CONVERTING JUPYTER NOTEBOOKS TO PYTHON

Jacalyn Huband
UVA Research Computing

Motivation for Today

- Jupyter Notebooks are great for exploring data and testing codes.
- When you are ready to perform a “production run”, you may not need to have the interaction of a Notebook.
- It would be nice if you could submit a job, log out, and come back later to see the results.

Today's Notes

- Notes for today are on the git repository:

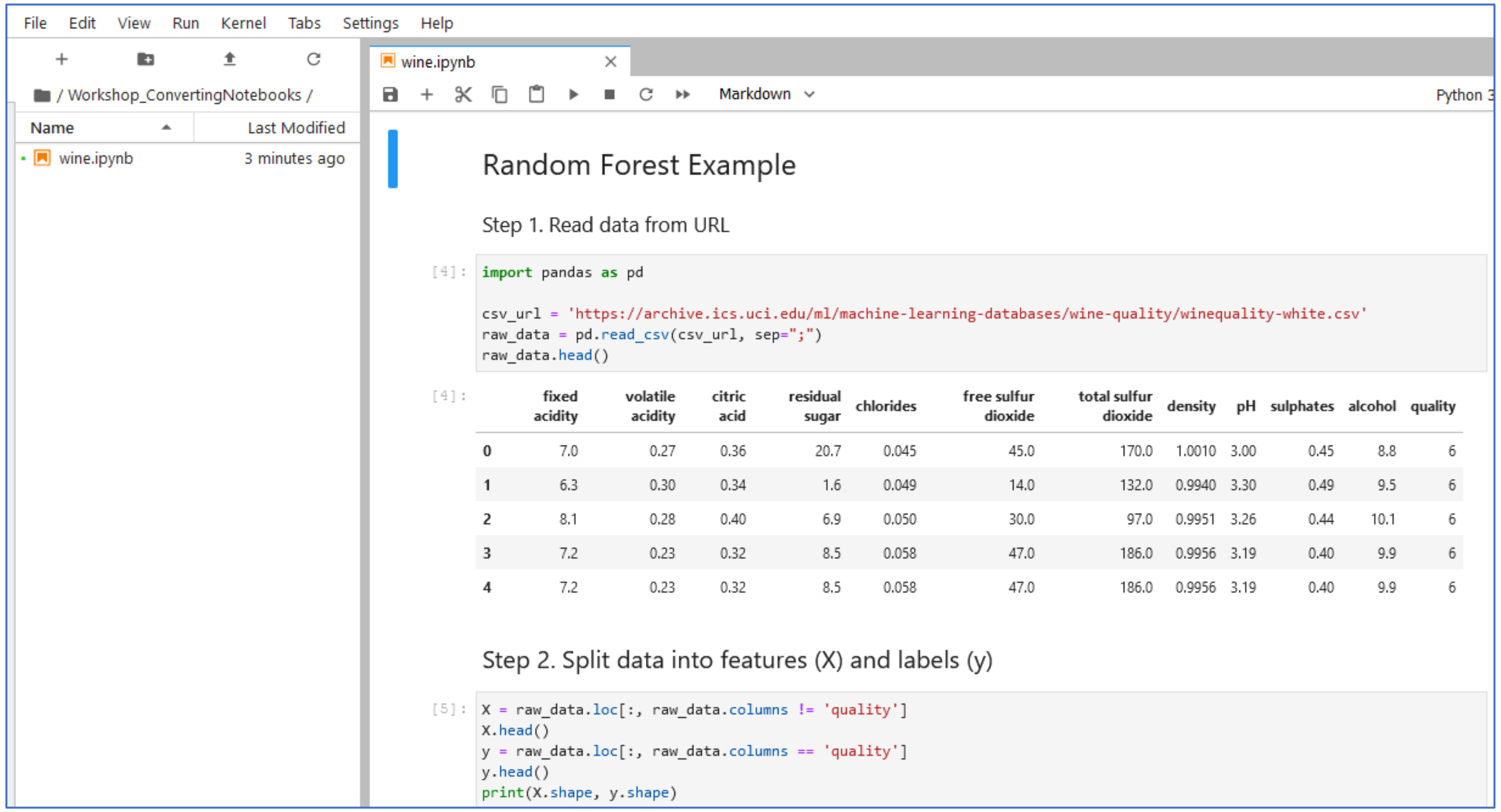
https://github.com/jhuband/Converting_Notebooks_to_Script.git

METHOD #1

Notebook is not on Rivanna

First Sample Notebook

The first sample notebook, wine.ipynb, is a Random Forest algorithm that tries to determine the quality of wine based on several features.


