T2

Friday, May 21, 2021 1:31 AM

Applied

1. ✓ Complete all tasks in the lecture note. If you would like any task to be explained in the tutorial, please post it on Padlet: https://padlet.com/xiaochenyang/94nywkzv5f8nw4li

2. Wolford and Hollingsworth (1974) were interested in the confusions made when a person attempts to identify letters of the alphabet viewed for some milliseconds only. A confusion matrix was constructed that shows the frequency with which each stimulus letter was mistakenly called something else. A section of this matrix is shown in the table below.

Letter	B	C	D	F	G
B	–				
C	3	–			
D	7	5	–		
F	3	5	7	–	
G	7	12	2	2	–

The dataset is available from 'Datasets for week 2', under the name 'letter.csv'. The task is to visualise the letters as points in one/two/three dimensions and discover if there is anything interesting. Below are some specific questions which may help your analysis.

(1) Is this a dissimilarity matrix? No. It is not square and symmetric.

(2) Which multidimensional scaling (MDS) method is more appropriate? *(Can we use principal component analysis to visualise the data?)*

(3) How many dimensions would you keep?

(4) Do you see any clusters?

(3) ndim=2

(2) non-metric MDS
when covert simi. to dissimi., add a new parameter, subtract entry by large value, not quantity, use non-parametric MDS. just need to keep the rank.
Cannot use PCA b/c no original data.

(5) Apart from the number of dimensions, have you introduced any other parameter in your analysis? If so, study the influences by trying different values. different max value z[ij]

(6) Finally, as mentioned in the lecture note, MDS are sensitive to initial configurations. Try multiple initial configurations and compare the results. mds(..., init="random") *(You will need to look into the help page of `mds` to find out how to change the initial configuration.)*

3. We have looked at European employment dataset in Week 1 (Example 5). Let's now analyse this dataset again using MDS. MDS allows us to visualise how similar two countries are as well as visualising the variables. You may start the analysis by thinking about the following questions.

(1) For this dataset, is it more appropriate to use the original variables or the standardised variables when calculating the distance matrix?

(2) How to compute a distance matrix from the dataset? You will need two distance matrices, one for visualising the countries and the other for visualising the variables. *(Hint: the `dist` command).*

(3) Which MDS method is more appropriate?

(4) How many dimensions would you keep?

(5) Do you see any clusters when visualising the countries, or any country distinct from others?

