Advantages of Docker

--Sekhar Mekala

In Python based Data Science projects we quite often encounter a situation where we need to install a new package, and the new package is often dependent on the versions of the other existing packages. Sometimes the new package also requires the downgrade/upgrade other packages which were already existing in the Python environment.

Downgrading/Upgrading some of the already existing packages might have an adverse effect on already existing Python based application programs (including machine learning programs), and we may end up breaking critical applications. This problem is not only predominant in Python based systems, but also in most of the open source systems such as R.

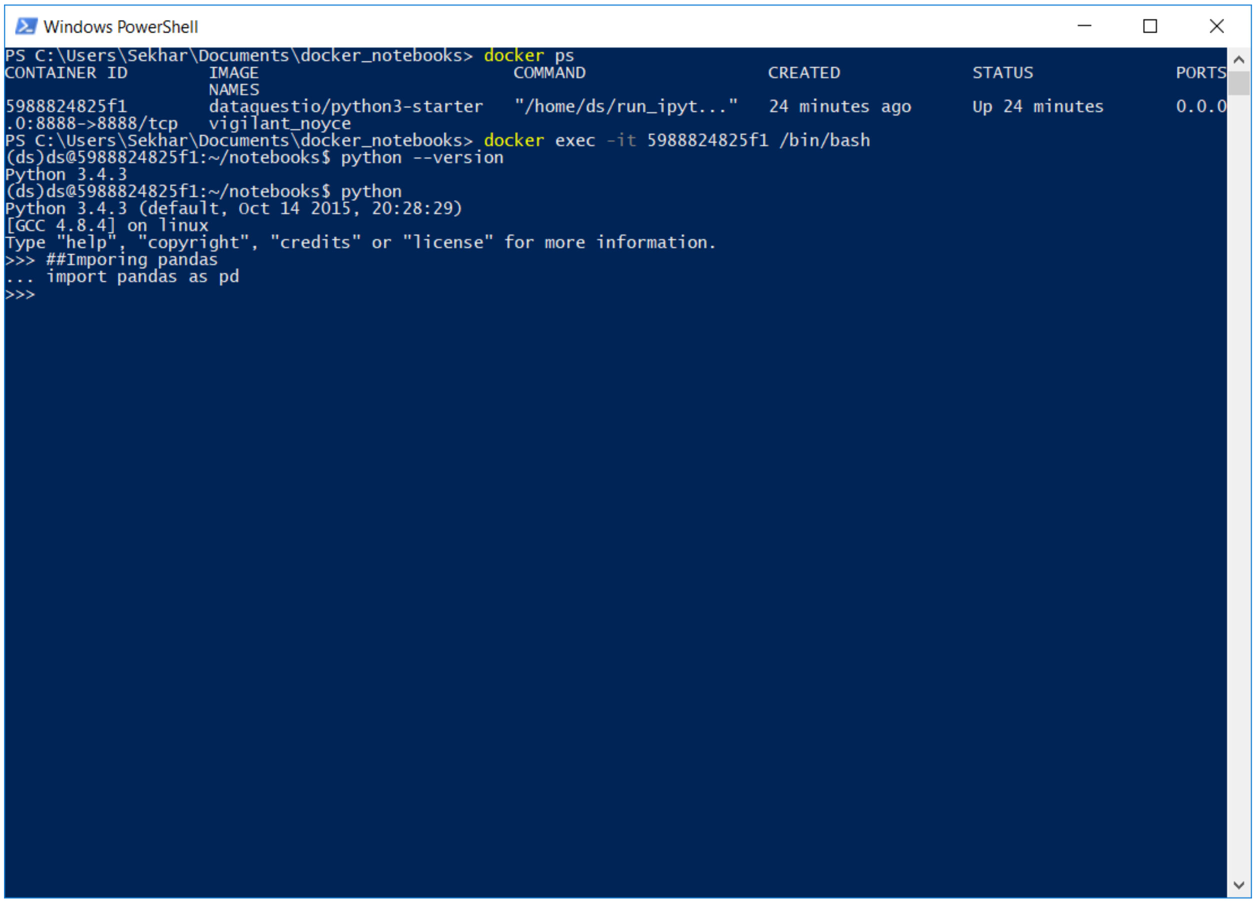
Python provides an elegant way to develop/test/deploy Python based applications using the concept of virtual environment (see http://docs.python-guide.org/en/latest/dev/virtualenvs/). But Python's `virtualenv` package is applicable only to the creation of *Python virtual environments* (and not the operating system environments).

