AI Tech Labs 0⇒1

Zhenhua He

happidence1@tamu.edu

HPRC Short Course

06/02/2021

Original slides created by Dr. Jian Tao

High Performance Research Computing
DIVISION OF RESEARCH

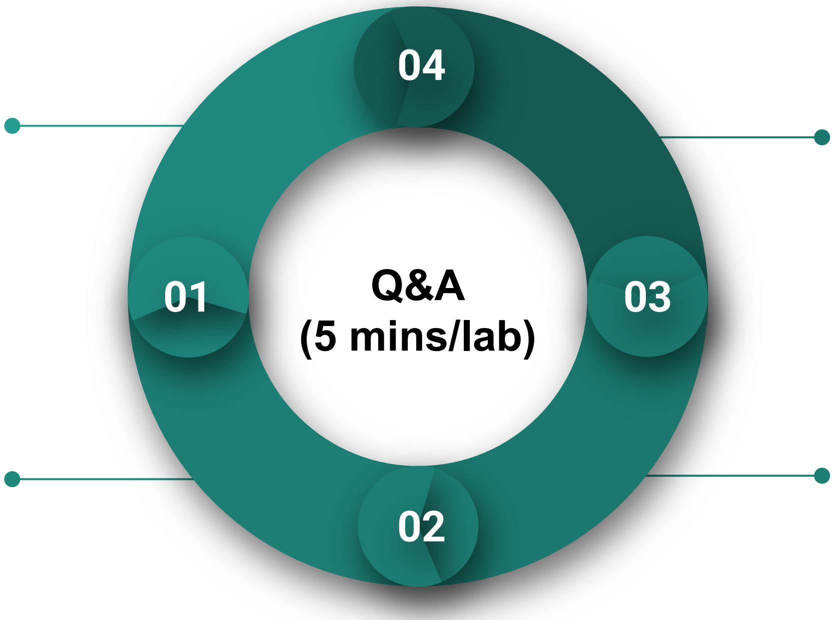
AI Tech Labs

Lab I. JupyterLab (30 mins)

We will set up a Python virtual environment and run JupyterLab on the HPRC Portal.

Lab II. Data Exploration (30 mins)

We will go through simple examples with two popular Python modules: Pandas and Matplotlib for simple data exploration.



04

01

Q&A
(5 mins/lab)

03

02

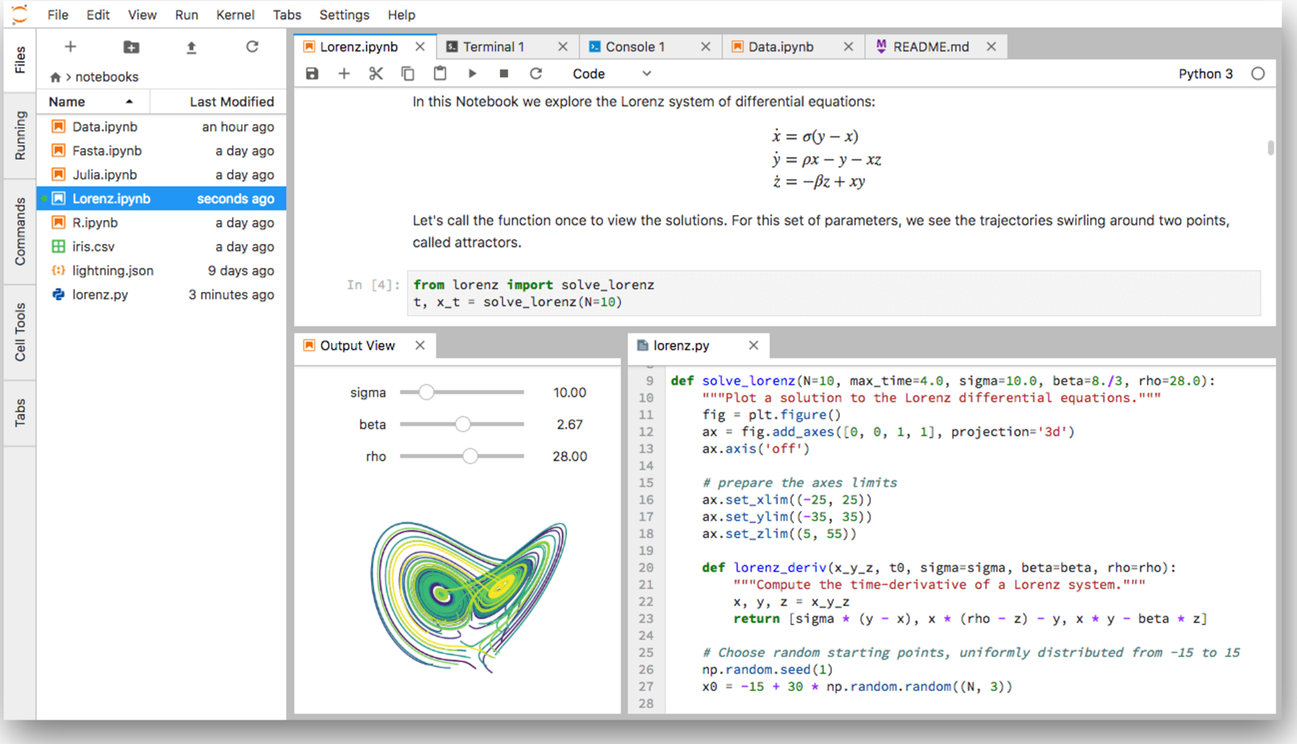
Lab IV. Deep Learning (30 minutes)

We will learn how to use Keras to create and train a simple image classification model with deep neural network (DNN).

Lab III Machine Learning (30 minutes)

We will learn to use scikit-learn for linear regression and classification applications.

Lab I. JupyterLab



notebooks

Name	Last Modified
Data.ipynb	an hour ago
Fasta.ipynb	a day ago
Julia.ipynb	a day ago
Lorenz.ipynb	seconds ago
R.ipynb	a day ago
iris.csv	a day ago
lightning.json	9 days ago
lorenz.py	3 minutes ago

In this Notebook we explore the Lorenz system of differential equations:

$$\begin{aligned} \dot{x} &= \sigma(y - x) \\ \dot{y} &= \rho x - y - xz \\ \dot{z} &= -\beta z + xy \end{aligned}$$

Let's call the function once to view the solutions. For this set of parameters, we see the trajectories swirling around two points, called attractors.

```
In [4]:  from lorenz import solve_lorenz
         t, x_t = solve_lorenz(N=10)
```

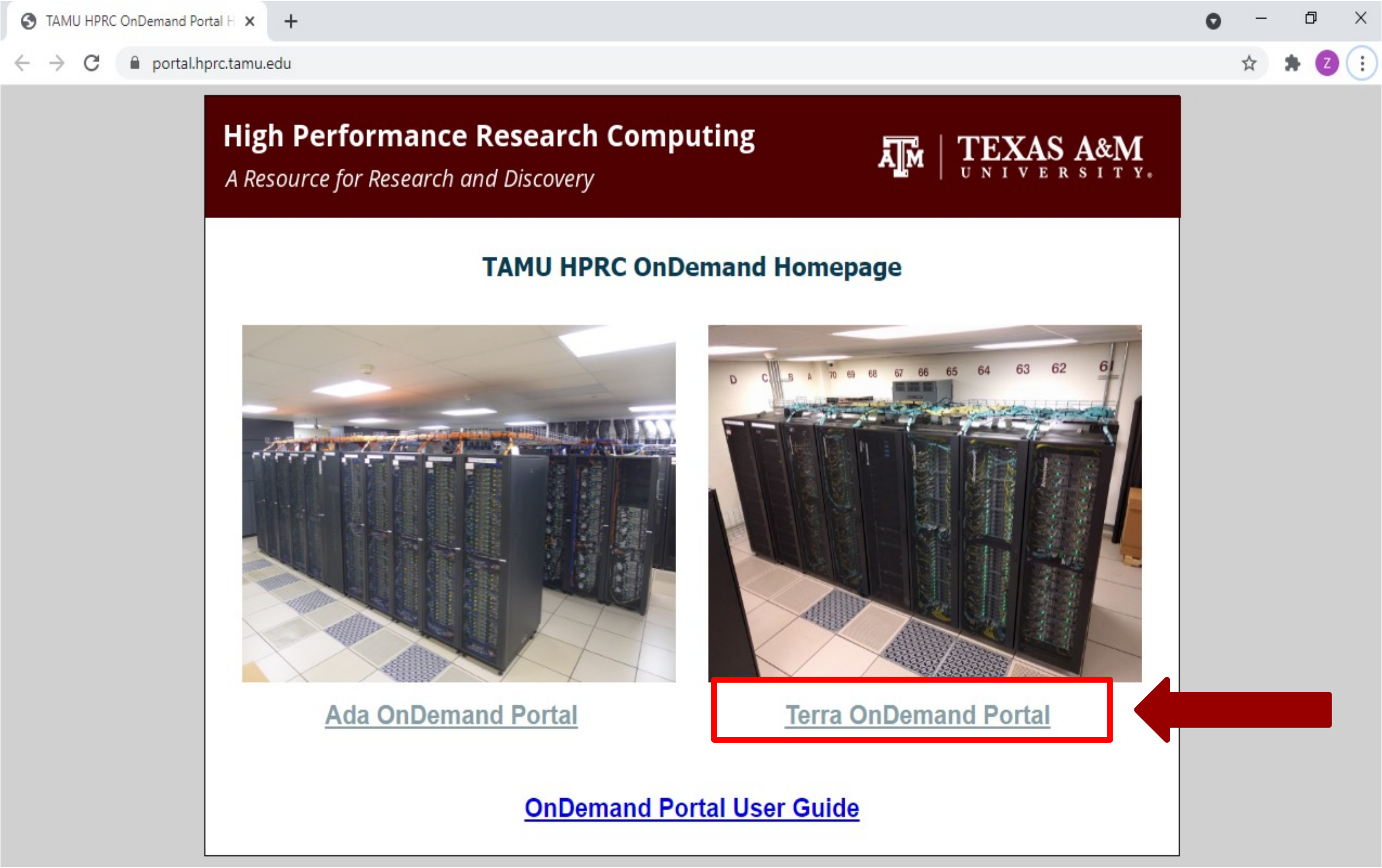
```
sigma   10.00
beta    2.67
rho     28.00
```

```
def solve_lorenz(N=10, max_time=4.0, sigma=10.0, beta=8./3, rho=28.0):
    """Plot a solution to the Lorenz differential equations."""
    fig = plt.figure()
    ax = fig.add_axes([0, 0, 1, 1], projection='3d')
    ax.axis('off')

    # prepare the axes limits
    ax.set_xlim((-25, 25))
    ax.set_ylim((-35, 35))
    ax.set_zlim((5, 55))

    def lorenz_deriv(x_y_z, t0, sigma=sigma, beta=beta, rho=rho):
        """Compute the time-derivative of a Lorenz system."""
        x, y, z = x_y_z
        return [sigma * (y - x), x * (rho - z) - y, x * y - beta * z]

    # Choose random starting points, uniformly distributed from -15 to 15
    np.random.seed(1)
    x0 = -15 + 30 * np.random.random((N, 3))

```

L1 - Resources

- Texas A&M High Performance Research Computing (HPRC)
- Terra Quick Start Guide
- HPRC Portal
- HPRC YouTube Channel
- Jupyter Project

