

AI Tech Labs $0 \Rightarrow 1$

Jian Tao

jtao@tamu.edu

HPRC Short Course

10/30/2020



High Performance
Research Computing
DIVISION OF RESEARCH

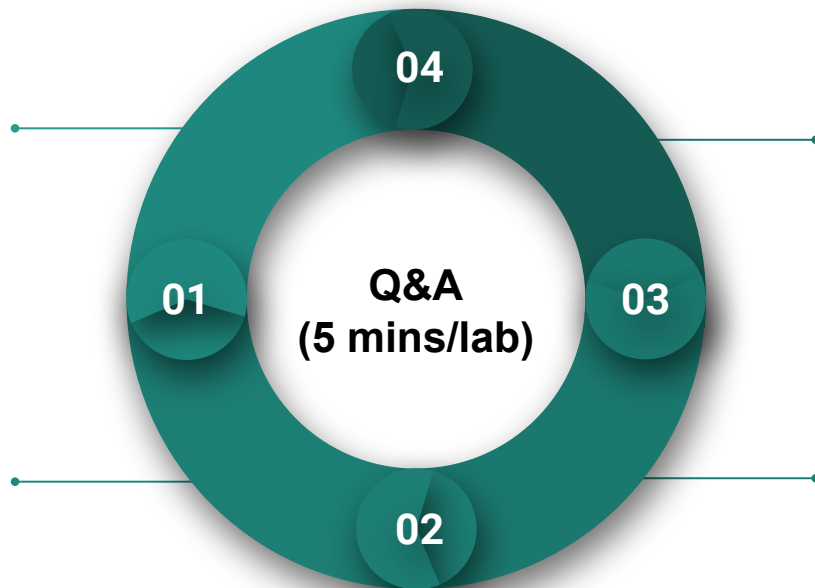
AI Tech Labs

Lab I. JupyterLab (15 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.



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



File Edit View Run Kernel Tabs Settings Help

Files

- notebooks
- Data.ipynb
- Fasta.ipynb
- Julia.ipynb
- Lorenz.ipynb** (seconds ago)
- R.ipynb
- iris.csv
- lightning.json
- lorenz.py (3 minutes ago)

