R version 3.4.3 (2017-11-30) -- "Kite-Eating Tree"

Copyright (C) 2017 The R Foundation for Statistical Computing

Platform: x86\_64-w64-mingw32/x64 (64-bit)

R is free software and comes with ABSOLUTELY NO WARRANTY.

You are welcome to redistribute it under certain conditions.

Type 'license()' or 'licence()' for distribution details.

R is a collaborative project with many contributors.

Type 'contributors()' for more information and

'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or

'help.start()' for an HTML browser interface to help.

Type 'q()' to quit R.

[Workspace loaded from ~/.RData]

Restarting R session...

> library(psych)

> library(GPArotation)

Error in library(GPArotation) : there is no package called ‘GPArotation’

> library(readxl)

> library(vcd)

Loading required package: grid

> library(ggplot2)

Attaching package: ‘ggplot2’

The following objects are masked from ‘package:psych’:

%+%, alpha

> library(grid)

> 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

> library(gridExtra)

Attaching package: ‘gridExtra’

The following object is masked from ‘package:dplyr’:

combine

> library(reshape2)

> installed.packages("GPArotation")

Package LibPath Version Priority Depends Imports LinkingTo Suggests Enhances License License\_is\_FOSS

License\_restricts\_use OS\_type Archs MD5sum NeedsCompilation Built

> install.packages("GPArotation")

Installing package into ‘C:/Users/Qubix/Documents/R/win-library/3.4’

(as ‘lib’ is unspecified)

trying URL 'https://cran.rstudio.com/bin/windows/contrib/3.4/GPArotation\_2014.11-1.zip'

Content type 'application/zip' length 132542 bytes (129 KB)

downloaded 129 KB

package ‘GPArotation’ successfully unpacked and MD5 sums checked

The downloaded binary packages are in

C:\Users\Qubix\AppData\Local\Temp\RtmpEhY3E4\downloaded\_packages

> library(psych)

> library(GPArotation)

> library(readxl)

> library(vcd)

> library(ggplot2)

> library(grid)

> library(dplyr)

> library(gridExtra)

> library(reshape2)

> #Import data and global variables

> df <- read\_excel("Desktop/CEFR/CEFR Database Simplified 2017I.xlsx")

Error in read\_fun(path = path, sheet = sheet, limits = limits, shim = shim, :

Evaluation error: zip file 'Desktop/CEFR/CEFR Database Simplified 2017I.xlsx' cannot be opened.

> alpha = 0.01 # Used for hypothesis testing.

> options(warn=-1) #suppress warnings

> #Import data and global variables

> df <- read\_excel("C:/Users/Qubix/Desktop/GitHub/CEFRAnalysis2017/CEFR Database Simplified 2017I.xlsx")

> alpha = 0.01 # Used for hypothesis testing.

> #options(warn=-1) #suppress warnings

> BEGIN FUNCTIONS ----------

Error: unexpected symbol in "BEGIN FUNCTIONS"

>

> #---------- END FUNCTIONS ----------

>

> # Be sure to copy the data to "data" and then use the Scree plot to find the

> # number of factors (nfact).

> data<-df[c(7:19)]

> fap<-fa.parallel(data, fm = 'minres', fa = 'fa', plot = TRUE)

The estimated weights for the factor scores are probably incorrect. Try a different factor extraction method.

The estimated weights for the factor scores are probably incorrect. Try a different factor extraction method.

The estimated weights for the factor scores are probably incorrect. Try a different factor extraction method.

The estimated weights for the factor scores are probably incorrect. Try a different factor extraction method.

The estimated weights for the factor scores are probably incorrect. Try a different factor extraction method.

Parallel analysis suggests that the number of factors = 1 and the number of components = NA

>

> # Check for correlation--Kendall. If there is correlation, then the fa()

> # will use the oblimin or promax method; else, varimax.