Login HPRC Portal



TAMU HPRC OnDemand Portal H

portal.hprc.tamu.edu

High Performance Research Computing

A Resource for Research and Discovery

TEXAS A&M UNIVERSITY

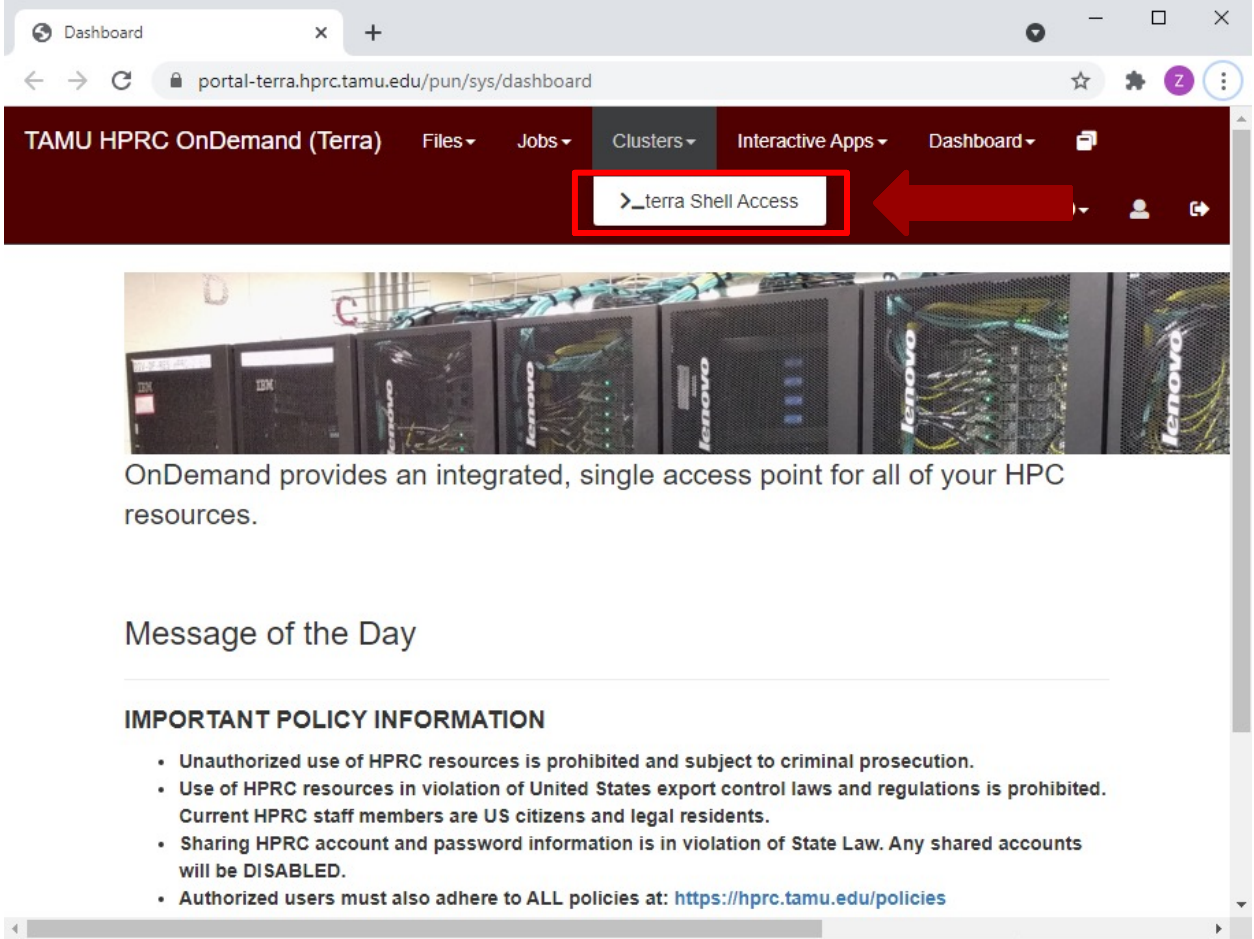
TAMU HPRC OnDemand Homepage

Ada OnDemand Portal

Terra OnDemand Portal

OnDemand Portal User Guide

Shell Access - I



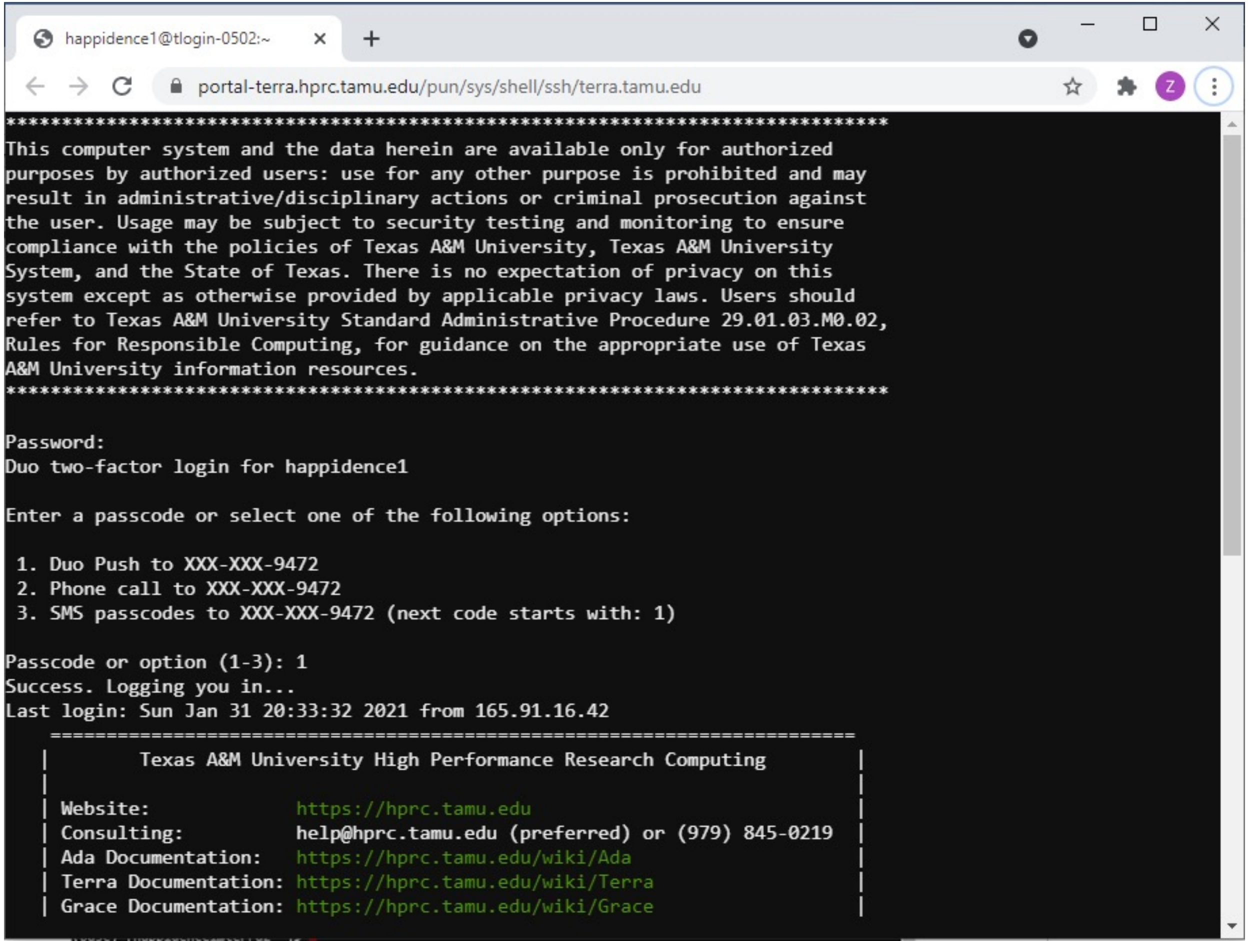
OnDemand provides an integrated, single access point for all of your HPC resources.

Message of the Day

IMPORTANT POLICY INFORMATION

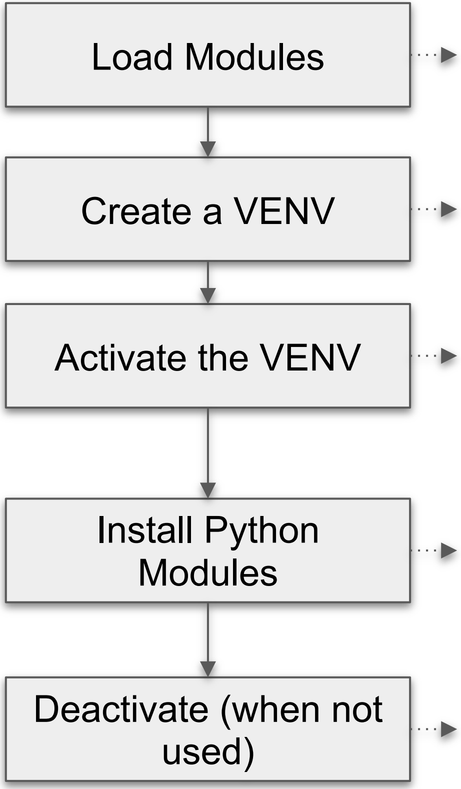
- Unauthorized use of HPRC resources is prohibited and subject to criminal prosecution.
- Use of HPRC resources in violation of United States export control laws and regulations is prohibited. Current HPRC staff members are US citizens and legal residents.
- Sharing HPRC account and password information is in violation of State Law. Any shared accounts will be DISABLED.
- Authorized users must also adhere to ALL policies at: https://hprc.tamu.edu/policies

Shell Access - II


happidence1@tlogin-0502:~
portal-terra.hprc.tamu.edu/pun/sys/shell/ssh/terra.tamu.edu
Z

```
*******************************************************************************
This computer system and the data herein are available only for authorized
purposes by authorized users: use for any other purpose is prohibited and may
result in administrative/disciplinary actions or criminal prosecution against
the user. Usage may be subject to security testing and monitoring to ensure
compliance with the policies of Texas A&M University, Texas A&M University
System, and the State of Texas. There is no expectation of privacy on this
system except as otherwise provided by applicable privacy laws. Users should
refer to Texas A&M University Standard Administrative Procedure 29.01.03.M0.02,
Rules for Responsible Computing, for guidance on the appropriate use of Texas
A&M University information resources.
*******************************************************************************


Password:
Duo two-factor login for happidence1

Enter a passcode or select one of the following options:

 1. Duo Push to XXX-XXX-9472
 2. Phone call to XXX-XXX-9472
 3. SMS passcodes to XXX-XXX-9472 (next code starts with: 1)

Passcode or option (1-3): 1
Success. Logging you in...
Last login: Sun Jan 31 20:33:32 2021 from 165.91.16.42
    ==========================================================================
   |        Texas A&M University High Performance Research Computing          |
   |                                                                          |
   | Website:              https://hprc.tamu.edu                              |
   | Consulting:           help@hprc.tamu.edu (preferred) or (979) 845-0219   |
   | Ada Documentation:    https://hprc.tamu.edu/wiki/Ada                     |
   | Terra Documentation:  https://hprc.tamu.edu/wiki/Terra                   |
   | Grace Documentation:  https://hprc.tamu.edu/wiki/Grace                   |
```

Python Virtual Environment (VENV)



Load Modules

Create a VENV

Activate the VENV

Install Python Modules

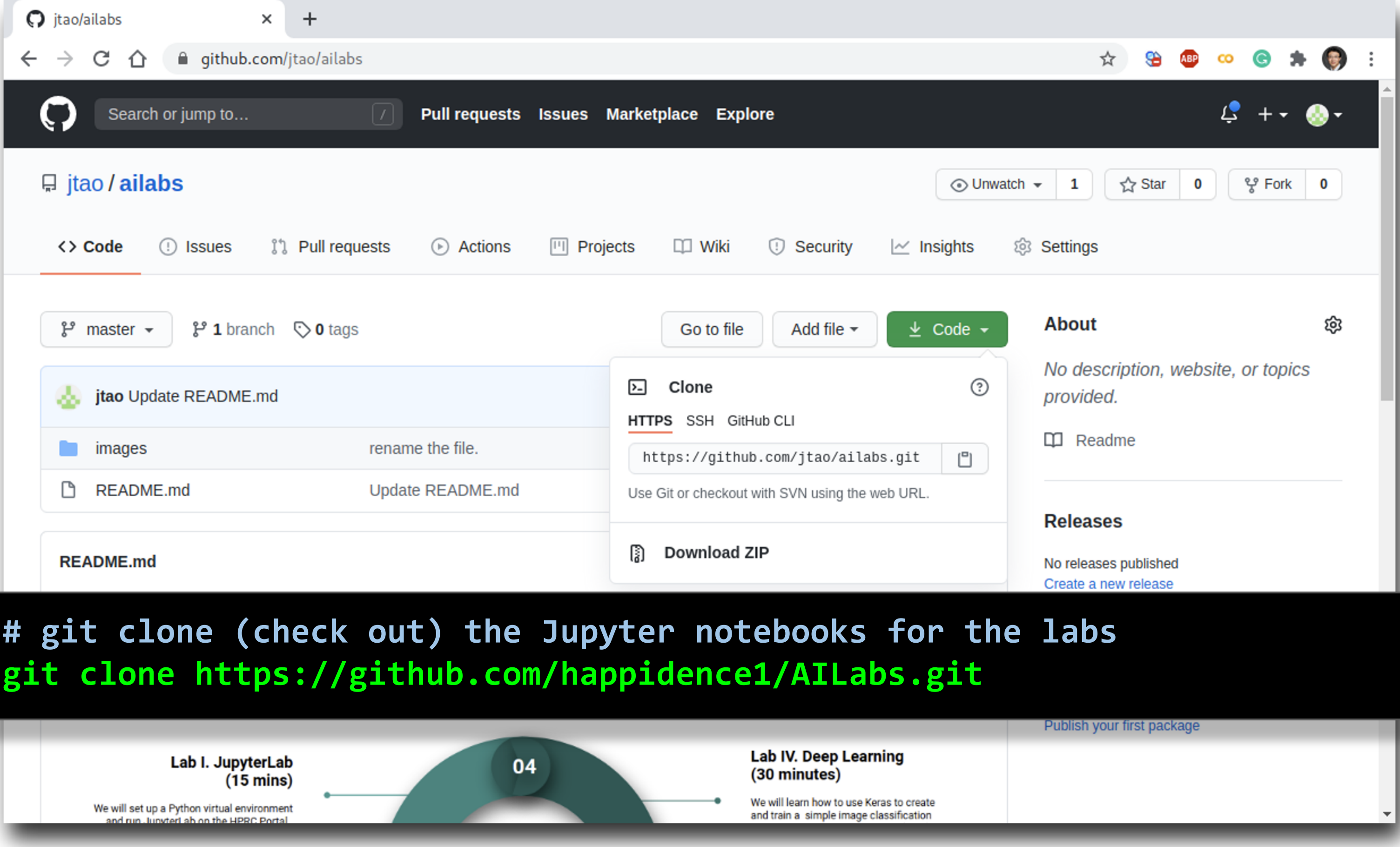
Deactivate (when not used)

```
# clean up and load Anaconda
cd $SCRATCH
module purge
module load Anaconda/3-5.0.0.1

# create a Python virtual environment
conda create -n mylab

# activate the virtual environment
source activate mylab

# install required package to be used in the portal
conda install -c conda-forge jupyterlab=1.2.2
conda install pandas matplotlib
conda install scikit-learn
conda install tensorflow

# deactivate the virtual environment
# source deactivate
```

Common Anaconda Commands

```
# Conda virtual environment
conda info                           # show Conda installation
conda create -n VENV                 # create a virtual environment
conda create -n VENV python=3.4      # create a venv with a py version
conda env list                       # list installed venv

# Conda package management
conda list                           # list all installed packages
conda search  PACKAGENAME            # search a Conda package
conda install PACKAGENAME            # install a Conda package
conda update  PACKAGENAME            # update a Conda package
conda remove  PACKAGENAME            # remove a Conda package
```

Check out Exercises



jtao Update README.md

images	rename the file.
README.md	Update README.md

README.md

https://github.com/jtao/ailabs.git

Use Git or checkout with SVN using the web URL.

