Worksheet 5 Missing Data

We will use the Visualization and Imputation of Missing values package (VIM) package for R that has been developed to deal with missing values in data sets. First we need to install the package

install.packages(“VIM”)

Next we load the packages

library(VIM)

Reference manuals for each package are available on the cran website.

We will use a built in data set to explore these packages. The data() command lists all the data sets available.

data()

Have a quick look at the different data sets, we wish to load the sleep dataset from the VIM package.

data(sleep, package="VIM")

Notice that the data set does not load into the R workspace, it is described as a ‘promise’, which means that R can load in the data set as it is needed but the data is not stored in the working memory.

If you would like to load the data directly into R and view it, use

sleep <- sleep

If a data set is not too large we can locate missing data using:

is.na(sleep$Span)

This function produces a vector of TRUE or FALSE values where TRUE indicates missing data.

Arithmetic functions on missing values produce missing values. The following command creates a vector where the final value is non numeric, NA.

x<-c(4, 8, 2, NA)

mean(x)

[1] NA

The following command excludes the missing value before calculating the mean.

mean(x, na.rm=TRUE)

**If there are only a few missing values and you wish to work with complete cases only then the function complete.cases()** returns a logical vector indicating which cases are complete.

First we create a matrix of data:

x<-c(4,8,2,NA,NA,3,7,5) # this creates a vector of data

x<-matrix(x,4,2)

x

The matrix(x,4,2) command re shaped the vector x into a matrix with four rows and two columns.

To find out which rows are complete cases use:

complete.cases(x)

Then we can get the matrix consisting of complete cases only:

x[complete.cases(x),]

this can also be achieved using na.omit:

x<-c(4,8,2,NA,NA,3,7,5)

x<-matrix(x,4,2)

x<-na.omit(x)

x

Visualising Missing Data

The function aggr aggregates missing data and can be used to count the *amount* of missingness for each variable.

a <- aggr(sleep)

a

Missings in variables:

Variable Count

NonD 14

Dream 12

Sleep 4

Span 4

Gest 4

The aggr function can also plot the *amount* of missing-ness for each variable.

> aggr(sleep)



The barplot on the left shows the proportions of missing values in each of the variables.

The combination plot on the right shows all existing combinations of missing (red) and non-missing (blue) values in the observations. The frequencies of the combinations are visualized by small horizontal bars. For example, there were 42 observations with no missing values, 9 observations with missing values in the variables NonD and Dream, there were 3 observations with missing values in the variable Gest, etc.

The barMiss function creates a barplot for a given variable of interest that highlights missing values in other variables by splitting each bar into two parts (red for missing). Additionally, information about missing values in the variable of interest is shown on the right hand side.

First we define the data frame x (it is defined to contain all the rows of the sleep data (i.e. all the observations) but just two columns, the variables Exp and Sleep.

x <- sleep[, c("Exp", "Sleep")]

barMiss(x)

To create a bar chart showing where the missing values in the Sleep variable occur in relation to the Exp variable use:

barMiss(x)



This bar chart shows us that Sleep has four missing values. One of the missing values occurs when the variable Exp has the value 1, and three of the missing values occur when Exp has the value 5. The number of missing values in the Exp variable is shown by the missing bar on the right hand side (no missing values).

Try this for the variable Span instead of Sleep.

How many missing values are there for the variable Span?

How do the missing variables in Span relate to the variable Exp?

The histMiss() function produces the same information as the barMiss() function but, produces histograms instead of bar graphs.

y <- sleep[, c("Span", "Sleep")]

histMiss(y)



This histogram shows us that sleep has four missing values. Two of the missing values occur when the variable Span lies between10 and 20, and two of the missing values occur when Span lies between 20 and 30.

Why do we use a histogram rather than a bar chart to examine the missing values in the variable Sleep in relation to the variable Span?

Create a histogram showing where the missing values in the Dream variable occur in relation to the Span variable.

Try this for other variables.

The function marginplot() creates a two variable scatter plot with additional information about the missing values in the margins. Boxplots for available and missing data are shown in the margins, as well as univariate scatterplots for missing values.

marginplot(sleep[,c("Span", "Sleep")])

Note that the argument for marginplot is a matrix with two variables and here rather than creating a new variable containing that matrix we put the subset of the sleep data set straight into the marginplot function. We could have used:

y <- sleep[, c("Span", "Sleep")]

marginplot(y)



The red numbers tell us how many missing values there are for each variable (4 for Sleep and 4 for Span) The red number in the corner tells us of many observations are missing values for both observations (0).

The pairs of boxplots show how the missing data (shown in red) from one variable is distributed in relation to the other variable (shown in blue). We see that the missing data for the Span variable all occur for larger values of the Sleep variable (the vertical pair of boxplots). The missing data for the Sleep variable occur when Span is approximately between 15 and 30.