Docker is an efficient solution since it helps us to span multiple Linux environments on the fly, without the need to perform multiple installations of Linux on virtual machines. For example, in VM Ware based virtualization, we need to install guest operating systems from the scratch on each of the virtual machine, and this process is usually laborious. In Docker based virtualization, we can span virtual Linux environment almost instantaneously, since we do not need to install the Linux operating system from the scratch. Another advantage of Docker is the portability of its virtual environments. Above all the Docker based virtualization is not just applicable to Python, but it is applicable to any situation where we need to maintain isolated environments to avoid the impact on other existing applications. However, it must be noted that

Docker provides only Linux virtual environments, and cannot provide any other operating system virtual environment.

As a part of Home Work-1, I installed Docker on Windows 10, downloaded the docker image (as explained in https://www.dataquest.io/blog/docker-data-science/) successfully performed the following:

1. Ran Jupyter Notebook using the docker image downloaded.
2. Logged into the virtual environment, and ran some commands (such as `python –version`).



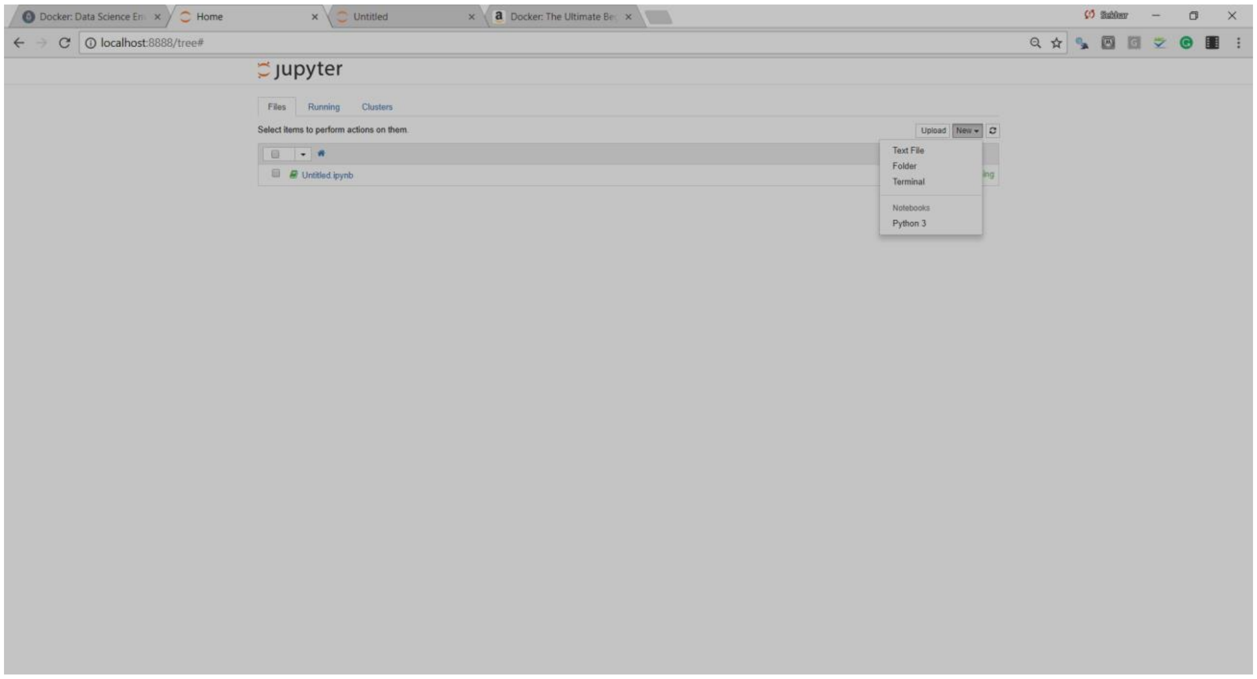
```
PS C:\Users\Sekhar\Documents\docker_notebooks> docker ps
CONTAINER ID        IMAGE                          COMMAND                  CREATED             STATUS              PORTS
                    NAMES
5988824825f1        dataquestio/python3-starter    "/home/ds/run_ipyt..."   24 minutes ago      Up 24 minutes       0.0.0
.0:8888->8888/tcp   vigilant_noyce
PS C:\Users\Sekhar\Documents\docker_notebooks> docker exec -it 5988824825f1 /bin/bash
(ds)ds@5988824825f1:~/notebooks$ python --version
Python 3.4.3
(ds)ds@5988824825f1:~/notebooks$ python
Python 3.4.3 (default, Oct 14 2015, 20:28:29)
[GCC 4.8.4] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> ##Imporing pandas
... import pandas as pd
>>>
```

