Computer lab 5

Måns Magnusson

September 12, 2016

Instructions

- This lab should be conducted by students **two by two**.
- The lab constists of writing a package that is version controlled on github.com.
- Both student should **contribute equally much** to the package.
- Commit continously your addition and changes.
- Collaborations should be done using github (ie you should commit using your own github account).
- In the lab some functions can be marked with an *. These parts is only mandatory for students taking the advanced course or students working together in groups of three.
- The deadline for the lab can be found on the **webpage**
- The lab should be turned in as a url to the repository containing the package on github using **LISAM**. This should also include name, github user names and liuid of the students behind the project.

Contents

Chapter 1

A package connecting to a web API

In this lab we will will create a package to connect to a web API using R. The purpose of this lab is to start the work with a API package you will use and develop further during the course. You can choose a connection to any API you find to have interesting data to analyze as well (you may use this data in a machine learning project in the last lab). If the project is good enough you will be able to submit this package to CRAN during the last week.

The API needs to be accepted by the teacher before you start. In **this document** you can find API:s suggestions to implement as an R package.

Master students will also implement a simple Shiny application using the package to connect to the API.

1.1 Create a package

Start out by creating a new package on github see lab 3 **here** for details on how to setup a package. Think about the naming of your package.

Read about the package and discuss whcih functions you should have to access the data in the API. There are often limits and configurations for a given API you need to access.

In genreal it is good to have as few functions as possible theat are exported in the package. But behind these functions you may need to have more functionality not accessed by the users of your package.

1.1.1 Write the R code and unit tests

Implement functions to connect to the API and download data to the R session. Do also write a unit test suite for your public functions in the package. Examples of test can be:

1. Downloading a big query
2. Testing the limits of the API configuration
3. Testing the inputs and outputs of your functions
4. Testing that the function allways return a given value for a specific set of inputs.

See chapter "Testing" in [**?**] or [**?**] for details.

1.1.2 Documentation and vignettes

Write a vignette that describes your package and how to use it with examples. For more information on how to write a vignette and include them in your package see "Vignettes" in [**?**].

Document all public functions using `roxygen2`. See chapter "Object documentation" in [**?**] for details.

1.2 * Create a Shiny application using your package

Create a new repository on github for a Shiny application. Create a simple Shiny application that makes it possible to interactively analyze data using your API package. At least one interactive widget should

be used. See "Lesson 6" in [**?**] for information on how to handle the situation of reactive programming in Shiny when "slow" data is used.

It should be possible to run your shiny app out using the `runGitHub()` function.

1.3 Seminar and examination

During the seminar you will bring your own computer and demonstrate your package and what you found difficult in the project.

We will present as many packages as possible during the seminar and you should

1. Show that the package can be built using R Studio and that all unit tests is passing.
2. Discuss why you implemented the functions as you did.
3. Give a short presentation of the testsuite.
4. Show your vignette/run the examples live.
5. * Present your Shiny application.

1.3.1 Examination

Turn in a the adress to your github repo with the package (and Shiny application) using LISAM.

The packages will be assigned to other groups to try your package out and return eventual problems as bugs using the issue tracker. The teacher will the decide if there are any bugs or corrections that is needed to correct to get the lab to pass.

Bibliography