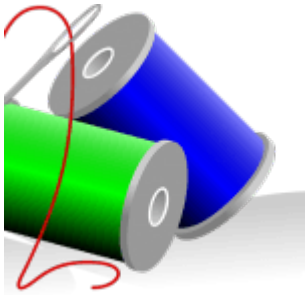
Character Functions (Advanced)



This set of exercises will help you to help you improve your skills with character functions in R. Most of the exercises are related with text mining, a statistical technique that analyses text using statistics. If you find them interesting I would suggest checking the library tm, this includes functions designed for this task. There are many applications of text mining, a pretty popular one is the ability to associate a text with his or her author, this was how J.K.Rowling (Harry potter author) was caught publishing a new novel series under an alias. Before proceeding, it might be helpful to look over the help pages for the nchar, tolower, toupper, grep, sub and strsplit. Take at the library stringr and the functions it includes such as str_sub.

Answers to the exercises are available here.

If you obtained a different (correct) answer than those listed on the solutions page, please feel free to post your answer as a comment on that page.

Before starting the set of exercises run the following code lines :

```
if (!'tm' %in% installed.packages()) install.packages('tm')
library(tm)
txt = system.file("texts", "txt", package = "tm")
ovid = VCorpus(DirSource(txt, encoding = "UTF-8"),
readerControl = list(language = "lat"))
OVID = c(data.frame(text=unlist(TEXT), stringsAsFactors = F))
TEXT = lapply(ovid[1:5], as.character)
TEXT1 = TEXT[[4]]
```

Exercise 1

Delete all the punctuation marks from TEXT1

Exercise 2

How many letters does TEXT1 contains?

Exercise 3

How many words does TEXT1 contains?

Exercise 4

What is the most common word in TEXT1?

Learn more about Text analysis in the online course Text Analytics/Text Mining Using R. In this course you will learn how create, analyse and finally visualize your text based data source. Having all the steps easily outlined will be a great reference source for future work.

Exercise 5

Get an object that contains all the words with at least one capital letter (Make sure the object contains each word only once)

Exercise 6

Which are the 5 most common letter in the object OVID?

Exercise 7

Which letters from the alphabet are not in the object OVID

Exercise 8

On the OVID object, there is a character from the popular sitcom 'FRIENDS' , Who is he/she? There were six main characters (Chandler, Phoebe, Ross, Monica, Joey, Rachel)

Exercise 9

Find the line where this character is mentioned

Exercise 10

How many words finish with a vowel, how many with a consonant?

Unit testing in R using testthat library Exercises

testthat is a testing framework developed by Hadley Wickham, which makes unit testing easy for developers.

Test scripts developed can be re-run after debugging or making changes to the functions without the hassle of developing the code for testing again.

testthat has a heirarchical structure made up of expectations, tests and contexts.
Visit this link to know more.

You should be familiar with creation of functions in R to know how this testing framework works.

Answers to the exercises are available here.

Exercise 1

Install and load the package **testthat** using the appropriate function.

Exercise 2

expect_that() is the function that makes the binary assertion of whether or not the value is as expected.
expect_that(x,equals(y)) reads as “it is expected that ‘a’ will be equal to ‘b'”.

Use this function to see if 5*2 equals 10

Learn more about Hadley Wickhams usefull packages in the online course R Data Pre-Processing & Data Management – Shape your Data!. In this course you will learn how to work with:

- tidyr, cleaning your data
- dplyr, shape your data
- And much more

Exercise 3

The function equals() checks for equality with a numerical tolerance. Let’s see what that tolerance level is

Use appropriate function to see if 5*2 equals 10 + (1e-7).
Does the test fail?
If no, change the value to 1e-6 and see what happens.

Exercise 4

To exactly match the values is_identical_to() can be used instead of equals()

Using the appropriate function, check if 2*2 is identical to 4 + (1e-8)

Please check the documentation of this package to learn more about the available functions.

Exercise 5

Let us create a function (m) to multiply two numbers (two arguments) and check if it throws an error with character input arguments.

