R for Complete Beginners
Dolores Romero Morales

The aim of this workshop is to get familiar with R using a brief guide to R (R for Complete Beginners) and a dataset (housing). Both can be found in the course directory. If you want to learn more about the dataset, please go to the well-known UCI Machine Learning repository (http://archive.ics.uci.edu/ml/), and find the housing dataset using the search tool (https://archive.ics.uci.edu/ml/machine-learning-databases/housing/).

If you are comfortable with the material in this guide, you are in a good position from which to learn quickly lots more about R. If not, go through the whole guide step by step first. Once you are ready with the guide, use the *housing* dataset to perform the following steps using R:

Step 1. Read the housing.txt file into a data frame.
Answer:
```
myhousing <- read.table(file.choose(),header=TRUE)
```
Note:
With file.choose() a new window will pop up and you will need to find the file.

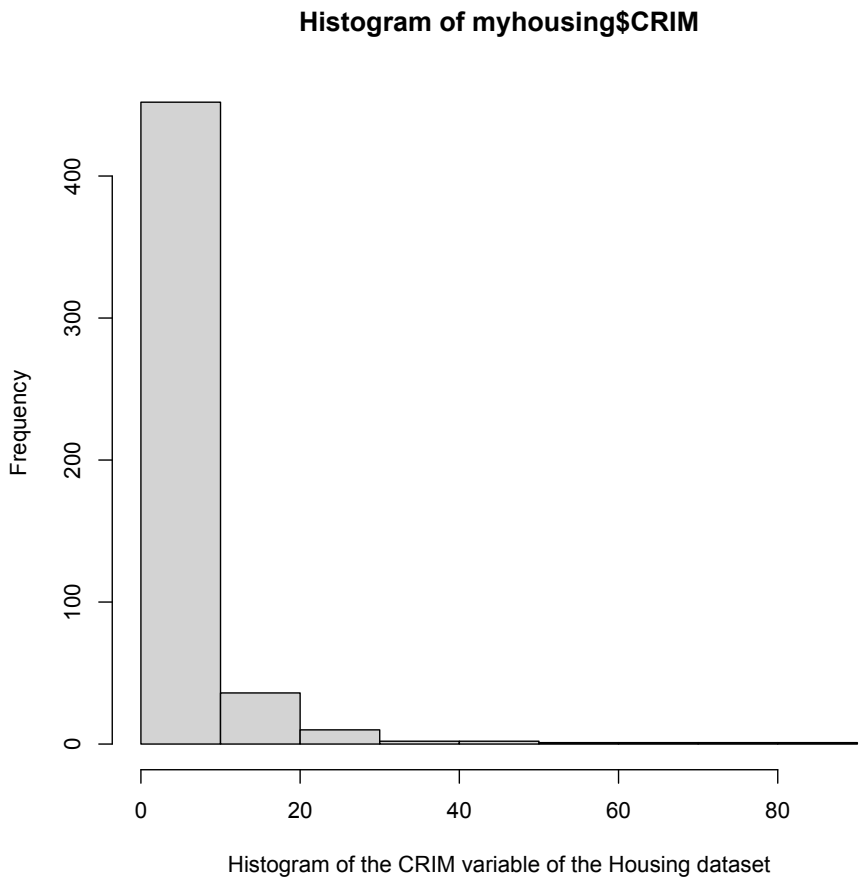
Step 2. Report the number of observations, the number of variables, and the type of data.
Answer:
```
nrow(myhousing)
ncol(myhousing)
str(myhousing)
```
Note1:
With dim(myhousing) you get both nrow(myhousing) and ncol(myhousing).
Note2:
You can see that R is able to distinguish between numeric variables and integer variables.
Note3:
For the binary variables, i.e., variables that can only take on the values 0 and 1, we may want to count how many 1's are there. You can simply use the function sum().

Step 3. Give descriptive statistics for each of the variables (mean, median, variance, standard deviation, quartiles).
Answer:
```
mysummary <- summary(myhousing)
myvar <- apply(myhousing,2,var)
mysd <- apply(myhousing,2,sd)
```

Note1:
The function apply(myhousing,2,var) repeats the function 'var' to each column ('2') in myhousing. The function apply() can be applied to other functions, as with did with 'sd'. There are other functions to perform repeated calculations.
Note2:
When having a lot of variables, it may be worthy transposing mysummary, that will show in a nice way. In order to do so, we can simply use the function t(), t(mysummary).

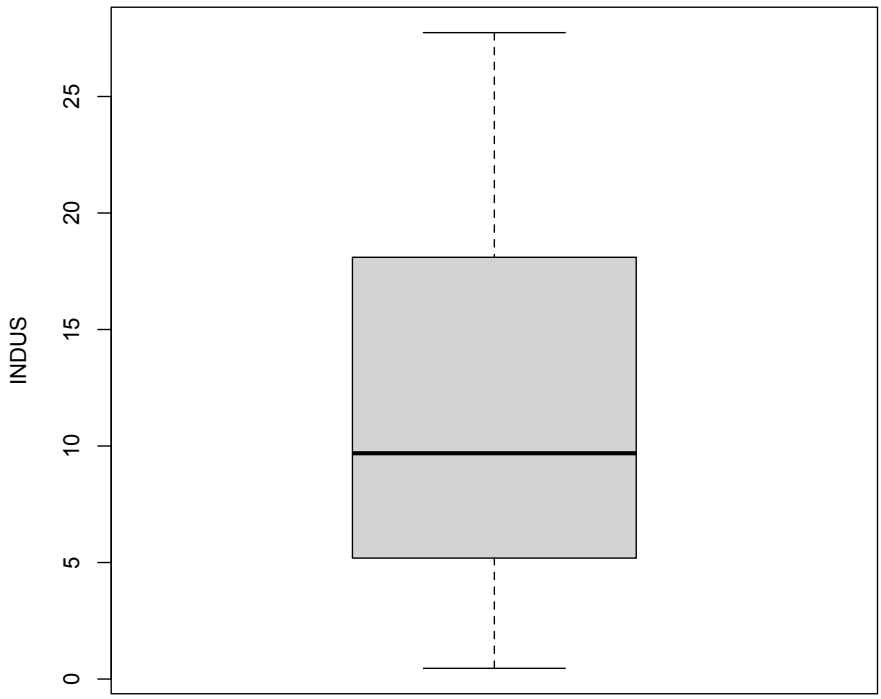
Step 4. Plot the histogram of the first variable and save it to a pdf file.
Answer:
```
hist(myhousing$CRIM)
```

Note:
This function creates a histogram that can be saved as a pdf file (click on the histogram, goto toolbar, click on File, goto Save as).



Step 5. Plot the boxplot of third variable, and save it to a pdf file.
Answer:

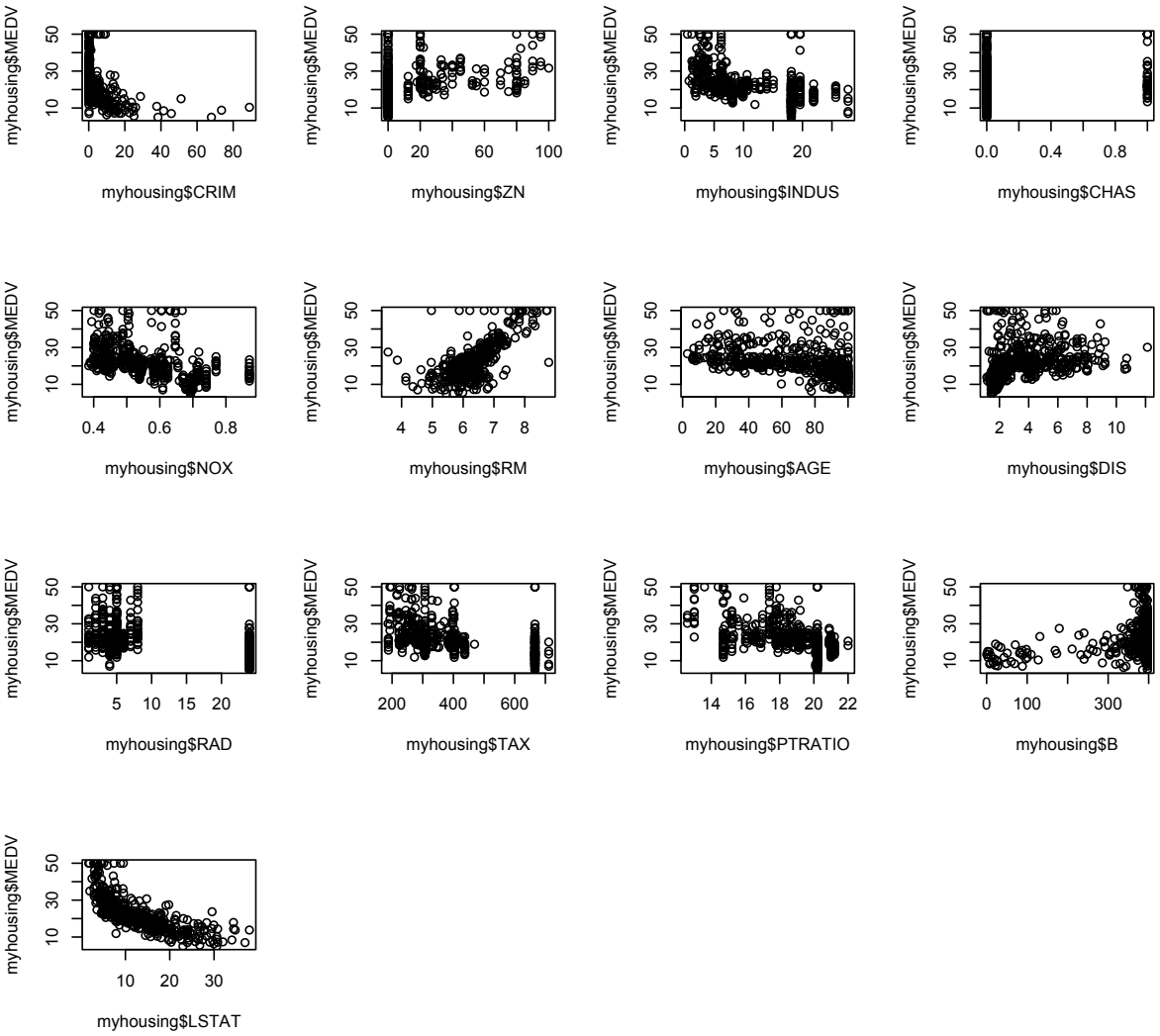
```
boxplot(myhousing$INDUS, ylab ="INDUS", xlab ="Boxplot of the INDUS variable of the Housing dataset")
```



Boxplot of the INDUS variable of the Housing dataset

Step 6. Plot each variable against the last one, and save it to a pdf file. (Why the last one?)
Answer:

```
par(mfrow=c(4,4))
plot(myhousing$CRIM, myhousing$MEDV)
plot(myhousing$ZN, myhousing$MEDV)
plot(myhousing$INDUS, myhousing$MEDV)
plot(myhousing$CHAS, myhousing$MEDV)
plot(myhousing$NOX, myhousing$MEDV)
plot(myhousing$RM, myhousing$MEDV)
plot(myhousing$AGE, myhousing$MEDV)
plot(myhousing$DIS, myhousing$MEDV)
plot(myhousing$RAD, myhousing$MEDV)
plot(myhousing$TAX, myhousing$MEDV)
plot(myhousing$PTRATIO, myhousing$MEDV)
plot(myhousing$B, myhousing$MEDV)
plot(myhousing$LSTAT, myhousing$MEDV)
```



Note1:
The function 'par(mfrow=c(a,b))' places the plots in a axb matrix.
Note2:
plot(myhousing) gives all plots between pairs of variables in the data frame.