Statistics with R – Advanced Level

Section 3

Grouping Methods

Lesson 20 - MDS - data are not distances

```
j <- read.csv("juices.csv")

View(j)

#########
### how to perform the multidimensional scaling procedure
### when the data are NOT distances between objects
#########

### we will generate two "hidden attributes" for our
brands,
### compute the brands coordinates on each attribute
### and build a perceptual map

### transpose the data set to compute the distance between
columns (brands)

j1 <- t(j)

### compute the distances between brands

prox <- dist(j1)

print(prox)
```

```
### we apply the cmdscale function to the new matrix prox
### requiring the program to generate two dimensions

model <- cmdscale(prox, k=2)

### this model contains the coordinates of each brand on
the two dimensions

print(model)

######### preparing the data in order to plot the brands
######## in a two-dimensional space (i.e. build a
perceptual map)

### make a data frame with the coordinates

coord <- as.data.frame(model)

View(coord)

### add a new column to the data frame coord, containing
the brand names

coord <- cbind(coord, c("b1", "b2", "b3", "b4", "b5", "b6",
"b7", "b8", "b9", "b10", "b11"))

### rename the columns

colnames(coord) <- c("attribute1", "attribute2", "brand")

### plot the points (with labels)

require(ggplot2)

ggplot()+geom_point(data=coord, aes(x=attribute1,
y=attribute2), color="gray")+
      geom_text(data=coord, aes(x=attribute1, y=attribute2,
label=brand, size=1))
```

Lesson 21 - MDS - data are distances

```
dest <- read.csv("destinations.csv")
```

```
View(dest)

#########
### how to perform the multidimensional scaling procedure
### when the data ARE distances between objects
#########

### we will generate two "hidden attributes" for our
destinations,
### compute the destinations coordinates on each attribute
### and draw a perceptual map

### the dest data frame is already a matrix
### so we can use it as an argument in the cmdscale
function

model <- cmdscale(dest, k=2)

print(model)

### store the coordinates as a data frame

coord <- as.data.frame(model)

View(coord)

### add the destination names and name the data frame
columns

coord <- cbind(coord, c("A", "B", "C", "D", "E", "F", "G"))

colnames(coord) <- c("attribute1", "attribute2",
"destination")

### plot the map with labels

require(ggplot2)

ggplot()+geom_point(data=coord, aes(x=attribute1,
y=attribute2), color="gray")+
  geom_text(data=coord, aes(x=attribute1, y=attribute2,
label=destination, size=10))
```

Lesson 22 - Factor analysis basics

```
brd <- read.csv("brandsurvey.csv")

View(brd)

#########
### how to perform an exploratory factor analysis
#########

###### run a principal component analysis first
###### to find the best number of factors

pcamodel <- princomp(brd, cor=T)

### Benzecri criterion
### proportion of explained variance should be at least 70%

summary(pcamodel)

### Kaiser criterion
### the eigenvalue should be higher than one

### compute the eigenvalues

eigenv <- pcamodel$sdev^2

print(eigenv)

### Evrard criterion
### visual inspection of the scree plot

### get the scree plot

screeplot(pcamodel, type="line")

### based on the pca results we decide to retain three
factors

######## run the factor analysis with the varimax rotation
######## and print the factor matrix, with a 0.3 cutoff
```

```
model <- factanal(brd, factors = 3, rotation = "varimax")

print(model, digits=2, cutoff=.3, sort=TRUE)

### compute the communalities

comm <- 1 - model$uniquenesses

print(comm)
```

Lesson 23 - Factor analysis sample adequacy measures

```
brd <- read.csv("brandsurvey.csv")

View(brd)

#########
### exploratory factor analysis - adequacy tests
#########

### how to compute the Kaiser-Meier-Olkin measure
### and the Bartlett's sphericity test

### get the correlation matrix for our variables

corm <- cor(brd)

View(corm)

#### compute the KMO indicator

require(psych)

KMO(corm)

### alternatively

KMO(brd)

### compute the Bartlett's test

cortest.bartlett(corm, 106)
```

```
### 106 is the sample size (number of respondents)
### 100 is the default
### the sample size must be specified only when the
argument is a matrix

### alternatively

cortest.bartlett(brd)
```