Check if m(“2″,”3”) throws an error “non-numeric argument to binary operator”

Exercise 6

Now that we know how to check for expectations, let us create tests.
Test is a collection of expectations, where these expectations test a single item of the functionality of a process.

test_that() is the function that encapsulates the description and the code to test each expectation.

The first argument is the description and the second argument is a collection of expectations.

Create a test for function ‘m’ with description “Testing multiplication function” and add a few scenarios to it.
1. Check if m(2,3) equals 6
2. Check if m(2,c(2,3)) equals c(4,6)
3. Check if m(2,”3″) throws an error “non-numeric argument to binary operator”

Exercise 7

The User can write his own expectation using the expect() function. This expectation should compare the input value and the expectation and report the result.
The syntax to write one is as below.

```
custom_expectation <- function() {function(x) {expectation(condition, "Failure message")}}
```

Now, write an expectation is_greater_10() to check if a number

is greater than 10

Exercise 8

Use the expectation defined above to check if 9 is greater than 10.

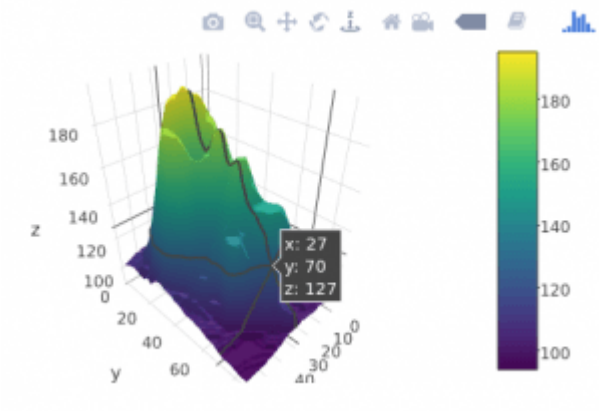
Exercise 9

tests can be put together in a file and run at once. Write tests of your choice and save them in a file.
Use the function test_file() to run all the tests in the file.

Exercise 10

Test files in a directory can be run at once using the function test_dir().
Create multiple test files and save them in a directory. Run all the tests at once using the function.

Plotly : Advanced plots and features



Plotly is a d3 based graphing library used to produce interactive and high quality graphs in R. In the following exercises, we will look at some advanced plots and features avaialable in the package. Please note that the list here is not exhaustive.
We will use datasets available in base R packages.

You are expected to have the knowledge of generating basic plots using plotly package before attempting this exercise set. It is recommended that you go through this exercise set to test your knowledge on plotly basics.

Answers to the exercises are available here.

Refer to this link for more help on the plotly functions.
For a quick info on the functions and arguments required to build basic plotly plots, refer to this cheat sheet.

Learn more about Plotly in Section 17 *Interactive Visualizations with Plotly* of the online course Data Science and Machine Learning Bootcamp with R.

Exercise 1

Install and load the latest version of plotly package.

Generate separate histograms for the first four columns of iris dataset and save the plots in objects p1, p2, p3 and p4.
HINT: Use plot_ly() function with x argument and type="histogram". Use name argument to give appropriate name for the trace.

Exercise 2

a. Use subplot() function to generate a plot with all the plot objects from previous exercise as the arguments.

b. Use the nrows argument to arrange 2 plots per row.

Exercise 3

a. Generate a scatter plot for the iris dataset with first column on the x-axis and second column on the y-axis. Save the plot object.

b. Generate a 2d histogram using the add_histogram2d() function.Save the plot object.

HINT: Use the function plot_ly() with the same x and y arguments and pass the plot object to the 2-d histogram function.

Exercise 4

Generate a subplot with the scatter plot and the 2-d histogram created in the previous exercise.

Notice how the scatter plot can be represented in a more interesting way. Cells in the 2-d histogram are binned and represented with the color on the scale based on the cell population/density.

Exercise 5

Set the value of shareX and shareY arguments in the subplot() function to scale the plots to the same range of x and y.