No description, website, or topics provided.

No releases published

Lab I. JupyterLab (15 mins)

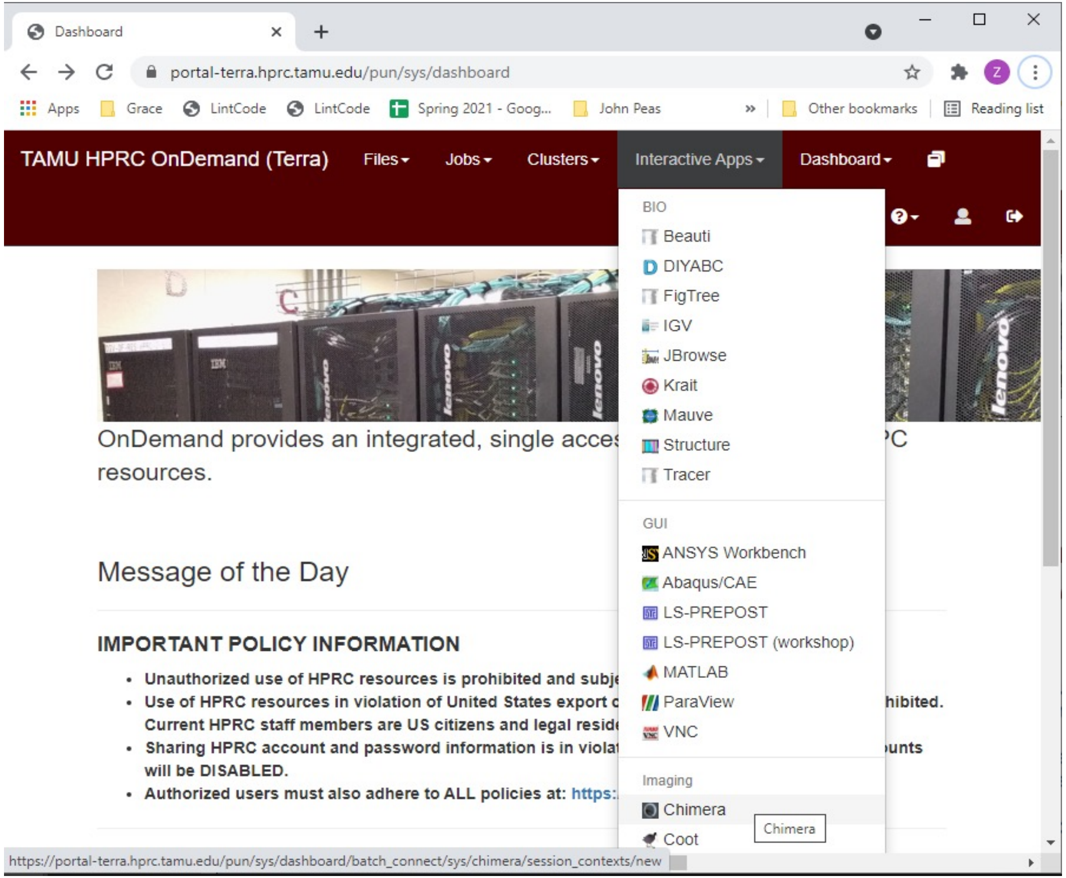
We will set up a Python virtual environment and run JupyterLab on the HPRC Portal

Lab IV. Deep Learning (30 minutes)

We will learn how to use Keras to create and train a simple image classification

```
# git clone (check out) the Jupyter notebooks for the labs
git clone https://github.com/happidence1/AILabs.git
```

Go to JupyterLab Page

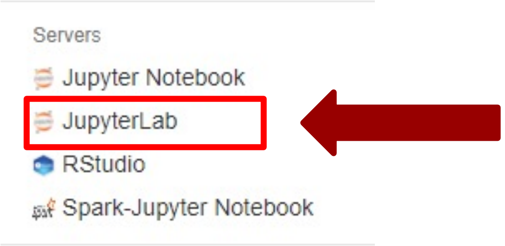


OnDemand provides an integrated, single acces PC resources.

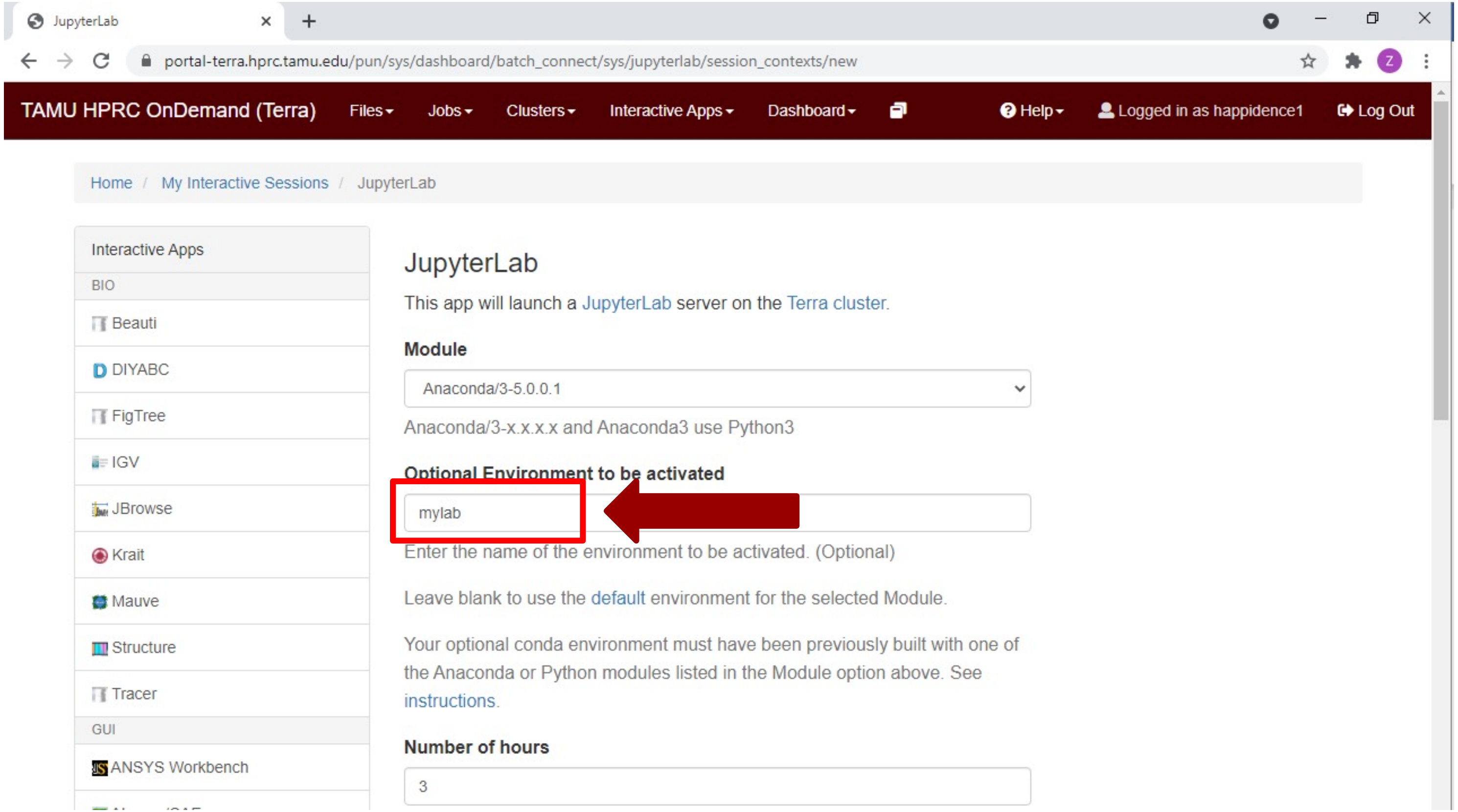
Message of the Day

IMPORTANT POLICY INFORMATION

- Unauthorized use of HPRC resources is prohibited and subj
- Use of HPRC resources in violation of United States export c hibited. Current HPRC staff members are US citizens and legal reside
- Sharing HPRC account and password information is in violat ounts will be DISABLED.
- Authorized users must also adhere to ALL policies at: https:



Set Virtual Environment



JupyterLab

This app will launch a JupyterLab server on the Terra cluster.

Module

Anaconda/3-5.0.0.1

Anaconda/3-x.x.x.x and Anaconda3 use Python3

Optional Environment to be activated

mylab

Enter the name of the environment to be activated. (Optional)

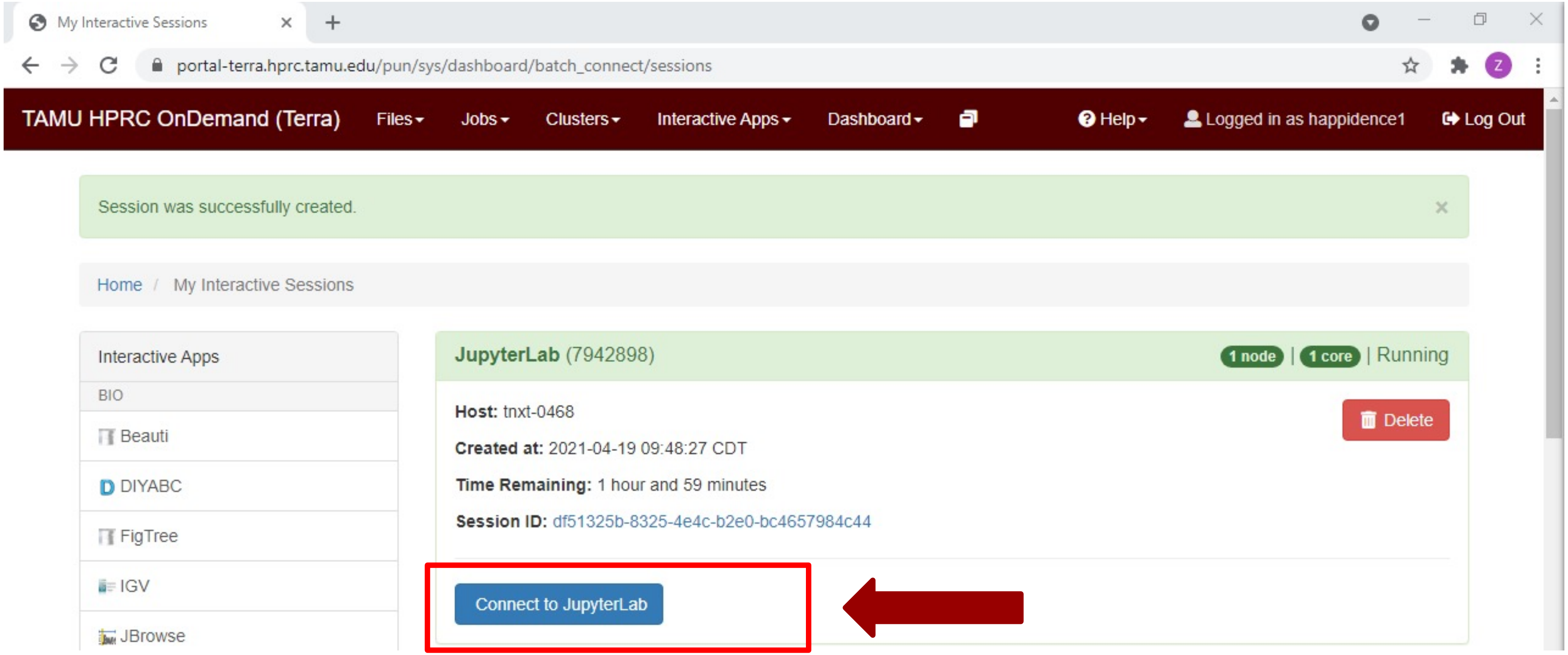
Leave blank to use the default environment for the selected Module.