Lesson 24 - Simple correspondence analysis

```
toyo <- read.csv("toyota.csv")

View(toyo)

#########
### how to perform a simple correspondence analysis
#########

### our goal is to determine the models that sell better on
each continent

### we will create a perceptual map with the profiles
(categories) of the two variables:
### car model and continent

### create a contingency table with our variables

tt <- xtabs(~model+continent, data=toyo)

print(tt)

###### run the analysis and display the results

require(ca)

model <- ca(tt)

summary(model)

### the eigenvalues show the proportion of the total
variance explained by each axis (attribute)
```

```
### mass - proportion of each row/column in the total

### inr (inertia) shows the amount of variance each
row/column
### accounts for the total inertia value

### mass, inr (inertia) and qlt (quality) are multiplied by
1000
### so are cor (squared correlations)

### k=1 and k=2: coordinates of each profile on each
dimension
### quality = cor(k=1) + cor(k=2)


############# prepare the data in order to plot the
profiles

### get the row and column coordinates

rco <- ca(tt)$rowcoord   ## coordinates of the car models

cco <- ca(tt)$colcoord   ## coordinates of the continents

print(rco)

print(cco)

### put the coordinates in dataframes

rcodata <- data.frame(rco)

ccodata <- data.frame(cco)

View(rcodata)

View(ccodata)


### plot the coordinates with ggplot2

require(ggplot2)

ggplot()+
```

```
  geom_point(data=rcodata, aes(x=Dim1, y=Dim2), size=3,
color="red", shape=16)+
  geom_point(data=ccodata, aes(x=Dim1, y=Dim2), size=3,
color="blue", shape=16)+
  geom_text(data=rcodata, aes(x=Dim1, y=Dim2-0.07,
label=rownames(rcodata), size=16))+
  geom_text(data=ccodata, aes(x=Dim1, y=Dim2-0.07,
label=rownames(ccodata), size=16))+
  geom_hline(yintercept = 0, colour = "black")+
  geom_vline(xintercept = 0, colour = "black")+
  theme(legend.position="none")
```

Lesson 25 - Multiple correspondence analysis

```
retail <- read.csv("retail.csv")

View(retail)

#########
### how to perform a multiple correspondence analysis
#########

### our goal is to determine whther there is an association
### between juice type, packaging and retail channel

### we will create a perceptual map with the profiles of
the three variables

### run the analysis (with two dimensions)

require(MASS)

model <- mca(retail, nf=2)

print(model)

########## prepare the data to plot the profiles

### create the table with the column coordinates

mtable <- model$cs

print(mtable)
```

```
### convert the table into a data frame

ccodata <- data.frame(mtable)

### rename the rows and columns

colnames(ccodata) <- c("Dim1", "Dim2")

rownames(ccodata) <- c("Apple Juice", "Strawberry Juice",
"Bottle", "Tetra Pak", "Convenience Store", "Supermarket")

View(ccodata)

### create the plot

require(ggplot2)

ggplot()+
  geom_point(data=ccodata, aes(x=Dim1, y=Dim2), size=3,
color="red", shape=16)+
  geom_text(data=ccodata, aes(x=Dim1, y=Dim2,
label=rownames(ccodata), size=16))+
  geom_hline(yintercept = 0, colour = "black")+
  geom_vline(xintercept = 0, colour = "black")+
  theme(legend.position="none")
```

Lesson 26 - Hierarchical cluster

```
car <- read.csv("cars.csv")

View(car)

#########
### how to perform a hierarchical cluster analysis
#########

### we will cluster the car models by their
characteristics:
### price, engine displacement, power, fuel consumption,
maximum speed

### create a new data set with the clustering variables
```

```
car2 <- cbind(car$price, car$engine, car$power,
car$fuelcons, car$speed)

View(car2)

### add column names and (important!) row names

colnames(car2) <- c("price", "engine", "power", "fuelcons",
"speed")

rownames(car2) <- car$carmodel

### compute the distance matrix

dm <- dist(car2, method = "euclidean")

### create the clustering model

model <- hclust(dm, method = "ward.D")

### plot the model (as a dendrogram)

plot(model, labels=rownames(car2))

### get cluster membership

cutree(model, k=2:4)

### visualize clusters on the dendrogram

rect.hclust(model, k=5, border="red")
```

Lesson 27 - K-means cluster

```
ctr <- read.csv("countries.csv")

View(ctr)

#########
### how to perform a k-means cluster analysis
#########
```

```
### we will cluster the countries by their
### demographic and economic characteristics

####### data preparation

### remove the missing values

ctr <- na.omit(ctr)

### create a matrix with all the clustering variables

ctr2 <- cbind(ctr$urban, ctr$flexp, ctr$mlexp,
ctr$literacy, ctr$infmort, ctr$gdp, ctr$density,
ctr$popincr)

View(ctr2)

### standardize the clustering variables (recommended)

ctr2 <- scale(ctr2)

### name the rows and the columns

rownames(ctr2) <- ctr$country

colnames(ctr2) <- c("urban", "flexp", "mlexp", "literacy",
"infmort", "gdp", "density", "popincr")

######### run the k-means algorithm, with three clusters

model <- kmeans(ctr2, 3)

print(model)

###### get some relevant information

### clustering vector

model$cluster

### size of clusters

model$size
```

```
### cluster centers

model$centers

####### sums of squares

### total sum of squares

model$totss

### within-cluster sum of squares

model$withinss

### between-cluster sum of squares, i.e. totss-tot.withinss

model$betweenss

### total within-cluster sum of squares, i.e. sum(withinss)

model$tot.withinss

##########################

###### plot the clusters and their centers (scatterplot
chart)
###### in a two dimensional space
###### the axes will be literacy and gross domestic product
(gdp)

### put the cluster centers in a data frame

centers <- data.frame(model$centers)

View(centers)

### convert the ctr2 matrix into a data frame

ctr3 <- data.frame(ctr2)

View(ctr3)

### build the scatterplot
```

```
require(ggplot2)

ggplot()+geom_point(data=ctr3, aes(x=literacy, y=gdp, color
= model$cluster))+
  geom_point(data=centers, aes(x=literacy[1], y=gdp[1],
size=10), color="red", shape=8)+
  geom_point(data=centers, aes(x=literacy[2], y=gdp[2],
size=10), color="green", shape=8)+
  geom_point(data=centers, aes(x=literacy[3], y=gdp[3],
size=10), color="magenta", shape=8)+
  theme(legend.position="none")

################ we can put labels to the centers

ggplot()+geom_point(data=ctr3, aes(x=literacy, y=gdp, color
= model$cluster))+
  geom_point(data=centers, aes(x=literacy[1], y=gdp[1],
size=10), color="red", shape=8)+
  geom_text(data=centers, aes(x=literacy[1], y=gdp[1]-0.09,
label="Cluster 1"))+
  geom_point(data=centers, aes(x=literacy[2], y=gdp[2],
size=10), color="green", shape=8)+
  geom_text(data=centers, aes(x=literacy[2], y=gdp[2]-0.09,
label="Cluster 2"))+
  geom_point(data=centers, aes(x=literacy[3], y=gdp[3],
size=10), color="magenta", shape=8)+
  geom_text(data=centers, aes(x=literacy[3], y=gdp[3]-0.09,
label="Cluster 3"))+
  theme(legend.position="none")
```