Exercise 6

Now, let us build a 3-d surface plot. The syntax to build such plot is as below.
`plot_ly(z = matrix(1:100, nrow = 10)) %>% add_surface()`
Click, hold and drag the cursor to see the plot surface.

Build a 3-d surface plot using the volcano dataset available in the base R distribution.

Exercise 7

Let’s look at few helpful and commonly used arguments from the layout() function.

Create and save a scatter plot object with first and second columns of iris dataset as x and y arguments respectively. Colour the markers based on the species

Exercise 8

a. Add an appropriate title to the plot using the layout

function and title argument.
b. Add an appropriate x-axis label using the xaxis argument. xaxis takes a list of attribute values. Refer to the R reference page for more help.
c. Add an appropriate y-axis label.

Exercise 9

a. Use the range attribute in the list of values given to the xaxis argument to set the x-axis range from 1 to 10.
b. Similarly, set the y-axis range from 1 to 5.

Exercise 10

Try different layout options to further customize the font, axes etc… of the plot.

Descriptive Analytics-Part 6: Interactive dashboard (2/2)

Descriptive Analytics is the examination of data or content, usually manually performed, to answer the question "What happened?".As this series of exercises comes to an end, the last part is going to be the development of a data product. Not everybody is able to code in R, so it is useful to be able to make GUIs in order to share your work with non-technical people. This part may be a little challenging, since it requires some basic knowledge of the shiny package. The outcome of this set of exercises will be almost like this web

app (some variables are missing because I had to reduce the size of the data set).

In order to be able to solve this set of exercises you should have solved the part 0, part 1, part 2,part 3, and part 4 of this series but also you should run this script which contain some more data cleaning. In case you haven't, run this script in your machine which contains the lines of code we used to modify our data set. This is the tenth set of exercise of a series of exercises that aims to provide a descriptive analytics solution to the '2008' data set from here. This data set which contains the arrival and departure information for all domestic flights in the US from 2008 has become the "iris" data set for Big Data. The goal of Descriptive analytics is to inform the user about what is going on at the dataset. Before proceeding, it might be helpful to look over the help pages for the fluidPage, pageWithSidebar, headerPanel , sidebarPanel, selectInput, mainPanel, tabPanel, observe, verbatimTextOutput, renderPrint, shinyApp.

For this set of exercises you will need to install and load the package shiny.

```
install.packages('shiny')
library(shiny)
```

I have also changed the values of the DaysOfWeek variable, if you wish to do that as well the code for that is :

```
install.packages('lubridate')
library(lubridate)
flights$DayOfWeek <-
wday(as.Date(flights$Full1_Date,'%m/%d/%Y'), label=TRUE)
```

Because the app requires some time to run, I have also removed the rows with missing values from the data set just to save some time.

```
flights <-flights[which(!is.na(flights['WeatherDelay'])),]
flights <-flights[which(!is.na(flights['ArrDelay'])),]
```

Answers to the exercises are available here.

If you obtained a different (correct) answer than those listed on the solutions page, please feel free to post your answer as a comment on that page. Moreover it would be really nice of you to share the links of the apps you have developed. It would be a great contribution the community.

Learn more about Shiny in the online course R Shiny Interactive Web Apps – Next Level Data Visualization. In this course you will learn how to create advanced Shiny web apps; embed video, pdfs and images; add focus and zooming tools; and many other functionalities (30 lectures, 3hrs.).

Exercise 1

Create the user interface and set as the header of the web app : “Descriptive Analysis”

Exercise 2

Create a side panel.

Exercise 3

Create two select list input control. The former will contain the variables: CarrierDelay, WeatherDelay, NASDelay, SecurityDelay, LateAircraftDelay. The latter will contain the variables :Dest, Origin, UniqueCarrier, TailNum, CancellationCode.

Exercise 4

Create a set of radio buttons used to select a plot from a list (Histogram, Scatter plot, Violin plot),and set as default plot the Histogram.