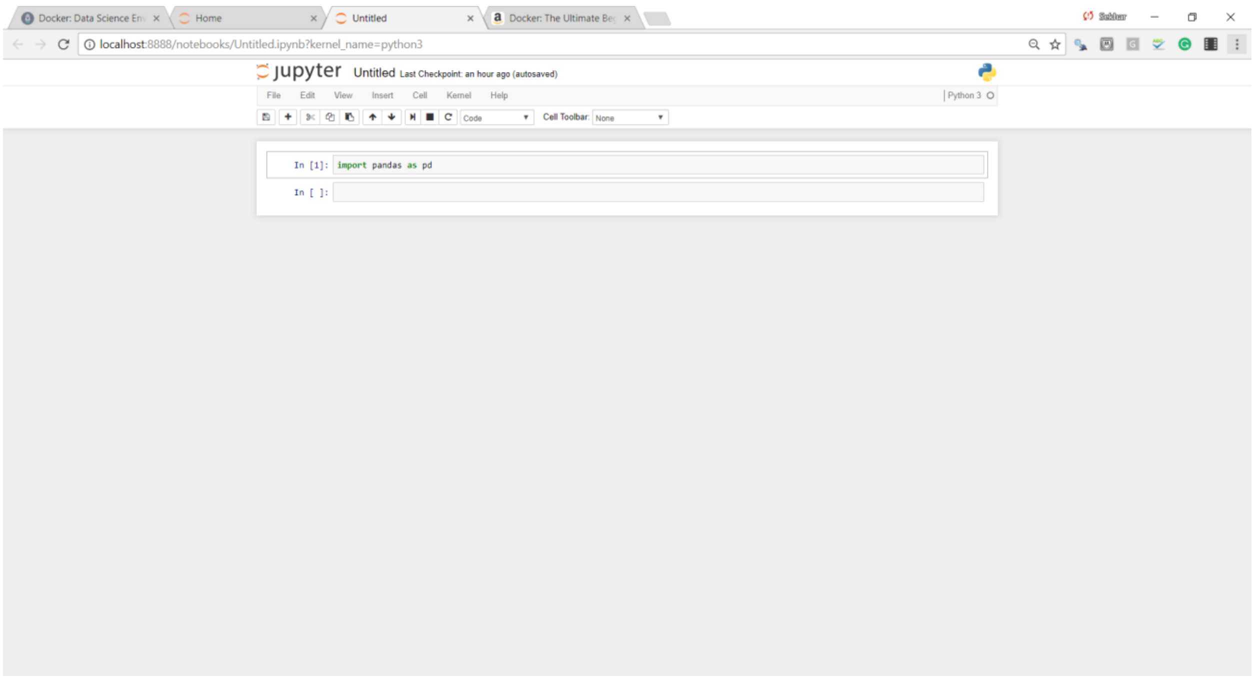
The Docker commands used in the above screen are shown below:

1. `docker ps` to display the containers information
2. `docker exec -it <container> /bin/bash` to login to virtual linux server
3. python –version to check the python’s version existing in the virtual environment.

Created a Jupyter notebook using the URL localhost:8888

See the below screen shots





```
In [1]: import pandas as pd

In [ ]:
```