>

> ken\_cor<-corr.test(data, method = "kendall")

> ken\_cor$p < alpha

WI WP CO CW ES RA CN VC GA OR TD CH SL

WI TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE

WP TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE

CO TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE

CW TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE

ES TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE

RA TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE

CN TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE

VC TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE

GA TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE

OR TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE

TD TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE

CH TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE

SL TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE

>

> # Read the resulting alpha-level correlational matrix and then run the fa() at

> # the number of factors using fap$nfact and fap$nfact+1.

>

> nfact\_fa<-fa(data,nfactors = fap$nfact, rotate = "promax",fm = "minres")

> nfact1\_fa<-fa(data,nfactors = fap$nfact+1, rotate = "promax",fm = "minres")

The estimated weights for the factor scores are probably incorrect. Try a different factor extraction method.

>

> # We have two models. Print them using a cutoff.

>

> fa\_cutoff <-function(n)

+ {

+ return(4.9123\*n^-0.476)

+ }

> print("----------BEGIN RESULTS----------")

[1] "----------BEGIN RESULTS----------"

> print("----------NFACT RESULTS----------")

[1] "----------NFACT RESULTS----------"

> print(paste0("Cutoff for this nfact model:", fa\_cutoff(nfact\_fa$n.obs)))

[1] "Cutoff for this nfact model:0.596112105963296"

> print(nfact\_fa$loadings, cutoff = fa\_cutoff(nfact\_fa$n.obs))

Loadings:

MR1

WI 0.908

WP 0.934

CO 0.946

CW 0.957

ES 0.924

RA 0.916

CN 0.871

VC 0.883

GA 0.920

OR 0.885

TD 0.927

CH 0.942

SL 0.828

MR1

SS loadings 10.801

Proportion Var 0.831

> print("----------NFACT1 RESULTS----------")

[1] "----------NFACT1 RESULTS----------"

> print(paste0("Cutoff for this nfact1 model:", fa\_cutoff(nfact1\_fa$n.obs)))

[1] "Cutoff for this nfact1 model:0.596112105963296"

> print(nfact1\_fa$loadings, cutoff = fa\_cutoff(nfact1\_fa$n.obs))

Loadings:

MR1 MR2

WI 0.688

WP 0.926

CO 0.825

CW 0.896

ES 0.808

RA

CN 0.644

VC 1.058

GA 0.770

OR 0.864

TD 0.789

CH 0.772

SL

MR1 MR2

SS loadings 6.538 2.166

Proportion Var 0.503 0.167

Cumulative Var 0.503 0.670

> print("----------END RESULTS----------")

[1] "----------END RESULTS----------"

>

> # Now validate both models and choose one.

>

> print("----------BEGIN VALIDATION----------")

[1] "----------BEGIN VALIDATION----------"

> print("----------NFACT VALIDATION----------")

[1] "----------NFACT VALIDATION----------"

> #print(paste0("RMS Error in Approx::",nfact\_fa$RMSEA))

> print(paste0("Tucker-Lewis Index (>0.95 is good fit):",nfact\_fa$TLI))

[1] "Tucker-Lewis Index (>0.95 is good fit):0.937815712044339"

> print("----------NFACT1 VALIDATION----------")

[1] "----------NFACT1 VALIDATION----------"

> #print(paste0("RMS Error in Approx::",nfact1\_fa$RMSEA))

> print(paste0("Tucker-Lewis Index (>0.95 is good fit):",nfact1\_fa$TLI))

[1] "Tucker-Lewis Index (>0.95 is good fit):0.973399936812376"

> print("----------COMPARISON VALIDATION----------")

[1] "----------COMPARISON VALIDATION----------"

> if(nfact1\_fa$TLI>nfact\_fa$TLI)

+ {

+ print("TLI finds nfact1 is the better model.")

+ png(file="./nfact1\_fa.png",width=800,height=800)

+ fa.diagram(nfact1\_fa, main = "FA Tree for nfact1")

+ dev.off()

+ } else {

+ print("TLI finds nfact is the better model.")

