02_01_Jupyter_Notebook

August 11, 2019

1 01. Jupyter Notebook

1.1 01.01 What is Jupyter Notebook?

According to Jupyter Notebook website: https://jupyter.org The Jupyter Notebook is an open-source web application that allows you to create and share documents that contain live code, equations, visualizations and narrative text. Uses include: data cleaning and transformation, numerical simulation, statistical modeling, data visualization, machine learning, and much more.

According to REF1 : The Jupyter notebook is a browser-based graphical interface to the IPython shell, and builds on it a rich set of dynamic display capabilities. As well as executing Python/IPython statements, the notebook allows the user to include formatted text, static and dynamic visualizations, mathematical equations, JavaScript widgets, and much more. Furthermore, these documents can be saved in a way that lets other people open them and execute the code on their own systems.

1.2 01.02 Basic Jupyter Notebook fuctionalities

1.2.1 01.02.01 Code cells

This is basic functionality when you execute code

[1]:
```
import numpy as np
print(np.linspace(0, 100, 26))
```

```
[  0.   4.   8.  12.  16.  20.  24.  28.  32.  36.  40.  44.  48.  52.
  56.  60.  64.  68.  72.  76.  80.  84.  88.  92.  96. 100.]
```

[2]:
```
from IPython.display import display, Markdown, Latex
display(Markdown("#### You can return a markdown as well"))
```

You can return a markdown as well Other langauges code (eg. HTML) can be executed as well. Below example to turn on/off raw code in Jupyter Notebook

[3]:
```
from IPython.display import HTML

HTML('''<script>
code_show=true;
```

```
function code_toggle() {
 if (code_show){
 $('div.input').hide();
 } else {
 $('div.input').show();
 }
 code_show = !code_show
}
$( document ).ready(code_toggle);
</script>
<form action="javascript:code_toggle()"><input type="submit" value="Click here␣
 ↪to toggle on/off the raw code."></form>''')
```

[3]: `<IPython.core.display.HTML object>`

1.2.2 01.02.02 Markdown cells

Text written in this type of cells will be intepreted as `.md` file.

Sample markdown cheat sheet can be found here

Bold text

Italics

- [x] TODO1
- [ ] TODO2
- [ ] TODO3

First Header	Second Header
Content from cell 1	Content from cell 2
Content in the first column	Content in the second column

You can use LATEX:

$$c = \sqrt{a^2 + b^2}$$

Code can be written as well

```
for k in range(10):
    print(k)
```

1.2.3 01.02.03 Jupyter Notebook menu

How to open the console: 1. Use `New` > `Terminal`

How to upload new file to the sever (when you work remotely): 1. Click `Upload`

How to download file from the sever (when you work remotely): 1. Click `File` > `Download as` > `notebook.ipnb`

1.2.4 01.02.04 Useful Jupyter Notebook shortcuts

The Jupyter Notebook has two different keyboard input modes. `Edit` mode allows you to type code or text into a cell and is indicated by a **green** cell border. `Command` mode binds the keyboard to notebook level commands and is indicated by a grey cell border with a **blue** left margin.

When you are in edit mode you can write a code in the cells. To switch to the `command` mode click `ESC`

In `command` mode you can use shortcuts (shortcuts can be Edited in `Help` > `Edit Keyboard Shortcuts`:

Shortcut	Action
`CTRL+ENTER`	run cell
`x`	Cut cell
`c`	Copy cell
`v`	Paste cell
`a`	Insert empty cell above
`b`	Insert empty cell below
`m`	Chenge cell into markdown cell
`“`	(Custom) Restart kernel

1.2.5 01.02.05 What are the caveats Jupyter users must be particularly aware of?

Jupyter Notebook is a great tool with a lot of functionality but when you use all these *magic* functions and command line tools you need to first test their behaviour to gain some intuition how does it work. In the example below we investigate how does the command line workflow look like:

[4]:
```
# this is current working directory for this
!pwd
```

```
/home/lcs123/DS/dstip/02_Jupyter_Notebook
```

[5]:
```
# we run the command to chenge working directory, everything seems to be OK
!cd ../dstip_utils/
```

[6]:
```
# unfortunately current working directory has not changed
!pwd
```

```
/home/lcs123/DS/dstip/02_Jupyter_Notebook
```

[7]:
```
# we can check that working directory is indeed changed but only temporarly
!cd ../dstip_utils/;pwd
```

```
/home/lcs123/DS/dstip/dstip_utils
```

[8]:
```
# in this case what we need is magic command `%cd`
# which permanently change the working directory for the notebook
%cd ../dstip_utils/
```

```
/home/lcs123/DS/dstip/dstip_utils
```

[9]:
```
# we may notice that the working directory has changed
# this condition will continue untill we change it explicite (or restart the␣
 ↪IPython kernel)
!pwd
```

```
/home/lcs123/DS/dstip/dstip_utils
```

1.3 01.03 Run notebooks via command line

Sometimes it is very helpful to run jupyter notebook via command line. To do this we use `nbconvert`. Run in the shell

```
jupyter nbconvert --to pdf '02_01_Jupyter_Notebook.ipynb'
```

```
jupyter nbconvert --to html '02_01_Jupyter_Notebook.ipynb'
```

[10]:
```
from dstip_utils.utils import yes_no_table
yes_no_dict = {'yes': ["Making a report (something to send/export/present)",
                       "Exploring, leaning, prototyping",
                       "Executing manually",
                       "... or making a training",
                       ],
               'no':["GIT Version Control is crucial",
                     "Meanwhile developing a lot of python sub-packages",
                     "Executing via tools (CI)"]}
yes_no_table(yes_no_dict)
```

```
<IPython.core.display.HTML object>
```