The Questions That were Repeated
#index 34 is Q1.2_KEY_A, goes with #21 but was not seen among other questions, index 77 is Q1.2KeyC, go

Repeated_Questions <- clean_no_dup_student$Question[c(6:7,8:10,16:17,23:24,25:26,30:31,32:33,35:36,56:5

First_time <- clean_no_dup_student$Question[c(6,8,16,23,25,30,32,35,56,61,67,70,74,82,85,89,93,21,77)]

First_repeat <- clean_no_dup_student$Question[c(7,9,17,24,26,31,33,36,57,62,68,71,75,83,86,90,94,34,92)

Second_repeat <- clean_no_dup_student$Question[c(10,87)]

#Proportion of Repeated Questions in Total
length(Repeated_Questions)/nrow(total_percent_correct)
```

```
## [1] 0.4123711
```

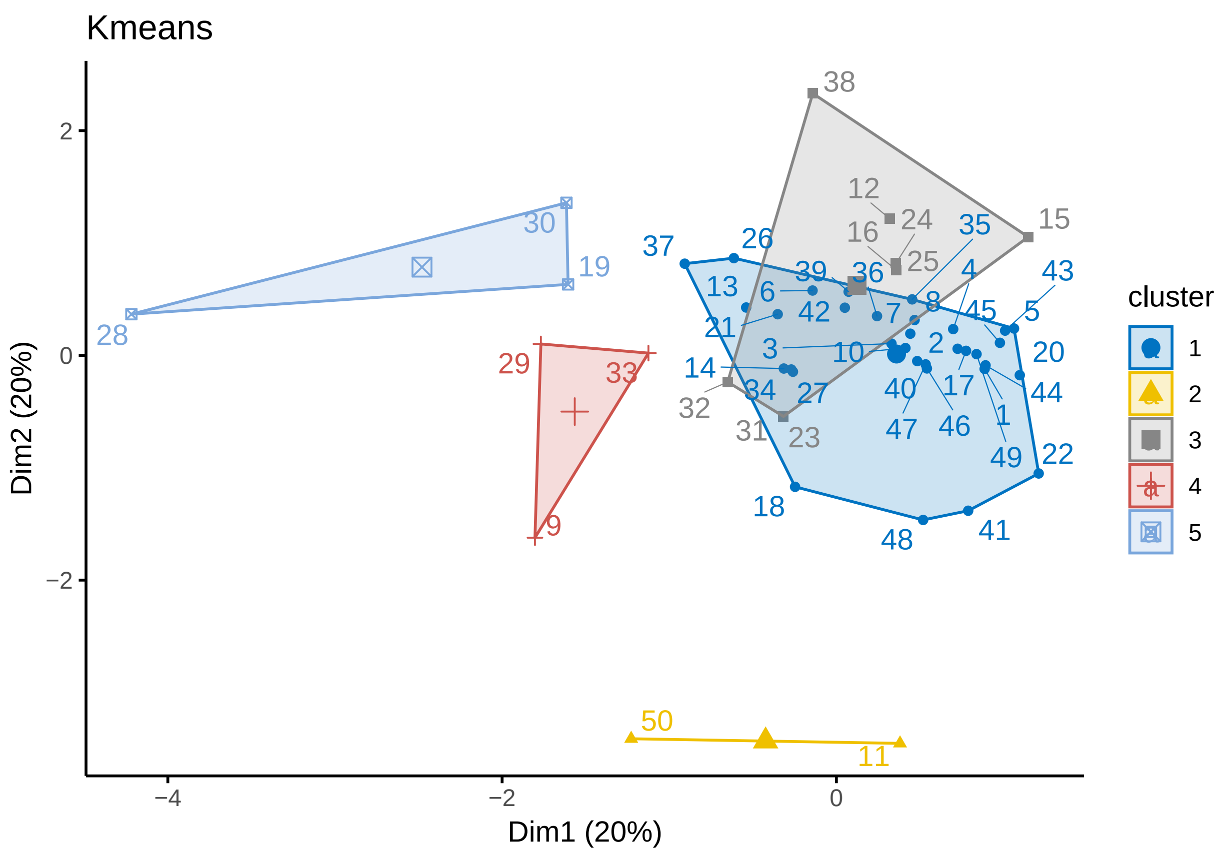
```r
#Proportion of Repeated Questions in Most Contribution
Repeated_contr_df <- data.frame( Type = c("Out of Highest Contribution","Out of Total"), Proportion = c


ggplot(Repeated_contr_df, aes(x = Type, y = Proportion, fill = Type)) + geom_bar(stat="identity")
```



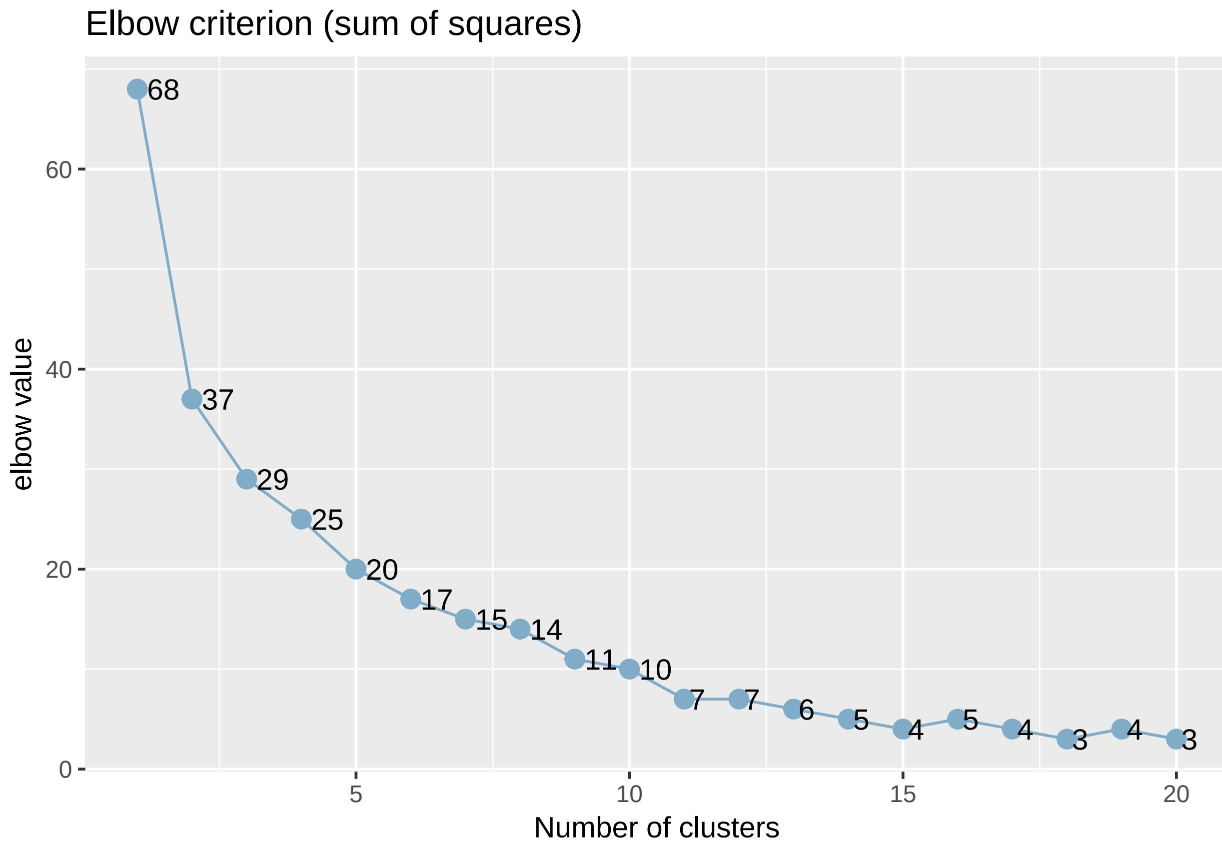

```r
#A Big Difference -- tells us that the variation can be mostly contributed to the repeated questions
```

```r
#Trying CA function instread of MCA for FactoMineRlibrary(FactoMineR)
library(factoextra)


#res<- CA(clean_no_dup[,c(-1,-2)])

#summary(res)

#plot(res)
#plot(res, invisible = c("var"),autoLab = "y")
#plot(res, invisible = c("ind"),label="none")


#fviz_screeplot(res)

#Gives coordinates for the individual students of the first 5 dimensions from MCA
km<- data.frame(res$ind$coord)
```

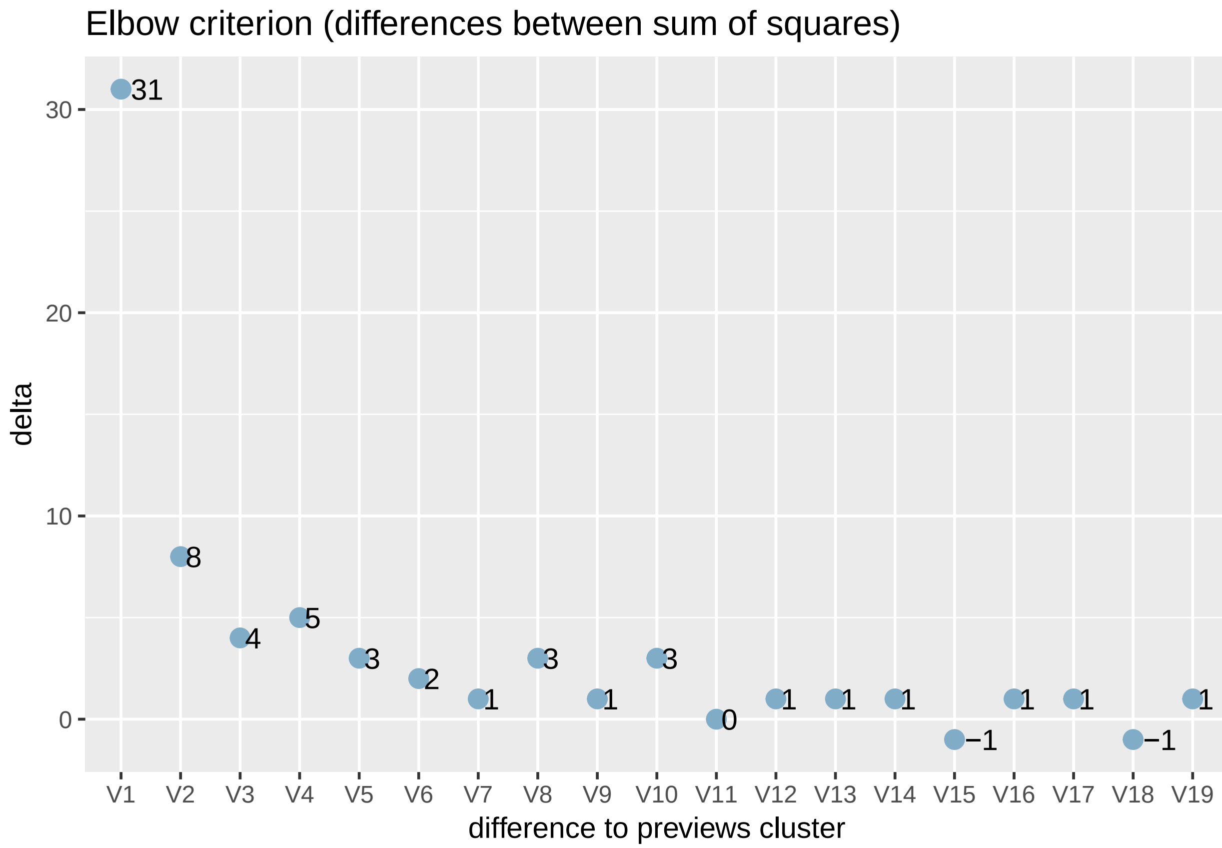
```
groupes.kmeans<- kmeans(km,centers = 5,nstart =5)

fviz_cluster(groupes.kmeans, data = km, palette = "jco", repel= TRUE, main = "Kmeans", ggtheme = theme_c
```



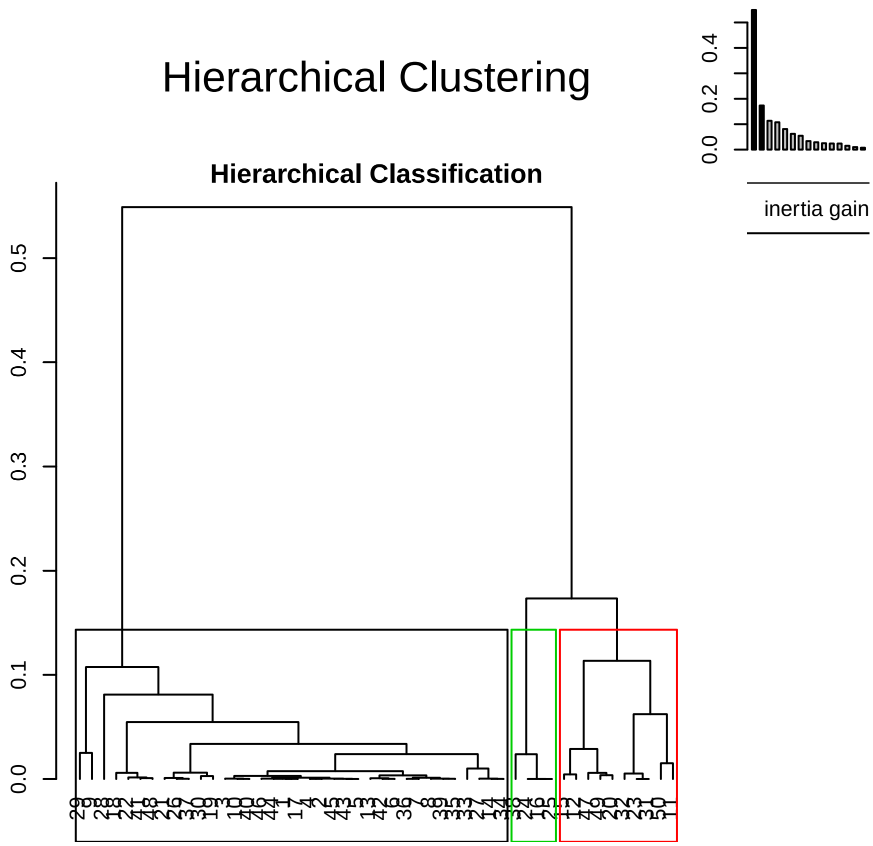