+ png(file="./nfact\_fa.png",width=800,height=800)

+ fa.diagram(nfact\_fa, main = "FA Tree for nfact")

+ dev.off()

+ }

[1] "TLI finds nfact1 is the better model."

RStudioGD

2

> print("----------END VALIDATION----------")

[1] "----------END VALIDATION----------"

>

> rm(fap, ken\_cor, nfact\_fa, nfact1\_fa, alpha, fa\_cutoff)

>

> # General various summary stats and visualizations.

> data2<-df[c(4:19)]

>

> # Box/Violin CEFR - Entire Year and by Term

> T18Box<-ggplot(stack(data2[c(4:16)]), aes(x = factor(ind, levels = names(data2[c(4:16)])), y = values, colours=values)) + geom\_boxplot()+scale\_x\_discrete(name="CEFR Category") +scale\_y\_continuous(name="Scale (A1=1, C2=6)")+geom\_jitter(alpha=0.07)+geom\_violin(alpha=0.1)+geom\_boxplot(alpha = 0.25)+ ggtitle("CEFR Scores for Academic Year") + theme(plot.title = element\_text(hjust = 0.5))+theme(text=element\_text(size=21))

> # Term 1

> rm(data2)

> data2<-filter(df,Term==1)

> data2<-data2[c(4:19)]

> T1Box<-ggplot(stack(data2[c(4:16)]), aes(x = factor(ind, levels = names(data2[c(4:16)])), y = values, colours=values)) + geom\_boxplot()+scale\_x\_discrete(name="CEFR Category") +scale\_y\_continuous(name="Scale (A1=1, C2=6)")+geom\_jitter(alpha=0.07)+geom\_violin(alpha=0.1)+geom\_boxplot(alpha = 0.25)+ ggtitle("CEFR Scores for T1") + theme(plot.title = element\_text(hjust = 0.5))+theme(text=element\_text(size=21))

> # Term 2

> rm(data2)

> data2<-filter(df,Term==2)

> data2<-data2[c(4:19)]

> T2Box<-ggplot(stack(data2[c(4:16)]), aes(x = factor(ind, levels = names(data2[c(4:16)])), y = values, colours=values)) + geom\_boxplot()+scale\_x\_discrete(name="CEFR Category") +scale\_y\_continuous(name="Scale (A1=1, C2=6)")+geom\_jitter(alpha=0.07)+geom\_violin(alpha=0.1)+geom\_boxplot(alpha = 0.25)+ ggtitle("CEFR Scores for T2") + theme(plot.title = element\_text(hjust = 0.5))+theme(text=element\_text(size=21))

> # Term 3

> rm(data2)

> data2<-filter(df,Term==3)

> data2<-data2[c(4:19)]

> T3Box<-ggplot(stack(data2[c(4:16)]), aes(x = factor(ind, levels = names(data2[c(4:16)])), y = values, colours=values)) + geom\_boxplot()+scale\_x\_discrete(name="CEFR Category") +scale\_y\_continuous(name="Scale (A1=1, C2=6)")+geom\_jitter(alpha=0.07)+geom\_violin(alpha=0.1)+geom\_boxplot(alpha = 0.25)+ ggtitle("CEFR Scores for T3") + theme(plot.title = element\_text(hjust = 0.5))+theme(text=element\_text(size=21))

> # Term 4

> rm(data2)

> data2<-filter(df,Term==4)

> data2<-data2[c(4:19)]

> T4Box<-ggplot(stack(data2[c(4:16)]), aes(x = factor(ind, levels = names(data2[c(4:16)])), y = values, colours=values)) + geom\_boxplot()+scale\_x\_discrete(name="CEFR Category") +scale\_y\_continuous(name="Scale (A1=1, C2=6)")+geom\_jitter(alpha=0.07)+geom\_violin(alpha=0.1)+geom\_boxplot(alpha = 0.25)+ ggtitle("CEFR Scores for T4") + theme(plot.title = element\_text(hjust = 0.5))+theme(text=element\_text(size=21))