File Edit View Run Kernel Tabs Settings Help
/ Workshop_ConvertingNotebooks /
Name
Last Modified
wine.ipynb
3 minutes ago
wine.ipynb
Markdown
Python 3

Random Forest Example

Step 1. Read data from URL

```
[4]: import pandas as pd

     csv_url = 'https://archive.ics.uci.edu/ml/machine-learning-databases/wine-quality/winequality-white.csv'
     raw_data = pd.read_csv(csv_url, sep=";")
     raw_data.head()
```

[4]:

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	alcohol	quality
0	7.0	0.27	0.36	20.7	0.045	45.0	170.0	1.0010	3.00	0.45	8.8	6
1	6.3	0.30	0.34	1.6	0.049	14.0	132.0	0.9940	3.30	0.49	9.5	6
2	8.1	0.28	0.40	6.9	0.050	30.0	97.0	0.9951	3.26	0.44	10.1	6
3	7.2	0.23	0.32	8.5	0.058	47.0	186.0	0.9956	3.19	0.40	9.9	6
4	7.2	0.23	0.32	8.5	0.058	47.0	186.0	0.9956	3.19	0.40	9.9	6

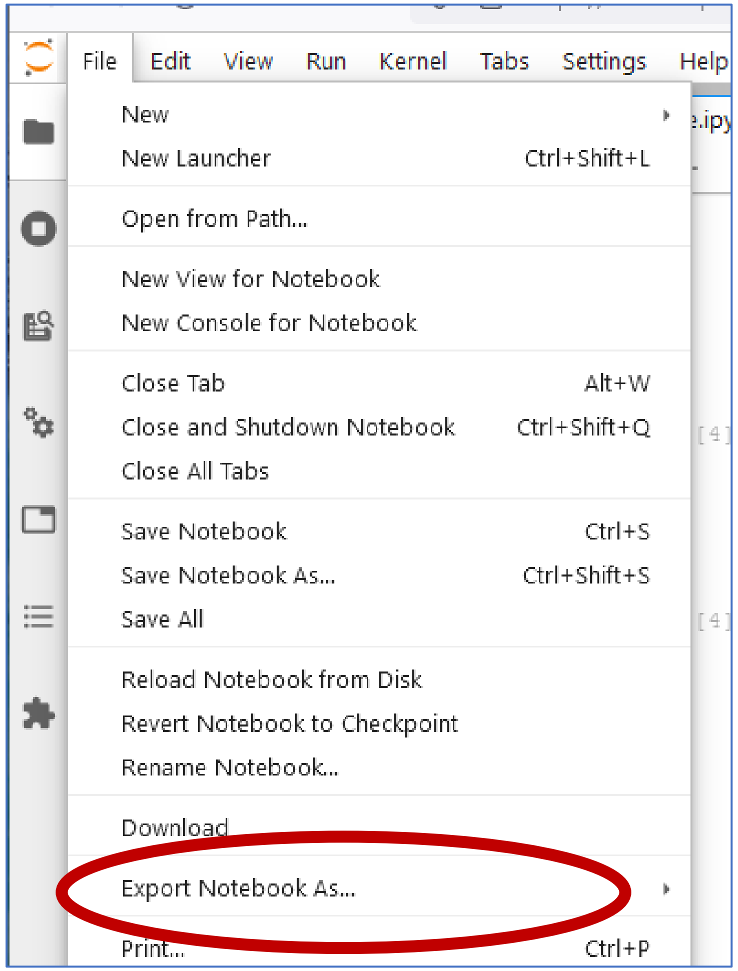
Step 2. Split data into features (X) and labels (y)

```
[5]: X = raw_data.loc[:, raw_data.columns != 'quality']
     X.head()
     y = raw_data.loc[:, raw_data.columns == 'quality']
     y.head()
     print(X.shape, y.shape)
```