Running

Commands

Cell Tools

Output View

sigma 10.00

beta 2.67

rho 28.00

Code

Python 3

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)
```

lorenz.py

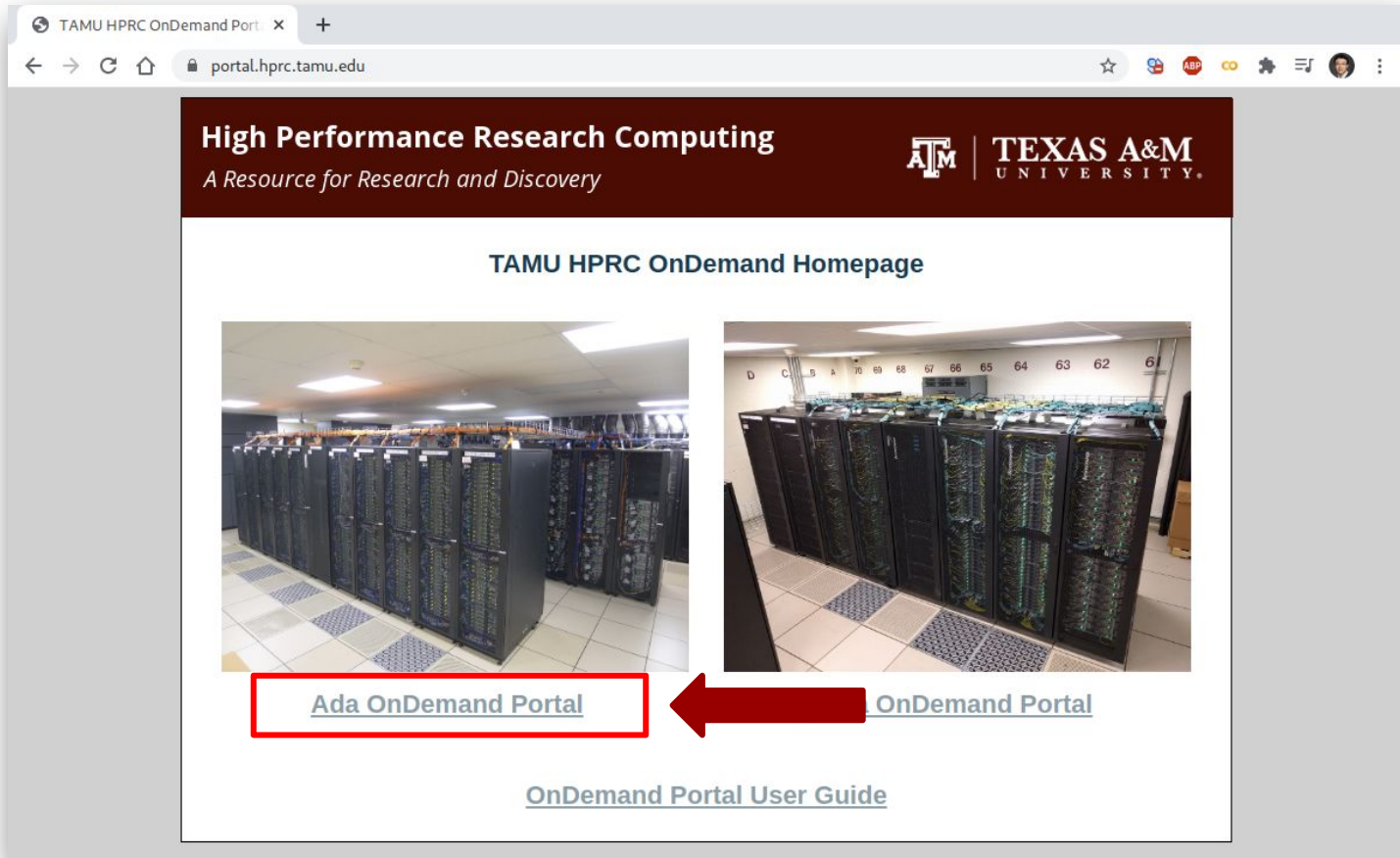
```
9
10 def solve_lorenz(N=10, max_time=4.0, sigma=10.0, beta=8./3, rho=28.0):
11     """Plot a solution to the Lorenz differential equations."""
12     fig = plt.figure()
13     ax = fig.add_axes([0, 0, 1, 1], projection='3d')
14     ax.axis('off')
15
16     # prepare the axes limits
17     ax.set_xlim((-25, 25))
18     ax.set_ylim((-35, 35))
19     ax.set_zlim((5, 55))
20
21     def lorenz_deriv(x_y_z, t0, sigma=sigma, beta=beta, rho=rho):
22         """Compute the time-derivative of a Lorenz system."""
23         x, y, z = x_y_z
24         return [sigma * (y - x), x * (rho - z) - y, x * y - beta * z]
25
26     # Choose random starting points, uniformly distributed from -15 to 15
27     np.random.seed(1)
28     x0 = -15 + 30 * np.random.random((N, 3))
```

A 3D plot of the Lorenz attractor, showing a complex, swirling trajectory in a 3D space. The plot is rendered with a green-to-yellow color gradient. The axes are labeled x, y, and z, with the z-axis pointing upwards. The attractor is centered around the origin and exhibits a characteristic butterfly-like shape.