> # Term 5

> rm(data2)

> data2<-filter(df,Term==5)

> data2<-data2[c(4:19)]

> T5Box<-ggplot(stack(data2[c(4:16)]), aes(x = factor(ind, levels = names(data2[c(4:16)])), y = values, colours=values)) + geom\_boxplot()+scale\_x\_discrete(name="CEFR Category") +scale\_y\_continuous(name="Scale (A1=1, C2=6)")+geom\_jitter(alpha=0.07)+geom\_violin(alpha=0.1)+geom\_boxplot(alpha = 0.25)+ ggtitle("CEFR Scores for T5") + theme(plot.title = element\_text(hjust = 0.5))+theme(text=element\_text(size=21))

> # Term 6

> rm(data2)

> data2<-filter(df,Term==6)

> data2<-data2[c(4:19)]

> T6Box<-ggplot(stack(data2[c(4:16)]), aes(x = factor(ind, levels = names(data2[c(4:16)])), y = values, colours=values)) + geom\_boxplot()+scale\_x\_discrete(name="CEFR Category") +scale\_y\_continuous(name="Scale (A1=1, C2=6)")+geom\_jitter(alpha=0.07)+geom\_violin(alpha=0.1)+geom\_boxplot(alpha = 0.25)+ ggtitle("CEFR Scores for T6") + theme(plot.title = element\_text(hjust = 0.5))+theme(text=element\_text(size=21))

> # Term 7

> rm(data2)

> data2<-filter(df,Term==7)

> data2<-data2[c(4:19)]

> T7Box<-ggplot(stack(data2[c(4:16)]), aes(x = factor(ind, levels = names(data2[c(4:16)])), y = values, colours=values)) + geom\_boxplot()+scale\_x\_discrete(name="CEFR Category") +scale\_y\_continuous(name="Scale (A1=1, C2=6)")+geom\_jitter(alpha=0.07)+geom\_violin(alpha=0.1)+geom\_boxplot(alpha = 0.25)+ ggtitle("CEFR Scores for T7") + theme(plot.title = element\_text(hjust = 0.5))+theme(text=element\_text(size=21))

> # Term 8

> rm(data2)

> data2<-filter(df,Term==8)

> data2<-data2[c(4:19)]

> T8Box<-ggplot(stack(data2[c(4:16)]), aes(x = factor(ind, levels = names(data2[c(4:16)])), y = values, colours=values)) + geom\_boxplot()+scale\_x\_discrete(name="CEFR Category") +scale\_y\_continuous(name="Scale (A1=1, C2=6)")+geom\_jitter(alpha=0.07)+geom\_violin(alpha=0.1)+geom\_boxplot(alpha = 0.25)+ ggtitle("CEFR Scores for T8") + theme(plot.title = element\_text(hjust = 0.5))+theme(text=element\_text(size=21))

> # Combine all into a stacked plot and render to YearCEFR.png

> png(file="./YearCEFR.png",width=4000,height=7000)

> grid.draw(rbind(ggplotGrob(T1Box), ggplotGrob(T2Box), ggplotGrob(T3Box), ggplotGrob(T4Box), ggplotGrob(T5Box), ggplotGrob(T6Box), ggplotGrob(T7Box), ggplotGrob(T8Box), ggplotGrob(T18Box), size="first"))

> dev.off()

RStudioGD

2

>

> rm(data2, T1Box, T2Box, T3Box, T4Box, T5Box, T6Box, T7Box, T8Box, T18Box)

>

> # Do the summary stats (overall and by term) as to wrap up.

> data2<-df[c(6:19)]

> print("----------PLOTTING SUMMARY DATA----------")

[1] "----------PLOTTING SUMMARY DATA----------"

> termsummary<- describeBy(data2[c(2:14)],data2$Term)

>

> # Make a chart of the median CEFRs by term

> cefr<-rownames(termsummary$`2`)

> t1md<-as.data.frame(termsummary$`1`$median)

> t2md<-as.data.frame(termsummary$`2`$median)

> # For 2017 ONLY

> t1md<-t2md #T1 has no data.