University of Virginia | Research Computing

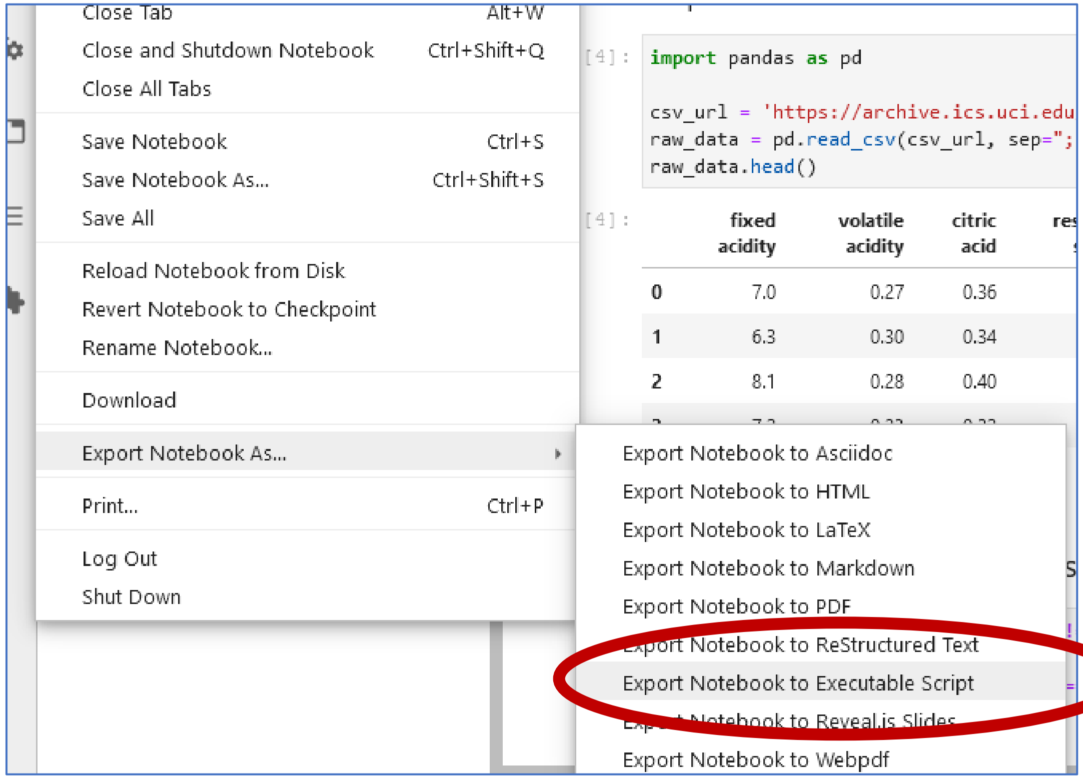
Converting the Notebook to Python

- Click on
  File > Export Notebook As >



Converting the Notebook to Python

- In the drop-down box, select
  Export Notebook to Executable Script


Close Tab Alt+W
Close and Shutdown Notebook Ctrl+Shift+Q
Close All Tabs
Save Notebook Ctrl+S
Save Notebook As... Ctrl+Shift+S
Save All
Reload Notebook from Disk
Revert Notebook to Checkpoint
Rename Notebook...
Download
Export Notebook As...
Print... Ctrl+P
Log Out
Shut Down
Export Notebook to Asciidoc
Export Notebook to HTML
Export Notebook to LaTeX
Export Notebook to Markdown
Export Notebook to PDF
Export Notebook to ReStructured Text
Export Notebook to Executable Script
Export Notebook to Reveal.js Slides
Export Notebook to Webpdf