L1 - Resources

- [Texas A&M High Performance Research Computing \(HPRC\)](#)
- [Ada Quick Start Guide](#)
- [HPRC Portal](#)
- [HPRC YouTube Channel](#)
- [Jupyter Project](#)

L1 - Login HPRC Portal



The screenshot shows a web browser window with the address bar displaying "portal.hprc.tamu.edu". The page header features the text "High Performance Research Computing" and "A Resource for Research and Discovery" alongside the Texas A&M University logo. The main content area is titled "TAMU HPRC OnDemand Homepage" and contains two photographs of server racks. Below the left photograph is a red-bordered box containing the text "Ada OnDemand Portal". Below the right photograph is the text "OnDemand Portal". A red arrow points from the right text to the left text. At the bottom center, there is a link for "OnDemand Portal User Guide".

TAMU HPRC OnDemand Portal

High Performance Research Computing
A Resource for Research and Discovery

ATM | TEXAS A&M UNIVERSITY

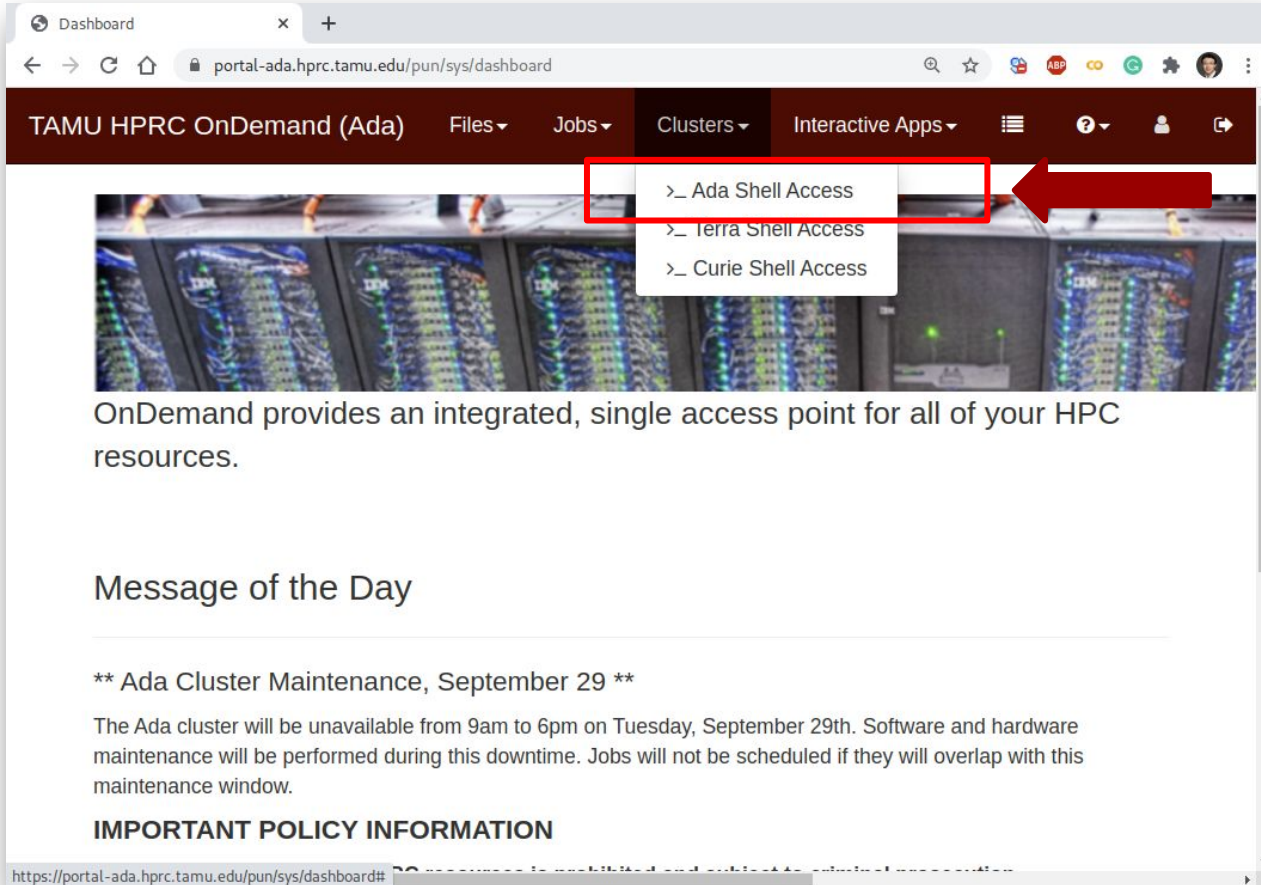
TAMU HPRC OnDemand Homepage

Ada OnDemand Portal

OnDemand Portal

[OnDemand Portal User Guide](#)

L1 - Shell Access - I