```
library(sjPlot)

sjc.elbow(km,steps=20,show.diff = T)
```



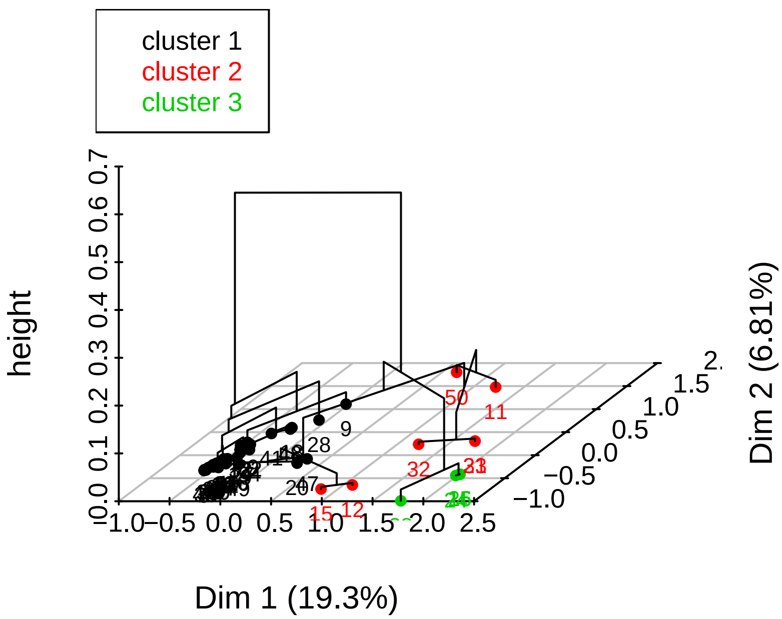

```
## geom_path: Each group consists of only one observation. Do you need to
## adjust the group aesthetic?
```



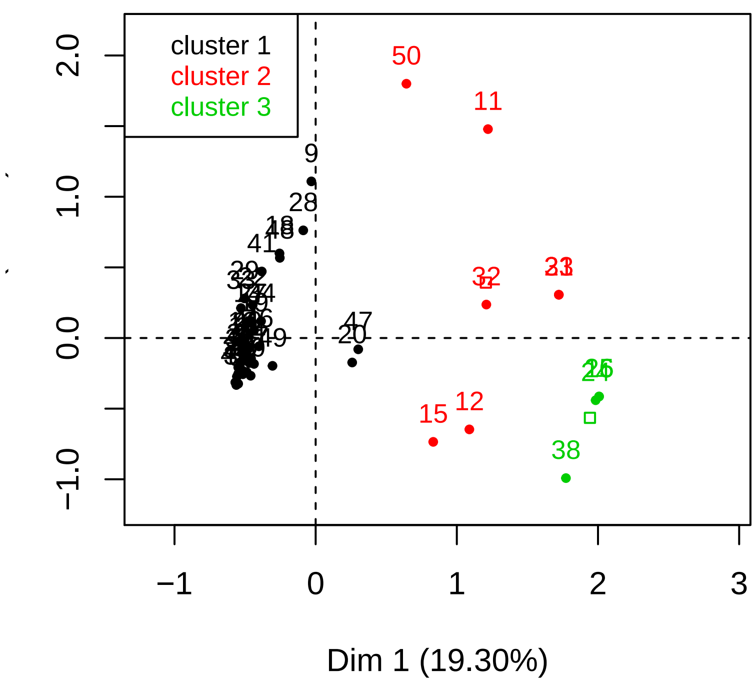

```
hier<- HCPC(res,nb.clust=-1)
```


Hierarchical Clustering
Hierarchical Classification
inertia gain
0.0
0.1
0.2
0.3
0.4
0.5

Hierarchical clustering on the factor map



Factor map




```r
plot(hier, choice=c("3D.map, tree, bar"))
```

```r
library(dplyr)

hierarchal_cluster <- data.frame(student = clean_no_dup[,1])

hierarchal_cluster
```

```
##    student
## 1     ab70
## 2    as160
## 3     bb33
## 4    cd102
## 5    ce138
## 6     cj85
## 7     cq54
## 8     cx93
## 9     dr56
## 10   eu125
## 11   ez119
## 12    fg66
## 13    fs47
## 14     gr7
## 15   gz132
## 16    hc97
## 17   ho117
## 18    ij80
```

```
## 19   il134
## 20   ix115
## 21   jh167
## 22    jr82
## 23   js147
## 24    js52
## 25    kj31
## 26    kk83
## 27   kq149
## 28     mk6
## 29   mn113
## 30    ne71
## 31    pr39
## 32    qf53
## 33     qk1
## 34   qs140
## 35   sa106
## 36   tb139
## 37   tl104
## 38   tn137
## 39    vk37
## 40    wh38
## 41    wz10
## 42   xu128
## 43    ya79
## 44     ya8
## 45   yb118
## 46   yd105
## 47     yi5
## 48   zf127
## 49    zu23
## 50   zz133
```

```
hierarchal_cluster<- cbind(hierarchal_cluster, cluster = hier$data.clust[,"clust"])


correct_cluster <- full_join(percent_correct_student_total, hierarchal_cluster, by = 'student')
```

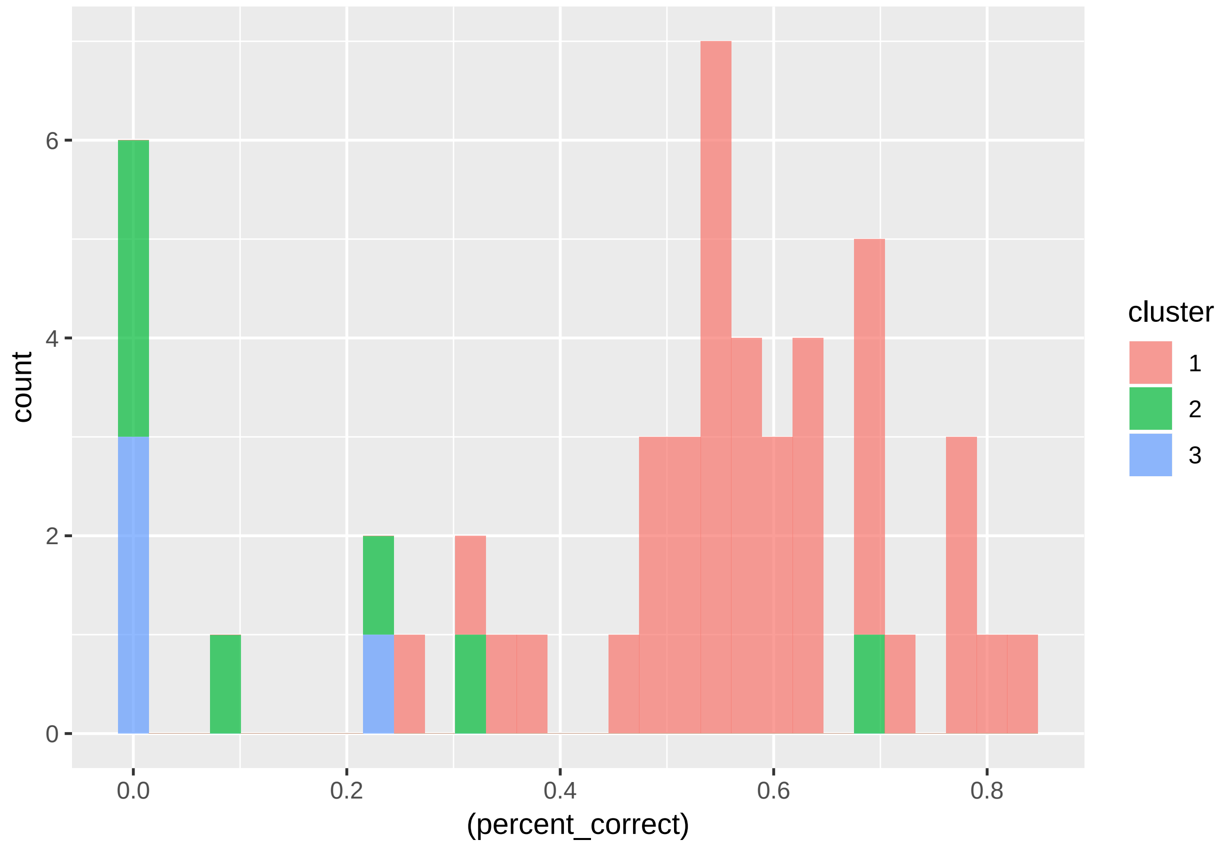
```
## Warning: Column `student` joining factors with different levels, coercing
## to character vector
```

```
mean_correct_clust <-correct_cluster %>% group_by(cluster) %>% summarise(mean_correct = mean(percent_co

mean_correct_clust
```

```
## # A tibble: 3 x 2
##   cluster mean_correct
##   <fct>          <dbl>
## 1 1              0.582
## 2 2              0.190
## 3 3              0.0565
```

```
#Create histograms facet_wrap by ggplot

ggplot(correct_cluster,aes(x=(percent_correct), fill=cluster)) + geom_histogram(stat="bin", alpha = 0.7
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```




```
#Density plots
correct_plot <- ggplot(correct_cluster, aes(x = percent_correct, color = cluster)) + geom_density()

#Density Plot of Question Difficulty
# plot2 <- ggplot(data = total_percent_correct, aes(x=percent_correct, color = "darkorange")) + geom_de



#Looking at the most NA students versus the clusters
most_na_student
```

```
## # A tibble: 12 x 2
## # Groups:   researchID [12]
##    researchID N_A_count
##    <fct>          <int>
##  1 ez119             20
##  2 ix115             20
##  3 yi5               20
##  4 qf53              36
##  5 fg66              45
##  6 gz132             45
##  7 js147             45
##  8 pr39              45
##  9 hc97              59
## 10 js52              59
```

```
## 11 kj31               59
## 12 tn137              65
```

```
hierarchal_cluster
```

```
##    student cluster
## 1     ab70       1
## 2    as160       1
## 3     bb33       1
## 4    cd102       1
## 5    ce138       1
## 6     cj85       1
## 7     cq54       1
## 8     cx93       1
## 9     dr56       1
## 10   eu125       1
## 11   ez119       2
## 12    fg66       2
## 13    fs47       1
## 14     gr7       1
## 15   gz132       2
## 16    hc97       3
## 17   ho117       1
## 18    ij80       1
## 19   il134       1
## 20   ix115       1
## 21   jh167       1
## 22    jr82       1
## 23   js147       2
## 24    js52       3
## 25    kj31       3
## 26    kk83       1
## 27   kq149       1
## 28     mk6       1
## 29   mn113       1
## 30    ne71       1
## 31    pr39       2
## 32    qf53       2
## 33     qk1       1
## 34   qs140       1
## 35   sa106       1
## 36   tb139       1
## 37   tl104       1
## 38   tn137       3
## 39    vk37       1
## 40    wh38       1
## 41    wz10       1
## 42   xu128       1
## 43    ya79       1
## 44     ya8       1
## 45   yb118       1
## 46   yd105       1
## 47     yi5       1
## 48   zf127       1
## 49    zu23       1
```

```
## 50   zz133       2
```

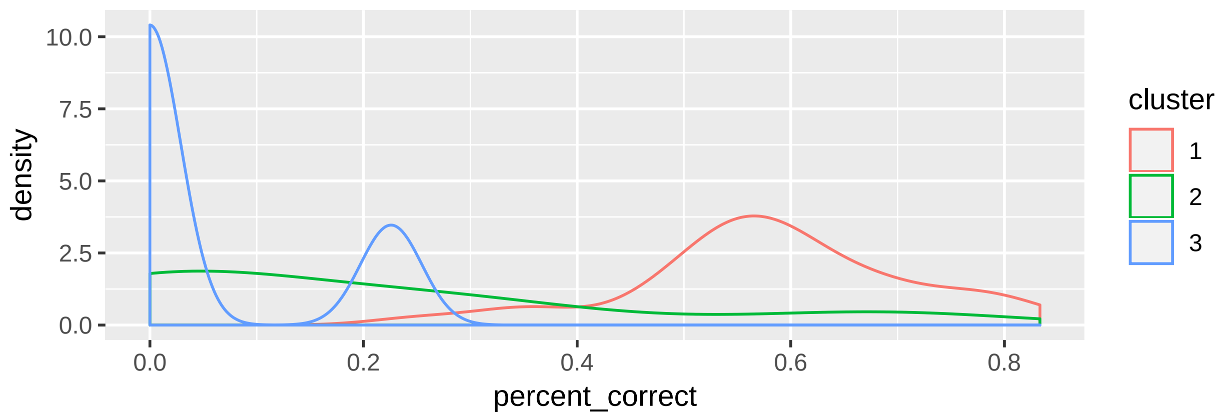
```r
student_na_count
```

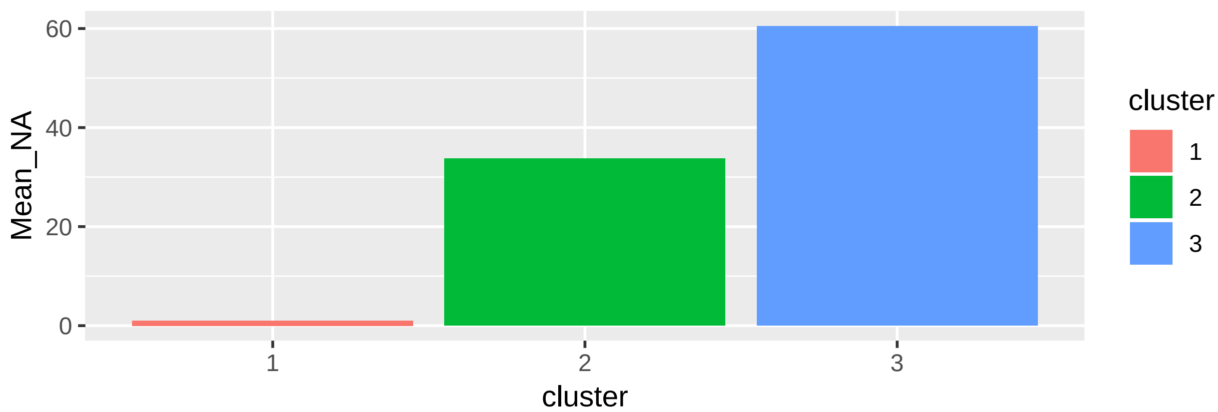
```
## # A tibble: 50 x 2
## # Groups:   researchID [50]
##    researchID N_A_count
##    <fct>          <int>
##  1 ab70               0
##  2 as160              0
##  3 bb33               0
##  4 cd102              0
##  5 ce138              0
##  6 cj85               0
##  7 cq54               0
##  8 cx93               0
##  9 dr56               0
## 10 eu125              0
## # ... with 40 more rows
```

```r
na_cluster_most <- hierarchal_cluster %>% filter(student %in% most_na_student$researchID)


na_cluster <- full_join(hierarchal_cluster, student_na_count, by = c("student"= "researchID"))
```