Your optional conda environment must have been previously built with one of the Anaconda or Python modules listed in the Module option above. See instructions.

Number of hours

3

Connect to JupyterLab



Session was successfully created.

JupyterLab (7942898) 1 node | 1 core | Running

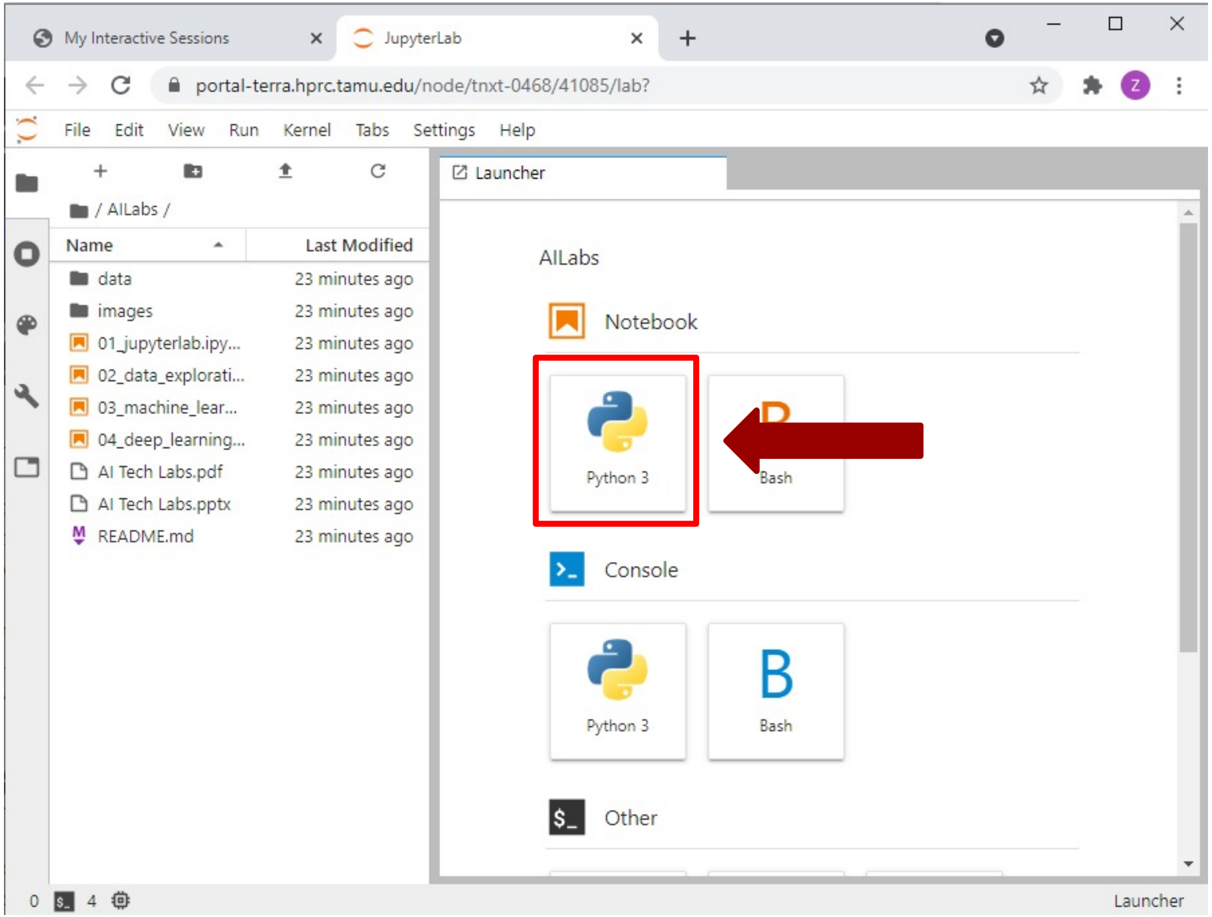
Host: tnxt-0468

Created at: 2021-04-19 09:48:27 CDT

Time Remaining: 1 hour and 59 minutes

Session ID: df51325b-8325-4e4c-b2e0-bc4657984c44

Create a Jupyter Notebook

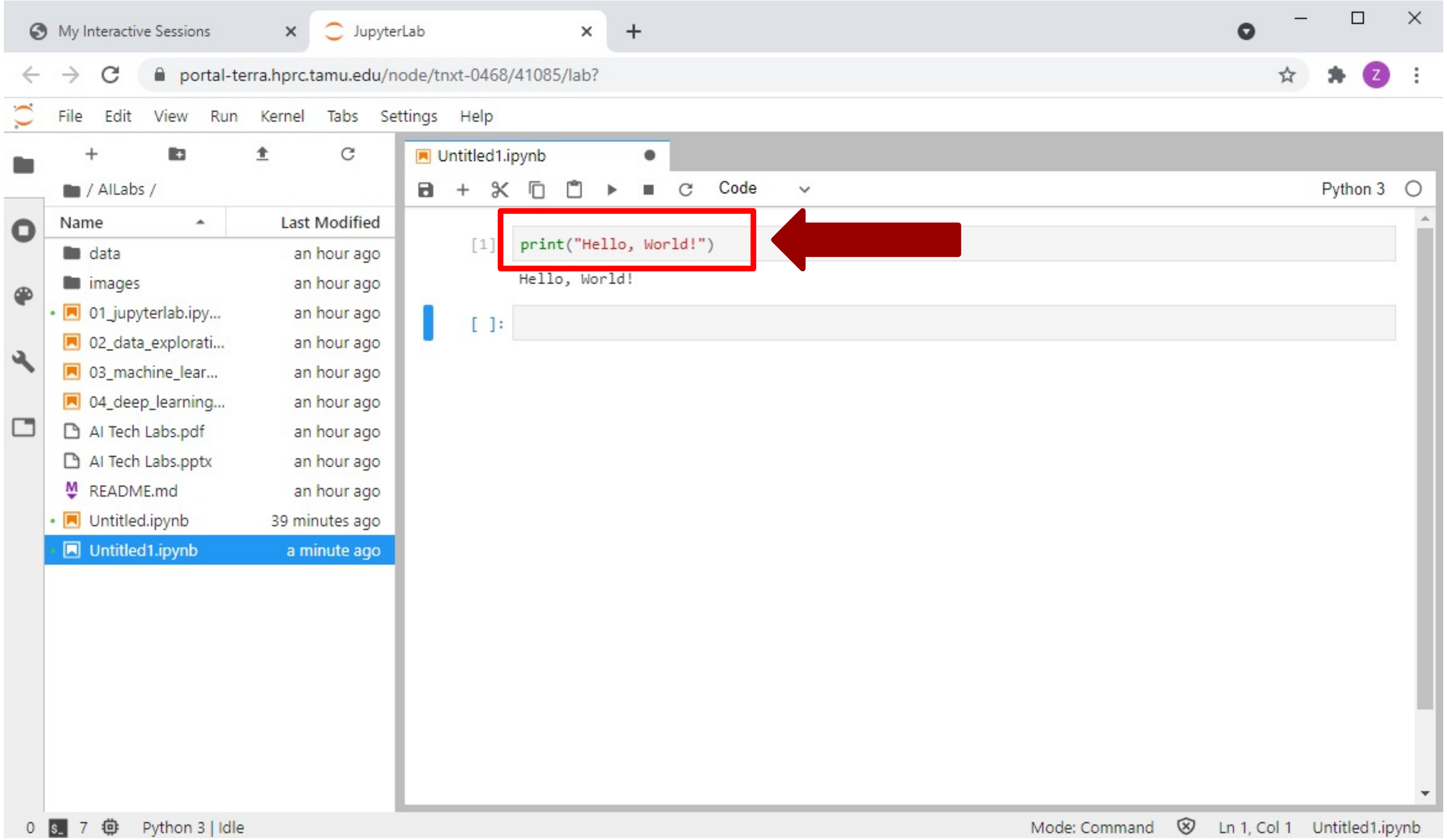


portal-terra.hprc.tamu.edu/node/tnxt-0468/41085/lab?

/ AILabs /

Name	Last Modified
data	23 minutes ago
images	23 minutes ago
01_jupyterlab.ipy...	23 minutes ago
02_data_explorati...	23 minutes ago
03_machine_lear...	23 minutes ago
04_deep_learning...	23 minutes ago
AI Tech Labs.pdf	23 minutes ago
AI Tech Labs.pptx	23 minutes ago
README.md	23 minutes ago

Test JupyterLab



Name	Last Modified
data	an hour ago
images	an hour ago
01_jupyterlab.ipy...	an hour ago
02_data_explorati...	an hour ago
03_machine_lear...	an hour ago
04_deep_learning...	an hour ago
AI Tech Labs.pdf	an hour ago
AI Tech Labs.pptx	an hour ago
README.md	an hour ago
Untitled.ipynb	39 minutes ago
Untitled1.ipynb	a minute ago

```
[1]: print("Hello, World!")
Hello, World!
[ ]:
```

Lab II. Data Exploration

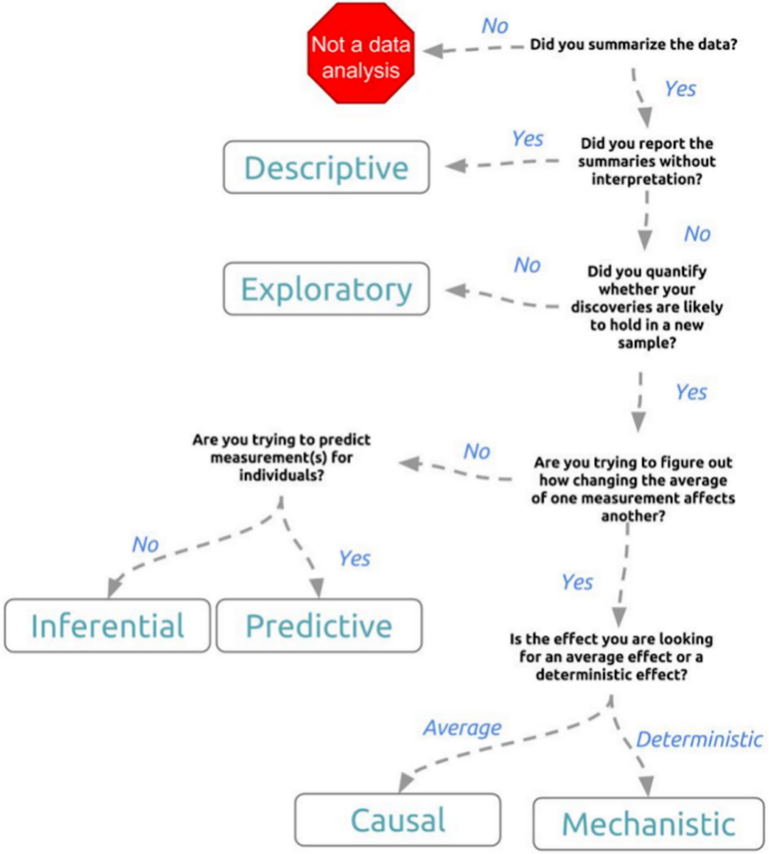
matplotlib

pandas

$y_{it} = \beta' x_{it} + \mu_i + \epsilon_{it}$

Types of Data Science Problems

- **Descriptive** (summaries, e.g., census)
- **Exploratory** (search for unknowns, e.g., four-planet solar system)
- **Inferential** (find correlations, e.g., many social studies)
- **Predictive** (make predictions, e.g., Face ID, Echo, Siri)
- **Causal** (explore causation, e.g., smoking versus lung cancer)
- **Mechanistic** (determine governing principles, e.g., experimental science)


Did you summarize the data?
No
Not a data analysis
Yes
Did you report the summaries without interpretation?
Yes
Descriptive
No
Did you quantify whether your discoveries are likely to hold in a new sample?
No
Exploratory
Yes
Are you trying to figure out how changing the average of one measurement affects another?
No
Are you trying to predict measurement(s) for individuals?
No
Yes
Inferential
Predictive
Yes
Is the effect you are looking for an average effect or a deterministic effect?
Average
Deterministic
Causal
Mechanistic

Credit: Jeff Leek - The Elements of Data Analytic Style

Data Structures

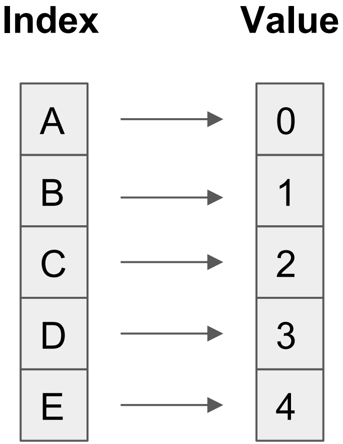
Pandas has two data structures that are descriptive and optimized for data with different dimensions.