The screenshot shows a web browser window with the URL `portal-ada.hprc.tamu.edu/pun/sys/dashboard`. The dashboard header includes navigation tabs for "Files", "Jobs", "Clusters", and "Interactive Apps". A red box highlights the "Clusters" dropdown menu, which is open to show three options: ">_ Ada Shell Access", ">_ Terra Shell Access", and ">_ Curie Shell Access". A red arrow points to the "Ada Shell Access" option. Below the navigation, there is a banner image of server racks with the text "OnDemand provides an integrated, single access point for all of your HPC resources." Below this is a "Message of the Day" section with a notice about "Ada Cluster Maintenance, September 29" and "IMPORTANT POLICY INFORMATION".

Dashboard

portal-ada.hprc.tamu.edu/pun/sys/dashboard

TAMU HPRC OnDemand (Ada) Files Jobs Clusters Interactive Apps

- >_ Ada Shell Access
- >_ Terra Shell Access
- >_ Curie Shell Access

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

Message of the Day

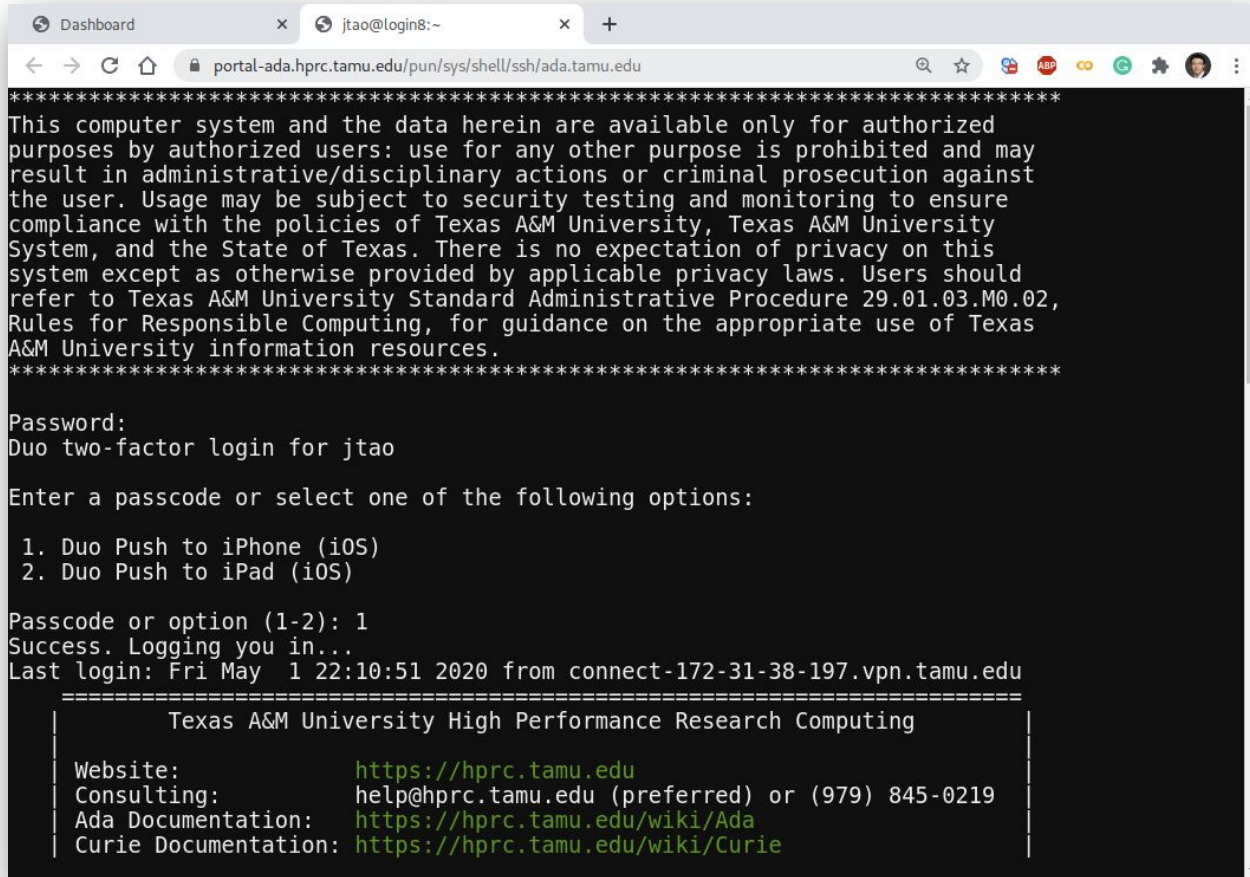
**** Ada Cluster Maintenance, September 29 ****

The Ada cluster will be unavailable from 9am to 6pm on Tuesday, September 29th. Software and hardware maintenance will be performed during this downtime. Jobs will not be scheduled if they will overlap with this maintenance window.

IMPORTANT POLICY INFORMATION

<https://portal-ada.hprc.tamu.edu/pun/sys/dashboard#>

L1 - Shell Access - II