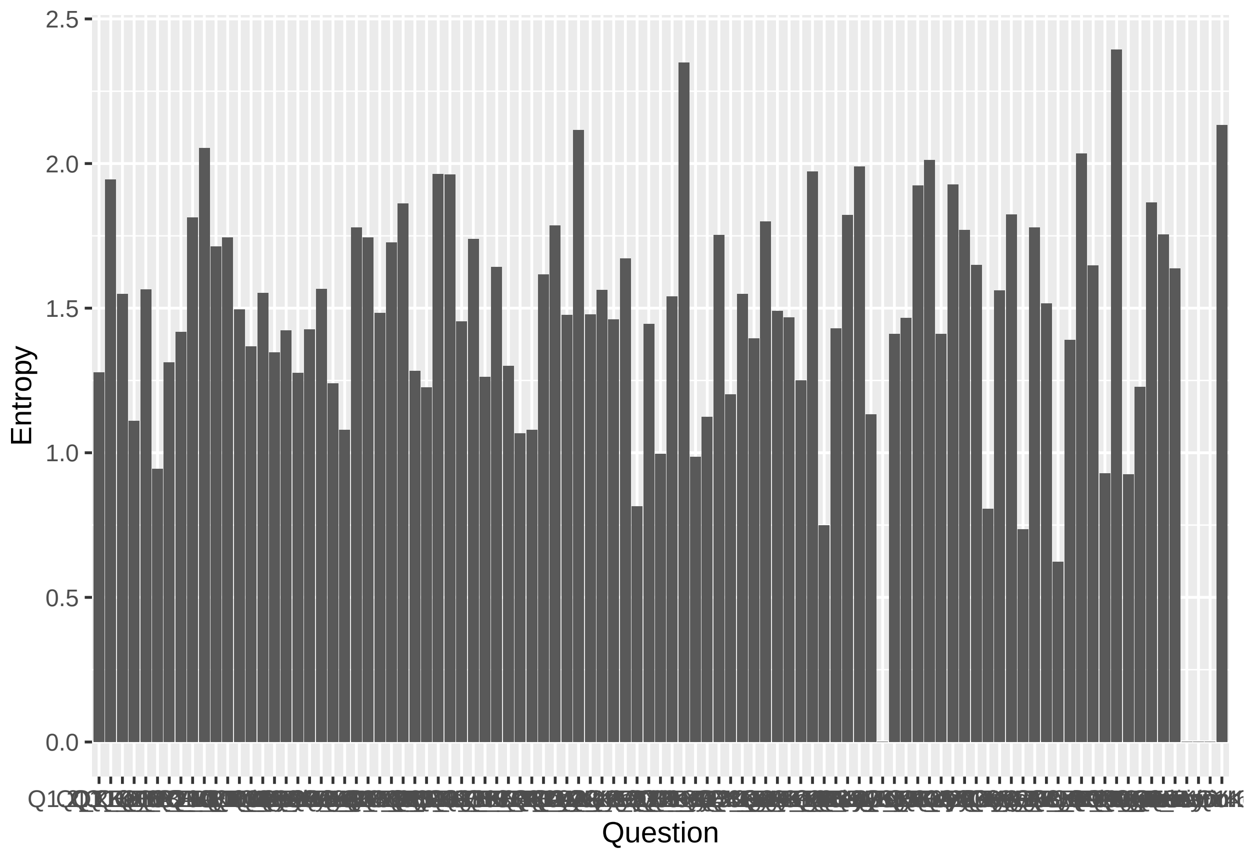
```
## Warning: Column `student`/`researchID` joining factors with different
## levels, coercing to character vector
```

```r
mean_na_cluster <- distinct(na_cluster %>% group_by(cluster) %>% mutate(Mean_NA = mean(N_A_count)), clus

na_plot <- ggplot(mean_na_cluster, aes(x = cluster, y = Mean_NA, fill = cluster))+ geom_bar(stat = "ider

grid.arrange(correct_plot, na_plot)
```






```
#Now, let's look at the correlations between the clusters


#Shannon's Entropy - Information thoery
library(DescTools)
#Entropy()
#MutInf()


#Splitting the Repeat Questions into 3 groups
#First_time
#First_repeat
#Second_repeat


Entropy_Q_vector<- numeric(length(3:99))

for(i in 3:ncol(clean_no_dup)){
  Entropy_Q_vector[i-2] <- Entropy(table(clean_no_dup[,i]))
}


Entropy_Q_df <- data.frame(Entropy = Entropy_Q_vector, Question = names(clean_no_dup)[3:99])


ggplot(Entropy_Q_df,aes(x = Question, y = Entropy)) + geom_bar(stat = 'identity')
```




```r
#Stat summary for Entropy of All Q's
summary(Entropy_Q_vector)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   0.000   1.251   1.483   1.449   1.754   2.394
```

```r
#Stat Summary for Entropy of Repeated Q's
Entropy_Rep_Q <- Entropy_Q_df %>% filter(Question %in% Repeated_Questions)
summary(Entropy_Rep_Q[,1])
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   0.000   1.274   1.515   1.513   1.773   2.350
```

```r
#Stat Summary for Entropy of Most Contrib to Dim 1
Entropy_Most_Cont_Q <- Entropy_Q_df %>% filter(Question %in% most_contribution)
summary(Entropy_Most_Cont_Q[,1])
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.0000  0.8974  1.2509  1.2140  1.6706  2.0536
```

```r
#Stat Summary for Entropy of First_time
Entropy_First_Q <- Entropy_Q_df %>% filter(Question %in% First_time)
summary(Entropy_First_Q[,1])
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   1.110   1.457   1.643   1.656   1.801   2.350
```

```r
#Stat Summary for Entropy of First_repeat
Entropy_First_R <- Entropy_Q_df %>% filter(Question %in% First_repeat)
```

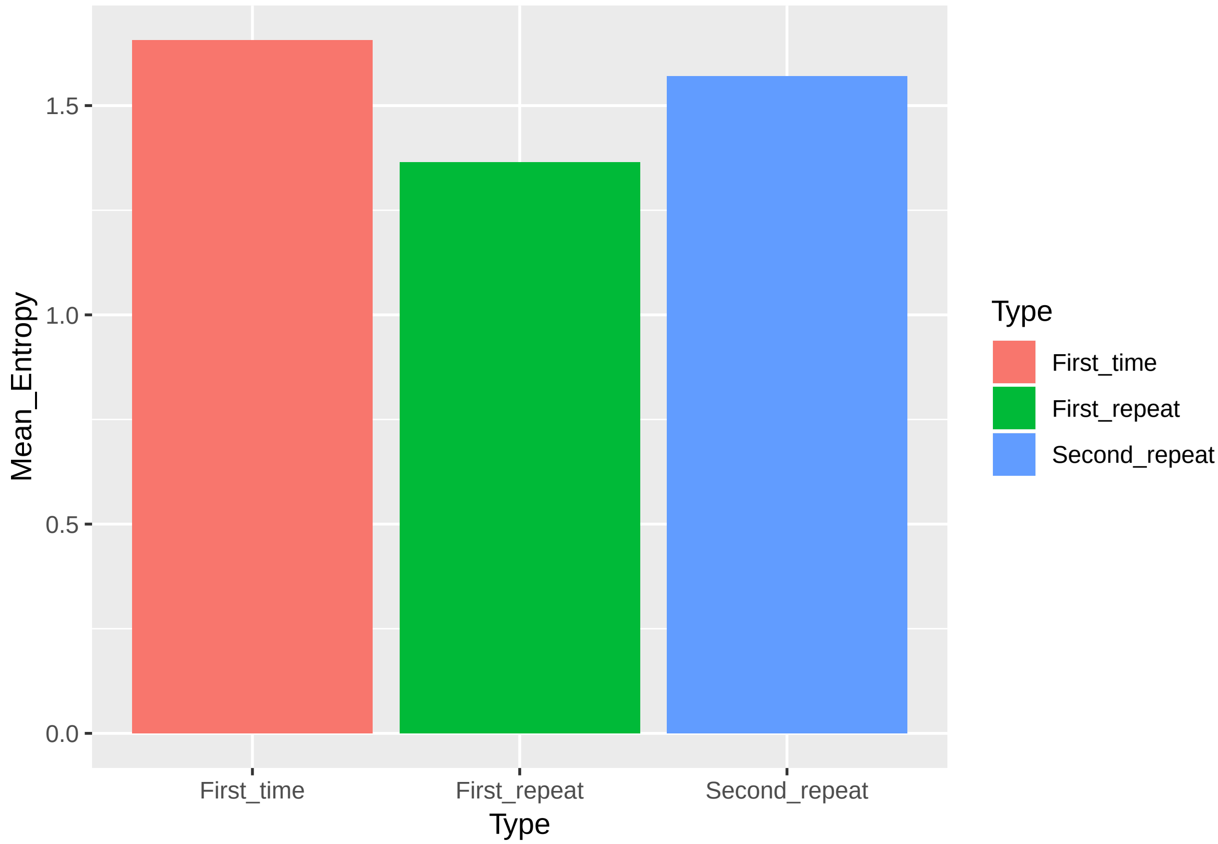
```r
summary(Entropy_First_R[,1])
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   0.000   1.179   1.417   1.364   1.669   2.116
```

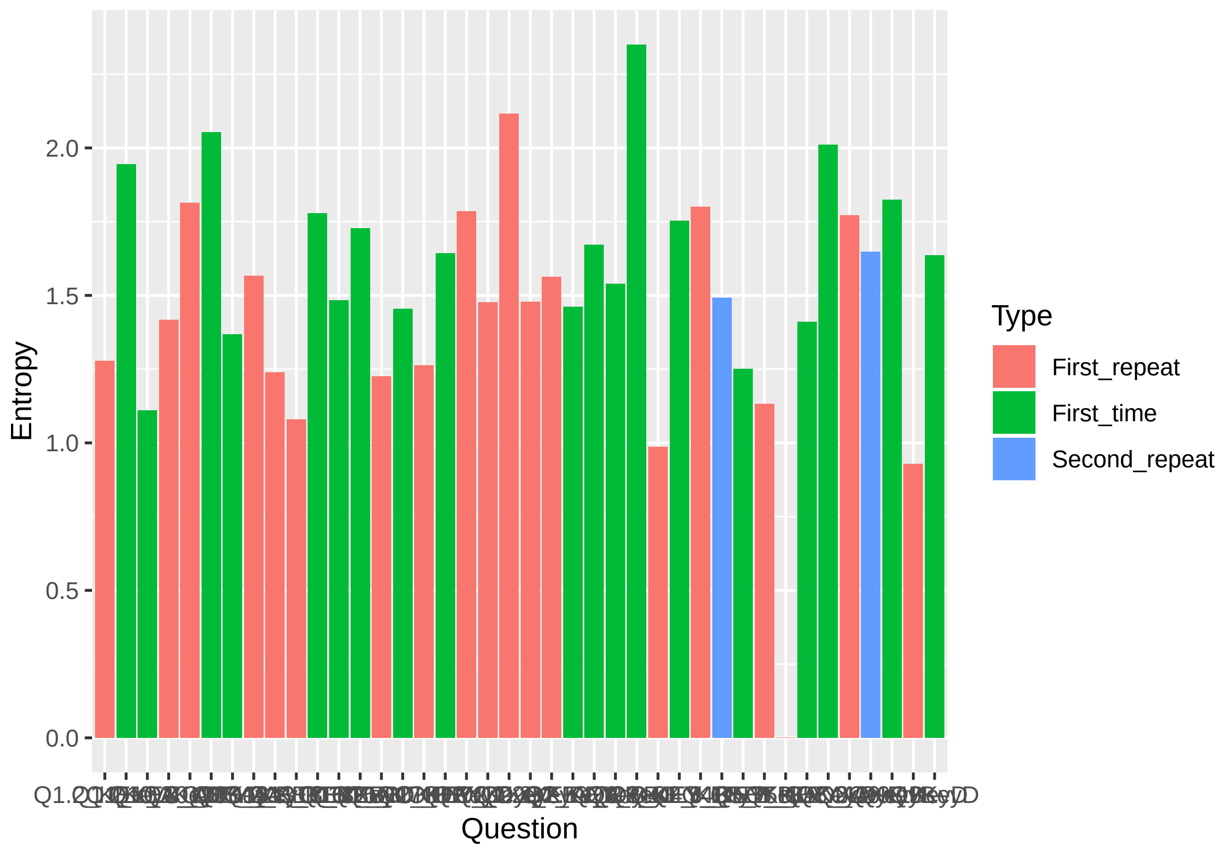
```r
#Stat Summary for Entropy of Second_repeat
Entropy_Second_R <- Entropy_Q_df %>% filter(Question %in% Second_repeat)
summary(Entropy_Second_R[,1])
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   1.491   1.530   1.570   1.570   1.609   1.648
```

```r
#Creating df and plot to compare Entropy and Repeat Question Group
Entropy_Rep_Q <- Entropy_Rep_Q %>% mutate(Type = NA)

Entropy_Rep_Q$Type[Entropy_Rep_Q$Question %in% First_time] <- "First_time"

Entropy_Rep_Q$Type[Entropy_Rep_Q$Question %in% First_repeat] <- "First_repeat"

Entropy_Rep_Q$Type[Entropy_Rep_Q$Question %in% Second_repeat] <- "Second_repeat"

Mean_Entropy_Type <- Entropy_Rep_Q %>% group_by(Type) %>% summarise(Mean_Entropy = mean(Entropy))

Mean_Entropy_Type <- Mean_Entropy_Type[c(2,1,3),]



Mean_Entropy_Type$Type <- factor(Mean_Entropy_Type$Type, levels = c("First_time","First_repeat","Second_


ggplot(Mean_Entropy_Type, aes(x = Type, y = Mean_Entropy, fill = Type)) + geom_bar(stat='identity')
```

