Introduction to R

Alex Rodriguez, Jorge Andrade

15 June 2012

Contents

List of Figures

1. Introduction

The data used for this exercise are from a study by the National Institute of Diabetes and Digestive and Kidney Diseases. The study was conducted on 768 adult female Pima Indians. The data was obtained through the 'faraway' R package. This exercise is intended to provide an introduction to preliminary data analysis in R. It will cover the following:

- Use the script provided to load the data into R.
- Initial analysis of the data.
- Cleaning the data.
- Visualizing the data.

2. Loading the Data

We have already downloaded the data and script. R recommends using one directory per analysis. It is helpful for the data and script to be in the directory for the analysis. We can move them or use an absolute path.

- Make a new directory for this analysis
- Find out what R's current working directory is

```
> getwd()
```

- Change R's current working directory to the new directory created for this analysis

```
> setwd("newdir")
```

- We are going to use an absolute path instead of moving the data. Use the source file via:

```
> source("/biodbA/WorkshopDir/R_BioConductor/EXPORT_BIOCONDUCTOR_DEMO.R")
```

The data are now loaded.

3. Initial Analysis

It is useful in all analyses to take a look at the data before using it. Numerical summaries, such as means, standard deviations, maximum and minimum values, correlations, and whatever else is appropriate for the dataset are a key first step. Graphical summaries are equally important. Using these methods, outliers, data-entry errors, and a number of other anomalies can be detected thus making the real analysis better and more accurate.

Using the R script supplied the data are loaded into an R object called a data.frame named 'data'. We can learn an object's class via the class() command:

```
> class(data)
```

Rename the data:

```
> pima<-data
```

3.1. Numerical Summary

It is a good idea to look at what was loaded into the data.frame:

```
> pima
> head(pima)
```

From this, we can see that all our variables were loaded properly.

To create a summary, it is convenient to use the `summary()` function:

```
> summary(pima)
```

Looking at `pregnant`, we see a maximum value of 17. This is large, but not impossible. We see that the minimum values for `glucose`, `diastolic`, `triceps`, `insulin`, and `bmi` are all 0. This is definitely a problem! Look at the sorted values:

```
> sort(pima$diastolic)
```

This shows us that the first 35 values are zero. It seems likely that zero has been used as a missing value code. Using sort on the other four variables leads to the same conclusion. While zero could be a valid value for a given variable, in this case we know the subjects were alive and thus must have had a blood pressure! In order to have an accurate summary, we must clean the data.

3.2. Cleaning the Data

Having 'dirty' data can throw off our analysis, so we must clean the data.

3.2.1. Replace the Zeros with NA

`NA` is the missing value code used by R.

```
> pima$diastolic[pima$diastolic==0] <-NA
> pima$glucose[pima$glucose==0] <-NA
> pima$triceps[pima$triceps==0] <-NA
> pima$insulin[pima$insulin==0] <-NA
> pima$bmi[pima$bmi==0] <-NA
```

If we use the `summary()` command again, things look better

```
> summary(pima)
```

3.2.2. Creating Factors

This dataset has one factor variable, `test`. The script creates a new factor variable called `test.factor` from `test`. If we look at the summary for this variable,

```
> summary(pima$test.factor)
```

We can see that the values are labeled with their appropriate level (positive and negative).

4. Graphical Summary

It is also useful to take a graphical look at the data. This will allow us to see anything we might have missed in the numerical summary and further investigate the anomaly or clean the data.