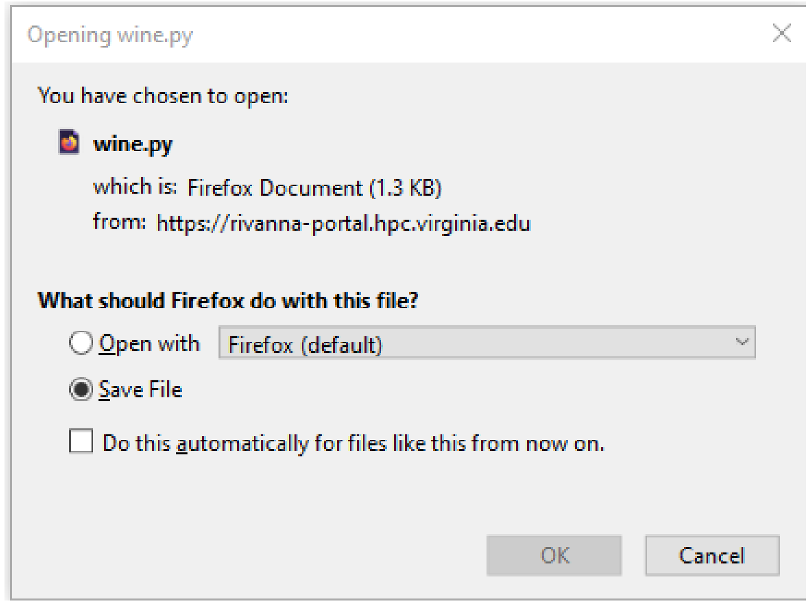
```
[4]: import pandas as pd

     csv_url = 'https://archive.ics.uci.edu
     raw_data = pd.read_csv(csv_url, sep=";
     raw_data.head()
```

[4]:

	fixed acidity	volatile acidity	citric acid	res
0	7.0	0.27	0.36	
1	6.3	0.30	0.34	
2	8.1	0.28	0.40	

Converting the Notebook to Python

- You may be asked about opening/saving the files



Opening wine.py

You have chosen to open:

wine.py

which is: Firefox Document (1.3 KB)
from: https://rivanna-portal.hpc.virginia.edu

What should Firefox do with this file?

( ) Open with Firefox (default)
(●) Save File
☐ Do this automatically for files like this from now on.

METHOD #2

Notebook is on Rivanna

Creating Python Script on Rivanna

- Open a terminal window and move to the directory where the nodebook is located.
- Type the following:

```
module load anaconda
jupyter nbconvert --execute  --to python wine.ipynb
```

- The *jupyter nbconvert* command will create a file with the same base name but with the `py` extension.

RUNNING A PYTHON SCRIPT ON RIVANNA

Modules
Slurm scripts

Using Modules on Rivanna

- Modules are our organizational tool for making software applications available on Rivanna.
- To have access to Python (at least a more current version of Python), you will need to type:

```
module load anaconda
```

- To see what you have loaded, you can type:

```
module list
```

Hands-on Activity

- Open a terminal window on Rivanna and try the following activities:

```
module purge
python --version

module load anaconda
python --version
```

Writing a Slurm script for the Code

- Slurm is the resources manager on Rivanna.
- We have to go through Slurm to run the code on a compute node
- Sample Slurm script, called submit_wine.slurm:

```
#!/bin/bash
#SBATCH --nodes=1                   #total number of nodes for the job
#SBATCH --ntasks=1                  #how many copies of code to run
#SBATCH --time=00:10:00             #amount of time for the whole job
#SBATCH --partition=standard        #the queue/partition to run on
#SBATCH --account=Rivanna-training #the account/allocation to use

module purge
module load anaconda/2020.11-py3.8 #load modules my job needs
python wine.py                    #command-line execution of my job
```

Hands-on Activity

- In your terminal window type:

```
sbatch submit_wine.slurm
```

- To check on its progress, type

```
sacct
```

- When it is done, you can look at the output. For example:

```
cat slurm-32631211.out
```

Tweaking the Results

- Notice that the plot which was in the Notebook, does not appear in the Slurm output file.
- The plots appear when the code runs in an interactive mode, like with Jupyter Notebooks.
- When running in a batch mode, you will need to instruct the code to write the plot to a separate file:

```
fig, ax = plt.subplots()
ax.barh(range(len(importances)), importances[indices])
ax.set_yticks(range(len(importances)))
_ = ax.set_yticklabels(np.array(X.columns)[indices])
## Save plot to a file
plt.savefig('Wine_Features.png')
```

Viewing the Image File

- To view the image file, you can
  - Go to the Open OnDemand Dashboard;
  - Click on Files > Home Directory;
  - Move into the appropriate folder;
  - Highlight the image file; and
  - Click “view”.
- Go back to your Jupyter Notebook. In the File Explorer column, double-click on the image file.