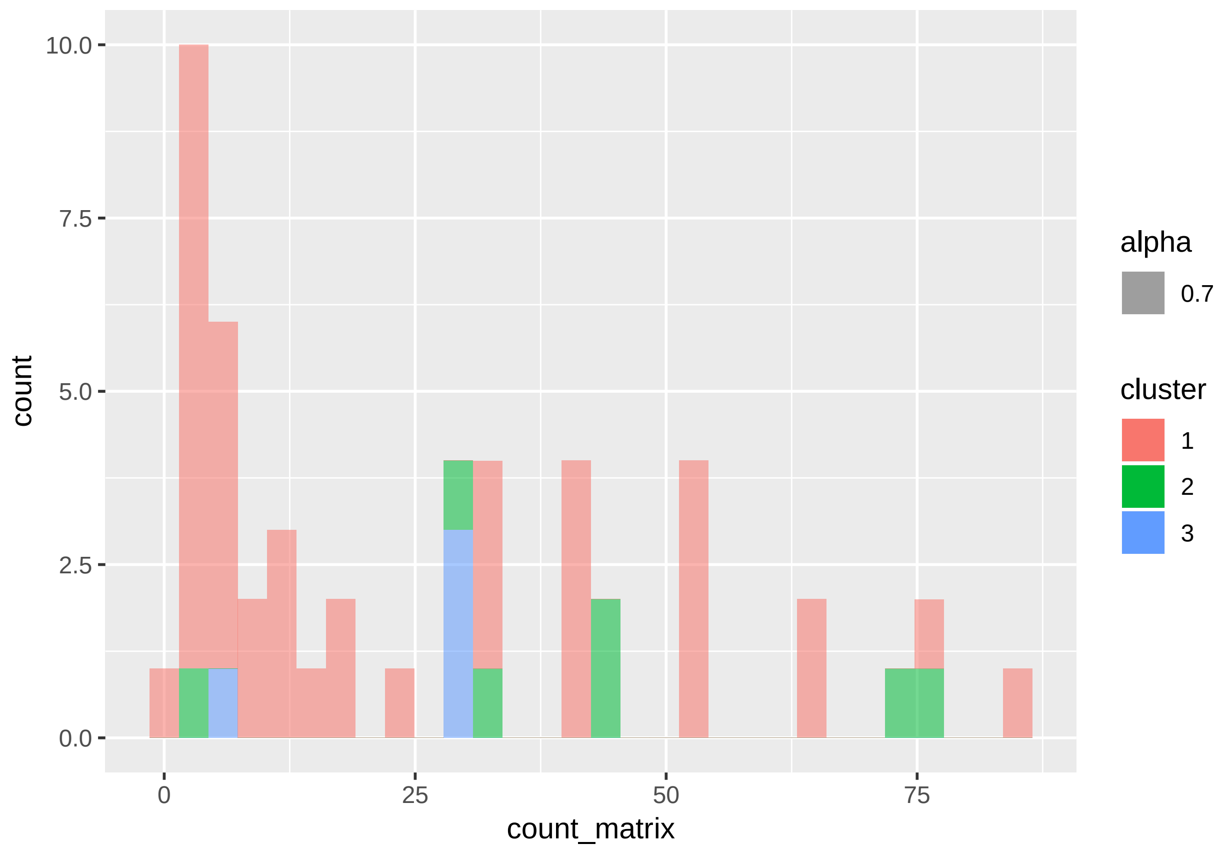



```
ggplot(Entropy_Rep_Q, aes(x = Question, y = Entropy, fill = Type))+ geom_bar(stat="identity")
```




```
#Creating df and plot to compare Percent_Correct and Repeat Question Group




#Start by seeing how many questions they actually answered -- look at number of NV's

clean_clust <- full_join(clean_no_dup, hierarchal_cluster, by = c("researchID"= "student"))

ggplot(clean_clust, aes(count_matrix, fill = cluster, alpha = 0.7)) + geom_histogram()
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```




```
#Next, look at the actual correlations between each other: 1 for same answers, 0 for different answers



#Look up if there is an easy way to measure correlations between categorical variables
#Cramer's V Derived from the Chi-Squared Test of Independence of Categorical Variables -- We are intere

#DO == to rowwise() of clean_clust and compute the mean as a measure of "correlation" between students



#Output will be a matrix of 'correlation' coefficients ranging from 0 to 1

#Test works
mean (as.character(clean_no_dup[1,3:99]) == as.character(clean_no_dup[2,3:99]) )
```

[1] 0.5360825

```
#Huge 50 by 50 matrix for loop

proportion_exactness <- matrix(nrow=50,ncol=50)


for(i in 1:50){
  for(j in 1:50){
    proportion_exactness[i,j] <- mean (as.character(clean_no_dup[i,3:99]) == as.character(clean_no_dup[
  }
```

```
}

#To test if the matrix is correct-- CORRECT
mean (as.character(clean_no_dup[1,3:99]) == as.character(clean_no_dup[2,3:99]) )== proportion_exactness
```

[1] TRUE

```
#Trying out the 2nd highest function
x <- c(1:10)
n <- length(x)
sort(x,partial=n-1)[n-1]
```

[1] 9

```
#Writing a manual function to insert to apply

second_highest <- function(df){
  n <- length(df)
  sort(df,partial=n-1)[n-1]
}

#Trying out the second_highest function -- CHECK
second_highest(x)
```

[1] 9

```
#Now, find the highest 50 values by row -- need to subset out the '1's

highest_sim <- apply(proportion_exactness, MARGIN = 1, second_highest)


#Now getting the assignments of students, while also filtering out the 1's

column_vector <- numeric(50)

for(i in 1:50){

  column_vector[i]<- which(proportion_exactness[i,] == second_highest(proportion_exactness[i,])  )

}
```

```
## Warning in column_vector[i] <- which(proportion_exactness[i, ] ==
## second_highest(proportion_exactness[i, : number of items to replace is not
## a multiple of replacement length

## Warning in column_vector[i] <- which(proportion_exactness[i, ] ==
## second_highest(proportion_exactness[i, : number of items to replace is not
## a multiple of replacement length

## Warning in column_vector[i] <- which(proportion_exactness[i, ] ==
## second_highest(proportion_exactness[i, : number of items to replace is not
## a multiple of replacement length

## Warning in column_vector[i] <- which(proportion_exactness[i, ] ==
## second_highest(proportion_exactness[i, : number of items to replace is not
## a multiple of replacement length
```

```
## Warning in column_vector[i] <- which(proportion_exactness[i, ] ==
## second_highest(proportion_exactness[i, : number of items to replace is not
## a multiple of replacement length

## Warning in column_vector[i] <- which(proportion_exactness[i, ] ==
## second_highest(proportion_exactness[i, : number of items to replace is not
## a multiple of replacement length

## Warning in column_vector[i] <- which(proportion_exactness[i, ] ==
## second_highest(proportion_exactness[i, : number of items to replace is not
## a multiple of replacement length

## Warning in column_vector[i] <- which(proportion_exactness[i, ] ==
## second_highest(proportion_exactness[i, : number of items to replace is not
## a multiple of replacement length

## Warning in column_vector[i] <- which(proportion_exactness[i, ] ==
## second_highest(proportion_exactness[i, : number of items to replace is not
## a multiple of replacement length

## Warning in column_vector[i] <- which(proportion_exactness[i, ] ==
## second_highest(proportion_exactness[i, : number of items to replace is not
## a multiple of replacement length

## Warning in column_vector[i] <- which(proportion_exactness[i, ] ==
## second_highest(proportion_exactness[i, : number of items to replace is not
## a multiple of replacement length

## Warning in column_vector[i] <- which(proportion_exactness[i, ] ==
## second_highest(proportion_exactness[i, : number of items to replace is not
## a multiple of replacement length

## Warning in column_vector[i] <- which(proportion_exactness[i, ] ==
## second_highest(proportion_exactness[i, : number of items to replace is not
## a multiple of replacement length

## Warning in column_vector[i] <- which(proportion_exactness[i, ] ==
## second_highest(proportion_exactness[i, : number of items to replace is not
## a multiple of replacement length

## Warning in column_vector[i] <- which(proportion_exactness[i, ] ==
## second_highest(proportion_exactness[i, : number of items to replace is not
## a multiple of replacement length
```

```r
#Check that these are the correct assignments -- CHECK

truth_vector <- numeric(50)


for(i in 1:50){
  truth_vector[i] <- proportion_exactness[i,column_vector[i]]
}
```

```
all(truth_vector == highest_sim)
```

[1] TRUE

```
pair_df <- data.frame(row = 1:50, column = column_vector, proportion = highest_sim)

pair_df <- pair_df  %>% arrange(desc(proportion))


head(pair_df)
```

row column proportion 1 16 16 1.0000000 2 23 23 1.0000000 3 25 16 1.0000000 4 31 23 1.0000000 5 24 16 0.9896907 6 4 43 0.8041237

```
#Pairs with Highest Proportions: (24, 16), (4,43), (5,43), (32,23)

table(clean_no_dup_tidy %>% group_by(researchID) %>% select(Response))
```

```
## Adding missing grouping variables: `researchID`
```

```
      Response
```

researchID A B C D E NV ab70 22 38 17 7 1 12 as160 27 33 17 7 2 11 bb33 26 32 13 11 2 13 ca62 0 0 0 0 0 0 cd102 26 34 19 11 2 5 ce138 24 35 18 8 5 7 cj85 32 31 16 12 0 6 cq54 25 32 15 12 3 10 cx93 27 36 19 7 2 6 dr56 16 19 6 7 1 48 eu125 34 24 11 15 1 12 ev22 0 0 0 0 0 0 ez119 0 0 0 0 0 77 fg66 14 15 9 1 0 13 fs47 24 30 21 9 2 11 gm172 0 0 0 0 0 0 gr7 27 22 17 12 0 19 gz132 18 11 12 5 2 4 hc97 0 0 0 0 0 38 hl94 0 0 0 0 0 0 ho117 28 31 11 13 3 11 ij80 28 23 12 7 1 26 il134 24 36 13 10 1 13 ix115 19 16 18 6 2 16 jh167 31 32 11 12 2 9 jr82 21 30 17 11 1 17 js147 0 0 0 0 0 52 js52 1 0 0 0 0 37 kc69 0 0 0 0 0 0 kj31 0 0 0 0 0 38 kk83 32 37 15 6 3 4 kq149 28 22 18 11 2 16 lp16 0 0 0 0 0 0 mk6 22 22 15 5 1 32 mn113 16 27 13 16 5 20 ne71 25 30 23 11 1 7 pr39 0 0 0 0 0 52 qf53 9 8 3 3 0 38 qk1 28 23 10 16 0 20 qs140 31 24 12 8 1 21 sa106 27 34 16 13 3 4 sc108 0 0 0 0 0 0 se121 0 0 0 0 0 0 tb139 28 33 18 10 3 5 tl104 26 37 17 9 2 6 tn137 5 6 4 2 0 15 vk37 23 33 21 14 2 4 wh38 33 27 14 11 1 11 wz10 28 30 10 5 1 23 xu128 29 32 18 9 1 8 ya79 26 34 20 9 3 5 ya8 23 33 19 8 3 11 yb118 24 29 22 14 2 6 yb87 0 0 0 0 0 0 yd105 27 30 12 11 2 15 yi5 16 28 16 2 1 14 za95 0 0 0 0 0 0 zf127 21 30 13 4 2 27 zu23 26 26 21 8 1 15 zz133 5 2 4 2 3 81

```
#Looking at the pairs that are also in the same cluster: cluster2: 11,12,15,23,31,32,50
# (50,11), (32,23), (11,23), (15,12), (31,23), (32,23): ALL SHARE SAME Cluster

#Cluster 3: 16,24,25,38
#(24,16), (25,16), (38,16): ALL SHARE SAME CLUSTER

stargazer(tail(pair_df, 48), summary=FALSE, rownames=FALSE)
```

% Table created by stargazer v.5.2.2 by Marek Hlavac, Harvard University. E-mail: hlavac at fas.harvard.edu
% Date and time: Sat, Jun 08, 2019 - 6:52:50 PM

```
#Interesting pairs to look at: (4,43), (5,43), (44,5), (45,5), (1,5), (2,43), NV's: (36,45), (40,4): Al


percent_correct_student_total[4,]$percent_correct - percent_correct_student_total[43,]$percent_correct
```

[1] -0.03571429

```
percent_correct_student_total[5,]$percent_correct - percent_correct_student_total[43,]$percent_correct
```

[1] 0.02380952

```
percent_correct_student_total[4,]$percent_correct - percent_correct_student_total[45,]$percent_correct
```

Table 6:

row	column	proportion
25	16	1
31	23	1
24	16	0.990
4	43	0.804
5	43	0.804
43	4	0.804
32	23	0.763
38	16	0.763
11	23	0.742
44	5	0.742
45	5	0.742
1	5	0.711
2	43	0.711
36	45	0.711
40	4	0.711
10	40	0.691
14	27	0.691
17	4	0.691
21	36	0.691
27	14	0.691
35	17	0.680
39	17	0.680
8	2	0.670
26	4	0.660
50	11	0.660
3	5	0.649
7	45	0.649
12	15	0.649
15	12	0.649
37	36	0.649
46	3	0.649
22	41	0.639
41	22	0.639
42	5	0.619
29	33	0.608
33	29	0.608
49	36	0.608
6	4	0.598
13	14	0.588
34	2	0.588
19	26	0.577
30	8	0.557
18	8	0.536
20	43	0.526
48	41	0.526
9	50	0.515
28	18	0.464
47	20	0.464

[1] 0

```r
percent_correct_student_total[5,]$percent_correct - percent_correct_student_total[45,]$percent_correct
```

[1] 0.05952381

```r
#Big difference between 1 and 5
percent_correct_student_total[5,]$percent_correct - percent_correct_student_total[1,]$percent_correct
```

[1] 0.2261905