- **Series:** 1D labeled homogeneously-typed array
- **DataFrame:** General 2D labeled, size-mutable tabular structure with potentially heterogeneously-typed columns

Series in pandas

"Series is a one-dimensional labeled array capable of holding any data type (integers, strings, floating point numbers, Python objects, etc.). The axis labels are collectively referred to as the index." - pandas site

```
In [3]: s = pd.Series(np.random.randn(5),
                 index=['a', 'b', 'c', 'd', 'e'])
In [5]: s.index
In [6]: pd.Series(np.random.randn(5))
In [7]: d = {'b': 1, 'a': 0, 'c': 2}
In [8]: pd.Series(d)
```

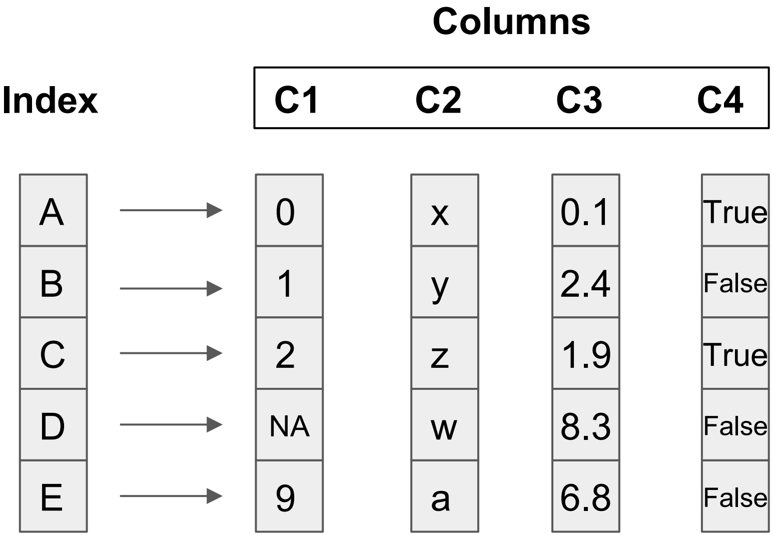


Index	Value
A	0
B	1
C	2
D	3
E	4

DataFrame in pandas

"Two-dimensional size-mutable, potentially heterogeneous tabular data structure with labeled axes (rows and columns). Arithmetic operations align on both row and column labels. Can be thought of as a dict-like container for Series objects. The primary pandas data structure." - pandas site

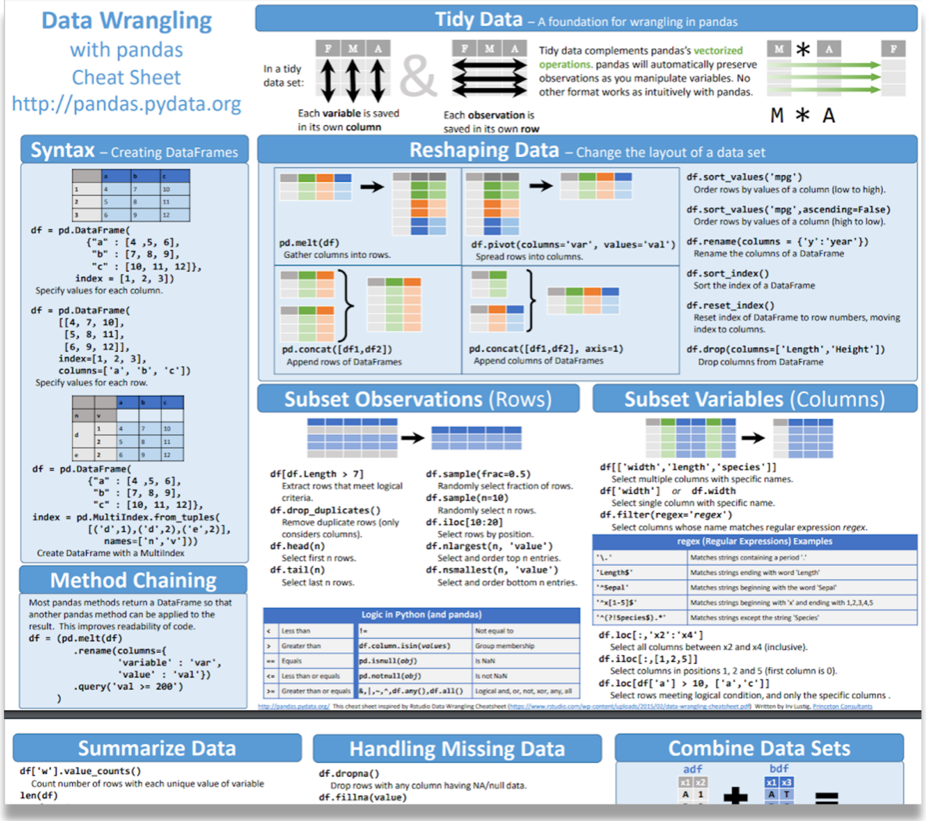
```
In [2]: d = {'col1': [1, 2], 'col2': [3, 4]}
In [3]: df = pd.DataFrame(data=d)
In [5]: df.index
In [6]: df =  pd.DataFrame(
          np.array([[1, 2, 3], [4, 5, 6], [7, 8, 9]]),
          columns=['a', 'b', 'c'])
```



Columns

Index	C1	C2	C3	C4
A	0	x	0.1	True
B	1	y	2.4	False
C	2	z	1.9	True
D	NA	w	8.3	False
E	9	a	6.8	False

Pandas Cheat Sheet



Data Wrangling with pandas Cheat Sheet http://pandas.pydata.org

Syntax – Creating DataFrames

	a	b	c
1	4	7	10
2	5	8	11
3	6	9	12

```
df = pd.DataFrame(
          {"a" : [4 ,5, 6],
           "b" : [7, 8, 9],
           "c" : [10, 11, 12]},
        index = [1, 2, 3])
```

Specify values for each column.

```
df = pd.DataFrame(
     [[4, 7, 10],
      [5, 8, 11],
      [6, 9, 12]],
     index=[1, 2, 3],
     columns=['a', 'b', 'c'])
```

Specify values for each row.

		a	b	c
n	v			
d	1	4	7	10
	2	5	8	11
e	2	6	9	12

```
df = pd.DataFrame(
          {"a" : [4 ,5, 6],
           "b" : [7, 8, 9],
           "c" : [10, 11, 12]},
index = pd.MultiIndex.from_tuples(
          [('d',1),('d',2),('e',2)],
             names=['n','v']))
```

Create DataFrame with a MultiIndex

Method Chaining

Most pandas methods return a DataFrame so that another pandas method can be applied to the result. This improves readability of code.

```
df = (pd.melt(df)
        .rename(columns={
                'variable' : 'var',
                'value' : 'val'})
        .query('val >= 200')
     )
```

Tidy Data – A foundation for wrangling in pandas

Each **variable** is saved in its own **column**

Each **observation** is saved in its own **row**

Tidy data complements pandas's vectorized operations. pandas will automatically preserve observations as you manipulate variables. No other format works as intuitively with pandas.

Reshaping Data – Change the layout of a data set

`pd.melt(df)`
Gather columns into rows.

`df.pivot(columns='var', values='val')`
Spread rows into columns.

`pd.concat([df1,df2])`
Append rows of DataFrames

`pd.concat([df1,df2], axis=1)`
Append columns of DataFrames

`df.sort_values('mpg')`
Order rows by values of a column (low to high).

`df.sort_values('mpg',ascending=False)`
Order rows by values of a column (high to low).

`df.rename(columns = {'y':'year'})`
Rename the columns of a DataFrame

`df.sort_index()`
Sort the index of a DataFrame

`df.reset_index()`
Reset index of DataFrame to row numbers, moving index to columns.

`df.drop(columns=['Length','Height'])`
Drop columns from DataFrame

Subset Observations (Rows)

`df[df.Length > 7]`
Extract rows that meet logical criteria.

`df.drop_duplicates()`
Remove duplicate rows (only considers columns).

`df.head(n)`
Select first n rows.

`df.tail(n)`
Select last n rows.

`df.sample(frac=0.5)`
Randomly select fraction of rows.

`df.sample(n=10)`
Randomly select n rows.

`df.iloc[10:20]`
Select rows by position.

`df.nlargest(n, 'value')`
Select and order top n entries.

`df.nsmallest(n, 'value')`
Select and order bottom n entries.

Logic in Python (and pandas)			
<	Less than	!=	Not equal to
>	Greater than	df.column.isin(*values*)	Group membership
==	Equals	pd.isnull(*obj*)	Is NaN
<=	Less than or equals	pd.notnull(*obj*)	Is not NaN
>=	Greater than or equals	&,|,~,^,df.any(),df.all()	Logical and, or, not, xor, any, all

Subset Variables (Columns)

`df[['width','length','species']]`
Select multiple columns with specific names.

`df['width']` *or* `df.width`
Select single column with specific name.

`df.filter(regex='regex')`
Select columns whose name matches regular expression *regex*.

regex (Regular Expressions) Examples	
'\.'	Matches strings containing a period '.'
'Length$'	Matches strings ending with word 'Length'
'^Sepal'	Matches strings beginning with the word 'Sepal'
'^x[1-5]$'	Matches strings beginning with 'x' and ending with 1,2,3,4,5
'^(?!Species$).*'	Matches strings except the string 'Species'

`df.loc[:,'x2':'x4']`
Select all columns between x2 and x4 (inclusive).

`df.iloc[:,[1,2,5]]`
Select columns in positions 1, 2 and 5 (first column is 0).

`df.loc[df['a'] > 10, ['a','c']]`
Select rows meeting logical condition, and only the specific columns .

http://pandas.pydata.org/ This cheat sheet inspired by Rstudio Data Wrangling Cheatsheet (https://www.rstudio.com/wp-content/uploads/2015/02/data-wrangling-cheatsheet.pdf) Written by Irv Lustig, Princeton Consultants

Summarize Data

`df['w'].value_counts()`
Count number of rows with each unique value of variable

`len(df)`

Handling Missing Data

`df.dropna()`
Drop rows with any column having NA/null data.