> # End For 2017 ONLY

> t3md<-as.data.frame(termsummary$`3`$median)

> t4md<-as.data.frame(termsummary$`4`$median)

> t5md<-as.data.frame(termsummary$`5`$median)

> t6md<-as.data.frame(termsummary$`6`$median)

> t7md<-as.data.frame(termsummary$`7`$median)

> t8md<-as.data.frame(termsummary$`8`$median)

> dfmd<-as.data.frame(t(bind\_cols(t1md, t2md, t3md, t4md, t5md, t6md, t7md, t8md)))

> colnames(dfmd)<-cefr

> rownames(dfmd)<-c("T1","T2","T3","T4","T5","T6","T7","T8")

> dfmd$Term<-rownames(dfmd)

>

> rm(t1md, t2md, t3md, t4md, t5md, t6md, t7md, t8md, cefr, termsummary, data, df)

>

> #Melt into a stacked table using Term

> long.dfmd<-melt(dfmd,id=c("Term"))

> png(file="./YearCEFRmeds.png",width=600,height=600)

> ggplot(long.dfmd,aes(x=Term,y=variable,color=value))+geom\_point(aes(size=value))+labs(y="Weight")+scale\_colour\_gradient(low = "red", high="green")

> dev.off()

RStudioGD

2

> source('C:/Users/Qubix/Desktop/GitHub/CEFRAnalysis2017/CEFRAnalysis.R')

The estimated weights for the factor scores are probably incorrect. Try a different factor extraction method.

The estimated weights for the factor scores are probably incorrect. Try a different factor extraction method.

Parallel analysis suggests that the number of factors = 1 and the number of components = NA

The estimated weights for the factor scores are probably incorrect. Try a different factor extraction method.

[1] "----------BEGIN RESULTS----------"

[1] "----------NFACT RESULTS----------"

[1] "Cutoff for this nfact model:0.596112105963296"

Loadings:

MR1

WI 0.908

WP 0.934

CO 0.946

CW 0.957

ES 0.924

RA 0.916

CN 0.871

VC 0.883

GA 0.920

OR 0.885

TD 0.927

CH 0.942

SL 0.828

MR1

SS loadings 10.801

Proportion Var 0.831

[1] "----------NFACT1 RESULTS----------"

[1] "Cutoff for this nfact1 model:0.596112105963296"

Loadings:

MR1 MR2

WI 0.688

WP 0.926

CO 0.825

CW 0.896

ES 0.808

RA

CN 0.644

VC 1.058

GA 0.770

OR 0.864

TD 0.789

CH 0.772

SL

MR1 MR2

SS loadings 6.538 2.166

Proportion Var 0.503 0.167

Cumulative Var 0.503 0.670

[1] "----------END RESULTS----------"

[1] "----------BEGIN VALIDATION----------"

[1] "----------NFACT VALIDATION----------"

[1] "Tucker-Lewis Index (>0.95 is good fit):0.937815712044339"

[1] "----------NFACT1 VALIDATION----------"

[1] "Tucker-Lewis Index (>0.95 is good fit):0.973399936812376"

[1] "----------COMPARISON VALIDATION----------"

[1] "TLI finds nfact1 is the better model."

[1] "----------END VALIDATION----------"

[1] "----------PLOTTING SUMMARY DATA----------"

> View(dfmd)

> View(long.dfmd)

> long.dfmd<-melt(dfmd,id=c("Term"))

> png(file="C:/Users/Qubix/Desktop/GitHub/CEFRAnalysis2017/YearCEFRmeds.png",width=600,height=600)

> ggplot(long.dfmd,aes(x=Term,y=variable,color=value))+geom\_point(aes(size=value))+labs(y="Weight")+scale\_colour\_gradient(low = "red", high="green")

> dev.off()

RStudioGD