Lesson 28 - Simple discriminant analysis

```
comp <- read.csv("company.csv")

View(comp)

#########
### how to perform a simple discriminant analysis
#########

### we will determine whether the employee gender can be
predicted
### based on other variables
```

```
### dependent variable: gender
### independent variables: education level (educ), salary,
### months at the current job (jobtime), previous
experience in months (prevexp)

### run the DA using the lda function

require(MASS)

### get the prior probabilities, the group means and
### the coefficients of the discriminant function (CV =
FALSE)

lmod1 <- lda(gender~educ+salary+jobtime+prevexp, data=comp,
CV = F, method="mle")

print(lmod1)

### get the predicted group membership and the posterior
probabilities (CV = TRUE)

lmod2 <- lda(gender~educ+salary+jobtime+prevexp, data=comp,
CV = T, method="mle")

print(lmod2)

### to compute the discriminant function scores

pred <- predict(lmod1)

### get the scores

pred$x

########## compute the Wilks' lambda

### create a matrix with the independent variables
### remove columns 1 and 3 (gender and job)

matr <- as.matrix(comp[c(-1, -3)])

View(matr)
```

```r
### apply manova to the new matrix
### requiring the Wilks test

summary(manova(matr~comp$gender), test = "Wilks")

####### create the classification table (to assess the prediction accuracy)

### create a new data set with the real and expected groups

comp2 <- cbind(comp$gender, lmod2$class)

comp2 <- data.frame(comp2)

View(comp2)

### give names to the variables

colnames(comp2) <- c("original", "predicted")

### build the cross table

require(gmodels)

CrossTable(comp2$original, comp2$predicted, digits = 2,
expected=FALSE, prop.r=TRUE, prop.c=FALSE,
           prop.t=FALSE, prop.chisq=FALSE, chisq = FALSE,
fisher = FALSE, mcnemar = FALSE,
           missing.include=FALSE)
```

Lesson 29 - Multiple discriminant analysis

```r
comp <- read.csv("company.csv")

View(comp)

#########
### how to perform a multiple discriminant analysis
#########

### dependent variable: job category (1 - employees, 2 -
middle managers, 3 - top managers)
```

```
### independent variables: education level (educ), salary,
### months at the current job (jobtime), previous
experience in months (prevexp)

### run the DA using the lda function

require(MASS)

### get the prior probabilities, the group means and
### the coefficients of the discriminant function (CV =
FALSE)

lmod1 <- lda(job~educ+salary+jobtime+prevexp, data=comp, CV
= F, method="mle")

print(lmod1)

### get the classes and the posterior probabilities (CV =
TRUE)

lmod2 <-lda(job~educ+salary+jobtime+prevexp, data=comp, CV
= T, method="mle")

print(lmod2)

### compute the Wilks lambda

matr <- as.matrix(comp[c(-1, -3)])

View(matr)

summary(manova(matr~comp$job), test = "Wilks")

### get the discriminant function scores

pred <- predict(lmod1)

pred$x
```

```
### build the classification table

comp2 <- cbind(comp$job, lmod2$class)

comp2 <- data.frame(comp2)

View(comp2)

colnames(comp2) <- c("original", "predicted")

require(gmodels)

CrossTable(comp2$original, comp2$predicted, digits = 2,
expected=FALSE, prop.r=TRUE, prop.c=FALSE,
           prop.t=FALSE, prop.chisq=FALSE, chisq = FALSE,
fisher = FALSE, mcnemar = FALSE,
           missing.include=FALSE)
```