`df.fillna(value)`

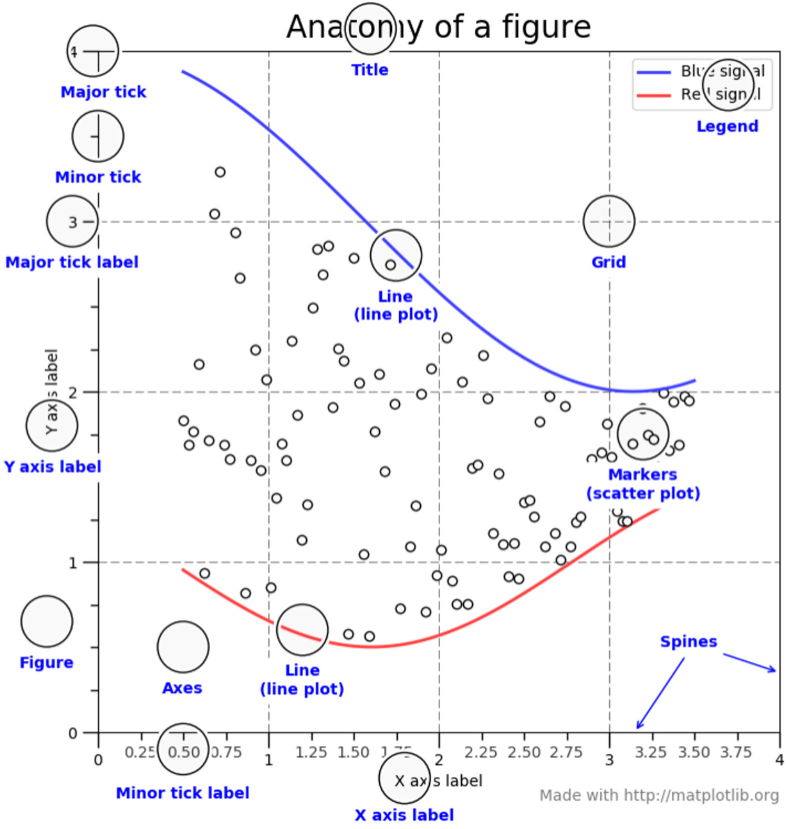
Combine Data Sets

https://pandas.pydata.org/Pandas_Cheat_Sheet.pdf

Key Plotting Concepts in Matplotlib

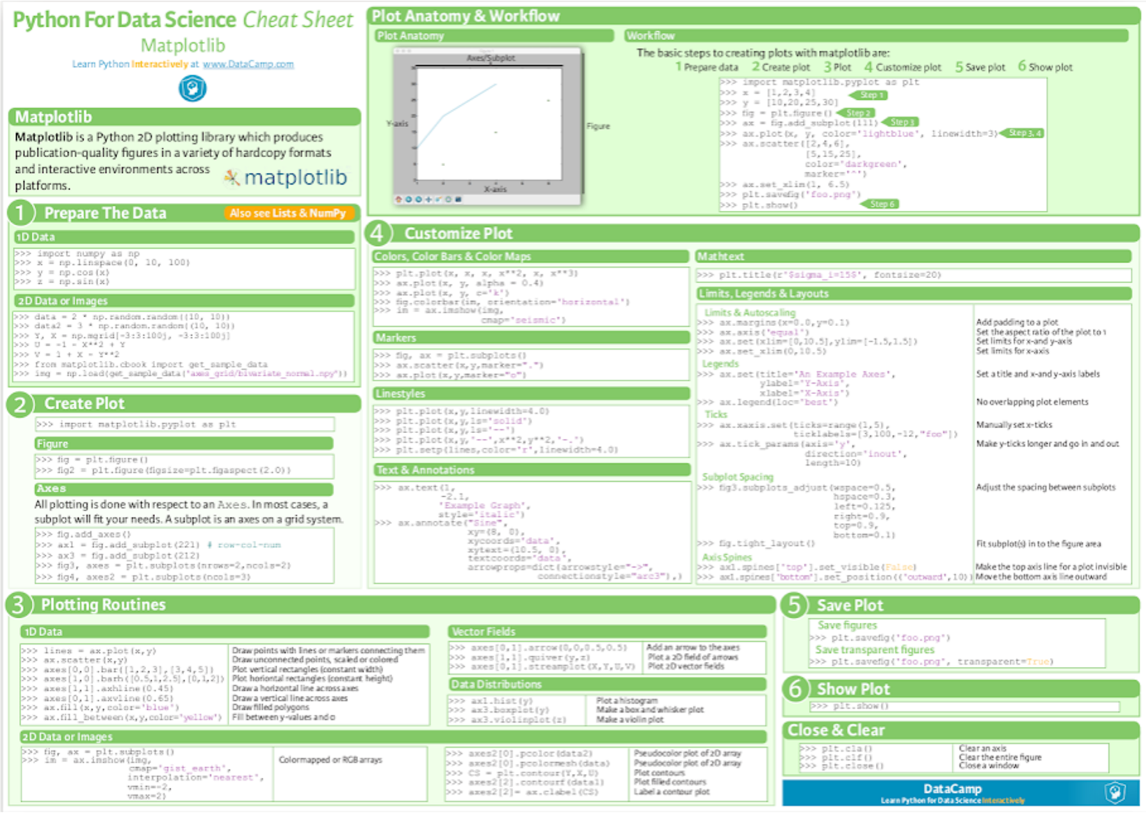
- **Matplotlib: Figure**
  Figure is the object that keeps the whole image output. Adjustable parameters include:
  1. Image size (set_size_inches())
  2. Whether to use tight_layout (set_tight_layout())

- **Matplotlib: Axes**
  Axes object represents the pair of axis that contain a single plot (x-axis and y-axis). The Axes object also has more adjustable parameters:
  1. The plot frame (set_frame_on() or set_frame_off())
  2. X-axis and Y-axis limits (set_xlim() and set_ylim())
  3. X-axis and Y-axis Labels (set_xlabel() and set_ylabel())
  4. The plot title (set_title())



(Credit: matplotlib.org)

Matplotlib Cheat Sheet



Python For Data Science *Cheat Sheet*

Matplotlib

Learn Python Interactively at www.DataCamp.com

Matplotlib

Matplotlib is a Python 2D plotting library which produces publication-quality figures in a variety of hardcopy formats and interactive environments across platforms.

matplotlib

1 Prepare The Data

Also see Lists & NumPy

1D Data

```
>>> import numpy as np
>>> x = np.linspace(0, 10, 100)
>>> y = np.cos(x)
>>> z = np.sin(x)
```

2D Data or Images

```
>>> data = 2 * np.random.random((10, 10))
>>> data2 = 3 * np.random.random((10, 10))
>>> Y, X = np.mgrid[-3:3:100j, -3:3:100j]
>>> U = -1 - X**2 + Y
>>> V = 1 + X - Y**2
>>> from matplotlib.cbook import get_sample_data
>>> img = np.load(get_sample_data('axes_grid/bivariate_normal.npy'))
```

2 Create Plot

```
>>> import matplotlib.pyplot as plt
```

Figure

```
>>> fig = plt.figure()
>>> fig2 = plt.figure(figsize=plt.figaspect(2.0))
```

Axes

All plotting is done with respect to an Axes. In most cases, a subplot will fit your needs. A subplot is an axes on a grid system.

```
>>> fig.add_axes()
>>> ax1 = fig.add_subplot(221) # row-col-num
>>> ax3 = fig.add_subplot(212)
>>> fig3, axes = plt.subplots(nrows=2,ncols=2)
>>> fig4, axes2 = plt.subplots(ncols=3)
```

Plot Anatomy & Workflow

Plot Anatomy

Workflow

The basic steps to creating plots with matplotlib are:

1 Prepare data 2 Create plot 3 Plot 4 Customize plot 5 Save plot 6 Show plot

```
>>> import matplotlib.pyplot as plt
>>> x = [1,2,3,4]                                        Step 1
>>> y = [10,20,25,30]
>>> fig = plt.figure()                                   Step 2
>>> ax = fig.add_subplot(111)                            Step 3
>>> ax.plot(x, y, color='lightblue', linewidth=3)        Step 3, 4
>>> ax.scatter([2,4,6],
               [5,15,25],
               color='darkgreen',
               marker='^')
>>> ax.set_xlim(1, 6.5)
>>> plt.savefig('foo.png')
>>> plt.show()                                           Step 6
```

4 Customize Plot

Colors, Color Bars & Color Maps

```
>>> plt.plot(x, x, x, x**2, x, x**3)
>>> ax.plot(x, y, alpha = 0.4)
>>> ax.plot(x, y, c='k')
>>> fig.colorbar(im, orientation='horizontal')
>>> im = ax.imshow(img,
                   cmap='seismic')
```

Markers

```
>>> fig, ax = plt.subplots()
>>> ax.scatter(x,y,marker=".")
>>> ax.plot(x,y,marker="o")
```

Linestyles

```
>>> plt.plot(x,y,linewidth=4.0)
>>> plt.plot(x,y,ls='solid')
>>> plt.plot(x,y,ls='--')
>>> plt.plot(x,y,'--',x**2,y**2,'-.')
>>> plt.setp(lines,color='r',linewidth=4.0)
```

Text & Annotations

```
>>> ax.text(1,
            -2.1,
            'Example Graph',
            style='italic')
>>> ax.annotate("Sine",
                xy=(8, 0),
                xycoords='data',
                xytext=(10.5, 0),
                textcoords='data',
                arrowprops=dict(arrowstyle="->",
                                connectionstyle="arc3"),)
```

Mathtext

```
>>> plt.title(r'$sigma_i=15$', fontsize=20)
```

Limits, Legends & Layouts

Limits & Autoscaling

`>>> ax.margins(x=0.0,y=0.1)`	Add padding to a plot
`>>> ax.axis('equal')`	Set the aspect ratio of the plot to 1
`>>> ax.set(xlim=[0,10.5],ylim=[-1.5,1.5])`	Set limits for x-and y-axis
`>>> ax.set_xlim(0,10.5)`	Set limits for x-axis

Legends

`>>> ax.set(title='An Example Axes', ylabel='Y-Axis', xlabel='X-Axis')`	Set a title and x-and y-axis labels
`>>> ax.legend(loc='best')`	No overlapping plot elements

Ticks

`>>> ax.xaxis.set(ticks=range(1,5), ticklabels=[3,100,-12,"foo"])`	Manually set x-ticks
`>>> ax.tick_params(axis='y', direction='inout', length=10)`	Make y-ticks longer and go in and out

Subplot Spacing

`>>> fig3.subplots_adjust(wspace=0.5, hspace=0.3, left=0.125, right=0.9, top=0.9, bottom=0.1)`	Adjust the spacing between subplots
`>>> fig.tight_layout()`	Fit subplot(s) in to the figure area

Axis Spines

`>>> ax1.spines['top'].set_visible(False)`	Make the top axis line for a plot invisible
`>>> ax1.spines['bottom'].set_position(('outward',10))`	Move the bottom axis line outward

3 Plotting Routines

1D Data

`>>> lines = ax.plot(x,y)`	Draw points with lines or markers connecting them
`>>> ax.scatter(x,y)`	Draw unconnected points, scaled or colored
`>>> axes[0,0].bar([1,2,3],[3,4,5])`	Plot vertical rectangles (constant width)
`>>> axes[1,0].barh([0.5,1,2.5],[0,1,2])`	Plot horiontal rectangles (constant height)
`>>> axes[1,1].axhline(0.45)`	Draw a horizontal line across axes
`>>> axes[0,1].axvline(0.65)`	Draw a vertical line across axes
`>>> ax.fill(x,y,color='blue')`	Draw filled polygons
`>>> ax.fill_between(x,y,color='yellow')`	Fill between y-values and 0

Vector Fields

`>>> axes[0,1].arrow(0,0,0.5,0.5)`	Add an arrow to the axes
`>>> axes[1,1].quiver(y,z)`	Plot a 2D field of arrows
`>>> axes[0,1].streamplot(X,Y,U,V)`	Plot 2D vector fields

Data Distributions

`>>> ax1.hist(y)`	Plot a histogram
`>>> ax3.boxplot(y)`	Make a box and whisker plot
`>>> ax3.violinplot(z)`	Make a violin plot

2D Data or Images

`>>> fig, ax = plt.subplots()`
`>>> im = ax.imshow(img, cmap='gist_earth', interpolation='nearest', vmin=-2, vmax=2)`
	Colormapped or RGB arrays

`>>> axes2[0].pcolor(data2)`	Pseudocolor plot of 2D array
`>>> axes2[0].pcolormesh(data)`	Pseudocolor plot of 2D array
`>>> CS = plt.contour(Y,X,U)`	Plot contours
`>>> axes2[2].contourf(data1)`	Plot filled contours
`>>> axes2[2]= ax.clabel(CS)`	Label a contour plot

5 Save Plot

Save figures

```
>>> plt.savefig('foo.png')
```

Save transparent figures

```
>>> plt.savefig('foo.png', transparent=True)
```

6 Show Plot

```
>>> plt.show()
```

Close & Clear

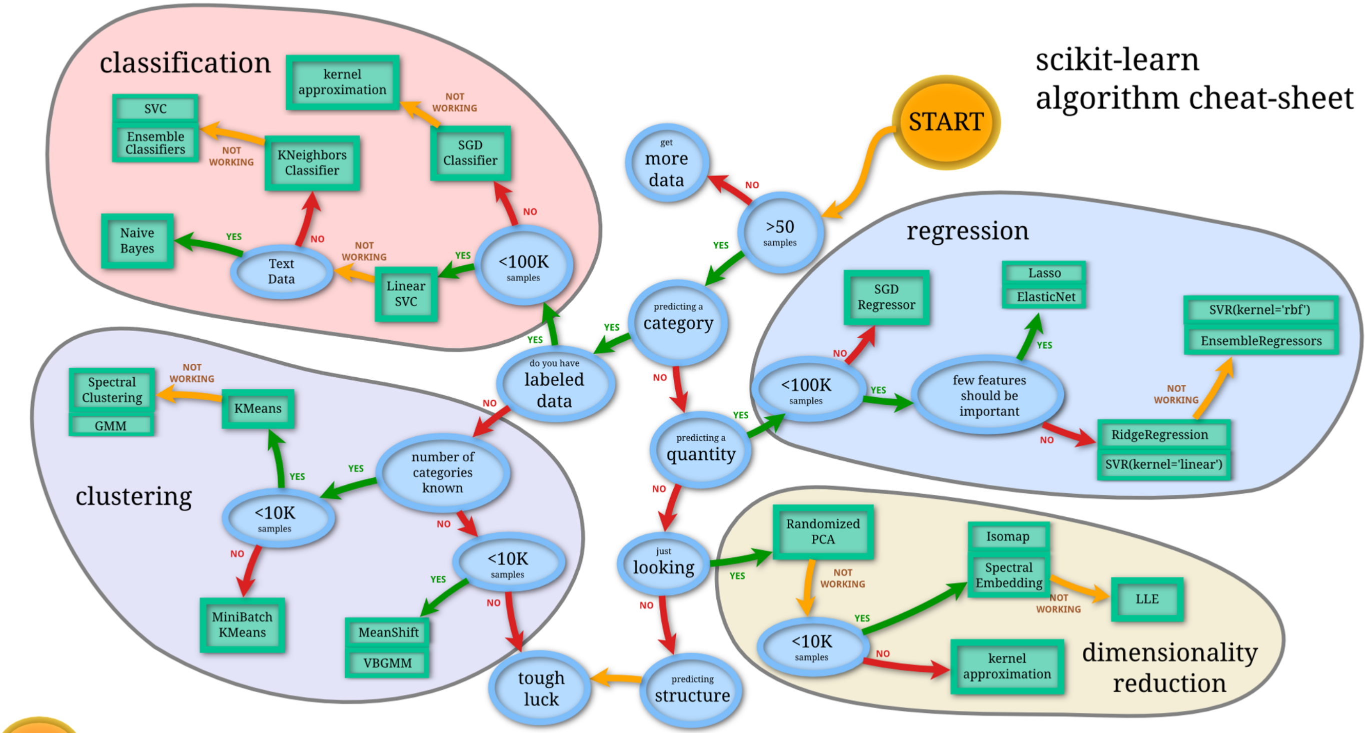
`>>> plt.cla()`	Clear an axis
`>>> plt.clf()`	Clear the entire figure
`>>> plt.close()`	Close a window

DataCamp

Learn Python for Data Science Interactively

https://s3.amazonaws.com/assets.datacamp.com/blog_assets/Python_Matplotlib_Cheat_Sheet.pdf

Lab III. Machine Learning



Back

Main Features of scikit-learn

Classification

Identifying category of an object

Applications: Spam detection, image recognition. **Algorithms**: SVM, nearest neighbors, random forest, and more...

