

Un poco más de Python e iPython Notebook

Carlos Malanche

8 de febrero de 2018

La clase pasada vimos un poco de lo que se puede hacer en Python con scripts y con su sesión interactiva. Ahora vamos a utilizar una herramienta un poco más sofisticada para hacer scripts con comentarios en html, además de ver lo que es un ambiente virtual y cómo instalar paquetes de internet.

Las instrucciones aquí escritas son para Windows 10 y Ubuntu 16.04, además de una guía tentativa para MacOS. En lo que llega la siguiente clase, voy a ver como consigo derechos de administrador en las computadoras del salón para que todo el software esté disponible en las máquinas de la universidad.

1. *Virtual environments*

¿Alguna vez han instalado algo que desearían no haber instalado? A veces la computadora funciona perfectamente, hacemos una actualización y algo deja de funcionar. Con **Python** podemos evitar este tipo de situaciones gracias a que tiene soporte para ambientes virtuales. Esto quiere decir que podemos hacer una pequeña instalación local de **Python** para probar a instalar paquetes o probar ciertas configuraciones, y si algo sale mal, borramos el folder de la instalación local de **Python** y la instalación original quedará intacta.

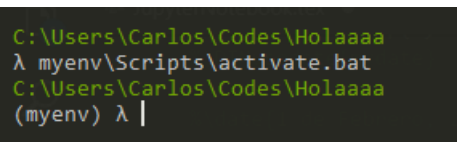
1.1. Windows

En Windows, **Python 3.6.4** ya viene con **venv**, la instrucción para generar ambientes virtuales. Para crear uno, basta con ejecutar:

```
python -m venv /ruta/al/folder
```

En particular, podemos ir a nuestro folder **adml-inicio** y dentro escribir **python -m venv myenv** para generar un ambiente virtual de **Python** en el folder **myenv**.

Una vez generado el ambiente virtual, hay que activarlo. Para ello, corremos el archivo **activate.bat** que está en **myenv/Scripts**.



```
C:\Users\Carlos\Codes\Holaaaa  
λ myenv\Scripts\activate.bat  
C:\Users\Carlos\Codes\Holaaaa  
(myenv) λ |
```

En el caso de **cmdr** podemos ver en paréntesis el nombre del ambiente virtual que estamos utilizando.

Dentro de nuestro ambiente virtual tenemos nuestros ejecutables **pip** y **pip3**, que sirven para instalar paquetes de Python. Por ejemplo, podemos instalar ahora **numpy** ejecutando **pip3 install numpy**. Numpy es un paquete que permite el manejo de matrices, vectores, y además contiene métodos numéricos usados frecuentemente en análisis numérico.

```

C:\Users\Carlos\Codes\Holaaaa
(myenv) λ pip3 install numpy
Collecting numpy
  Downloading numpy-1.14.0-cp36-none-win_amd64.whl (13.4MB)
    100% |#####| 13.4MB 49kB/s
Installing collected packages: numpy
Successfully installed numpy-1.14.0

C:\Users\Carlos\Codes\Holaaaa
(myenv) λ python
Python 3.6.4 (v3.6.4:d48eceb, Dec 19 2017, 06:54:40) [MSC v.1900 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> import numpy as np
>>> np.array([1,2,3])
array([1, 2, 3])
>>> |

```

Atención: Usen Python 3.6.4 para instalar esto en Windows, la versión 3.7 aún no cuenta con los archivos *wheel*, que son archivos precompilados, entonces les pedirá que tengan Visual Studio C++14 instalado.

Con el script `deactivate.bat` se puede verificar que el paquete `numpy` sólo ha sido instalado para nosotros.

```

C:\Users\Carlos\Codes\Holaaaa
(myenv) λ myenv\Scripts\deactivate.bat
C:\Users\Carlos\Codes\Holaaaa
λ python
Python 3.6.4 (v3.6.4:d48eceb, Dec 19 2017, 06:54:40) [MSC v.1900 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> import numpy as np
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
ModuleNotFoundError: No module named 'numpy'
>>> |

```

1.2. MacOS

Lamentablemente con MacOS no los puedo ayudar mucho, pero con unos amigos conseguimos hacerlo funcionar siguiendo las instrucciones de [esta página](#). A diferencia de Linux y Windows, es un poco difícil hacer funcionar el comando `venv`, por lo que existe `virtualenv`. Se necesita Python 3 junto con `pip3` (administrador de paquetes de Python).

Para instalar `virtualenv` ejecuten

```
pip3 install virtualenv
```

Ahora pueden crear un ambiente virtual ejecutando:

```
virtualenv myenv
```

Para activar el ambiente virtual, basta con ejecutar `source myenv/bin/activate`. Para salir del ambiente virtual, escribimos `deactivate` en la misma terminal y ya está.