Exercise 5

Create a set of radio buttons used to select a plot from a

list (bar plot, pie chart, rose wind),and set as default plot the bar plot.

Exercise 6

Create a main panel.

Exercise 7

Create in the main panel two tabs named “Delays” and “Categorical” that will contain the plots of the exercises 4 and 5 respectively.

Exercise 8

Now that we are done with the user interface, create the server side of the app. Create the output of the first tab, which will be the plots from exercise 4 in respect to the first set of variables from exercise 3 (notice that they are all continuous variables), bear in mind that at the scatter plot the x-axis should be the Full_Date and at the violin plot the x-axis should be the DayOfWeek as we did at the previous set of exercises. (please check out the first tab of the app, to make things more clear).

Exercise 9

Create the output of the second tab, , which will be the plots from exercise 5 in respect to the second set of variables from exercise 3 from the exercise 5, use the knowledge you applied (or acquired at the previous exercises for the plots, make them as interesting as you can).(please check out the second tab of the app, to make things more clear).

Exercise 10

Launch the app.

Descriptive Analytics-Part 6: Interactive dashboard (1/2)

Descriptive Analytics is the examination of data or content, usually manually performed, to answer the question "What happened?".As this series of exercises comes to an end, the last part is going to be the development of a data product. Not everybody is able to code in R, so it is useful to be able to make GUIs in order to share your work with non-technical people. This part may be a little challenging, since it requires some basic knowledge of the shiny package. The outcome of this set of exercises will be almost like this web app (some variables are missing because I had to reduce the size of the data set).

In order to be able to solve this set of exercises you should have solved the part 0, part 1, part 2,part 3, and part 4 of this series but also you should run this script which contain some more data cleaning. In case you haven't, run this script in your machine which contains the lines of code we used to modify our data set. This is the ninth set of exercise of a series of exercises that aims to provide a descriptive analytics solution to the '2008' data set from here. This data set which contains the arrival and departure information for all domestic flights in the US from 2008 has become the "iris" data set for Big Data. The goal of Descriptive analytics is to inform the user about what is going on at the dataset. Before proceeding, it might be helpful to look over the help pages for the fluidPage, pageWithSidebar, headerPanel ,

sidebarPanel, selectInput, mainPanel, tabPanel, verbatimTextOutput, renderPrint, shinyApp.

For this set of exercises you will need to install and load the package shiny.

```
install.packages('shiny')
library(shiny)
```

I have also changed the values of the DaysOfWeek variable, if you wish to do that as well the code for that is :

```
install.packages('lubridate')
library(lubridate)
flights$DayOfWeek <-
wday(as.Date(flights$Full1_Date,'%m/%d/%Y'), label=TRUE)
```

Because the app requires some time to run, I have also removed the rows with missing values from the data set just to save some time.

```
flights <-flights[which(!is.na(flights['WeatherDelay'])),]
flights <-flights[which(!is.na(flights['ArrDelay'])),]
```

Answers to the exercises are available here.

If you obtained a different (correct) answer than those listed on the solutions page, please feel free to post your answer as a comment on that page.

Exercise 1

Create the user interface and set as the header of the web app : “Descriptive Analysis”

Exercise 2

Create a side panel.

Exercise 3

Create a select list input control that contains the functions : summary, str, head, tail, names, summary.

Exercise 4

Create a select list input control that contains the functions : mean, median, max, min, range, sd.

Exercise 5

Create a select list input control that contains the variables : ActualElapsedTime, CRSElapsedTime, AirTime, ArrDelay, DepDelay, TaxiIn, TaxiOut.

Exercise 6

Create a main panel.

Exercise 7

Create in the main panel two tabs named “Content” and “Measures” that will contain the output of the functions of exercise 3 and exercise 4 respectively.

Exercise 8

Now that we are done with the user interface, create the server side of the app and the output that is supposed to print the functions of the exercise 3. (please check out the first tab of the app, to make things more clear).