```r
percent_correct_student_total[2,]$percent_correct - percent_correct_student_total[43,]$percent_correct
```

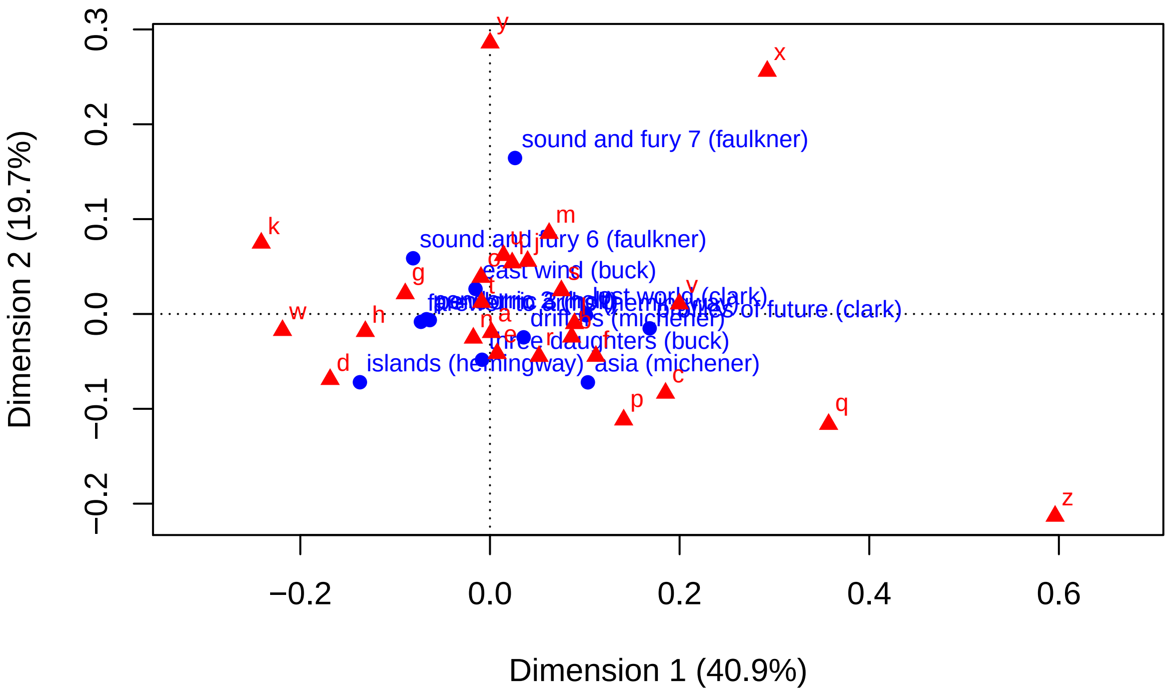
[1] -0.1190476

```r
library(ca)

#ca from ca package example
data("author")
ca(author)
```

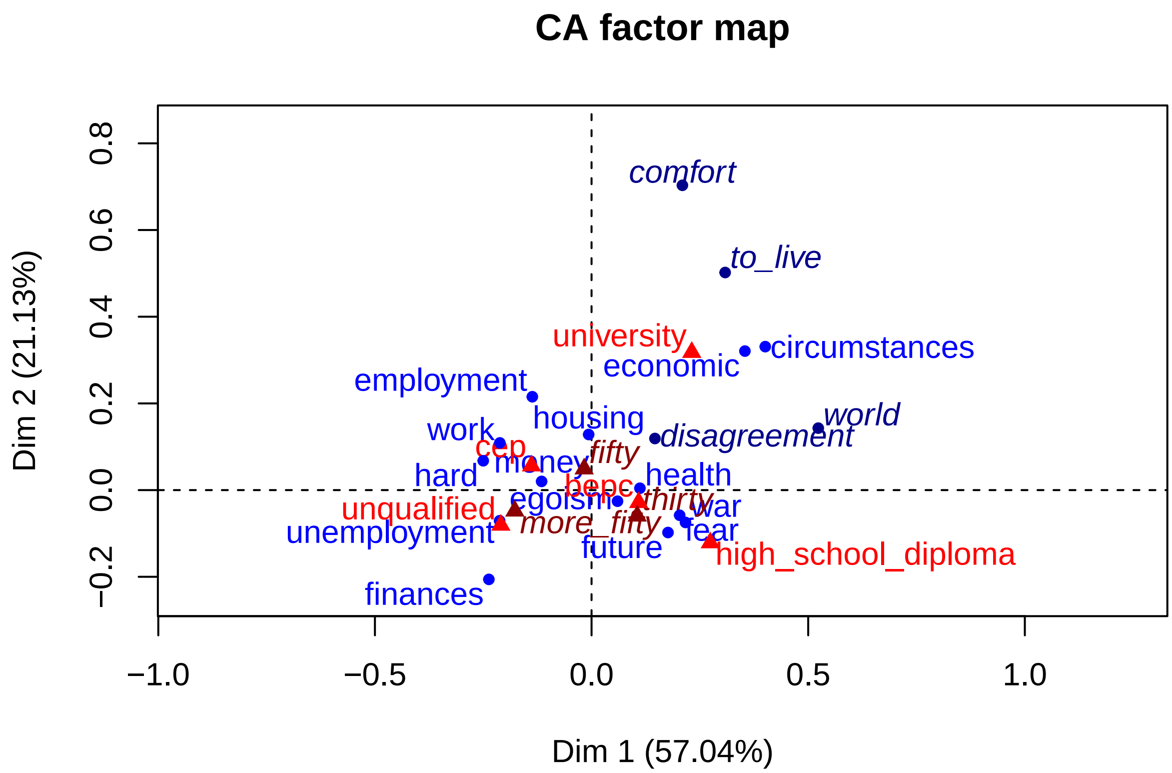
```
##
##  Principal inertias (eigenvalues):
##            1        2        3        4        5        6        7
## Value      0.007664 0.003688 0.002411 0.001383 0.001002 0.000723 0.000659
## Percentage 40.91%   19.69%   12.87%   7.38%    5.35%    3.86%    3.52%
##            8        9        10       11
## Value      0.000455 0.000374 0.000263 0.000113
## Percentage 2.43%    2%       1.4%     0.6%
##
##
##  Rows:
##         three daughters (buck) drifters (michener) lost world (clark)
## Mass                  0.085407            0.079728           0.084881
## ChiDist               0.097831            0.094815           0.128432
## Inertia               0.000817            0.000717           0.001400
## Dim. 1               -0.095388            0.405697           1.157803
## Dim. 2               -0.794999           -0.405560          -0.023114
##         east wind (buck) farewell to arms (hemingway)
## Mass            0.089411                     0.082215
## ChiDist         0.118655                     0.122889
## Inertia         0.001259                     0.001242
## Dim. 1         -0.173901                    -0.831886
## Dim. 2          0.434443                    -0.136485
##         sound and fury 7 (faulkner) sound and fury 6 (faulkner)
## Mass                       0.082310                    0.083338
## ChiDist                    0.172918                    0.141937
## Inertia                    0.002461                    0.001679
## Dim. 1                     0.302025                   -0.925572
## Dim. 2                     2.707599                    0.966944
##         profiles of future (clark) islands (hemingway) pendorric 3 (holt)
## Mass                      0.089722            0.082776           0.079501
## ChiDist                   0.187358            0.165529           0.113174
## Inertia                   0.003150            0.002268           0.001018
## Dim. 1                    1.924060           -1.566481          -0.724758
## Dim. 2                   -0.249310           -1.185338          -0.106349
##         asia (michener) pendorric 2 (holt)
```

```
## Mass           0.077827          0.082884
## ChiDist        0.155115          0.101369
## Inertia        0.001873          0.000852
## Dim. 1         1.179548         -0.764937
## Dim. 2        -1.186934         -0.091188
##
##
##  Columns:
##                 a         b         c         d         e         f
## Mass     0.079847  0.015685  0.022798  0.045967  0.127070  0.019439
## ChiDist  0.048441  0.148142  0.222783  0.189938  0.070788  0.165442
## Inertia  0.000187  0.000344  0.001132  0.001658  0.000637  0.000532
## Dim. 1   0.017623  0.984463  2.115029 -1.925632  0.086722  1.276526
## Dim. 2  -0.320271 -0.398032 -1.373448 -1.135362 -0.684785 -0.732952
##                 g         h        i        j         k         l        m
## Mass     0.020025  0.064928 0.070092 0.000789  0.009181  0.042667 0.025500
## ChiDist  0.156640  0.154745 0.086328 0.412075  0.296727  0.120397 0.159747
## Inertia  0.000491  0.001555 0.000522 0.000134  0.000808  0.000618 0.000651
## Dim. 1  -1.020713 -1.501277 0.267473 0.453341 -2.755177  1.018257 0.712695
## Dim. 2   0.353017 -0.302413 0.889546 0.916032  1.231557 -0.165020 1.400966
##                 n         o         p         q         r        s
## Mass     0.068968  0.076572  0.015159  0.000669  0.051897 0.060660
## ChiDist  0.075706  0.088101  0.250617  0.582298  0.111725 0.123217
## Inertia  0.000395  0.000594  0.000952  0.000227  0.000648 0.000921
## Dim. 1  -0.200364 -0.108491  1.610807  4.079786  0.591372 0.860202
## Dim. 2  -0.417258  0.637987 -1.837948 -1.914791 -0.734216 0.405610
##                 t        u        v         w        x        y         z
## Mass     0.093010 0.029756 0.009612  0.025847 0.001160 0.021902  0.000801
## ChiDist  0.050630 0.119215 0.269770  0.232868 0.600831 0.301376  0.833700
## Inertia  0.000238 0.000423 0.000700  0.001402 0.000419 0.001989  0.000557
## Dim. 1  -0.100464 0.163295 2.281333 -2.499232 3.340505 0.001519  6.808100
## Dim. 2   0.203141 1.017140 0.177022 -0.284722 4.215355 4.706083 -3.509223
```

```
plot(ca(author))
```




```
#Example from documentation for CA
data(children)
res.ca <- CA (children, row.sup = 15:18, col.sup = 6:8)
```

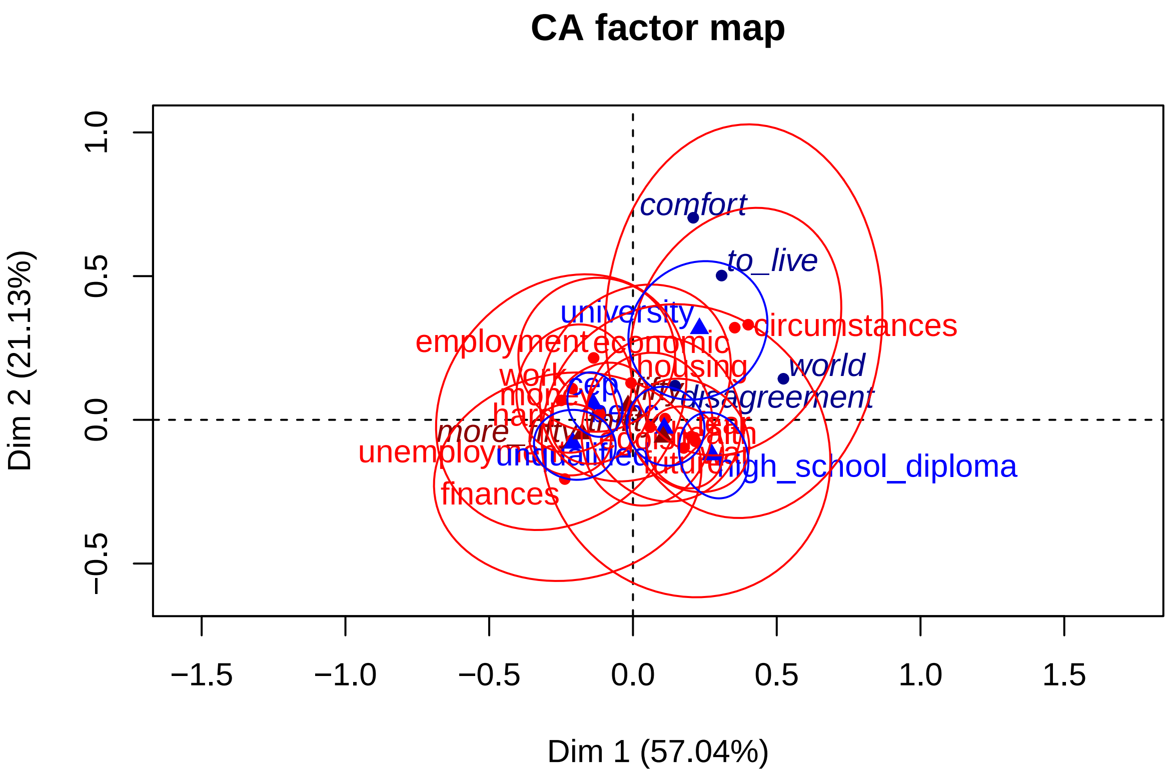



