R-Tutorials Using Titanic Data

Josemari Feliciano

7/10/2017

There are four major parts to this tutorials:

1. How to read and extract data from RData file
2. How to create basic tally and graphs
3. How to run basic descriptive statistics
4. How to run and read results of inferential statistics

Tutorial Notes:

The Beauty of R Studio is the ability to have notebooks for data analysis. It allows us to run R-code inline similar to IPython Notebooks.

Think of it as a fancy chemistry notebook where you can somehow run the experiments itself in the notebook – but for data analysis! For this notebook and R tutorial, I will use the titanic data for analysis. This assumes that you have R and RStudio loaded.

Note : The mosaic package is required for this if you want to run the data yourself in your R environment and not to simply view them from my github repo. If you do not have this installed, run this code in your r-console independently:

```
install.packages("ggplot2")
```

R-Tutorials Using Titanic Data

Part 1: Reading data

We probably want to load the titanic data first. We probably want to load the mosaic library out of the way.

Loading Data:

We can accomplish both data loading and library call with the following script:

```
load("Titanic.Rdata")
library("mosaic")
```

Note: Always rerun/replay the code above when entering this file in your local environment. The pre-ran scripts will remain intact but rerunning them might display errors. So you might have to rerun this code eventually. If you see an error, know that this code might be the culprit.

Which Variables Are In Data?

Now that the data has been loaded, we probably want to see which data variables we will deal with! Below, we will use `names()` to print the variable names in our data to have them handy.

```
names(Titanic)
```

```
## [1] "Gender"   "Age"      "Name"     "Fare"     "Class"    "Survived"
```

The `names()` function we just ran displayed the 6 variables within the Titanic Data which include Gender, Age, Name, Fare, Class and Survived.

Of course, you could have looked at the actual CSV file or RData file directly. But functions like `names()` are very useful when wrangling data such as JSON and similar data types.

Exploring Data Types:

The `sapply()` function is very useful for this. The data types do make sense.

```
sapply(Titanic,class)
```

```
##    Gender       Age      Name      Fare     Class  Survived
##  "factor" "numeric"  "factor" "numeric"  "factor"  "factor"
```

The output above does make sense; data types seem to match what we would expect. Age is numeric. And Gender is a 'factor', commonly known as string in other computer langauges.

Another function we could have used is `str()`, but the output can be messy and dense. `str()` does provide more information on the variables in our data.

```
str(Titanic)
```

```
## 'data.frame':    1045 obs. of  6 variables:
##  $ Gender  : Factor w/ 2 levels "Female","Male": 1 2 1 2 1 2 1 2 1 2 ...
##  $ Age     : num  29 1 2 30 25 48 63 39 53 71 ...
##  $ Name    : Factor w/ 1307 levels "Abbing, Mr. Anthony",..: 22 24 25 26 27 31 46 47 51 55 ...
##  $ Fare    : num  211 152 152 152 152 ...
##  $ Class   : Factor w/ 3 levels "Lower","Middle",..: 3 3 3 3 3 3 3 3 3 3 ...
##  $ Survived: Factor w/ 2 levels "No","Yes": 2 2 1 1 1 2 2 1 2 1 ...
```

```
##  - attr(*, "na.action")=Class 'omit'  Named int [1:264] 16 38 41 47 60 70 71 75 81 107 ...
##   .. ..- attr(*, "names")= chr [1:264] "16" "38" "41" "47" ...
```

As we can see, more information are provided. For instance, the output shows that there are 2 levels for Gender, "Famale" and "Male". But `str()` can look a bit messy.

Note: We could have used `View(Titanic)` to open the spreadsheet in a difference pane or tab in RStudio. But that will require sifting through the spreadsheet. It's good to familiarize with both `str()` and `sapply()` functions so we don't have leave our tab.

Basic Data Tallying

There are multiple ways of tallying variables.

Using count format of `tally()`:

```
tally(~Gender, format = "count", data = Titanic)
```

```
## Gender
## Female   Male
##    388    657
```

As you can see from above output, the count format will give us the raw number of count. For instance, 388 passengers from the Titanic are female.

Using proportion format of `tally()`:

```
tally(~Class, format = "proportion", data = Titanic)
```

```
## Class
##     Lower    Middle     Upper
## 0.4784689 0.2497608 0.2717703
```

Another format that statisticians use is the proportion format of `tally()`. For instance, the Lower class output implies that roughly 47.85% of Titanic passengers bought the low-class tickets.

Using percent format of `tally()`:

Most of Statistics will deal with proportion format like those displayed in the previous section. So the percent format is really unnecessary. But if you will not use any other statistical test, this might be handy to use.

```
tally(~Survived, format = "percent", data = Titanic)
```

```
## Survived
##       No      Yes
## 59.13876 40.86124
```

Just a quick note: had we ran the proportion format, the output would have displayed 0.5914 for No.

The output above clearly shows that roughly 40.86% of Titanic passengers survived the infamous incident.

Comparing Genders: