BIG DATA, MACHINE LEARNING, AND THEIR REAL WORLD APPLICATIONS

Jonathan Wayne Korn

2021-04-15

Prepare the Tools

Follow the guidelines to prepare your local machine for the course and future data science work:

---

(1) Clone Course Github

To download course resources locate them on our course GitHub, please clone the repo and perform the following steps from cu-hsp-learning:

---

(2) Install R <=3.5 and Rstudio.

To Install R<=3.5 follow the link https://cran.r-project.org/bin/ and select the correct OS from the directory lists on the webpage. Locate a version 3.5 or less and install it.

- https://cran.r-project.org/bin/windows/base/old/3.5.0/
- https://cran.r-project.org/bin/macosx/R-3.3.3.pkg

TO install Rstudio you just need to navigate to the page located in the embedded link in the title above. On the page that populates scroll down a bit until you see a table and the phrase `FREE` in big bold letters. A button labeled `DOWNLOAD` is below that phrase. Click on the button and a new page will populate with all the links to download Rstudio for Windows, Linux, or Mac Operating System *(OS)*.

---

(3) Install R Packages

Open Rstudio which starts an R session and set the working directory to the cloned folder. The `setwd()` function will set the working directory to any pathway of your choosing. It is best practice to set a working directory that is the main location that will contain all your sub-elements.

```
setwd("C:/Users/Jonathan Korn/Desktop/cu_hsp-learning")
```

If you notice there is a quick way to capture the working directory of your choosing. By using the bottom left window you will see a selection from the header labeled `Files`. Click on that header and it will show you a navigator that you are able to use and navigate to the folder of your choosing. When you have located the folder you can use the button indicated by the settings symbol and the label `More` and you will see a selection labeled `Set as Working Directory`.

For our course I have taken the liberty to creating some resources that may make the preparation of your R enviorments a little easier. Open the `install.packages.r` script to install the dependency packages for the course.

- Note, some of the packages will be installed from source files stored in the `packages` folder.

If the script runs successful you should have every package that we need for all the remaining code portions in R. Most likley you may have noticed that some of the packages did not install succesfully. That is because some of are R packages are dependent on some Python modules. Let us get Python prepared and we can install all our R resources again. Next time we should see a successful installtion and loading of all the R packages.

---

(4) Install Python <=3.7 and Anaconda.

Make sure to select the Anaconda version that matches your Python version for the best results. I suggest using the following options:

- Anaconda3-2018.12-Linux-ppc64le.sh
- Anaconda3-2018.12-Linux-x86.sh

- Anaconda3-2018.12-Linux-x86_64.sh
- Anaconda3-2018.12-MacOSX-x86_64.pkg
- Anaconda3-2018.12-MacOSX-x86_64.sh

- Anaconda3-2018.12-Windows-x86.exe
- Anaconda3-2018.12-Windows-x86_64.exe

(5) Open the Anaconda terminal or local machines commmand prompt.

If you choose your local machine make sure to change the directory to the pathway of your *./python.exe* to ensure any modules installed are directed properly. The Anaconda terminal is already connectto the Python pathway.

- Later when we have our Rstudio Enviorment prepared we can use the R package `reticulate` to access the function `py_config()` and it will direct us to the *./python.exe* file location.

Install the following modules:

- `pip install tensorflow==1.15` or `conda install tensorflow==1.15`

- `pip install keras`
  - Optionally you can unsleep the lines #() in script `install.packages.r` to install the modules in python because the package `reticulate` enables the connection between R and Python to communicate commands.

Note: *All this may not work the first time. We may have to attempt to install package sin R several times or modules in python. We may even run into some issues with the OS system you are using for some packages.*

Debugging Potential Issues in R

To help debug some issues during the installation of R packages you could attempt the following options. *Note*, even after these steps there are some issues that may need investigating to find the solution, if there is one yet.

Another Version of R

A truly simple method is to try a different version of R. After installing another version of R from one of the following examples you will be able to change your Rstudio to that version.

- R-3.6.3 for Windows
- R for Mac OS X

Click on the `Tools` dropdown from before and click on `Global Options...` again, but this time right from the original window that populates you will see `R Version`, click on the `change` button and from the list select the version you would like R to be utilizing. Make sure to click apply before clicking ok. Also it may be best to close Rstudio and re-open it.

Changing the R CRAN

Sometimes you are able to select a different CRAN which is the server you are connecting with when you install packages. Changing the server you are connected to may help install the package, just follow the following steps:

(1) On the list of dropdowns in Rstudio's header click on `Tools` and from the list that appears select the option labeled `Global Options...`.

(2) From the window that appears you will see in the sidebar menu a selection labeled `packages`, click on this and you will see the page change.

(3) On the new page after clicking `packages` in the sidebar you should see a widget box which is labeled `Primary CRAN repository` next to a button labeled `change`, click on `change` and you will see a new window populate.

(4) On the new page you will see a list of CRAN repositories you can select from. I would suggest the closet to where you are located.

Source File Installation

Another method of debugging troublesome package installations is by finding the R documentation with the link to the actual packages resources, which are usually stored in a *.tar* file. An example of this can be found for the `rowr` package on the following page http://cran.nexr.com/web/packages/rowr/index.html. You should see that they provide a current `Package Source` and even a list of `Old sources` which many times need to be installed depending on the version R you have installed.

A example of how to install a R package from a *.tar* file is shown below:

```
if (!require("rowr")) {
  install.packages('./packages/rowr_1.1.3.tar.gz', repos = NULL, type ='source')
  library(rowr)
}
```

Debugging Potential Issues in Python

Most of the issues that come with installing Python are the requirement of another module that the one your installing is dependent on. Simply install the modules that the error populates suggest with the same process discussed above. The best method is to use the Anaconda Terminal and use `conda install` because it goes through a process to fix any issues in the enviroment before installing the module. Another method that is useful and may be practically useful is by oprning a python IDLE from the Anaconda Navigator on your desktop and use a `! pip install` right in the python terminal. The `!` will direct the command to the terminal to execute. I suggest using Jupyter Notebooks and Labs for your Python IDLE. You can build report files and python programs from the same front end.