```
Dashboard x jtao@login8:~ x +
portal-ada.hprc.tamu.edu/pun/sys/shell/ssh/ada.tamu.edu
*****
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 jtao

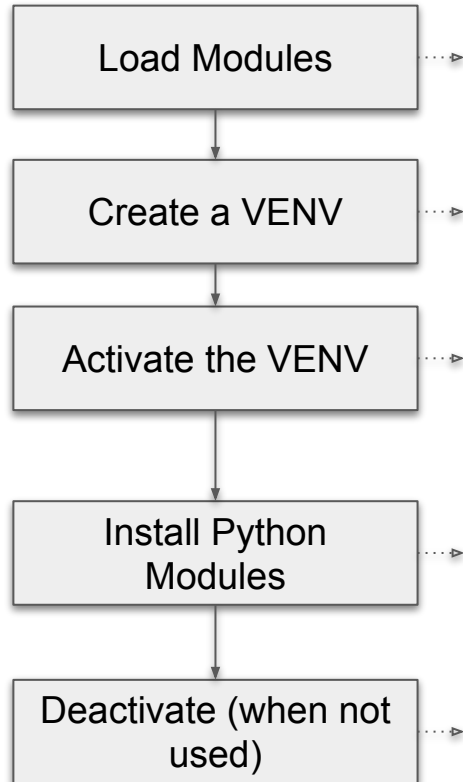
Enter a passcode or select one of the following options:

  1. Duo Push to iPhone (iOS)
  2. Duo Push to iPad (iOS)

Passcode or option (1-2): 1
Success. Logging you in...
Last login: Fri May  1 22:10:51 2020 from connect-172-31-38-197.vpn.tamu.edu

=====
|               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 |
| Curie Documentation:    https://hprc.tamu.edu/wiki/Curie |
|-----|
```

L1 - Python Virtual Environment (VENV)



```
# 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 jupyterlab=1.2.2
conda install pandas matplotlib
conda install scikit-learn
conda install tensorflow