4.1. Histogram

This is one of the most well-known plots.

```
> hist(pima$diastolic)
```

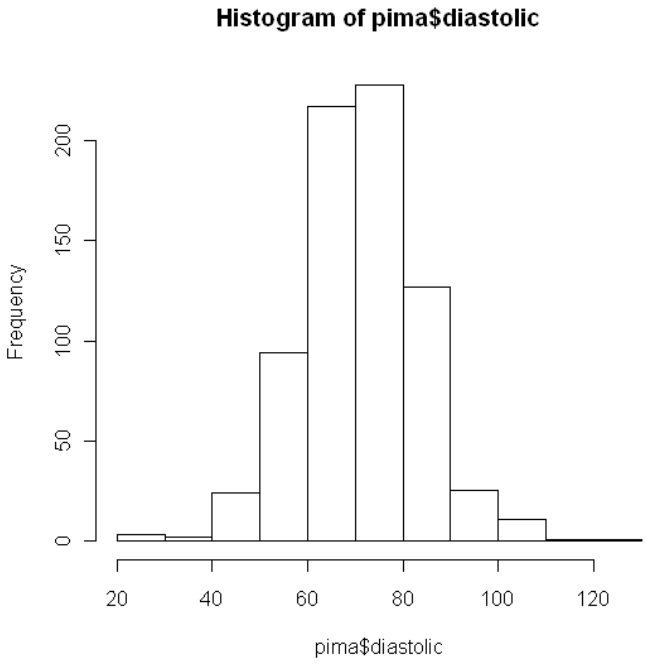

Figure 1 - Histogram of diastolic blood pressure

4.2. Kernel Density Plot

This is essentially a smoothed version of the histogram. Here, we remove the NAs so we do not plot them.

```
> plot(density(pima$diastolic, na.rm=TRUE))
```

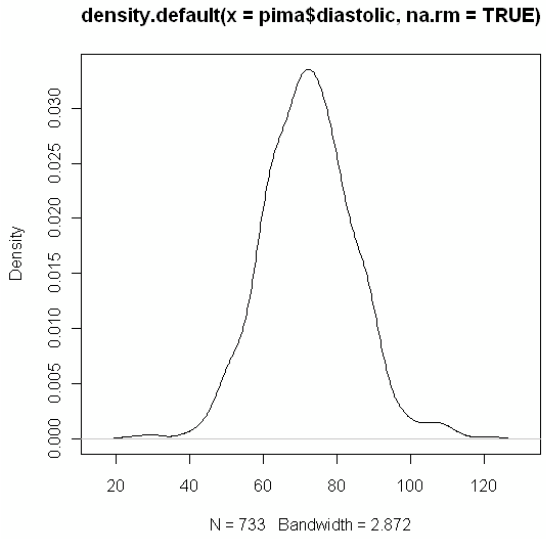

Figure 2 - Kernel density plot of diastolic blood pressure

4.3. Index Plot

We plot the sorted data against its index. This is advantageous because we can see all the cases individually and can see the distribution and possible outliers.

```
> plot(sort(pima$diastolic),pch=".")
```

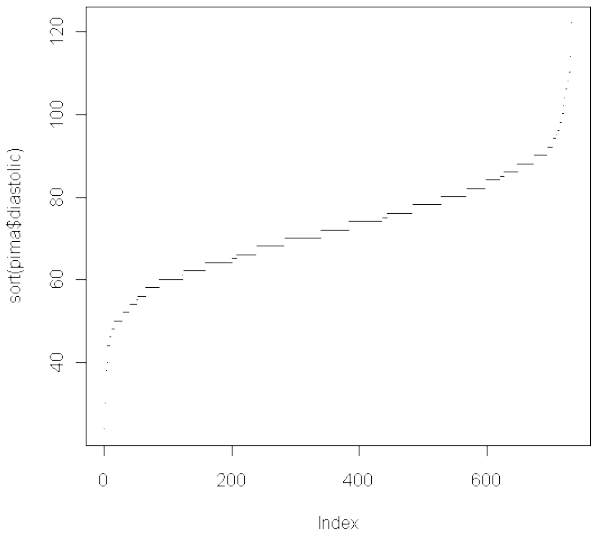

Figure 3 - Index plot of diastolic blood pressure

4.4. Bivariate Plots

If I want a rough idea of how one variable relates to another I can do bivariate plots.