Regression

Predicting a attribute for an object

Applications: Drug response, Stock prices. **Algorithms**: SVR, nearest neighbors, random forest, and more...

Clustering

Grouping similar objects into sets

Applications: Customer segmentation, Grouping experiment outcomes **Algorithms:** k-Means, spectral clustering, mean-shift, and more...

Dimension Reduction

Reducing the number of dimensions

Applications: Visualization, Increased efficiency **Algorithms:** k-Means, feature selection, non-negative matrix factorization, and more...

Model Selection

Selecting models with parameter search

Applications: Improved accuracy via parameter tuning **Algorithms:** grid search, cross validation, metrics, and more...

Preprocessing

Preprocessing data to prepare for modeling

Applications: Transforming input data such as text for use with machine learning algorithms. **Algorithms:** preprocessing, feature extraction, and more...

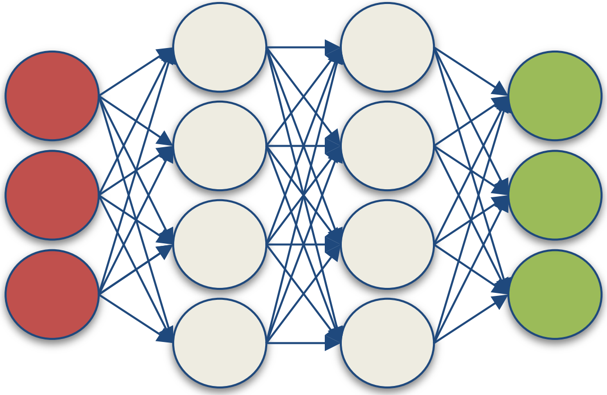
Credit: icons are from The Noun Project under Creative Commons Licenses

Lab IV. Deep Learning

Deep Learning
by Ian Goodfellow, Yoshua Bengio, and Aaron Courville
http://www.deeplearningbook.org/

Animation of Neutron Networks
by Grant Sanderson
https://www.3blue1brown.com/

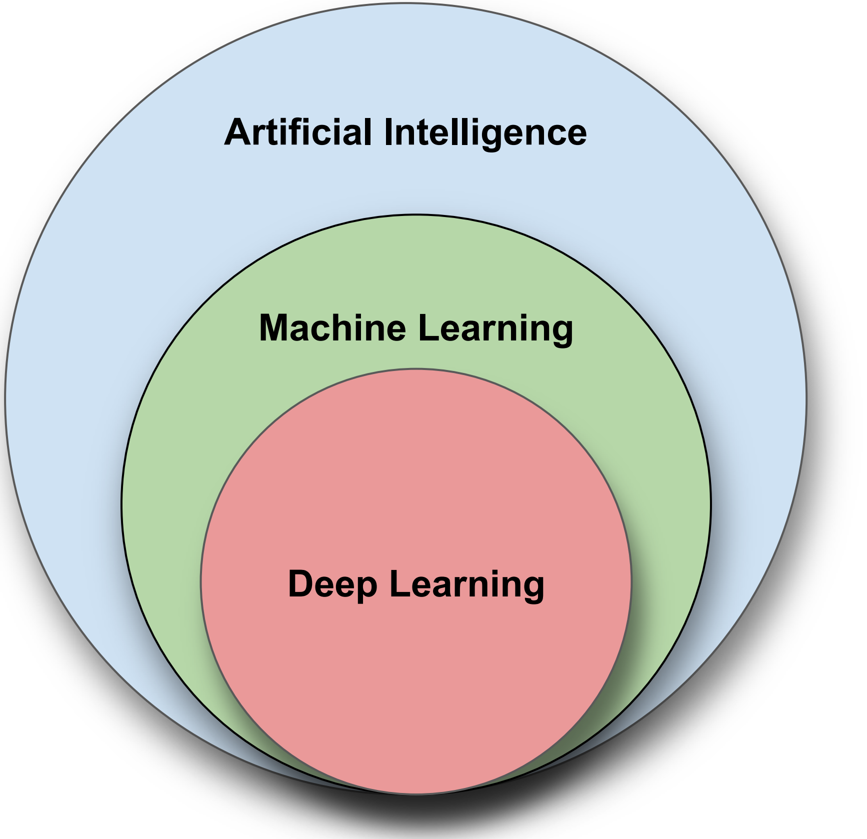
Visualization of CNN
by Adam Harley
https://www.cs.ryerson.ca/~aharley/vis/conv/



TensorFlow 2.0   Keras

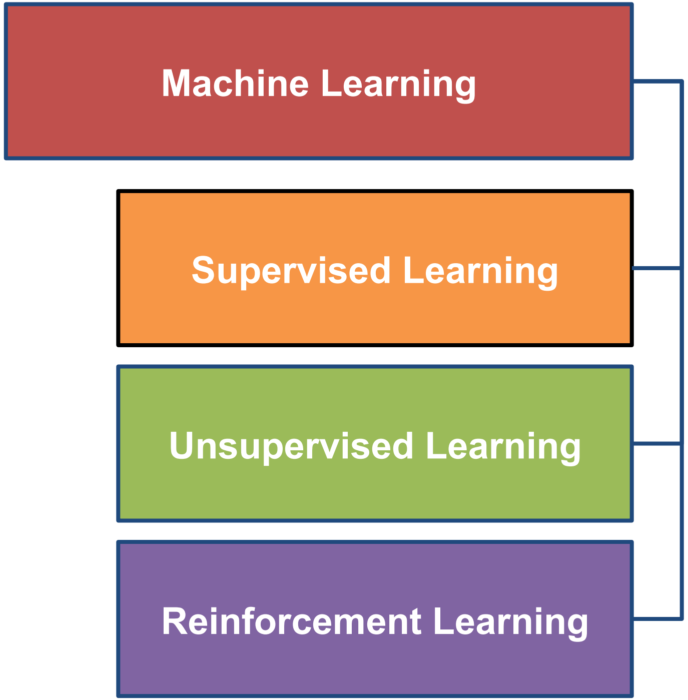
Relationship of AI, ML, and DL

- **Artificial Intelligence (AI)** is anything about man-made intelligence exhibited by machines.
- **Machine Learning (ML)** is an approach to achieve **AI**.
- **Deep Learning (DL)** is one technique to implement **ML**.

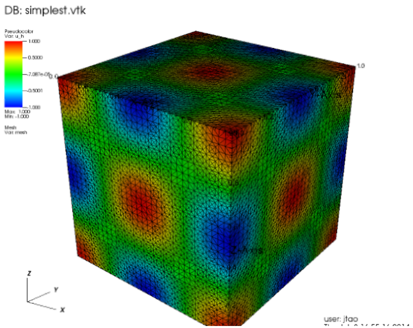


Types of ML Algorithms

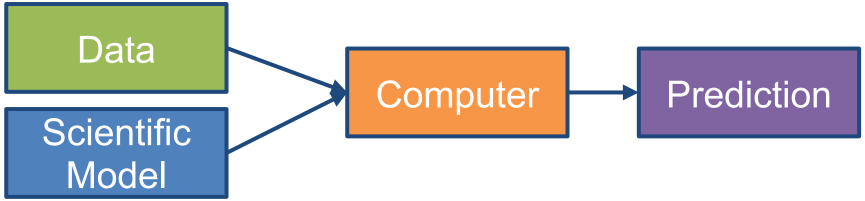
- **Supervised Learning**
  - trained with labeled data; including regression and classification problems
- **Unsupervised Learning**
  - trained with unlabeled data; clustering and association rule learning problems.
- **Reinforcement Learning**
  - no training data; stochastic Markov decision process; robotics and self-driving cars.

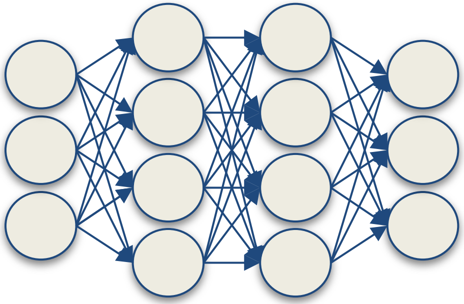


Machine Learning

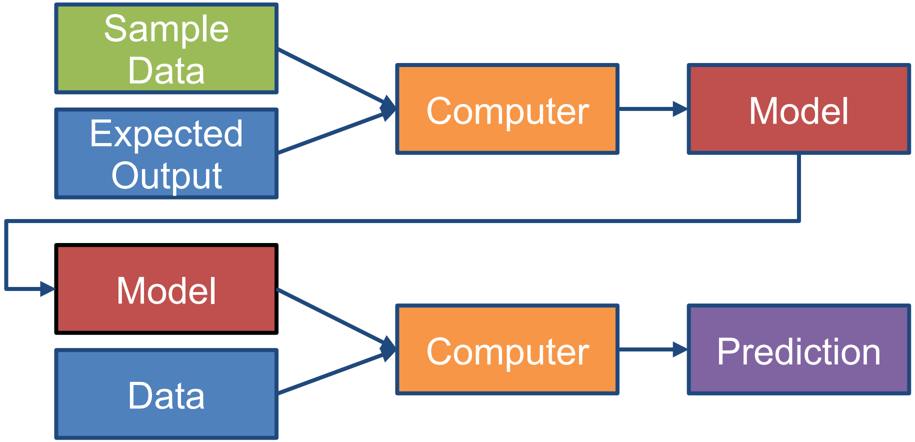


Traditional Modeling





Machine Learning (Supervised Learning)



Inputs and Outputs

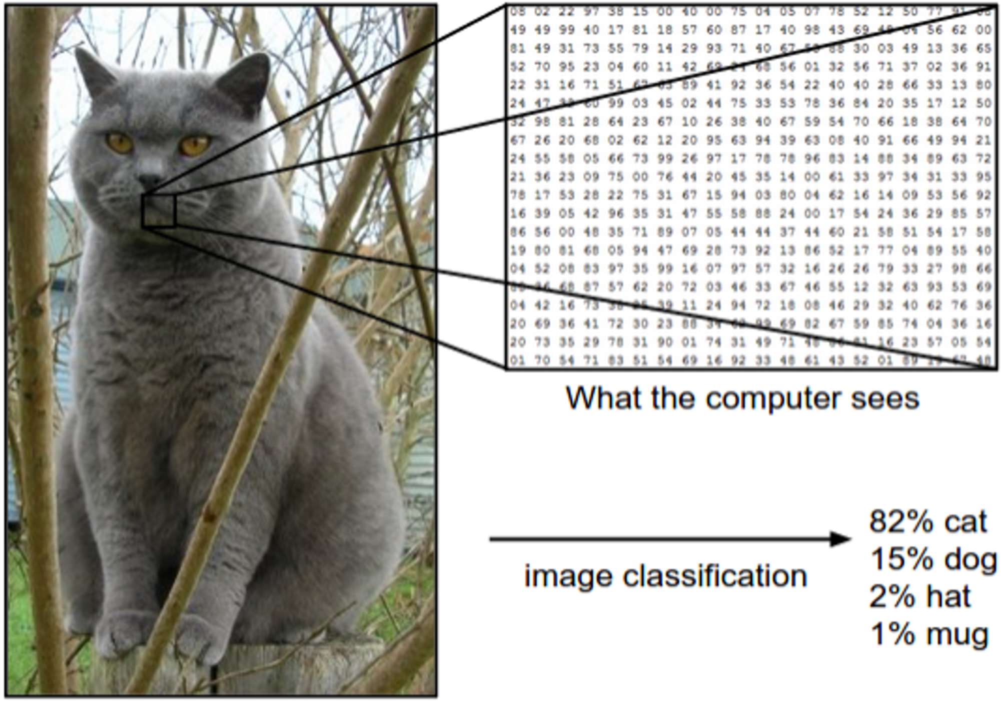
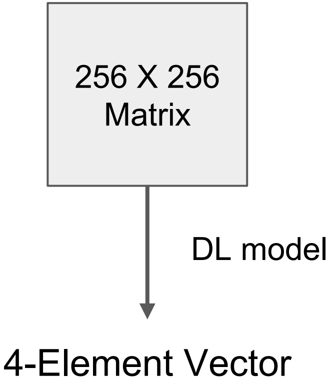


