

[Getting Started Guide]

Contents

What is Jupyter Notebook?
How To Use Jupyter Notebook?
Log in to the UofT JupyterHub4
Jupyter file explorer
Create a new Jupyter notebook for python6
Markdown cell7
Installing packages8
Docstring (Package, class, and function descriptions)9
Magic commands
How to share/distribute files?
One-click download (nbgitpuller)11
Upload/download files manually12
Read files from URL using supported Python modules13

What is Jupyter Notebook?

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.

Jupyter supports more than 40 programming languages, including Python, R, and Julia. The notebook can be shared just like any other document. The code within Jupyter notebook can produce rich, interactive output, including HTML, images, videos, and LaTeX. JupyterLab, Jupyter's Next Gen Notebook Interface, extends notebooks into fully interactive and customizable development environment for scientific computing applications.

The above information is summarized from Project Jupyter's official website. Learn more at <u>https://jupyter.org/</u>



How To Use Jupyter Notebook?

Log in to the UofT JupyterHub

To use the Jupyter hub environment, open any browser (Chrome, Edge, Safari, etc.) and search <u>https://jupyter.utoronto.ca</u>. The page will look like the image as shown below.



A proof of concept service, developed as a partnership between the Office of the CIO (Information Technology Services), the Faculty of Arts & Science's new Computational and Data Science Education initiative, the 2i2c Consortium, and Microsoft Canada.

site.



Select the interface you want to use and click on 'Log in to continue'. This will redirect you to the sign-in page. Please use your UTORid to sign-in.

Jupyter file explorer

After logging in, a new instance/server will be allocated to you. You can store your data files, code files here. The file structure is similar to what you see on your own system (laptop/pc). A sample structure is shown below. The files created/uploaded to the Hub will remain stored until the end of a current academic term. Please make sure to back-up files on your local computer if you want them after the end of an academic term. (Instructions on how to upload and download files are given later in this document).

💭 Jupyterhub	Logo	ut Cor	trol Panel
Files Running Clusters			
Select items to perform actions on them.	Download Directory	Upload	New 🗸 🕄
	Name 🕹 Last N	/lodified	File size
C Desktop	3 mo	nths ago	
Couments	3 mo	nths ago	
Downloads	3 mo	nths ago	
	2 mo	nths ago	
Ca Music	3 mo	nths ago	
	3 mo	nths ago	
C Public	3 mo	nths ago	
Co Templates	3 mo	nths ago	
C Videos	3 mo	nths ago	
🗌 🛢 Algo.ipynb	3 0	lays ago	17.6 kB
🗌 🖉 Assignment-1.ipynb	a m	onth ago	3.59 kB
🗌 🛢 Assignment-2.ipynb	a m	onth ago	1.07 MB
🗌 🖉 Assignment-3.ipynb	3 mo	oths ago	17.5 kB
🗋 🖉 Welcome to Jupyter.ipynb	a m	onth ago	7.35 kB
Dataset.csv	3 mo	oths ago	16.3 MB
Dataset_2.csv	3 mo	nths ago	292 kB

Create a new Jupyter notebook for python

On the top right corner of the file explorer, you can find a dropdown menu: "**New**", which will give you an option to either create a python, R, and text file or a folder, as shown below in the image. Click on '**Python 3**' to create a new python ipynb (iPython notebook) file.

Download D	irectory	Upload N	ew 🗸 🎗
	Noteboo	ok:	
Name 🔸	Pytho	n 3	:e
	R		
	Other:		
	Text F	ile	
	Folder		
	Termir	nal	
	Shiny		
	RStud	io	
	deskto	р	
	0 110	nina ago	
	3 mo	nths ago	

After creating a new python 3 file, an '**Untitled**' file will open, as shown in the image below.

Click here to change the name of the file			
Jupyterhub Untitled Last Checkpoint: 2 minutes ago (unsaved changes)	4	Logout	Control Panel
File Edit View Insert Cell Kernel Navigate Wildgets Help		Trusted	Python 3 O
B + %			
In []: Randoom Random Randoom Randoom Randoom Randoom Randoom Randoom Randoom Randoom		-	Cell
Click here to run the selected cell			

The file can be renamed by clicking on 'Untitled'. Just below the filename are menu bar and tool bar. The body of the page contains rectangle bar like structure called cell. This is where the code/markdown text is written. The toolbar has various buttons to add a new cell, copy a cell, run cell, change the type of cell (code, markdown, Raw NBConvert and heading), and so on.

Markdown cell

The markdown cell is generally used to write explanation or description of the code. These include headings, titles, bullet points, numbering, etc. The cell mode can be changed to Markdown type by selecting Markdown from the dropdown button in the tool bar.

