HUDM6122 Homework_01

Chenguang Pan

Jan 28, 2023

0.1 Exercise 1.1

First, I made a xlsx. version of `Table 1.1` to let R read it directly using the package 'readxl". This table is in 10x7 size. The first column is just the index of each observation, so I drop it here. Finally this dataset is in 9x7 size.

One should notice that the **sex**, **depression**, and **health** are categorical variables. The Pearson Correlation Coefficient is used for continuous rather than categorical variables. Therefore, when calculate the correlation matrix we should drop the categorical ones.

Note, there are some parameters need to be set. Since the original dataset contains missing value, I construct the correlation matrix based on all complete observations.

```
> library(readxl)
> table_11 <- read_excel("table_1.1.xlsx")
> my_data <- table_11[,c(2:7)]
> # drop the discrete vars and use only the complete observations
> my_data_cor <- round(cor(my_data[,c(2,3,6)], use = "complete"),2)
> # the output is rounded in two decimals.
> my_data_cor
         age    IQ weight
age     1.00 -0.15  -0.12
IQ     -0.15  1.00   0.75
weight -0.12  0.75   1.00
```

0.2 Exercise 1.2

Fill the `NA` with the column's mean, and recalculate the correlation matrix.

```
> # to impute the NA with mean using a for-loop
> for (cols in c(2,3,6)) {
+   my_data[,cols][is.na(my_data[,cols])] <- mean(my_data[,cols], na.rm=T)
+ }
> # create the correlation matrix
> my_data_cor_2 <- round(cor(my_data[,c(2,3,6)]),2)
> my_data_cor_2
         age    IQ weight
age     1.00 -0.14  -0.10
IQ     -0.14  1.00   0.52
weight -0.10  0.52   1.00
```

0.3 Exercise 1.3

The authors may not clearly say where can readers find the original dataset. After seeing the code underlying in this book's R package `MVA`, and run this argument `demo("Ch-MVA")`, one can find the `table 1.3`'s dataset information at the paragraph named `code chunk number 5`. It is in another package called `HSAUR2`. First, I draw the normal probability plots of each variable.

```
> # load the Table 1.3's original dataset
> library(HSAUR2)
> dim(pottery)
```

[1] 45 10

```
> names(pottery) # the dataset is the same
```

[1] "Al2O3" "Fe2O3" "MgO" "CaO" "Na2O" "K2O" "TiO2" "MnO" "BaO"
[10] "kiln"

```
> #layout(matrix(1:10, nc = 3))
> sapply(colnames(pottery)[1:9], function(x){
+   # use double bracket can directly return the col
+   qqnorm(pottery[[x]], main = x)
+   qqline(pottery[[x]])
+ })
```

$Al2O3 NULL

$Fe2O3 NULL

$MgO NULL

$CaO NULL

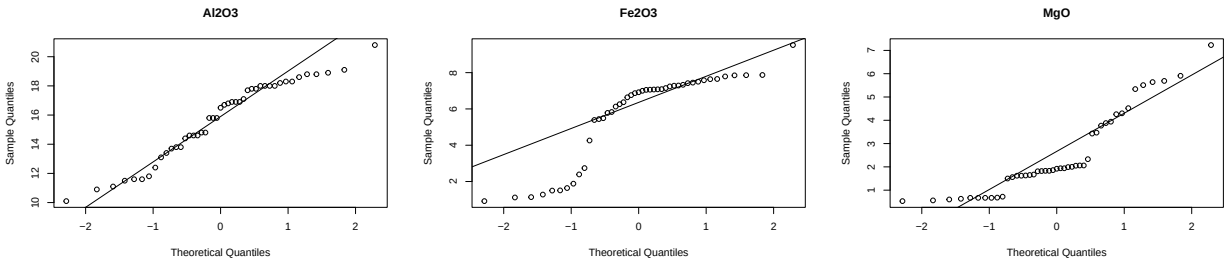
$Na2O NULL

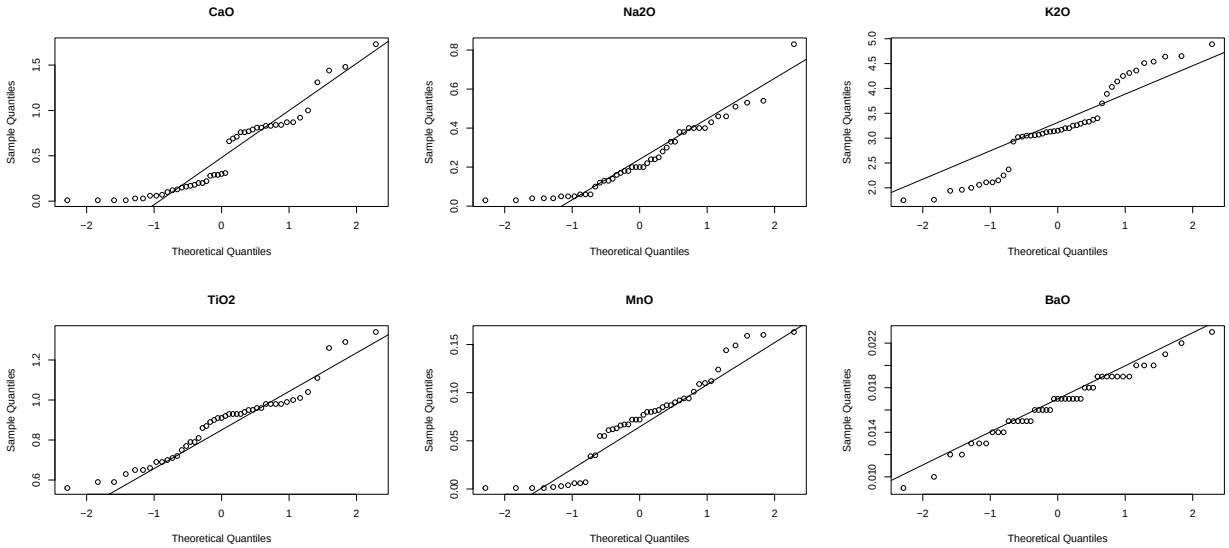
$K2O NULL

$TiO2 NULL

$MnO NULL

$BaO NULL





Second, I draw the chi-square plot of the data as followed.

```
> # to drop the last discrete variable
> X <- pottery[,1:9]
> # get each col's mean
> col_mean <-  colMeans(X)
> # get the cov matrix
> S <- cov(X)
> # solve() function can get inverse matrix directly
> # compute the generalised distance
> d <- apply(X, 1, # array; 1= row 2 = col
+            function(X)
+              t(X - col_mean) %*% solve(S) %*% (X - col_mean))
> plot(qc <- qchisq((1:nrow(X) - 1/2)/ nrow(X), df = 9),
+      sd<- sort(d),
+      xlab = expression(paste(chi[9]^2," Quantile")),
+      ylab = "Odered Distance", xlim = range(qc)*c(1, 1.1))
> oups <- which(rank(abs(qc-sd), ties="random") > nrow(X) - 3)
> text(qc[oups], sd[oups]-1.5, names(oups))
> abline(a=0, b=1)
```