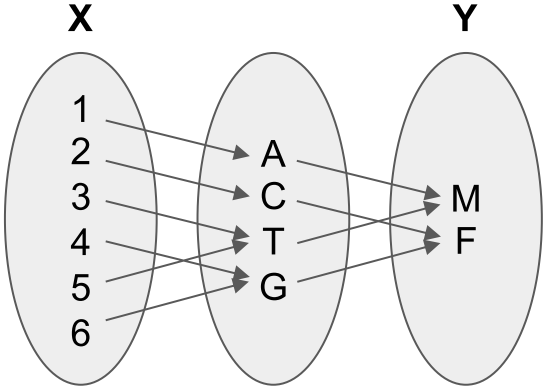
Image from the Stanford CS231 Course



256 X 256 Matrix

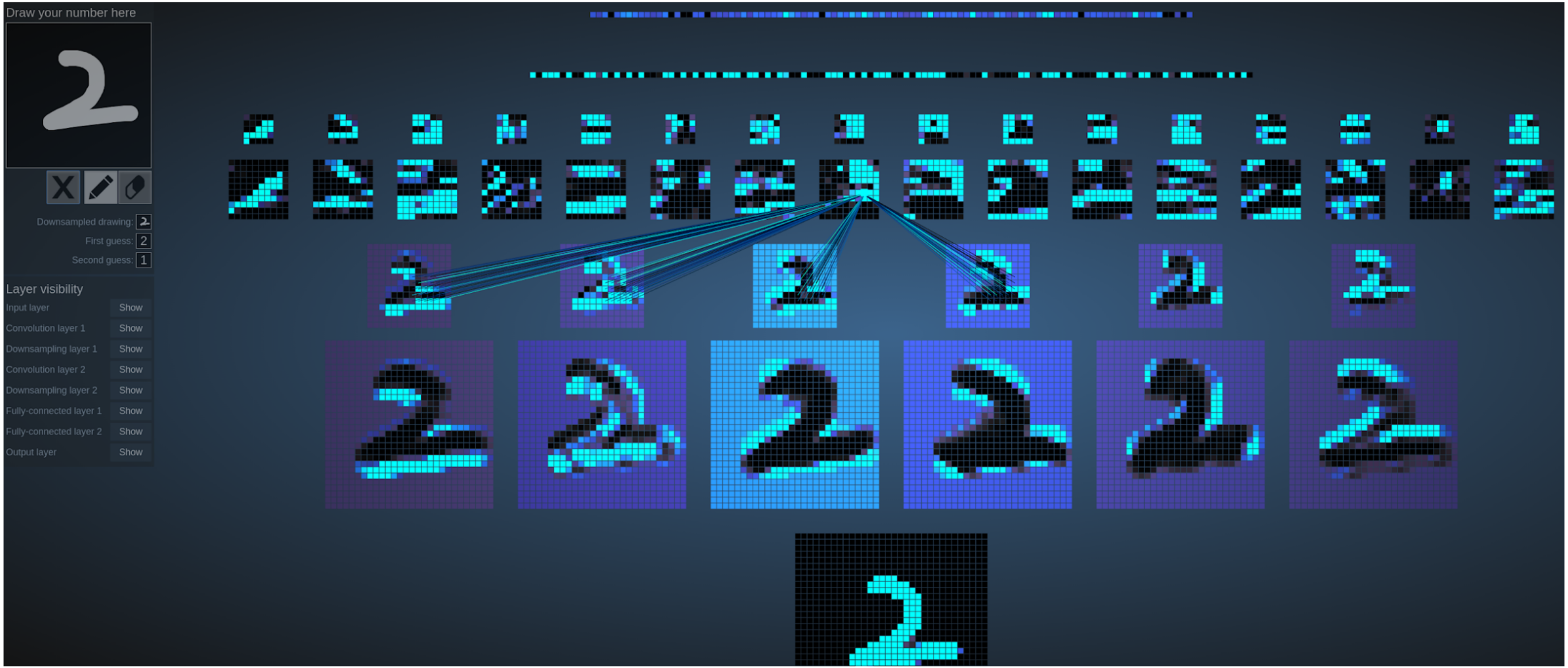
DL model

4-Element Vector



With deep learning, we are searching for a **surjective** (or **onto**) function **f** from a set **X** to a set **Y**.

MNIST - CNN Visualization



Draw your number here

Downsampled drawing:

First guess: 2

Second guess: 1

Layer visibility

Input layer Show

Convolution layer 1 Show

Downsampling layer 1 Show

Convolution layer 2 Show

Downsampling layer 2 Show

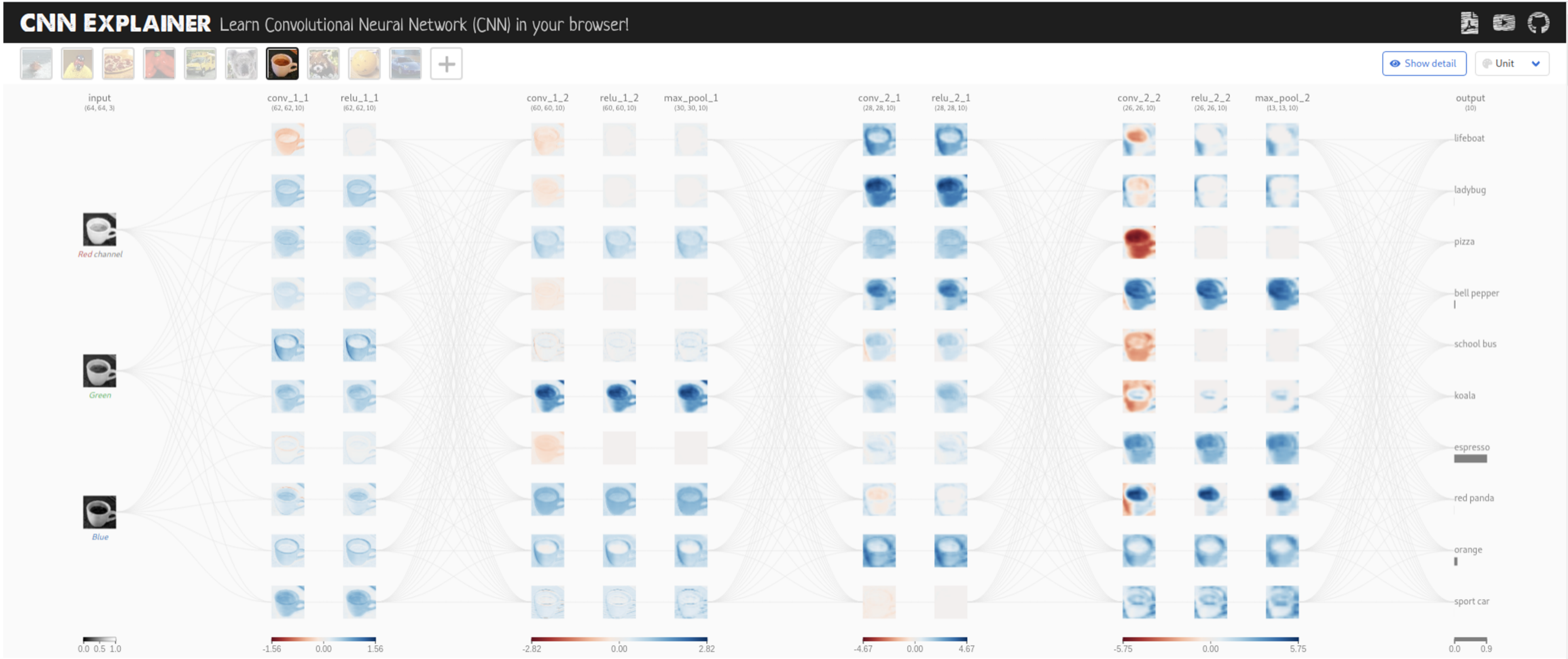
Fully-connected layer 1 Show

Fully-connected layer 2 Show

Output layer Show

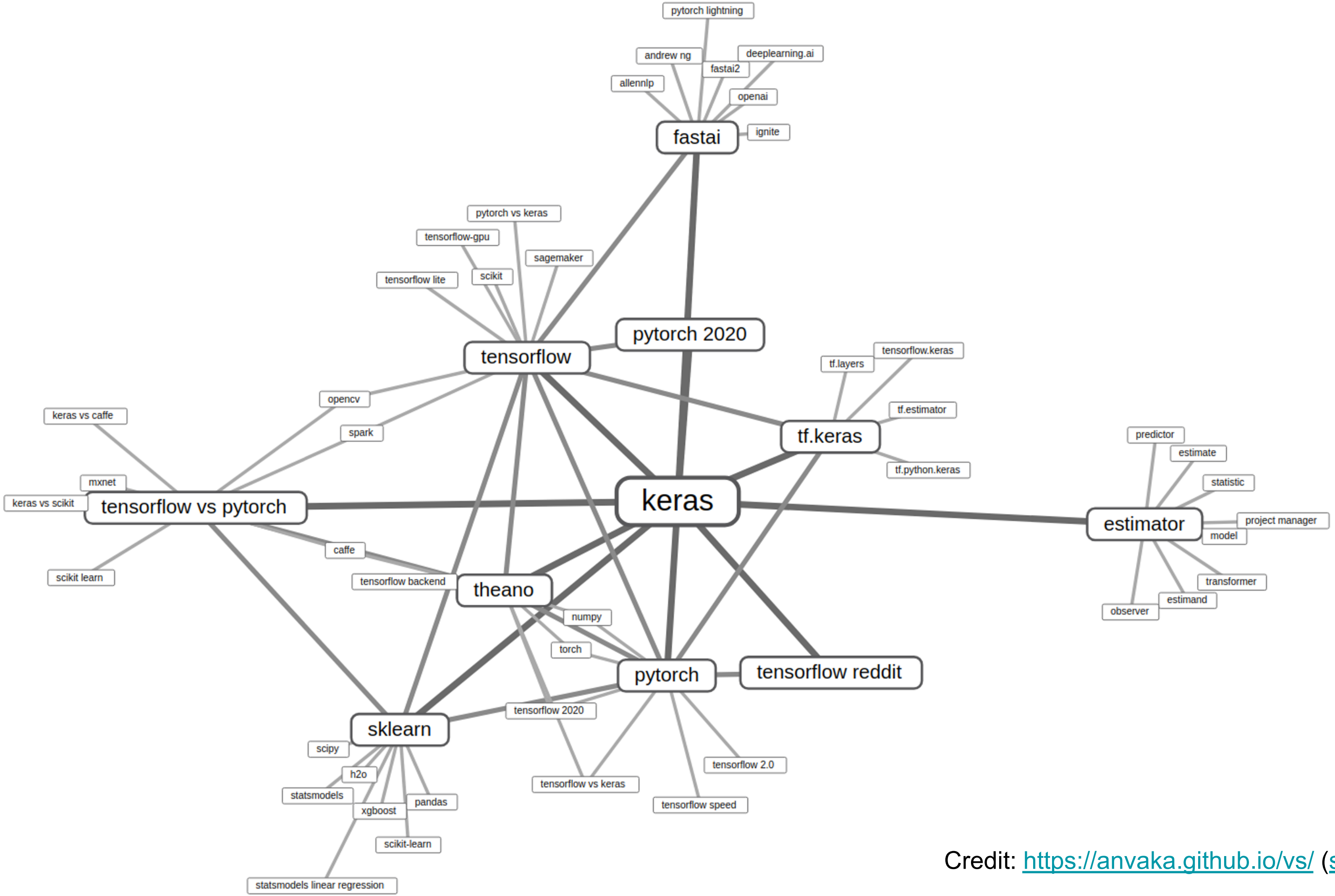
(Image Credit: http://scs.ryerson.ca/~aharley/vis/)

CNN Explainer



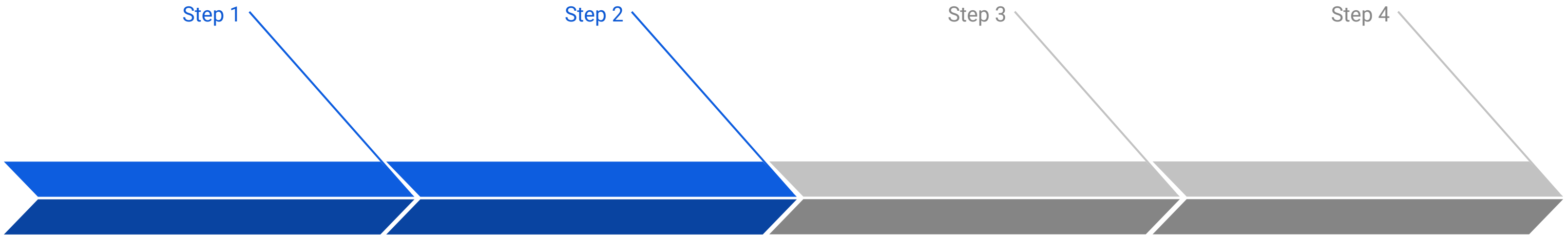
CNN EXPLAINER Learn Convolutional Neural Network (CNN) in your browser!

(Image Credit: https://poloclub.github.io/cnn-explainer/)



Credit: https://anvaka.github.io/vs/ (source)

Machine Learning Workflow with Keras



Step 1

Prepare Train Data

The preprocessed data set needs to be shuffled and splitted into training and testing data.

Step 2

Define Model

A model could be defined with Keras Sequential model for a linear stack of layers or Keras functional API for complex network.

Step 3

Training Configuration

The configuration of the training process requires the specification of an optimizer, a loss function, and a list of metrics.

Step 4

Train Model

The training begins by calling the fit function. The number of epochs and batch size need to be set. The measurement metrics need to be evaluated.