An example of Markdown cell is shown below. When the cell is run, the output is rendered as shown on the right in the image below. To learn more about Markdown, refer <u>The Ultimate Markdown Guide (for</u> <u>Jupyter Notebook) | by Hannan Satopay | Analytics Vidhya | Medium</u>

Run C Markdown Voila Line III	1 Markdown
INTRODUCTION TO JUPYTER	Note: This text will be italicized
1 Markdown	1.1 Headline 2
Note: This text will be italicized	This is a normal paragraph.
<pre>### Headline 2 This is a normal paragraph. <uvythis an="" is="" pre="" text<="" underlined="" uvy<=""></uvythis></pre>	This is an underlined text
#### This notebook shows:	1.1.1 This notebook shows:
1. Get started with jupyter notebooks	1. Get started with jupyter notebooks
3. How to add markdown cells	2. How to execute code
	3. How to add markdown cells

Installing packages

The UofT JupyterHub is setup with Anaconda Python distribution and comes with common packages preinstalled. The packages can be imported into current notebook as usual by typing "**import <packagename>**".

You can install your own packages or packages that do not exist in the UofT hub directly from the code cell by using the Jupyter line magic command, %.

To install a new package, run "**%pip install <package- name>**" in a new cell. By doing this, you do not need to open the terminal to install new packages. This is done from within the notebook itself.

The manually installed packages do not persist in the system after logging out. This means that if you had installed a package from within the notebook, the next time you log in and try to run the code, you will have to re-install the package. To avoid this, it is better to implement the try and except method offered by python. An example of such manner is shown below and indicated by red rectangle.

Packages not installed on the server

except:

%pip install bcrypt import bcrypt

import Derypt	
ModuleNotFoundError <ipython-input-9-c2a86f184093> in <r > 1 import bcrypt</r </ipython-input-9-c2a86f184093>	Traceback (most recent call last)
ModuleNotFoundError: No module named	d 'bcrypt'
%pip install bcrypt	
Collecting bcrypt Using cached bcrypt-3.2.0-cp36-abi Requirement already satisfied: six> Requirement already satisfied: cffi: Requirement already satisfied: pycpa Installing collected packages: bcryp Successfully installed bcrypt-3.2.0 Note: you may need to restart the ke	i3-manylinux2010_x86_64.whl (63 kB) =1.4.1 in /opt/conda/lib/python3.8/site-packages (from bcrypt) (1.15.0) >=1.1 in /opt/conda/lib/python3.8/site-packages (from bcrypt) (1.14.4) arser in /opt/conda/lib/python3.8/site-packages (from cffi>=1.1->bcrypt) (2.20 pt ernel to use updated packages.
terrest because	

Docstring (Package, class, and function descriptions)

Another advantage of a Jupyter notebook is that you can easily read the descriptions of packages or python objects such as class, methods, and functions, if the description is provided in the code by developer. Just type '?' after the function/package name and run the code cell.

In [9]:	# displays docstring
	np.max?
-	
Signature:	
np.max(
a,	
axis=None,	
out=None.	
keendims= <n< th=""><td>o values</td></n<>	o values
initial-/no	value
	value,
where= <no th="" v<=""><td>alue>,</td></no>	alue>,
)	
Docstring:	
Return the maxi	mum of an array or maximum along an axis.
Parameters	
a : arrav like	
Input data	
avic , None on	int on tunlo of into ontional
axis : wone of	int of tuple of ints, optional
used.	s along which to operate. By default, flattened input is
versiona	dded · · 1 7 0

Magic commands

The '%'sign used while installing new packages and '?' used to view the documentation of certain packages or functions are examples of magic commands. These commands can be used in Jupyter notebooks for enhancing the functionality and understanding of the code. Magics are specific to and provided by the IPython kernel. There are many Magic functions available with python kernel. Two examples are shown below. To learn more about other available commands, please refer the documentation at <u>Built-in magic commands</u> — IPython 7.19.0 documentation

Examples of such magic commands are:

- 1. **%%time** or **%time** : These are used to measure the time taken by a function or a line of code to execute. To use this command in line mode prefix with single % sign. To use it in cell mode, prefix with %%.
- 2. **%who_ls** : This command shows the sorted list of all the interactive variables/aliases used in the notebook.

```
def fact(n):
    out=1
    for i in range(1, n+1):
        out = out*i
    return out
%%time
fact(5)
CPU times: user 3 µs, sys: 2 µs, total: 5 µs
Wall time: 8.58 µs
120
```

```
%who_ls
```

['bcrypt', 'fact', 'np', 'pd', 'time']

How to share/distribute files?

The 3 ways to distribute files for use in JupyterHub are described below:

One-click download (nbgitpuller)

This method requires files to be uploaded in a GitHub repository. A sharable link can be created using nbgitpuller. When clicked on this link, students will be taken directly to the UofT JupyterHub and files will be fetched automatically into students file explorer inside JupyterHub. To distribute files using this method, follow the steps as described:

Step 1: Create public repository on *GitHub* and upload files to the repository.