```
summary(res.ca)
```

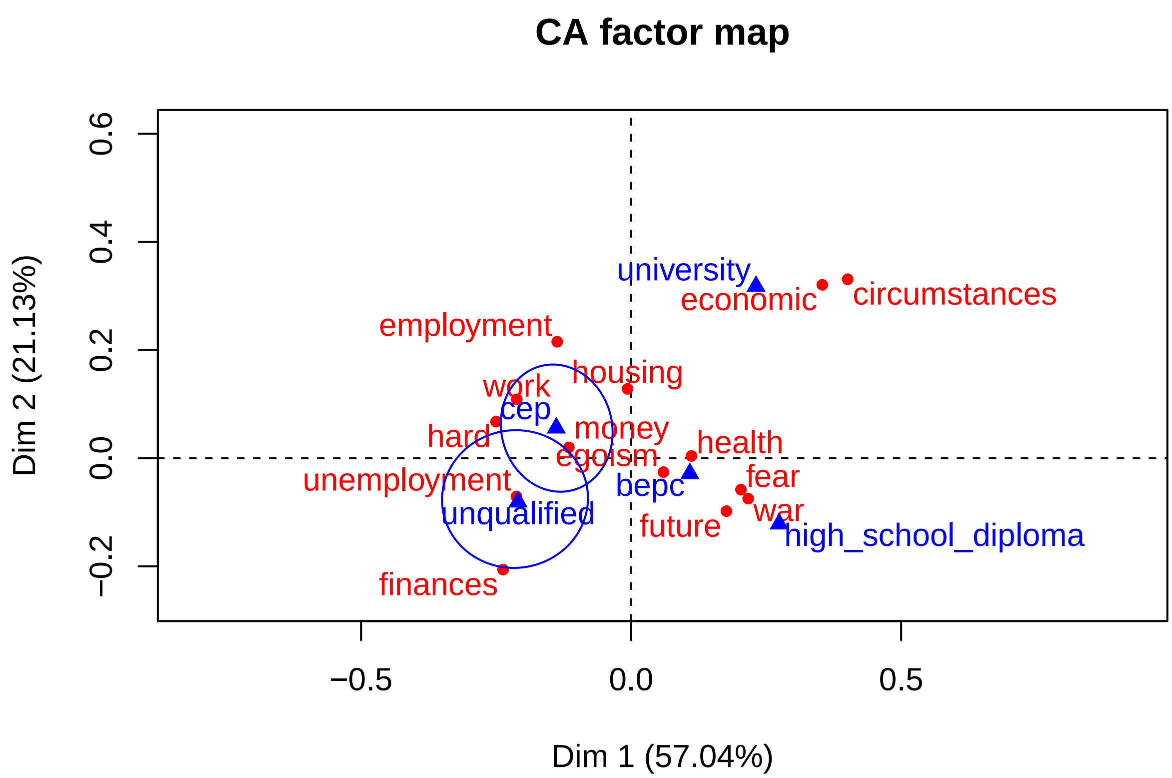
```
##
## Call:
## CA(X = children, row.sup = 15:18, col.sup = 6:8)
##
## The chi square of independence between the two variables is equal to 98.80159 (p-value =  9.748064e-0
##
## Eigenvalues
##                        Dim.1   Dim.2   Dim.3   Dim.4
## Variance               0.035   0.013   0.007   0.006
## % of var.             57.043  21.132  11.764  10.061
## Cumulative % of var.  57.043  78.175  89.939 100.000
##
## Rows (the 10 first)
##                       Iner*1000    Dim.1    ctr   cos2    Dim.2    ctr
## money               |     3.759 | -0.115  4.550  0.428 |  0.020  0.371
## future              |     8.690 |  0.176 17.567  0.716 | -0.098 14.587
## unemployment        |     9.151 | -0.212 22.616  0.875 | -0.071  6.779
## circumstances       |     3.804 |  0.401  6.274  0.584 |  0.331 11.544
## hard                |     1.199 | -0.250  2.994  0.884 |  0.068  0.592
## economic            |     8.787 |  0.354 12.005  0.484 |  0.321 26.604
## egoism              |     3.287 |  0.060  0.681  0.073 | -0.026  0.338
## employment          |     5.648 | -0.137  2.621  0.164 |  0.215 17.555
## finances            |     3.576 | -0.237  2.790  0.276 | -0.206  5.690
## war                 |     1.025 |  0.217  2.169  0.749 | -0.075  0.694
```

```
##                         cos2    Dim.3    ctr   cos2
## money                  0.013 |  0.101 16.884  0.328 |
## future                 0.220 | -0.053  7.568  0.064 |
## unemployment           0.097 | -0.004  0.046  0.000 |
## circumstances          0.398 | -0.016  0.046  0.001 |
## hard                   0.065 |  0.060  0.845  0.051 |
## economic               0.397 |  0.084  3.280  0.027 |
## egoism                 0.013 |  0.179 29.496  0.655 |
## employment             0.408 | -0.213 30.815  0.398 |
## finances               0.209 | -0.044  0.469  0.010 |
## war                    0.089 | -0.098  2.139  0.152 |
##
## Columns
##                         Iner*1000    Dim.1    ctr   cos2    Dim.2    ctr
## unqualified          |    13.146 | -0.209 25.110  0.676 | -0.081 10.082
## cep                  |    10.044 | -0.139 18.297  0.645 |  0.056  8.079
## bepc                 |     7.670 |  0.109  6.758  0.312 | -0.028  1.251
## high_school_diploma  |    17.732 |  0.274 37.976  0.758 | -0.121 20.099
## university           |    13.468 |  0.231 11.859  0.312 |  0.318 60.488
##                         cos2    Dim.3    ctr   cos2
## unqualified            0.101 |  0.073 14.659  0.081 |
## cep                    0.105 | -0.018  1.520  0.011 |
## bepc                   0.021 | -0.147 59.874  0.570 |
## high_school_diploma    0.149 |  0.077 14.407  0.059 |
## university             0.589 |  0.094  9.540  0.052 |
##
## Supplementary rows
##                         Dim.1  cos2   Dim.2  cos2   Dim.3  cos2
## comfort              | 0.210 0.069 | 0.703 0.775 | 0.071 0.008 |
## disagreement         | 0.146 0.131 | 0.119 0.087 | 0.171 0.180 |
## world                | 0.523 0.876 | 0.143 0.065 | 0.084 0.023 |
## to_live              | 0.308 0.139 | 0.502 0.369 | 0.521 0.397 |
##
## Supplementary columns
##                          Dim.1    cos2    Dim.2    cos2    Dim.3    cos2
## thirty               |  0.105   0.138 | -0.060   0.044 | -0.103   0.132 |
## fifty                | -0.017   0.011 |  0.049   0.090 | -0.016   0.009 |
## more_fifty           | -0.177   0.286 | -0.048   0.021 |  0.101   0.093 |
```

```
## Ellipses for all the active elements
ellipseCA(res.ca)
```




```
## Ellipses around some columns only
ellipseCA(res.ca,ellipse="col",col.col.ell=c(rep("blue",2),rep("transparent",3)),
     invisible=c("row.sup","col.sup"))
```

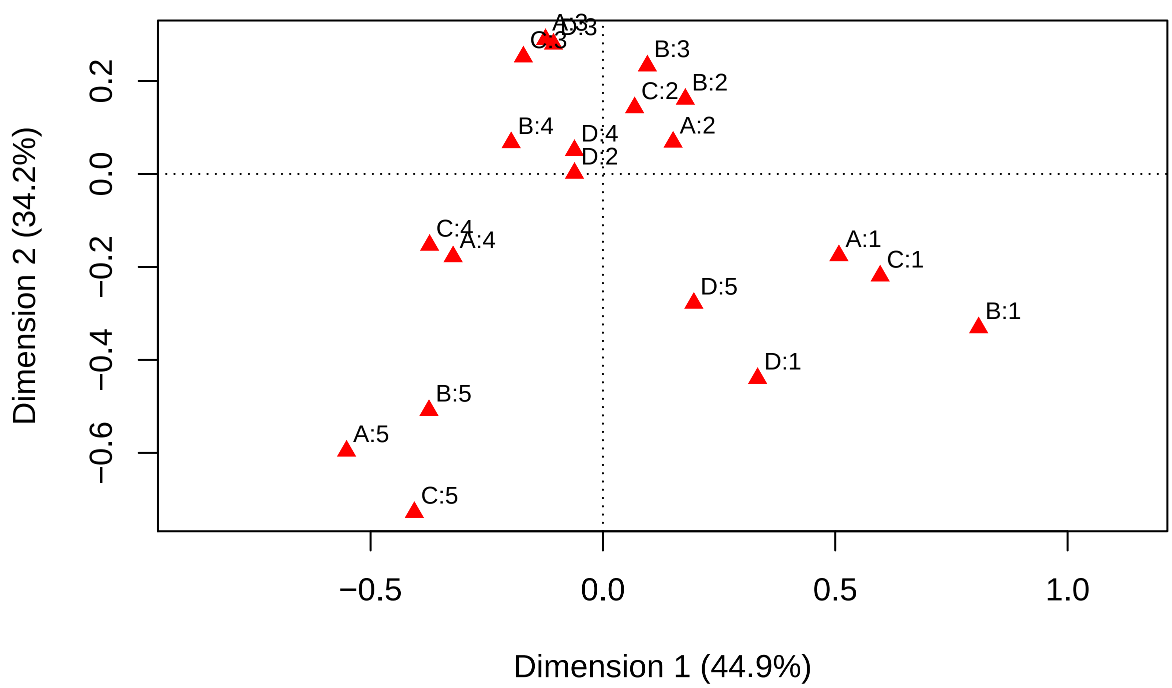



```
#Multiple CA with CA package
#mjca

data("wg93")
mjca(wg93[,1:4])
```

```
##
##  Eigenvalues:
##            1        2       3        4       5        6
## Value      0.076455 0.05822 0.009197 0.00567 0.001172 7e-06
## Percentage 44.91%   34.2%   5.4%     3.33%   0.69%    0%
##
##
##  Columns:
##                 A:1      A:2       A:3       A:4       A:5       B:1
## Mass     0.034156 0.092423  0.058553  0.051091  0.013777  0.020379
## ChiDist  1.343394 0.676433  0.947274  1.049164  2.214898  1.856041
## Inertia  0.061642 0.042289  0.052542  0.056238  0.067588  0.070203
## Dim. 1   1.836627 0.546240 -0.446797 -1.165903 -1.995217  2.924321
## Dim. 2  -0.727459 0.284443  1.199439 -0.736782 -2.470026 -1.370078
##               B:2      B:3       B:4       B:5       C:1      C:2       C:3
## Mass    0.049943 0.058840  0.080654  0.040184  0.043628 0.090700  0.056544
## ChiDist 1.034203 0.933288  0.760011  1.294006  1.241063 0.688137  0.977789
## Inertia 0.053417 0.051252  0.046587  0.067286  0.067197 0.042950  0.054060
## Dim. 1  0.641516 0.346050 -0.714126 -1.353725  2.157782 0.246828 -0.618996
## Dim. 2  0.666938 0.963918  0.280071 -2.107677 -0.908553 0.591611  1.044412
```

```
##               C:4       C:5       D:1       D:2       D:3       D:4
## Mass      0.044202  0.014925  0.017222  0.066590  0.057979  0.064868
## ChiDist   1.148345  2.132827  1.915937  0.843092  0.962007  0.860714
## Inertia   0.058289  0.067895  0.063217  0.047333  0.053657  0.048056
## Dim. 1   -1.348858 -1.467582  1.203782 -0.221151 -0.384656 -0.221635
## Dim. 2   -0.634647 -3.016588 -1.821975  0.006935  1.158694  0.210513
##               D:5
## Mass      0.043341
## ChiDist   1.136559
## Inertia   0.055986
## Dim. 1    0.707750
## Dim. 2   -1.151804
```

```r
plot(mjca(wg93[,1:4]))
```

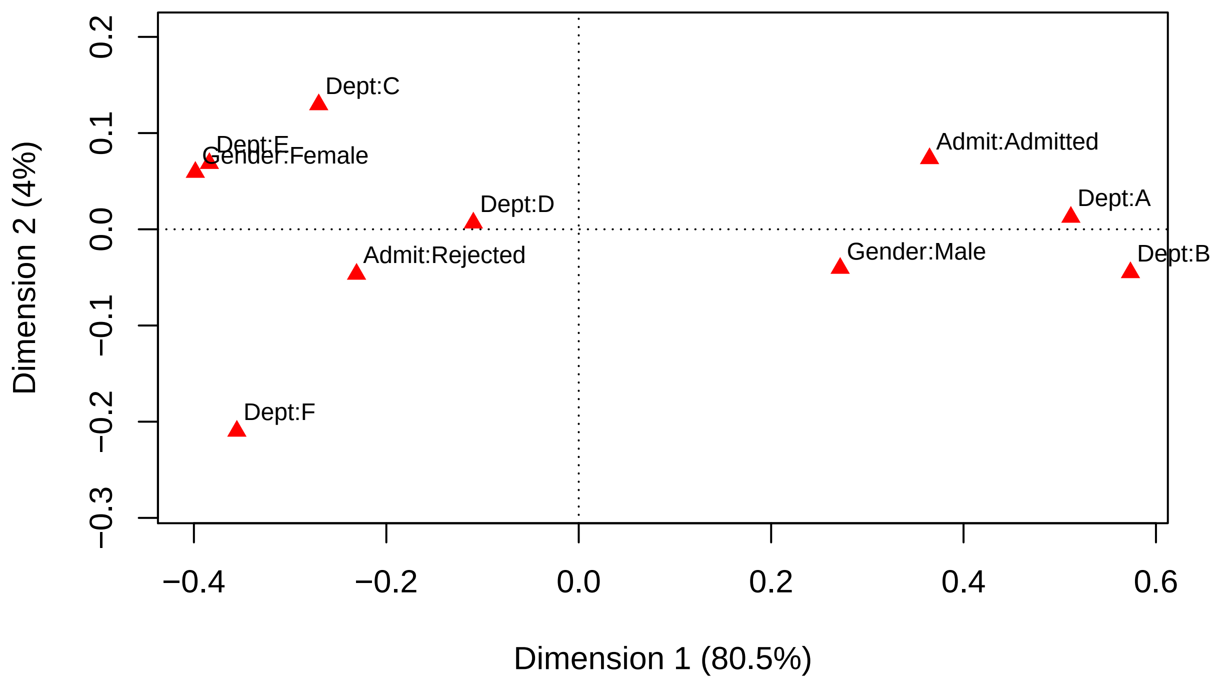



```r
data(UCBAdmissions)
mjca(UCBAdmissions)
```

```
##
##  Eigenvalues:
##            1        2        3  4  5
## Value      0.114945 0.005694 0  0  0
## Percentage 80.47%   3.99%    0% 0% 0%
##
##
##  Columns:
##          Admit:Admitted Admit:Rejected Gender:Female Gender:Male   Dept:A
```

```
## Mass           0.129253       0.204080      0.135145     0.198188 0.068714
## ChiDist        0.792214       0.501745      0.783694     0.534403 1.221480
## Inertia        0.081120       0.051377      0.083003     0.056600 0.102522
## Dim. 1         1.075594      -0.681222     -1.175453     0.801545 1.508902
## Dim. 2         0.974981      -0.617499      0.786218    -0.536124 0.167342
##             Dept:B    Dept:C    Dept:D    Dept:E    Dept:F
## Mass      0.043084  0.067609  0.058330  0.043011  0.052585
## ChiDist   1.584797  1.179839  1.257459  1.541085  1.390552
## Inertia   0.108210  0.094114  0.092231  0.102148  0.101680
## Dim. 1    1.691541 -0.797163 -0.323064 -1.132444 -1.048107
## Dim. 2   -0.595723  1.717196  0.088564  0.908371 -2.779621
```

```
plot(mjca(UCBAdmissions))
```




```
#Package By Nenadine and Greenacre
# library(ca)
#
# #EXAMPLE ##############
# #Where A and B are categorical factors
# # Correspondence Analysis
# library(ca)
 # mytable <- with(mydata, table(A,B)) # create a 2 way table
 # prop.table(mytable, 1) # row percentages
 # prop.table(mytable, 2) # column percentages
 # fit <- ca(mytable)
 # print(fit) # basic results
 # summary(fit) # extended results
```