The function marginmatrix creates a scatter plot matrix with information about missing values in the plot margins of each panel. In the margins, box plots in blue represent the (non-missing) data. Single variable scatter plots and boxplots in red represent missing data and are located along the axis for each variable (same as marginplot)

Next we create a margin plot for the first 5 variables of the sleep data set:

data(sleep)

z <- sleep[, 1:5]

First we create a subset of the sleep data, z. The data frame z contains all the rows of the sleep data set but just the first five columns.

z[,c(1,2,3)] <- log10(z[,c(1,2,3)])

[**Aside:** A log10 transform is applied to the first three variables of the matrix z, to see why this was necessary plot a histogram of the first column of the sleep data set. Then plot a histogram of the transformed data]

hist(sleep$BodyWgt)

hist(z$BodyWgt)

To create the margin plot:

windows(10,10)

marginmatrix(z)



Examine the scatterplot matrix and then produce a single marginplot for any combination of variables where you think there might be a pattern between the missing values.

Dealing with Missing Data

If there are very few missing data we can perform analyses on the complete cases only e.g. we can create a linear model:

model<-lm(z$sleep~z$BodyWgt, data=na.omit(z))

summary(model)

Try

model<-lm(z$sleep~z$BodyWgt)

summary(model)

we see that using complete cases only is the default method for the lm function.

Other actions that can be taken for missing data are:

• na.fail() - issue an error if the object contains missing values

• na.exclude() - same as na.omit() but will result in NA predictions for missing values)

There is not always an option for performing analyses using the ‘all available approach’ to dealing with missing data, e.g. the lm function does not have an option for the all available approach.

If we need to impute missing data, then the VIM package can be used to impute data using the irmi() function which stands for, Iterative Robust Model-based Imputation. The function runs iterative regression analysis in which each iteration uses one variable as the response variable and the remaining variables as explanatory variables. If the outcome has any missing values, the predicted values from the regression are imputed. Iterations end when all variables in the data frame have served as an outcome.

We can use the irmi function to create a new sleep data set where all the missing values are imputed.

imputed.sleep <-irmi(sleep)

We get the following output:

Imputation performed on the following data set:

type #missing

BodyWgt "numeric" "0"

BrainWgt "numeric" "0"

NonD "numeric" "14"

Dream "numeric" "12"

Sleep "numeric" "4"

Span "numeric" "4"

Gest "numeric" "4"

Pred "integer" "0"

Exp "integer" "0"

Danger "integer" "0"

Data has been replaced for the NonD, Dream, Sleep, Span and Gest variables.

Let’s compare the new data set where the missing values have been imputed with the old data set that contains missing values:

summary(sleep)

summary(imputed.sleep)

For example the NonD variable summary was originally:

Min. 1st Qu. Median Mean 3rd Qu. Max. NA's

2.100 6.250 8.350 8.673 11.000 17.900 14.000

and is now:

Min. 1st Qu. Median Mean 3rd Qu. Max.

2.100 5.800 8.400 8.429 10.750 17.900

Let’s see how the imputed values affect a simple linear regression model:

model.sleep<-lm(sleep$NonD~sleep$BodyWgt)

summary(model.sleep)

Call:

lm(formula = sleep$NonD ~ sleep$BodyWgt)

Residuals:

Min 1Q Median 3Q Max

-5.5992 -2.6794 -0.5426 2.0016 8.8996

Coefficients:

Estimate Std. Error t value Pr(>|t|)

(Intercept) 9.000419 0.509775 17.656 < 2e-16 \*\*\*

sleep$BodyWgt -0.003658 0.001329 -2.752 0.00845 \*\*

---

Signif. codes: 0 ‘\*\*\*’ 0.001 ‘\*\*’ 0.01 ‘\*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 3.434 on 46 degrees of freedom

(14 observations deleted due to missingness)

Multiple R-squared: 0.1413, Adjusted R-squared: 0.1227

F-statistic: 7.572 on 1 and 46 DF, p-value: 0.008455

model.imputed.sleep<-lm(imputed.sleep$NonD~imputed.sleep$BodyWgt)

summary(model.imputed.sleep)

Call:

lm(formula = imputed.sleep$NonD ~ imputed.sleep$BodyWgt)

Residuals:

Min 1Q Median 3Q Max

-5.9095 -2.5578 -0.1802 2.1917 9.2135

Coefficients:

Estimate Std. Error t value Pr(>|t|)

(Intercept) 8.6865535 0.4491758 19.339 <2e-16 \*\*\*

imputed.sleep$BodyWgt -0.0012995 0.0004916 -2.644 0.0105 \*

---

Signif. codes: 0 ‘\*\*\*’ 0.001 ‘\*\*’ 0.01 ‘\*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 3.452 on 60 degrees of freedom

Multiple R-squared: 0.1043, Adjusted R-squared: 0.08939

F-statistic: 6.988 on 1 and 60 DF, p-value: 0.01045

The Intercept is slightly smaller for the model fitted to the imputed.sleep data set and the slope is reduced by about a third.