4.4.1. Scatterplot

A good way to compare two quantitative variables is the scatterplot.

```
> plot(diastolic ~ bmi, pima)
```

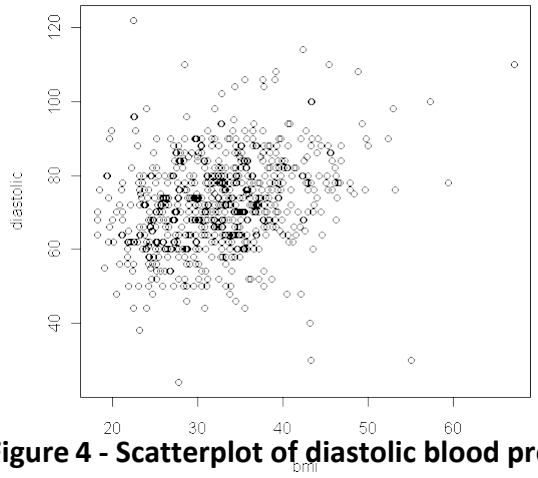

Figure 4 - Scatterplot of diastolic blood pressure vs bmi

4.4.2. Boxplot

A good way to compare a qualitative variable to a quantitative variable is the boxplot.

```
> plot(pima$bmi ~ pima$test.factor)
```

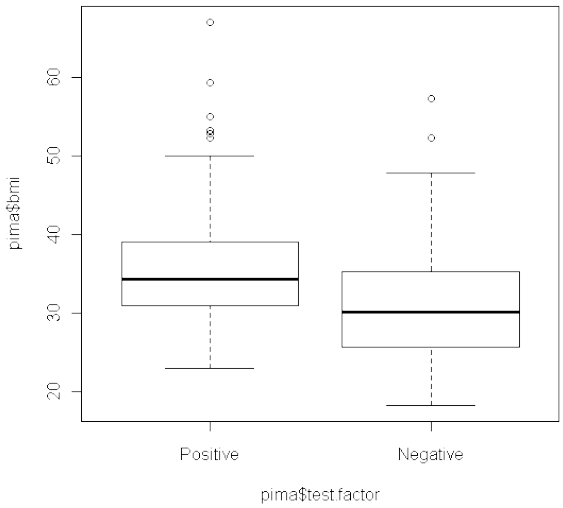

Figure 5 - Boxplot of bmi and test.factor

4.4.3. Scatterplot Matrix

We can also view scatterplots of all the quantitative variables.

```
> pairs(pima)
```

5. HeatMap

Another useful way to view the data is with a heatmap.

```
> pimaMat <- as.matrix(pima[2:9])
> heatmap(pimaMat)
```

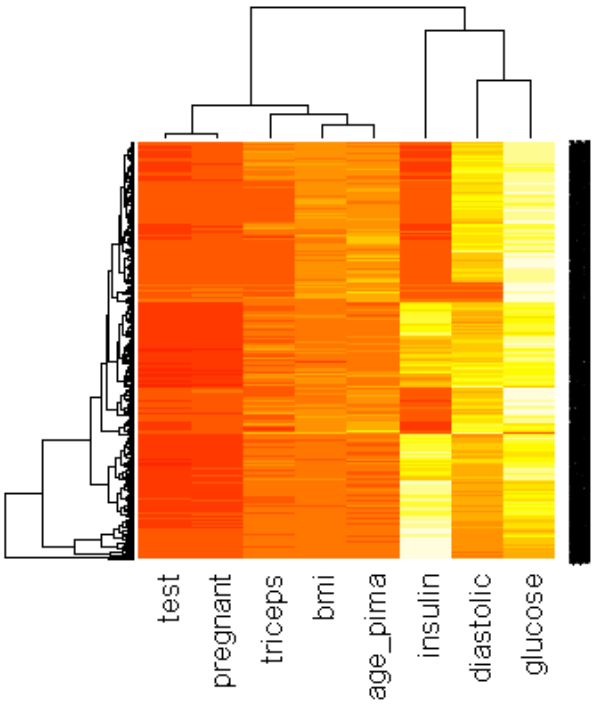

Figure 6 - Heatmap of pima data