Exercise 9

Create the output of the second tab, combining the functions of exercise 4 and the variables from the exercise 5.(please check out the second tab of the app, to make things more

clear).

Exercise 10

Launch the app.

Descriptive Analytics-Part 5: Data Visualisation (Spatial data)

Descriptive Analytics is the examination of data or content, usually manually performed, to answer the question "What happened?".

In order to be able to solve this set of exercises you should have solved the part 0, part 1, part 2,part 3, and part 4 of this series but also you should run this script which contain some more data cleaning. In case you haven't, run this script in your machine which contains the lines of code we used to modify our data set. This is the eighth set of exercise of a series of exercises that aims to provide a descriptive analytics solution to the '2008' data set from here. This data set which contains the arrival and departure information for all domestic flights in the US from 2008 has become the "iris" data set for Big Data.In order to solve this set of exercises, you have to download this data set which provide us the coordinates of each airport.Please find the script used to create a merged dataset here . I don't expect you to do the

pre-processing yourself since it is beyond the scope of this set but I highly encourage you to give it a try, in case you did that with a better or more efficient way than I did, please post your solution at the comment section(it will be highly appreciated). Moreover we will remove the rows with missing values (various delays) because the methods that we will use are computationally expensive so having a big data set is just a waste of time. The goal of Descriptive analytics is to inform the user about what is going on at the dataset. A great way to do that fast and effectively is by performing data visualisation. Data visualisation is also a form of art, it has to be simple, comprehended and full of information. On this set of exercises we will explore different ways of visualising spatial using the famous ggmap package. Before proceeding, it might be helpful to look over the help pages for the get_map, ggmap, facet_wrap.

For this set of exercises you will need to install and load the packages ggplot2, dplyr, and ggmap.

```
install.packages('ggplot2')
library(ggplot2)
install.packages('dplyr')
library(dplyr)
install.packages('ggmap')
library(ggmap)
```

I have also changed the values of the DaysOfWeek variable, if you wish to do that as well the code for that is :

```
install.packages('lubridate')
library(lubridate)
flights$DayOfWeek <-
wday(as.Date(flights$Full1_Date,'%m/%d/%Y'), label=TRUE)
```

Answers to the exercises are available here.

If you obtained a different (correct) answer than those listed on the solutions page, please feel free to post your answer as

a comment on that page.

Exercise 1

Query the map of United States using the get_map function.
It is recommended to experiment with the various types of maps and select the one that you think is the best. (I have used the toner-lite from Stamen Maps.)

Exercise 2

Print the map that you have selected.

Exercise 3

Modify the printed map in order to print out a bigger image(extent) and assign it to a m object.

Exercise 4

Plot the destination airports of the flights on the map.

Exercise 5

Plot the destination airports of the flights on the map, the size of the points should be based on the number of flights that arrived to the destination airports.

Exercise 6

Plot the destination airports of the flights on the map, the colour of the points should be based on the number of flights that arrived to the destination airport. Make it a bit prettier, use the scale_colour_gradient and set the lows and the highs of your preferences.

Exercise 7

Plot the destination airports of the flights on the map, the colour of the points should be based on the number of flights that arrived to the destination airport and the size of the

points should be based on the total delay of arrival of the flights that arrived at the destination airport.
Something is not right here, right?

Exercise 8

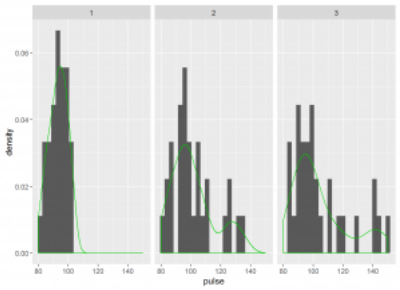
Plot the destination airports of the flights on the map, the colour of the points should be based on the number of flights that arrived to the destination airport and the size of the points should be based on the total delay of arrival divided by the number of flights per destination.