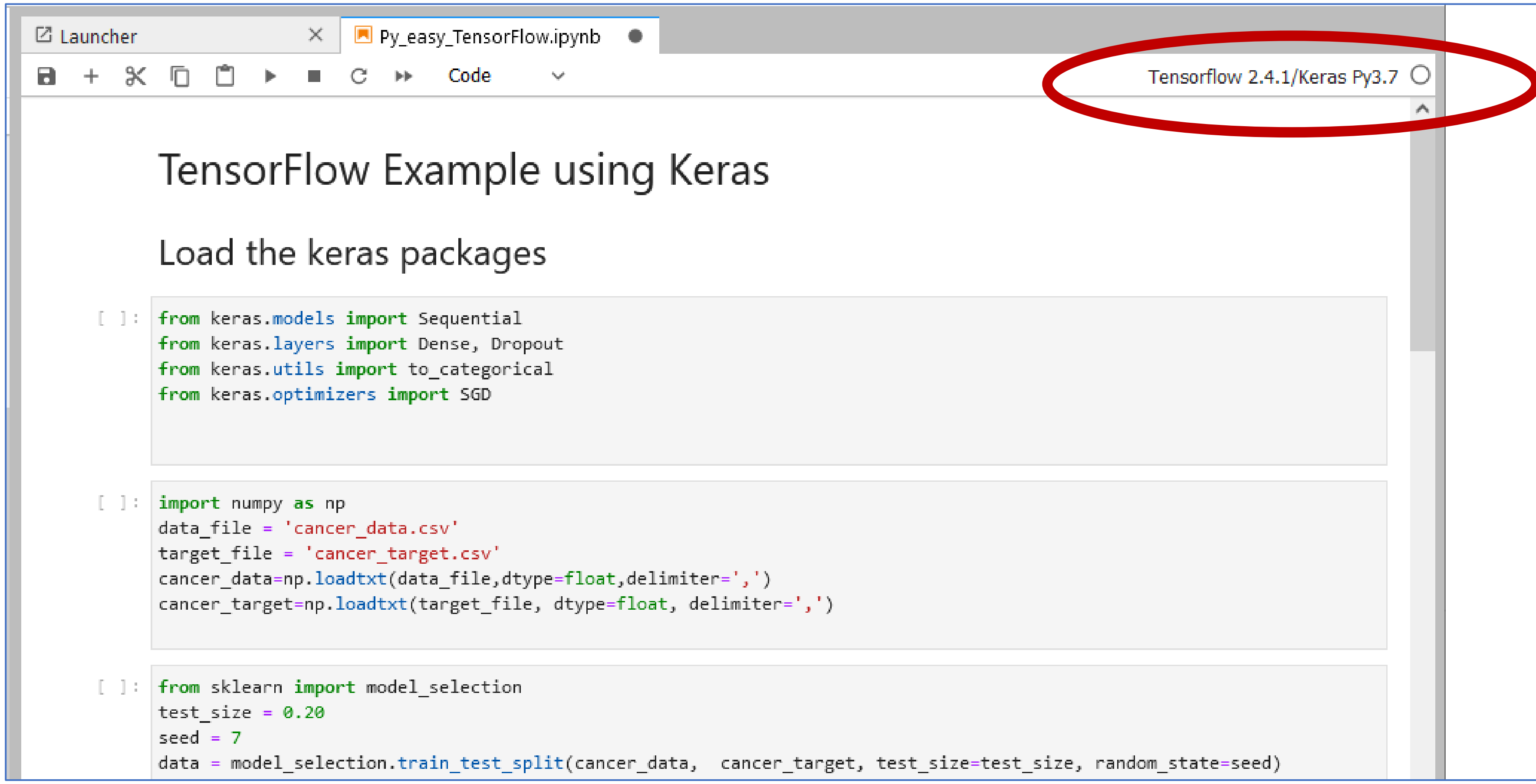
RUNNING A MORE COMPLICATED SCRIPT ON RIVANNA

Kernels
Containers

Second Sample Notebook

The next sample notebook, Py_easy_TensorFlow.ipynb, is a TensorFlow algorithm that tries to predict if a tumor is malignant or benign.

A major difference here is the kernel that the notebook uses.