Step 2: Go to "<u>https://jupyterhub.github.io/nbgitpuller/link</u>" and fill out all the required details as shown in the image below.

	nbgitpulle	r link generator			
	Use the following form to cr	eate your own nbgitpuller links.			
			JupyterHub	Launch from Canvas	Binder
Sharable link (URL)	https://jupyter.utoronto	o.ca/hub/user-redirect/git-pull?repo=https%3A%2F%2Fgithub.com%2FVIjayMarav	riya%2FAPS007	-Assignment-1&urlpat	:h=tree%
UofT JupyterHub URL ——	JupyterHub URL	https://jupyter.utoronto.ca/			~
		The JupyterHub to send users to. nbgitpuller must be installed in this hub.			
link to GitHub repository	 Git Repository URL 	https://github.com/VijayMaraviya/APS007-Assignment-1	branch mai	n	~
with files	File to open	index.ipynb			~
		This file or directory from within the repo will open when user clicks the link.	fy the corre	ct branch	
Select which application	 Application to Open 	Classic Jupyter Notebook JupyterLab	ing the corre		
to open		O RStudio			
		Relative URL to redirect user to			

Upload/download files manually

Professors can post all the files required for the assignment on Quercus from where the students can download and upload manually to their JupyterHub directory.

Clicking on the upload button will open the file picker to select and upload files from student's computer to JupyterHub.

	Click upload button for file picker 🔨	
💭 Jupyter <mark>hub</mark>	Logout Cor	trol Panel
Files Running Clusters Nbextensions		
Select items to perform actions on them.	Download Directory Upload	New 🗸 🎗
0 - 1	Name 🕹 Last Modified	File size
□ □ A2	6 minutes ago	
□ □ R	5 days ago	
C C Rdemoassign	an hour ago	
E test.Rmd	5 days ago	831 B
Untitled.ipynb	5 days ago	555 B
Untitled1.ipynb	5 days ago	551 kB
Untitled2.ipynb	5 days ago	20.8 kB
Untitled3.ipynb	2 days ago	17.8 kB
test.pdf	5 days ago	151 kB

Read files from URL using supported Python modules

Many python modules provide the functions than can read specific file formats from a URL. This is not a feature of JupyterHub. Example for one such module, Pandas, is shown below which reads csv file in Pandas DataFrame object.

Read table data from URL in Python

Jupytern	ub Untitled Last Checkpoint: 2 minutes ago (unsaved changes)	Logout Control Panel
ile Edit V	View Insert Cell Kernel Widgets Help	Trusted Python 3 (
+ 🔀 🗠		Memory: 208.3 MB / 2 GB
	Read CSV from URL using Pandas	
In [1]:	import pandas as pd	
In [1]:	import pandas as pd	
In [1]:	<pre>import pandas as pd url="https://raw.githubusercontent.com/cs109/2014_data/master/countries.csv" countries_df = pd.read_csv(url)</pre>	
In [1]:	<pre>import pandas as pd unl="https://raw.githubusercontent.com/cs109/2014_data/master/countries.csv" countries_df = pd.read_csv(url) countries_df.head()</pre>	
In [1]: Out[1]:	<pre>import pandas as pd url="https://raw.githubusercontent.com/cs109/2014_data/master/countries.csv" countries_df = pd.read_csv(url) countries_df.head() Country Region</pre>	
<pre>In [1]: Out[1]:</pre>	<pre>import pandas as pd url="https://raw.githubusercontent.com/cs109/2014_data/master/countries.csv" countries_df = pd.read_csv(url) countries_df.head() Country Region 0 Algeria AFRICA</pre>	
<pre>In [1]: Out[1]:</pre>	<pre>import pandas as pd unl="https://naw.githubusercontent.com/cs109/2014_data/master/countries.csv" countries_df = pd.read_csv(url) countries_df.head() Country Region Algeria AFRICA Angola AFRICA Angola AFRICA</pre>	
In [1]: Out[1]:	<pre>import pandas as pd url="https://raw.githubusercontent.com/cs109/2014_data/master/countries.csv" countries_df = pd.read_csv(url) countries_df.head() Country Region 0 Algeria AFRICA 1 Angola AFRICA 2 Benin AFRICA </pre>	
<pre>In [1]: Out[1]:</pre>	<pre>import pandas as pd url="https://raw.githubusercontent.com/cs109/2014_data/master/countries.csv" countries_df = pd.read_csv(url) countries_df.head() Country Region Algeria AFRICA Angola AFRICA Benin AFRICA Botswana AFRICA </pre>	

Additional Resources

• A sample Jupyter notebook: <u>Jupyter Notebook Viewer | Exploratory data analysis in Python</u>