1.3. Ubuntu 16.04

Ubuntu no viene con `venv`, pero se puede instalar rápidamente ejecutando `sudo apt-get install python3-venv`.

Una vez instalado, podemos crear un ambiente virtual ejecutando:

```
python3 -m venv /ruta/al/folder
```

Una vez generado el folder con el ambiente virtual, lo podemos activar con el comando `source` y el script `activate` que está en `myenv/bin/` (es decir, `source myenv/bin/activate`). Se nos va a nunciar que funcionó pues la terminal mostrará entre paréntesis el nombre del ambiente virtual. Para dejar el ambiente virtual, basta con escribir `deactivate` en la terminal. Todo esto se puede ver en la siguiente captura de pantalla:

```

carlos@raven:~/Documents$ sudo apt-get install python3-venv
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  python-pip-whl python3.5-venv
The following NEW packages will be installed:
  python-pip-whl python3-venv python3.5-venv
0 upgraded, 3 newly installed, 0 to remove and 114 not upgraded.
Need to get 1 118 kB of archives.
After this operation, 1 260 kB of additional disk space will be used.
Do you want to continue? [Y/n] y
Get:1 http://mx.archive.ubuntu.com/ubuntu xenial-updates/universe amd64 python-pip-whl all 8.1.1-2ubuntu0.4 [1 110 kB]
Get:2 http://mx.archive.ubuntu.com/ubuntu xenial-updates/universe amd64 python3.5-venv amd64 3.5.2-2ubuntu0~16.04.4 [5 998 B]
Get:3 http://mx.archive.ubuntu.com/ubuntu xenial/universe amd64 python3-venv amd64 3.5.1-3 [1 106 B]
Fetched 1 118 kB in 1s (587 kB/s)
Selecting previously unselected package python-pip-whl.
(Reading database ... 223721 files and directories currently installed.)
Preparing to unpack .../python-pip-whl_8.1.1-2ubuntu0.4_all.deb ...
Unpacking python-pip-whl (8.1.1-2ubuntu0.4) ...
Selecting previously unselected package python3.5-venv.
Preparing to unpack .../python3.5-venv_3.5.2-2ubuntu0~16.04.4_amd64.deb ...
Unpacking python3.5-venv (3.5.2-2ubuntu0~16.04.4) ...
Selecting previously unselected package python3-venv.
Preparing to unpack .../python3-venv_3.5.1-3_amd64.deb ...
Unpacking python3-venv (3.5.1-3) ...
Processing triggers for man-db (2.7.5-1) ...
Setting up python-pip-whl (8.1.1-2ubuntu0.4) ...
Setting up python3.5-venv (3.5.2-2ubuntu0~16.04.4) ...
Setting up python3-venv (3.5.1-3) ...
carlos@raven:~/Documents$ python3 -m venv myenv
carlos@raven:~/Documents$ myenv/bin/activate
carlos@raven:~/Documents$ source myenv/bin/activate
(myenv) carlos@raven:~/Documents$ deactivate
carlos@raven:~/Documents$

```

Ahora pueden instalar los paquetes que quieran (mientras el ambiente virtual esté activo) con `pip3 install`.

1.4. Común a todos

Son 4 los paquetes que vamos a instalar: `pandas`, `numpy`, `matplotlib` y `jupyter`. Eso será suficiente para la siguiente parte del curso.

2. iPython notebook

Ahora conocido como Jupyter notebook, es una herramienta que permite ejecutar Python desde un navegador de internet. La ventaja de utilizar el navegador de internet es que podemos añadir texto en formato, además de permitirle a otras personas ejecutar nuestro código instrucción por instrucción.

Un poco más a detalle, Jupyter Notebook es un servidor de Python con el que se puede acceder por un navegador.