(6) How would you interpret the plot on variables (i.e. variables as points)?

```
#mining_tut_2
setwd("/Users/kurisuuu/Documents/glasgow_stats_2021/semester\ 2/mining/Datasets\ for\ week\ 2-20210520")
#--------------------------------------------------------------------------------
# STATS5099 Data Mining Tutorial 2
#--------------------------------------------------------------------------------
# Task 2
#--------------------------------------------------------------------------------

letter <- read.csv("letter.csv",row.names=1)
letter

########################
########## Q1 ##########
########################

#convert into a dissimilarity matrix
library(smacof)
letter.dist <- sim2diss(letter,method=max(letter)+1) #z-s_ij
letter.dist <- as.dist(letter.dist)
letter.dist

########################
######## Q2,Q3 #########
########################
#Which method to choose?
#Because we have deduced dissimilarities from similarities, the absolute
#dissimilarities delta_ij depend on the value of personally chosen z.
#This is the case where the non-metric MDS makes most sense.
#However, we will also see that metric scalings do the job as well.

#nonmetric MDS
#2D
set.seed(1)
letter.nmds2 <- mds(letter.dist, ndim=2, type="ordinal")
plot(letter.nmds2,asp=1)
plot(letter.nmds2,plot.type="Shepard")
letter.nmds2$stress
# Kruskal (1964) gave following advise about stress values
# based on his experience:
# Stress Goodness-of-fit
# 0.200  poor
# 0.100  fair
# 0.050  good
# 0.025  excellent
# 0.000  perfect
# More recent articles caution against using a table like this since
# acceptable values of stress depends on the quality of
# the distance matrix and the number of objects in that matrix.

#3D
set.seed(1)
letter.nmds3 <- mds(letter.dist, ndim=3, type="ordinal")
library(rgl)
# plot3d(letter.nmds3$conf[,1],letter.nmds3$conf[,2],
#       letter.nmds3$conf[,3],type="",
#       xlab="Axis 1",ylab="Axis 2",zlab="Axis 3",asp=1)
text3d(letter.nmds3$conf[,1],letter.nmds3$conf[,2],
      letter.nmds3$conf[,3],texts=names(letter.dist),asp=1)
plot(letter.nmds3,plot.type="Shepard")
letter.nmds3$stress

#record stress for the scree plot
N_dim <- 1:(nrow(letter)-1)
letter.nmds <- matrix(nrow=length(N_dim),ncol=2)
for (i in N_dim){
  letter.nmds[i,1] <- i
  letter.nmds[i,2] <- mds(letter.dist, ndim=i, type="ordinal")$stress
}
plot(letter.nmds, type="b", main="scree plot",
    xlab="number of dimensions", ylab="stress-1")

#metric MDS
library(smacof)
set.seed(1)
letter.mds2 <- mds(letter.dist, ndim=2, type="interval")
plot(letter.mds2)
plot(letter.mds2,plot.type="Shepard")
print(c(letter.nmds2$stress,letter.mds2$stress))

N_dim <- 1:(nrow(letter)-1)
letter.mds <- matrix(nrow=length(N_dim),ncol=2)
for (i in N_dim){
  letter.mds[i,1] <- i
  set.seed(1)
  letter.mds[i,2] <- mds(letter.dist, ndim=i, type="interval")$stress
}
plot(letter.mds, type="b", main="scree plot",
    xlab="number of dimensions", ylab="stress")
letter.mds[3,]
#metric MDS is worse than nonmetric MDS
#caution: nonmetric MDS is more prone to overfitting than MDS

########################
########## Q5 ##########
########################

#When converting confusion to distance, we have introduced the parameter z.
#Now we will create a function to investigate the effect of z.

Z <- seq(max(letter)+1, by=10, length.out = 20)
letter.z <- matrix(nrow=length(Z),ncol=2)
for (i in 1:length(Z)){
  letter.dist <- sim2diss(letter,method=Z[i])
  set.seed(1)
  letter.z[i,1] <- Z[i]
  letter.z[i,2] <- mds(letter.dist, ndim=3, type="ordinal")$stress
}
plot(letter.z, xlab="integer z", ylab="stress-1")

########################
########## Q6 ##########
########################

#To test the sensitivity to initial configuration, we need to change
#the current way of initialisation, which is classical MDS by default.
#Specially, we need to use the argument: init="random"

Seed <- 1:100
letter.seed <- matrix(nrow=length(Seed),ncol=1)
for (i in Seed){
  letter.dist <- sim2diss(letter,method=max(letter)+1)
  set.seed(i)
  letter.seed[i] <- mds(letter.dist,ndim=3,type="ordinal",init="random")$stress
}
hist(letter.seed)



#--------------------------------------------------------------------------------
# Task 3
#--------------------------------------------------------------------------------

employ <- read.table("eurojob.txt",header=TRUE,row.names=1)

########################
########## Q1 ##########
########################

apply(employ,2,var)

#As the variables have different variances, it would be better to use
#standardised variables when computing the pairwise distances.

########################
########## Q2 ##########
########################

employ.sd <- scale(employ)
employ.dist <- dist(employ.sd)
employ.dist #to visualise countries as data points
employ.dist <- dist(t(employ.sd))
employ.dist #to visualise variables as data points

########################
########## Q3 ##########
########################

#As we have created the distance matrix directly from the data,
#metric MDS would be a better choice.

########################
######### Q4-6 #########
########################

#There are three types of metric MDS covered in the lecture.

#cmdscale is omitted as it is equivalent to PCA (check that!)

#metric MDS computed using 'mds' are explained in Task 2.
#The steps for this task is similar and hence details are omitted.

#There are a few differences when using Sammon mapping, which are explained below.

#Sammon
employ.dist <- dist(t(employ.sd))
library(MASS)
employ.sm <- sammon(employ.dist, k=2)
plot(employ.sm$points, type="n", xlab="Axis 1", ylab="Axis 2",asp=1)
text(employ.sm$points, labels=names(employ.dist))

#Shepard diagram needs to be created manually
plot(as.vector(employ.dist),as.vector(dist(employ.sm$points)),pch=16,
    xlab="Dissimilarities", ylab="Configuration Distances")
```