TensorFlow Example using Keras

Load the keras packages

```
[ ]: from keras.models import Sequential
     from keras.layers import Dense, Dropout
     from keras.utils import to_categorical
     from keras.optimizers import SGD
```

```
[ ]: import numpy as np
     data_file = 'cancer_data.csv'
     target_file = 'cancer_target.csv'
     cancer_data=np.loadtxt(data_file,dtype=float,delimiter=',')
     cancer_target=np.loadtxt(target_file, dtype=float, delimiter=',')
```

```
[ ]: from sklearn import model_selection
     test_size = 0.20
     seed = 7
     data = model_selection.train_test_split(cancer_data,  cancer_target, test_size=test_size, random_state=seed)
```

Jupyter Kernels Overview

- The kernel defines the underlying environment in which the Notebook will run.
- In our first example, the kernel was Python 3.
- But, other kernels exist that run as containers or conda environments. These kernels provide customized environments or even languages such as R or Julia.
- When working with kernels other than Python 3, we will need to identify the environment needed to run the code.

Kernels & Containers

- We have several kernels, such as Tensorflow 2.4.1/Keras Py3.7, that run as containers.

- To recreate this environment in a Slurm script, we will need to
  - Find the appropriate modules for the container, and
  - Invoke the container with Singularity to recreate the kernel environment.

Searching for Kernel Modules

- Pick a key word in the kernel name and search on it with the `module spider` command:

```
module spider tensorflow
```

Searching for Kernel Modules

- The module spider command will give you a list of possible modules

```
-------------------------------------------------------------------------------------------------------
 tensorflow:
-------------------------------------------------------------------------------------------------------
  Description:
   TensorFlow is an open-source software library for Machine Intelligence.

   Versions:
     tensorflow/1.12.0-py36
     tensorflow/2.1.0-py37
     tensorflow/2.4.1
     tensorflow/2.7.0
     tensorflow/2.8.0

-------------------------------------------------------------------------------------------------------
```

Closest to version of our kerrnel

Searching Kernel Modules Versions

- For more details, we can include the version number in the module spider command:

```
module spider tensorflow/2.4.1
```

- Resulting Output:

```
----------------------------------------------------------------------------------------------------
  tensorflow: tensorflow/2.4.1
----------------------------------------------------------------------------------------------------
. . .
    You will need to load all module(s) on any one of the lines below before the
"tensorflow/2.4.1" module is available to load.

      singularity/3.7.1
```

Important Information

University of Virginia | Research Computing

Running Singularity Modules

- If the version information shows “singularity” as a requirement for the module, then the module sets up a container.
- Loading the modules will tell us the name of the container.
- To run a container, we have to run singularity with the name of the container and the name of our script:

```
singularity run --nv <container_name> <script_name>
```

- Plus, if we are running on GPUs, we need to use the --nv option to make singularity aware of the available hardware.

Hands-on Activity

- In your terminal window type:

```
module load singularity/3.7.1
module load tensorflow/2.4.1
```

- Notice what is written to the screen. This is the command to run the container.

- Go ahead and convert the Notebook Py_easy_TensorFlow.ipynb to a Python script.

Slurm script for a Container on a GPU

- In addition to requesting the GPU partition, we need to ensure that a GPU device is requested.
- Sample Slurm script:

```
#!/bin/bash
#SBATCH --nodes=1                    #total number of nodes for the job
#SBATCH --ntasks=1                   #how many copies of code to run
#SBATCH --time=00:10:00              #amount of time for the whole job
#SBATCH --partition=gpu              #the queue/partition to run on
#SBATCH --gres=gpu:1
#SBATCH –account=rivanna-training  #the account/allocation to use

module purge
module load singularity/3.7.1 tensorflow/2.4.1
singularity run --nv $CONTAINERDIR/tensorflow-2.4.1.sif  Py_easy_TensorFlow.py
```

Hands-on Activity

- Submit the Slurm script to run the Py_easy_TensorFlow.py code

NEED HELP?

Research Computing Zoom Office Hours

https://www.rc.virginia.edu/support/#office-hours

Tuesdays:	3 pm – 5 pm
Thursdays:	10 am – noon

Or, contact us through the forms at:

https://www.rc.virginia.edu/support/