exploring EDA

Vince Croft

February 23, 2015

Abstract

Code fragments designed to explain the examples given in the inverted CERN School of Computing in february 2015. Examples given in R.

1 Introduction

1.1 Introduction to R

R is a GNU project widely used in statistical computing and graphics.

1.2 Exploratory Data Analysis

Exploratory Data Analysis is the way we display data to learn more about what it can tell us.

2 What Does Data Look Like?

Here we discus what data looks like and how to make simple plots.

2.1 Stock Tickers, Vectors and Distributions

When most people think of statistics they think of endless streams of numbers. The stock ticker machine printing of line after line of in comprehensible numbers. When we veiw the world of data like this no wonder it looks confusing. Let's take a simple example, people's heights...

```
heights<-c(172,210,200,180,178,145,163,187,160,175,155,168,184,165,171,164,179,173,188,172)
```

From first glance it's pretty tough to see what the average height of the people in this group are. It's pretty tough to see what the range of heights are. It's actaully fairly impossible to obtain anything useful from this plot at all.

In R we have a very useful command called summary() which can be used to get a first feel for the data.

```
summary(heights)
```

We can now use this data to construct one of the easiest to read plots of all time. A pie chart.

```
#first we move our heights into a data frame for easier processing
people<-data.frame(heights=heights)

#now to access the heights we use people$heights
summary(people$heights)

#this tells us that the most of the data lies between the 1st and 3rd
    quartile
#let's use this to register short and tall people
short<-subset(data,heights<164.8)
average<-subset(data,heights>=164.8 & heights < 181.0)
tall<-subset(data,heights>=181.0)

#now we can get some propertie of these sets. e.g. the number in each
    relative to the whole

quantities=c(length(short$heights),length(average$heights),length(tall$heights))
pie(quantities,labels=c(``short'',''average'',''tall''))
```

As you can see; a pie chart, though very easy to read, may only represent summary data. A more descriptive representation is the histogram.

```
hist(people$heights)
```

As we can see from the output a histogram gives a quick and simple way of visualising all of that summary data in one quick glance. The mean and median, range and a good feel for the distribution.

2.2 Generating Data in R

Here we will show how quick and easy it is to generate data in R.

Firstly let's generate some random numbers.

```
uniform_distribution=runif(100,min=-1,max=1)
hist(uniform_distribuion)
mean(uniform_distribution)
```

Here the options we specify are the range (min and max) and the number of random numbers we would like, here I picked 100.

The average of this distribution will be half way between the min and max.

The higher the quantity of numbers we produce the flatter the distribution will become.

The next distribution I want to demonstrate is the gaussian distribution, also known as the normal distribution or the bell curve.

```
norm_distribution=rnorm(100,0,1)
hist(norm_distribution)
mean(norm_distribution)
```

Here once again we see the distinctive bell shape become more visable with increasing numbers. The other options give the mean and the variance.

When we ask R for the values of these though often we won't get back the same mean or variacnce we put in. This is because there my not be enough events in the histogram to get an accurate enough value for this quantity. More events leads to a more accurate mean value.

The last distribtion (since it's kinda obvious how to generate it) is the exponential distribution.

```
exp_dist=rexp(100,1)
hist(exp_dist)
mean(exp_dist)
```

Here we only provide one additional piece of information to the distribution but this is enough due to the memoryless nature of the distribution.

2.3 more on histograms

There is much much more that can be said about histograms which delves deep into the realms of probability. Here I will just outline a few simple R options that can be used in the hist()

R defines bin sizes with a series of breakpoints. This number can be modified using the option, breaks.

```
hist(exp_dist,breaks=100)
\end{lstlistng}

We can also define the range in a similar way by using one of R's built
    in vector function.
\begin{lstlisting}
hist(exp_dist,breaks=10,xlim=c(2,12))
```

3 Distribution Information

So now that we know how to generate and plot (simply) how about the information that we can extract? Well we've already seen the summary command. Let's get some values.

3.1 Mean, Mode and Median

For a true gaussian distribution these three values will be the same. When we look at the probability interpretation, the mean is the value that if this same process produces another value it is most likely to be the mean, however the most common value in the distribution is the mode.

```
norm_dis=rnorm(100,0,1)
mean(norm_dis)
median(norm_dis)

# the mode is slightly more complicated because of restrictions on data
    opperations
# moving the vector into a data frame is I think the best way to
    continue.
norm_hist=hist(norm_dis)
norm_data=data.frame(counts=norm_hist$counts)
mode=max(norm_data$counts)
mode
```

3.2 Variance and Standard Deviation

This again is very simple in R.

```
norm_dis=rnorm(100,0,1)
sd(norm_dis)
var(norm_dis)
```

4 Exploring Two Variables

First lets generate a 2 dimensional distribution. This uses a special libaray.

```
library(MASS)

#for a multivariate distribution we have a 2D mean, and a covariance
    matrix instead of a single mean and variance
mu<-c(0,0)
sigma<-matrix(c(1,0,0,1),nrows=2)
mva_norm=mvrnorm(100,mu,sigma)
```

4.1 Marginal Distribution

A 2 dimensional distribution doesn't only show the information in x and y directions but also the relationship between them. Usually however we're only interested in a single value. So how do we get from a 2 dimensional distribution to the variable that we measure? We sweep all the data to one side and look at how the 2 dimensional distribution looks projected onto that one axis.

```
#make vector for each x and y within the mva_norm data.frame
mva_norm.x<-c(mva_norm[,1])
mva_norm.y<-c(mva_norm[,2])
#note that indicies in r start counting at 1

hist(mva_norm.x)
mean(mva_norm.y)
```

4.2 Covariance and Correlation

Again this seems very simple now but the information contained in a plot can be used for all sorts of very lovely algorithms later on.

```
cov(mva_norm)
cor(mva_norm)
```

5 Transformations

5.1 axis transformation

If we now think of our data as vectors this one becomes trivial. If you know that the first bin has the most data you can transform it to be the biggest

5.2 Rotation

rotations are more complicated. if you can see that two vectors are strongly correlated, you can rotate the axis of the plot to the angle of correlation and histogram the results

```
hist(mva_norm.x*mva_norm.y*cor(mva_norm.x,mva_norm.y)
```