```
# plot(fit) # symmetric map
# plot(fit, mass = TRUE, contrib = "absolute", map =
#   "rowgreen", arrows = c(FALSE, TRUE)) # asymmetric map

##############################
```

Questions:

Focusing on session 2.1

Maually cleaning data via excel

Starting with the session2.1 file only first. Questions before merging: What type of merge? What to do with the multiple ID's appearing, the "Total" column, and the percentage column seems incorrect as well. . . .

If we group the data by researchID: Do we have to relabel the variables by session # as well to keep track of which questions are asked?

dplyr: we can do group_by(Question) to see how students across the board fared on particular questions

Reading in the data

The way the data was collected: Were questions not asked for every session the class met? How many answer choices per question? Is it the same number of choices per question or does it vary? I saw some 'E' responses. . .

How does the 'Total' column work? Only seems to be recording a 0 or a 2, regardless of the number of questions answered.

I wasn't too sure for the multiple ID's appearing. . . So I manually began to remove them for cleaning.

case of ca62 researchID: 3 rows, 2 blanks, 1 3 answers, rest blank? Removed the 2 blanks rows, kept the 3 answers, rest blank row.

Removed the 3 NA values for the researchID's in the last 3 rows.

How many individual students were registered to take this class? Were any added on after the start? This could help us answer these questions we have about the data.

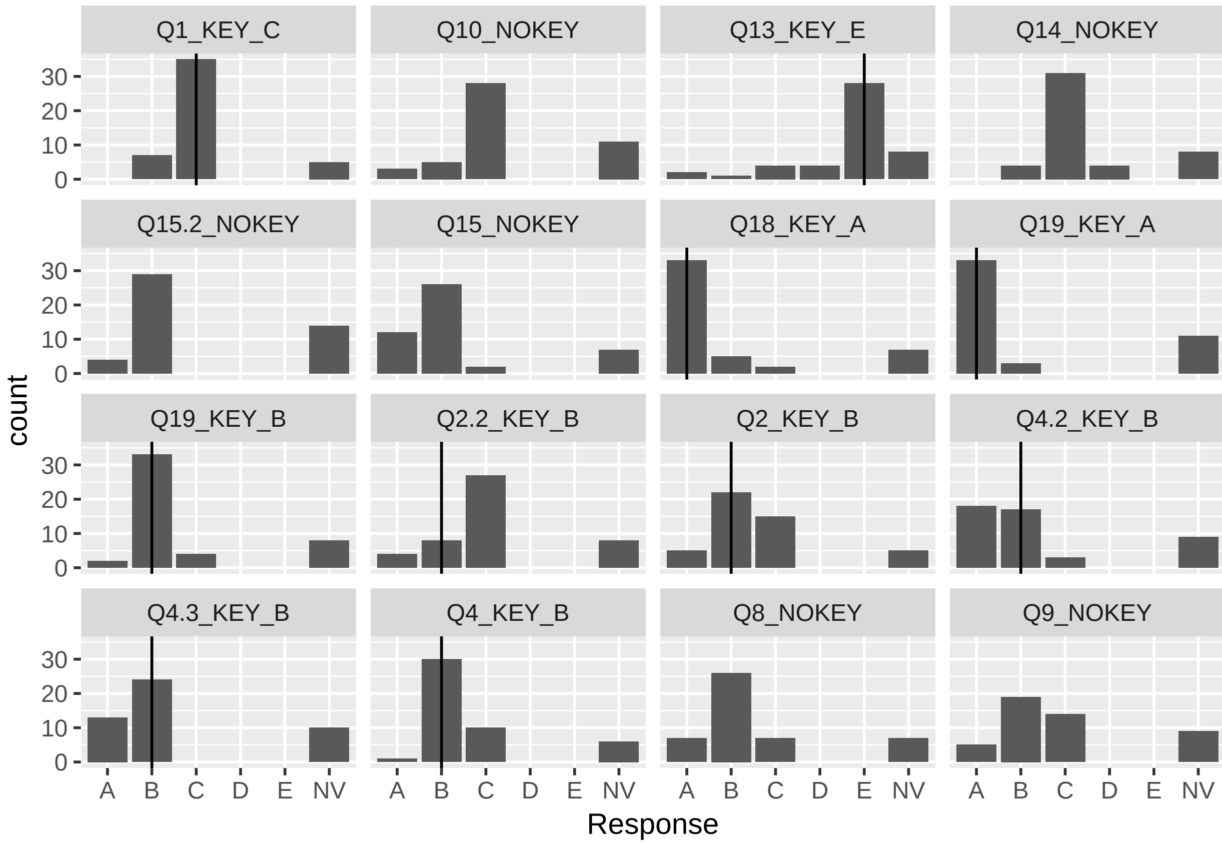
Why are some question numbers skipped?

47 unique ID's

Insights from the histograms This graph reveals the most difficult answer to be Question 2. What seems strange is that upon answering the questions a 2nd or 3rd time, the students seem to be getting a worse score(e.g. Questions 2 and 4).

Upon first glance, it seems that even when there is no correct answer(No Key Questions), the students tend to all select the same answer.

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

## 'data.frame':    752 obs. of  5 variables:
##  $ Total     : int  2 2 2 2 2 2 2 2 2 2 0 ...
##  $ Percentage: Factor w/ 2 levels "0%","100%": 2 2 2 2 2 2 2 2 2 2 1 ...
##  $ researchID: Factor w/ 47 levels "ab70","as160",..: 1 2 3 4 5 6 7 8 9 10 ...
##  $ Question  : Factor w/ 16 levels "Q1_KEY_C","Q10_NOKEY",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ Response  : Factor w/ 6 levels "A","B","C","D",..: 3 3 2 3 3 2 2 2 3 6 ...

## Warning: Ignoring unknown parameters: binwidth, bins, pad
```



Additional Exploration:

Try to find percentages of the correct answer for each Question

```
##
##    Q1_KEY_C   Q10_NOKEY   Q13_KEY_E   Q14_NOKEY Q15.2_NOKEY   Q15_NOKEY
##          47          47          47          47          47          47
##   Q18_KEY_A   Q19_KEY_A   Q19_KEY_B  Q2.2_KEY_B    Q2_KEY_B  Q4.2_KEY_B
##          47          47          47          47          47          47
##  Q4.3_KEY_B    Q4_KEY_B    Q8_NOKEY    Q9_NOKEY
##          47          47          47          47
```

```
##
##   A   B   C   D   E  NV
## 142 259 182   8  28 133
```

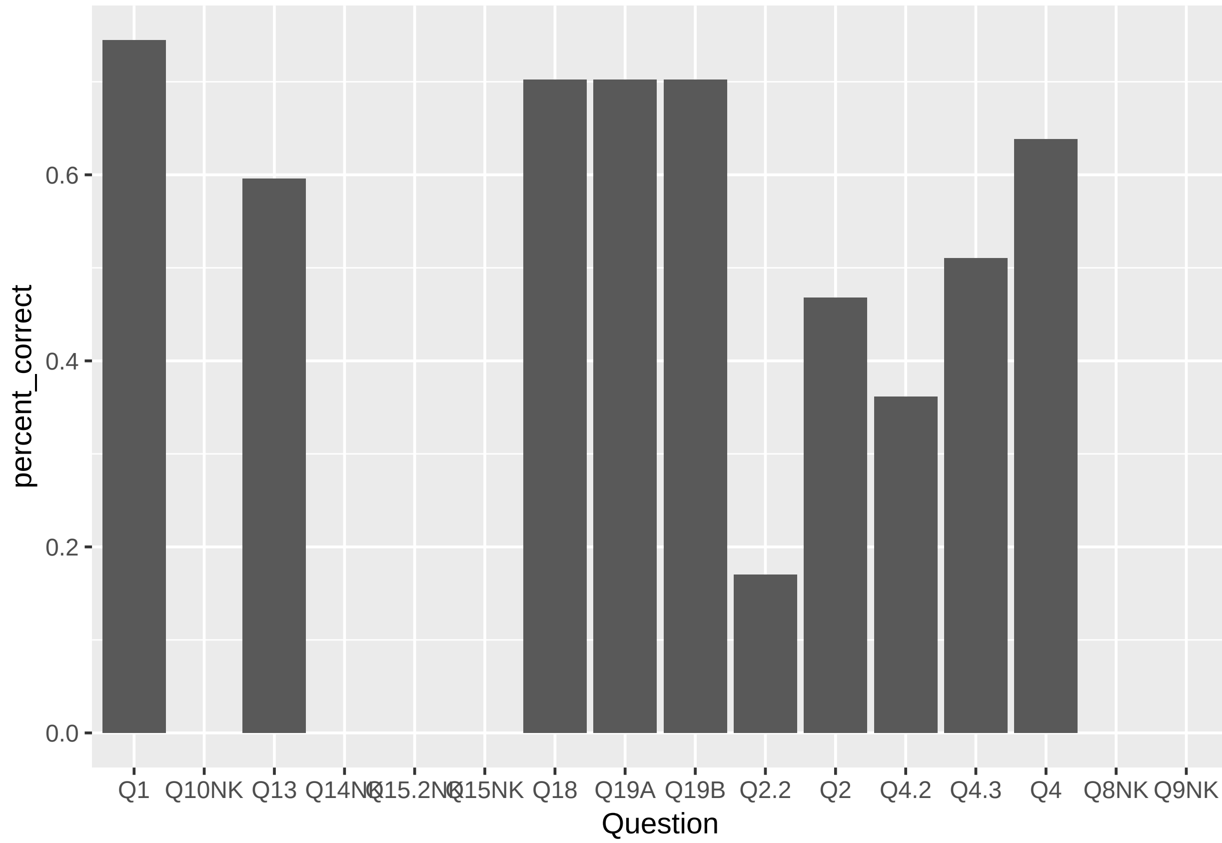
```
## [1] 0.5595745
```

```
## [1] 0.7446809
```

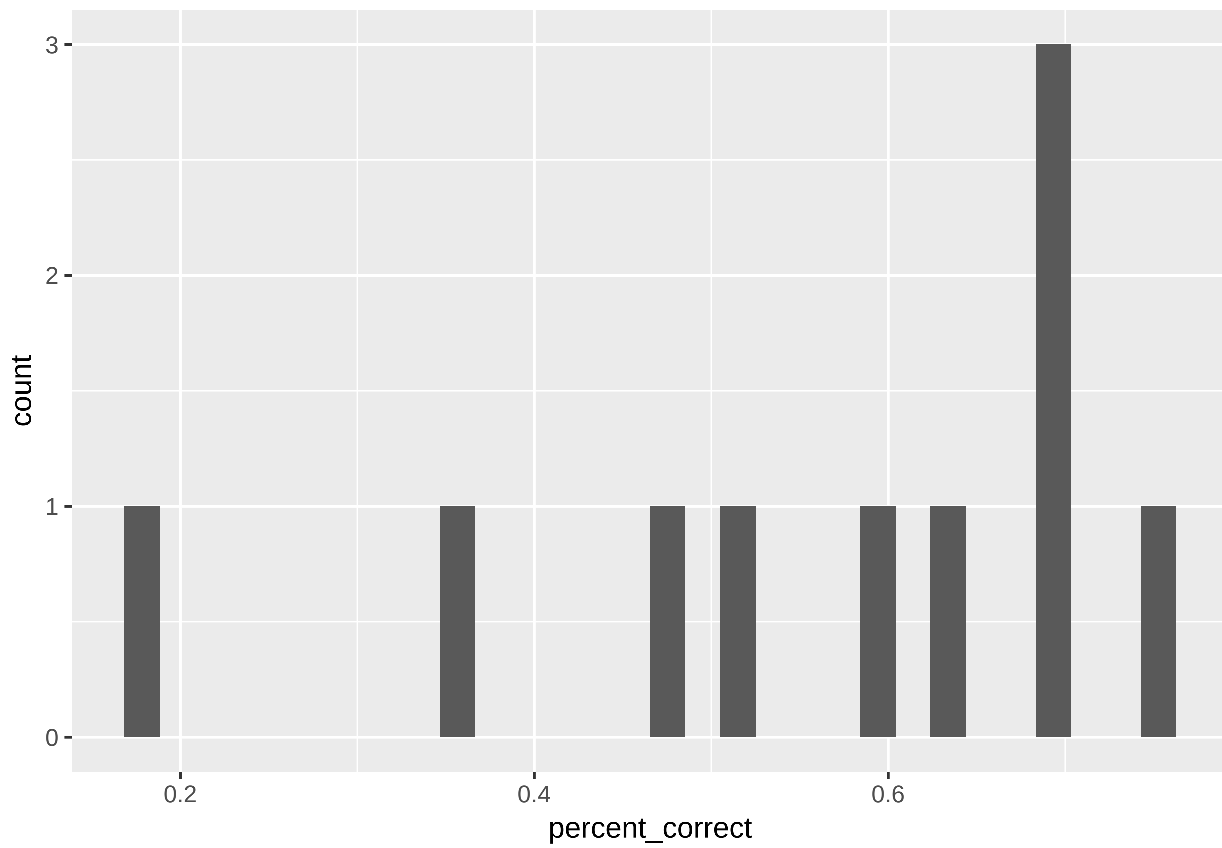
```
## [1] 0.4680851
```

```
## [1] 0.1702128
```