Exercise 9

Plot the destination airports for everyday of the week (hint : facet_wrap)

Exercise 10
Plot the destination airports of the flights on the map, the colour of the points should be based on the number of flights that arrived to the destination airports, the size of the points should be based on the total delay of arrival of the flights that arrived at the destination airport for everyday of the week.
(This may be a bit more challenging , if you can't solve it go to the solutions and try to understand the reason I did what I did, if you have any questions please post them at the comment section).

Repeated measures ANOVA in R Exercises



One way, two way and n way ANOVA are used to test difference in means when we have one, two and n factor variables. A key assumption when performing these ANOVAs is that the measurements are independent. When we have repeated measures this assumption is violated, so we have to use repeated measures ANOVA. Repeated measures designs occur often in longitudinal studies where we are interested in understanding change over time. For example a medical researcher would be interested in assessing the level of depression before and after a surgery procedure. Repeated measures designs are not limited to longitudinal studies, they can also be used when you have an important variable you would like to repeat measures. For example in a fitness experiment you can repeat your measures at different intensity levels. Repeated measures ANOVA can be considered an extension of the paired t test.

Before diving deeper into repeated measures ANOVA you need to understand terminology used. A ***subject*** is a member of the sample under consideration. In our medical study introduced earlier an individual patient is a subject. The ***within-subjects factor*** is the variable that identifies how the dependent variable has been repeatedly measured. In our medical study we would measure depression 4 weeks before surgery, 4 weeks after surgery and 8 weeks after surgery. The different conditions when repeated measurements are made are referred to as trials. A ***between-subjects factor*** identifies independent groups in the study. For example if we had two different procedures this would be the between subjects factor. These conditions are referred to as groups. Repeated measures analysis requires balance in between-subjects factor. For example subjects in each of surgery procedures need to be

equal.

With a repeated measures design we are able to test the following hypotheses.

1. There is no within-subjects main effect
2. There is no between-subjects main effect
3. There is no between subjects interaction effect
4. There is no within subject by between subject interaction effect

There are two assumptions that need to be satisfied when using repeated measures.

1. The dependent variable is normally distributed in each level of the within-subjects factor. Repeated measures analysis is robust to violations of normality with a large sample size which is considered at least 30 subjects. However the accuracy of p values is questionable when the distribution is heavily skewed or thick tailed.
2. The variance across the within subject factor is equal. This is the sphericity assumption. Repeated measures analysis is not robust to this assumption so when there is a violation power decreases and a corresponding increase in probability of a type II error occurs. A Mauchly's test assesses the null hypothesis variance is equal. The sphericity assumption is only relevant when there are more than 2 levels of the within subjects factor.

When the sphericity assumption is violated we make corrections by adjusting the degrees of freedom. Corrections available are Greenhouse-Geisser, Huynh-Feldt and Lower bound. To make a decision on appropriate correction we use a Greenhouse-Geisser estimate of sphericity (ξ). When $\xi < 0.75$ or we do not know anything about sphericity the Greenhouse-Geisser is the appropriate correction. When $\xi > 0.75$ Huynh-Feldt is the

appropriate correction.

For this exercise we will use data on pulse rate exer. People were randomized to two diets, three exercise types and pulse was measured at three different time points. For this data time points is the within-subjects factor. The between-subjects factors are diet and exercise type

The solutions to the exercises below can be found here

Exercise 1

Load the data and inspect its structure

Exercise 2

Check for missing values

Exercise 3

Check for balance in between-subjects factor

Exercise 4

Generate descriptive statistics for the sex variable which is a between subjects factor

Exercise 5

Generate descriptive statistics for the treatment level variable which is a between subjects factor

Exercise 6

Generate descriptive statistics for the weeks variable which is the within subjects factor

Exercise 7

Use histograms to assess distribution across within subjects factor.

Exercise 8

Perform a repeated measures analysis with only the within subjects factor

Exercise 9

Perform a repeated measures analysis with the within subjects factor and one between subjects factor

Exercise 10

Perform a repeated measures analysis with the within subjects factor and two between subjects factors