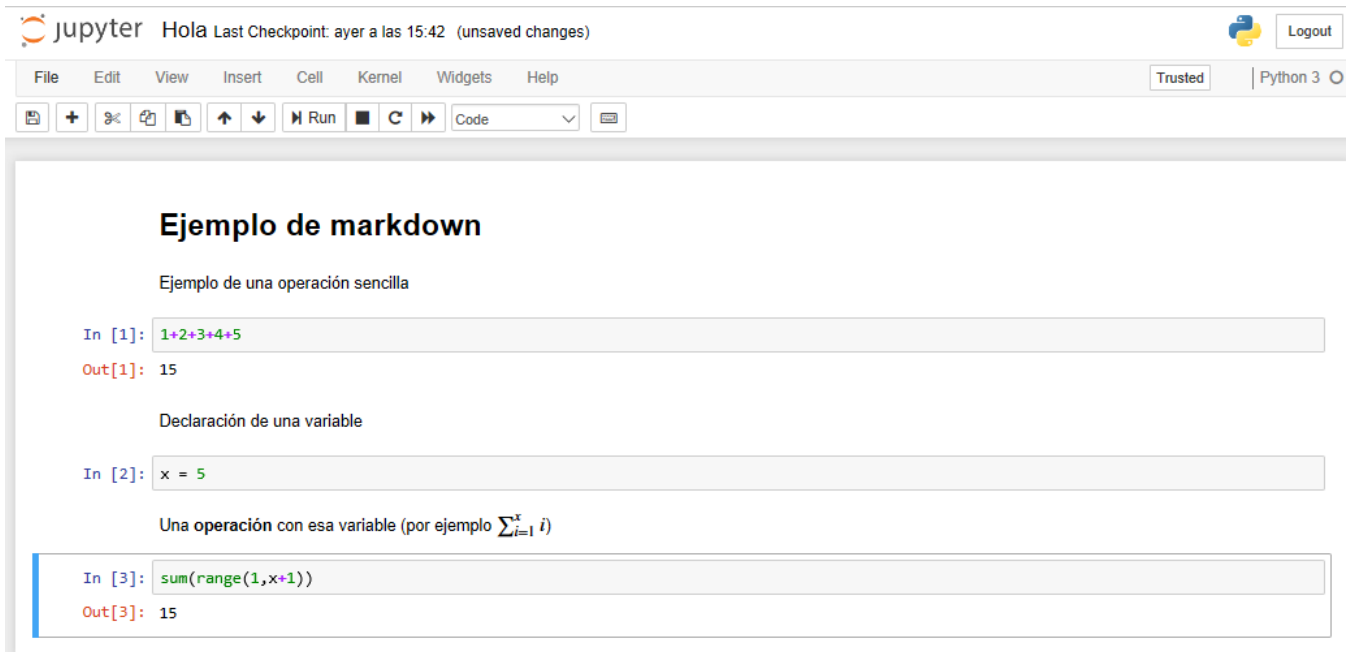
2.1. Instalación

Asumiendo que ya instalaron Python, basta con abrir una terminal y ejecutar `pip install jupyter` (**Ojo:** Los que usan windows tienen que añadir Python a las variables de entorno, el folder llamado Scripts, pues ahí se encuentra el comando `pip`. Los de Ubuntu pueden ejecutar `sudo apt install python3-pip`. Para los de MacOS debe haber una solución con `homebrew`).

2.2. Ejecución

En una terminal pueden escribir `jupyter notebook` para comenzar el servidor, u opcionalmente `jupyter notebook ejemplo.ipynb` para comenzar el servidor y comenzar a editar el archivo `ejemplo.ipynb` (el archivo debe existir. Si quieres empezar de cero, pueden escribir `touch ejemplo.ipynb` para generar un archivo vacío).

Al abrir jupyter sobre un documento (o hacer uno nuevo con el botón **new**) encontrarán una pequeña barra de tareas, y una barra vacía en el centro de la pantalla. Ahí pueden escribir código, texto en formato *markdown* e incluso cosas de \LaTeX (basta con rodear ecuaciones o lo que sea por símbolos de dinero, \$)



The screenshot shows a Jupyter Notebook interface. At the top, the Jupyter logo is followed by the text "Hola Last Checkpoint: ayer a las 15:42 (unsaved changes)". On the right, there is a "Logout" button. Below this is a menu bar with "File", "Edit", "View", "Insert", "Cell", "Kernel", "Widgets", and "Help". To the right of the menu bar are "Trusted" and "Python 3" indicators. Below the menu bar is a toolbar with icons for saving, adding, undo, redo, and other actions. The main area contains three code cells:

- Ejemplo de markdown**
Ejemplo de una operación sencilla
`In [1]: 1+2+3+4+5`
`Out[1]: 15`
- Declaración de una variable
`In [2]: x = 5`
- Una operación con esa variable (por ejemplo $\sum_{i=1}^x i$)
`In [3]: sum(range(1,x+1))`
`Out[3]: 15`

Para finalizar el servidor, se ejecuta `ctrl+c` en la terminal (y `cmd+c` en MacOS, *me supongo*).

Hay algunos *atajos* que les puedo compartir: si están en una celda, `shift+enter` ejecutará el código que contiene, y `alt+enter` hará lo mismo pero generará una nueva celda debajo. Hay otros truquitos, pero no se trata de que sean expertos *desde ahorita*, pues *uno nunca deja de aprender* e internet está ahí para ayudarnos si se nos olvida un comando. Python y sus paquetes son herramientas para nosotros, cuya utilidad vamos a ir descubriendo conforme necesitemos hacer cosas más específicas. Si tienen tiempo libre, pónganse a jugar con Python y con Jupyter, les va a ser útil la siguiente clase.