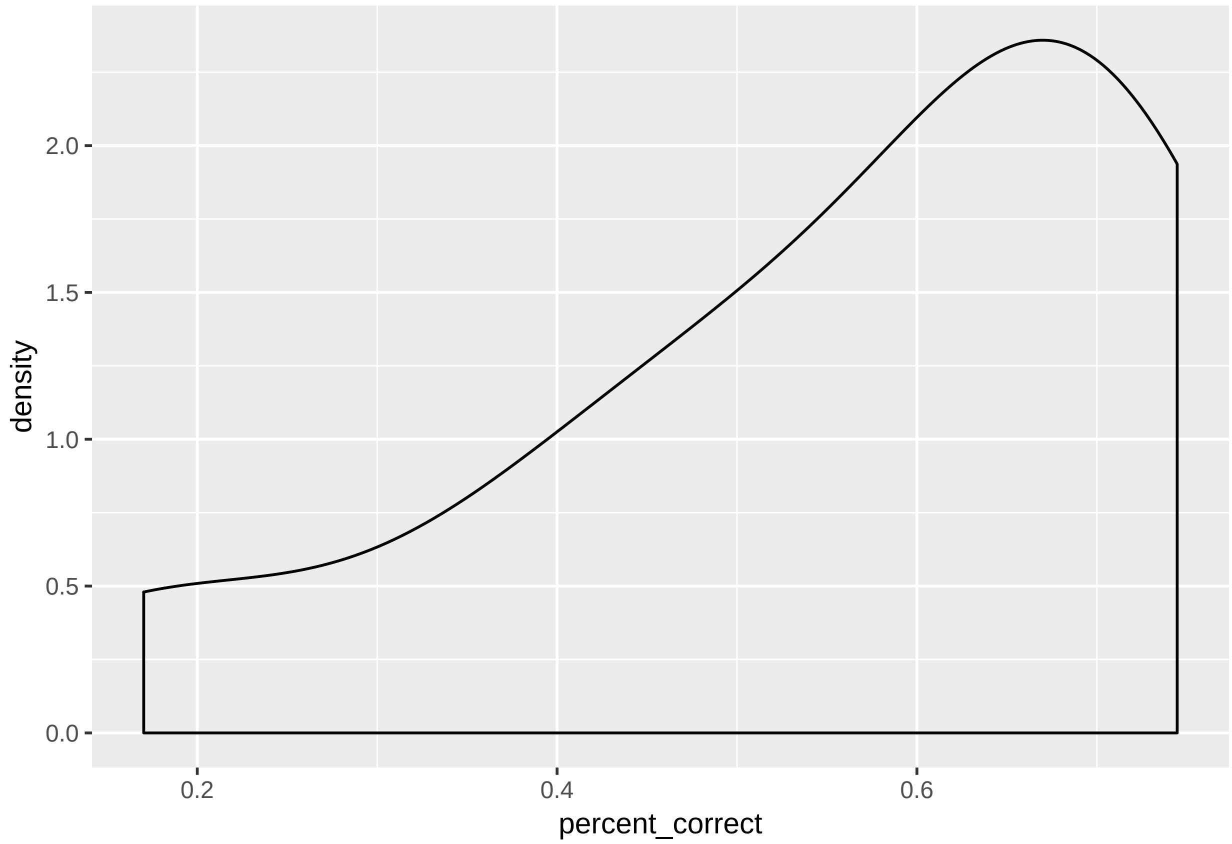
```
## Warning: Removed 6 rows containing missing values (geom_bar).
```




```
## Warning: Removed 6 rows containing non-finite values (stat_bin).
```




```
## Warning: Removed 6 rows containing non-finite values (stat_density).
```



Analysis by Student:

Try to find percentages of the correct answer for each Student

```
#Percent Correct Overall from Question Arrangement
mean(correct_vector == session2.1_tidy$Response, na.rm = TRUE)
```

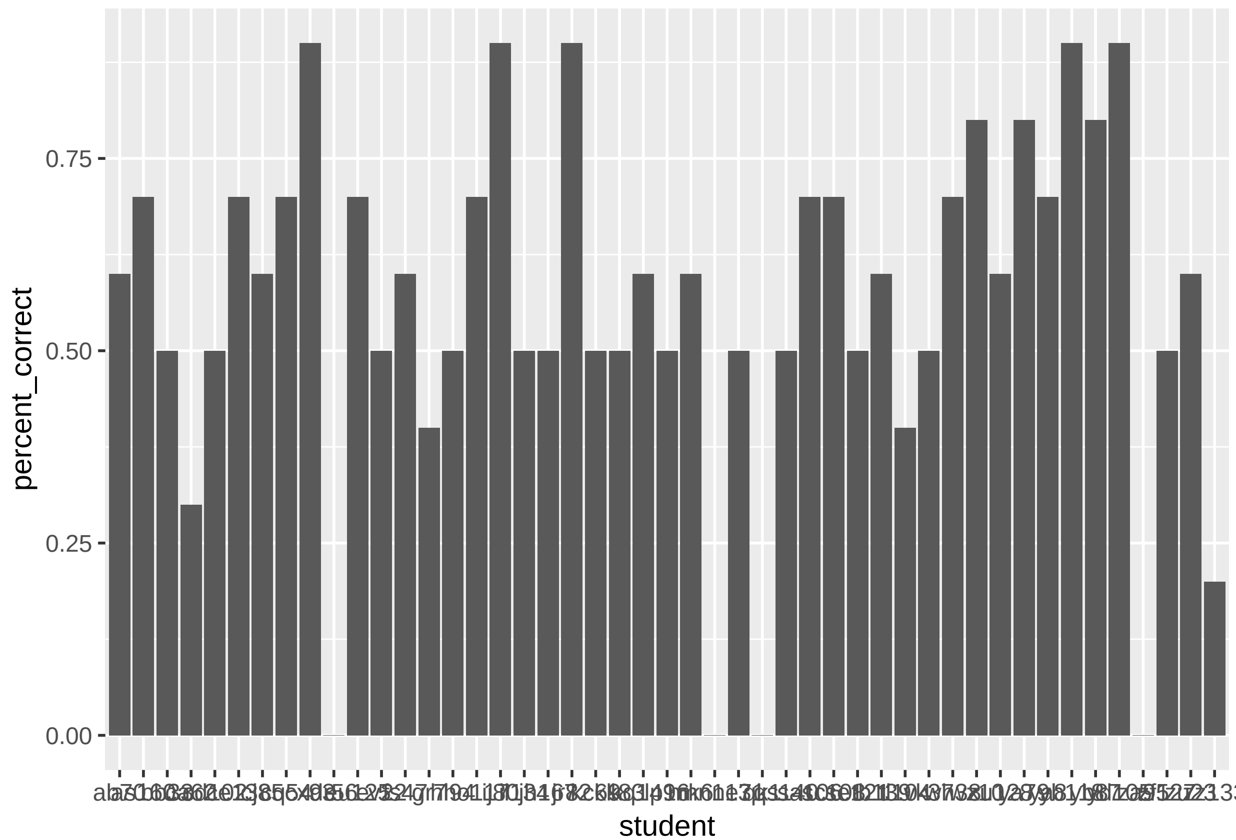
```
## [1] 0.5595745
```

```
correct_vector2 <- c(rep(c("C","B","B","B","B","B",NA,NA,NA,"E",NA,NA,NA,"A","A","B"),47))
session2.1_tidy_student <- session2.1_tidy %>% arrange(researchID)


#Percent Correct Overall from Student arrangement check YES
mean(correct_vector2 == session2.1_tidy_student$Response, na.rm = TRUE)
```

```
## [1] 0.5595745
```

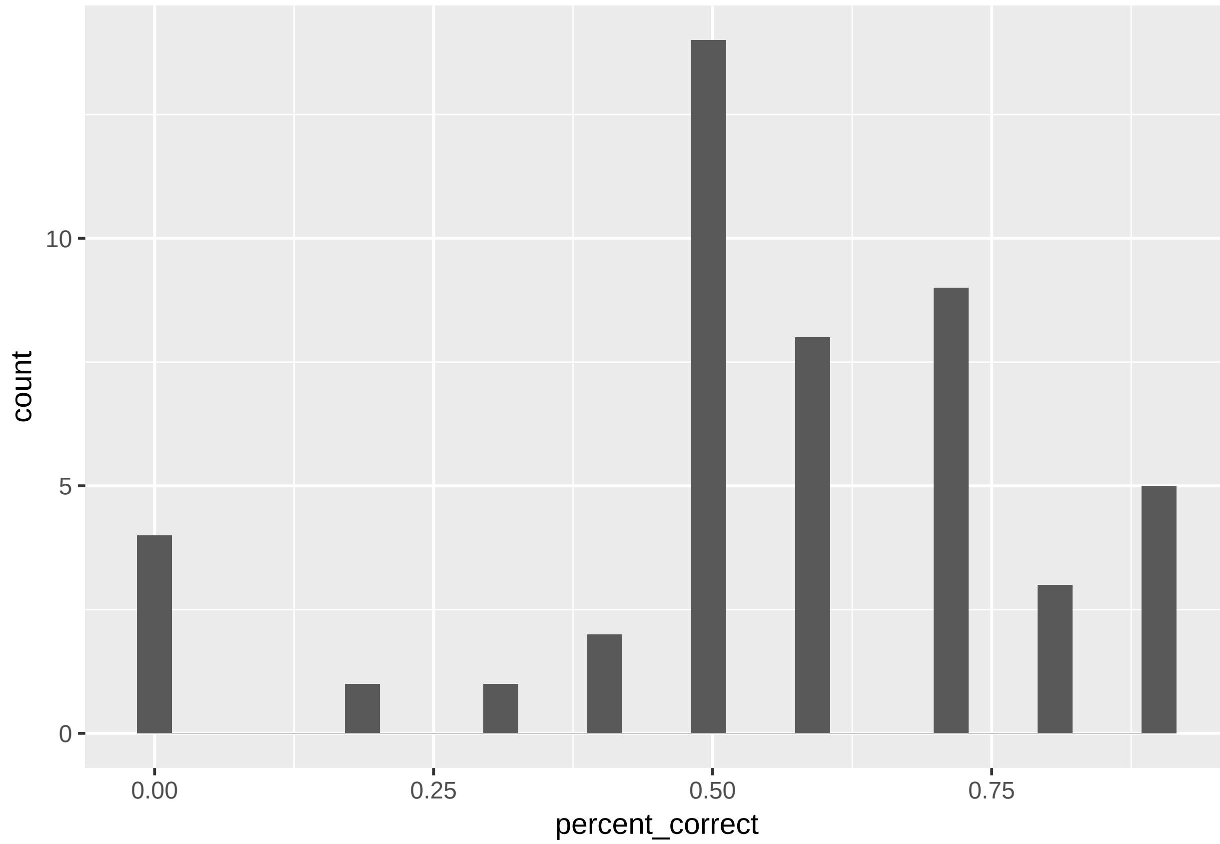
```
#New Student every 16 rows, 47 students total


temp<- c()
for(i in seq(from = 16, to = 752,by = 16)){
  temp[i] <- round(mean((correct_vector2 == session2.1_tidy_student$Response)[(i-15):i],na.rm=TRUE), dig
}

percent_correct_student<-data.frame(percent_correct=c(rep(NA,47)),student=unique(session2.1_tidy_studen

percent_correct_student$percent_correct<- temp[c(which(!is.na(temp)), which(is.nan(temp)))][order(c(whic
```

```r
#percent_correct_student is correct by checking the first student (correct_vector2 == session2.1_tidy_s

#Bar plot revealing student level
ggplot(data = percent_correct_student, aes(x = student, y = percent_correct)) + geom_bar(stat='identity
```

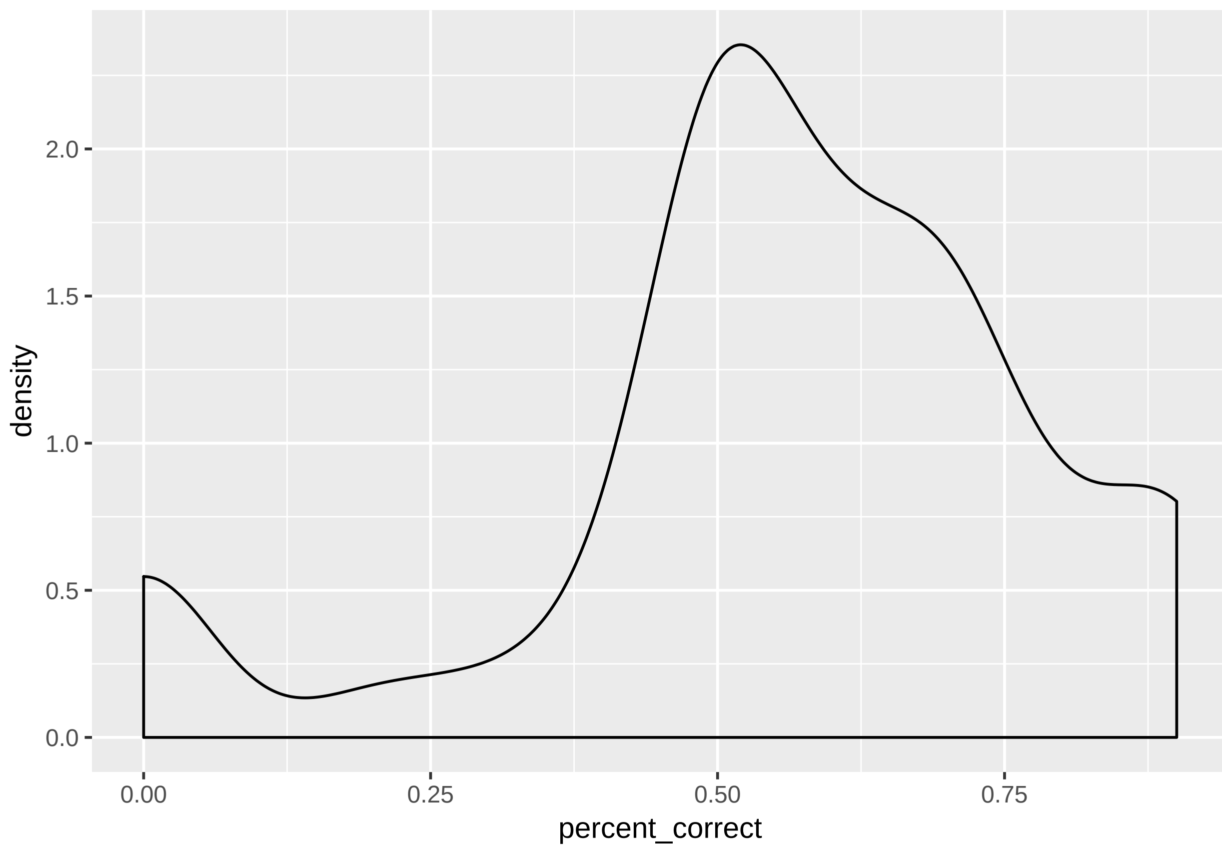



```r
#Histogram of Student Levels
ggplot(data = percent_correct_student, aes(x=percent_correct)) + geom_histogram()
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```




```
#Density Plot of Student Levels
ggplot(data = percent_correct_student, aes(x=percent_correct)) + geom_density()
```


density
2.0
1.5
1.0
0.5
0.0
0.00
0.25
0.50
0.75
percent_correct