# deactivate the virtual environment
# source deactivate
```


L1 - 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

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

L1 - Check out Exercises

The screenshot shows the GitHub interface for the repository `jtao/ailabs`. The repository has 1 pull request, 0 stars, and 0 forks. The 'Code' button is highlighted, and a dropdown menu is open, showing options for cloning the repository via HTTPS, SSH, or GitHub CLI. The repository name is `jtao/ailabs` and the URL is `https://github.com/jtao/ailabs.git`. The file list shows `README.md` and `images`.

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

Lab I. JupyterLab
(15 mins)

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

04

Lab IV. Deep Learning
(30 minutes)

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

[Publish your first package](#)

L1 - Go to JupyterLab Page

The screenshot shows the TAMU HPRC OnDemand portal dashboard. The browser address bar displays `portal-ada.hprc.tamu.edu/pun/sys/dashboard`. The navigation bar includes "TAMU HPRC OnDemand (Ada)", "Files", "Jobs", "Clusters", and "Interactive Apps". The "Interactive Apps" menu is open, listing various applications under categories: BIO (IGV, Mauve, Structure), GUI (ANSYS Workbench, Abaqus/CAE, LS-PREPOST, MATLAB, ParaView, VNC), Galaxy (Galaxy (maroon), Galaxy (reveille)), and Servers (Jupyter Notebook, JupyterLab, RStudio Server with R 3.4.3 (Singularity), RStudio Server with R 3.6.1 (Singularity)). The "JupyterLab" option is highlighted with a red box, and a red arrow points to it from the right. The main content area features a server rack image, a "Message of the Day" section with a maintenance notice for September 29, and "IMPORTANT POLICY INFORMATION" with a list of rules. A "Log Out" button is visible in the top right.

TAMU HPRC OnDemand (Ada) Files Jobs Clusters Interactive Apps ? jtao Log Out

BIO

- IGV
- Mauve
- Structure

GUI

- ANSYS Workbench
- Abaqus/CAE
- LS-PREPOST
- MATLAB
- ParaView
- VNC

Galaxy

- Galaxy (maroon)
- Galaxy (reveille)

Servers

- Jupyter Notebook
- JupyterLab**
- RStudio Server with R 3.4.3 (Singularity)
- RStudio Server with R 3.6.1 (Singularity)

OnDemand provides an integrated, single access po

Message of the Day

**** Ada Cluster Maintenance, September 29 ****

The Ada cluster will be unavailable from 9am to 6pm on Tuesday, September 29, 2023. Jobs will not be scheduled if they will overlap with this maintenance.

IMPORTANT POLICY INFORMATION

- Unauthorized use of HPRC resources is prohibited and subject to criminal and civil penalties.
- Use of HPRC resources in violation of United States export control regulations is prohibited for all users, including US citizens and legal residents.
- Sharing HPRC account and password information is in violation of HPRC policies.
- Authorized users must also adhere to ALL policies at: <https://hprc.tamu.edu/policies>

performed during this

RC staff members

L1 - Set Virtual Environment

The screenshot shows a web browser window displaying the TAMU HPRC OnDemand (Ada) interface. The browser address bar shows the URL: `portal-ada.hprc.tamu.edu/pun/sys/dashboard/batch_connect/sys/jupyterlab/session_contexts/new`. The interface includes a navigation bar with "Files", "Jobs", "Clusters", and "Interactive Apps" menus. The main content area is titled "JupyterLab" and contains the following information:

- Interactive Apps** sidebar with categories: BIO (IGV, Mauve, Structure), GUI (ANSYS Workbench, Abaqus/CAE, LS-PREPOST, MATLAB, ParaView, VNC), and Servers.
- JupyterLab** title and description: "This app will launch a JupyterLab server on the Ada cluster."
- Module** dropdown menu set to "Anaconda/3-5.0.0.1".
- Environment Selection**: A dropdown menu with "mylab" selected. This field is highlighted with a red box and a red arrow pointing to it from the right.
- Instructions**: "Enter the name of environment to be activated. Changing this field is optional." and "Use the default jupyterlab_v1.2.2 unless you have installed your own JupyterLab conda Environment."

L1 - Connect to JupyterLab

The screenshot shows a web browser window with the URL `portal-ada.hprc.tamu.edu/pun/sys/dashboard/batch_connect/sessions`. The page header includes "TAMU HPRC OnDemand (Ada)" and navigation menus for "Files", "Jobs", "Clusters", and "Interactive Apps". A green notification bar at the top states "Session was successfully created." Below this, a breadcrumb trail shows "Home / My Interactive Sessions".

On the left side, there is a sidebar titled "Interactive Apps" with a list of applications: BIO, IGV, Mauve, Structure, GUI, ANSYS Workbench, and Abaqus/CAE. The main content area displays a session for "JupyterLab (12695677)" with the status "1 node | 1 core | Running".

The session details include:

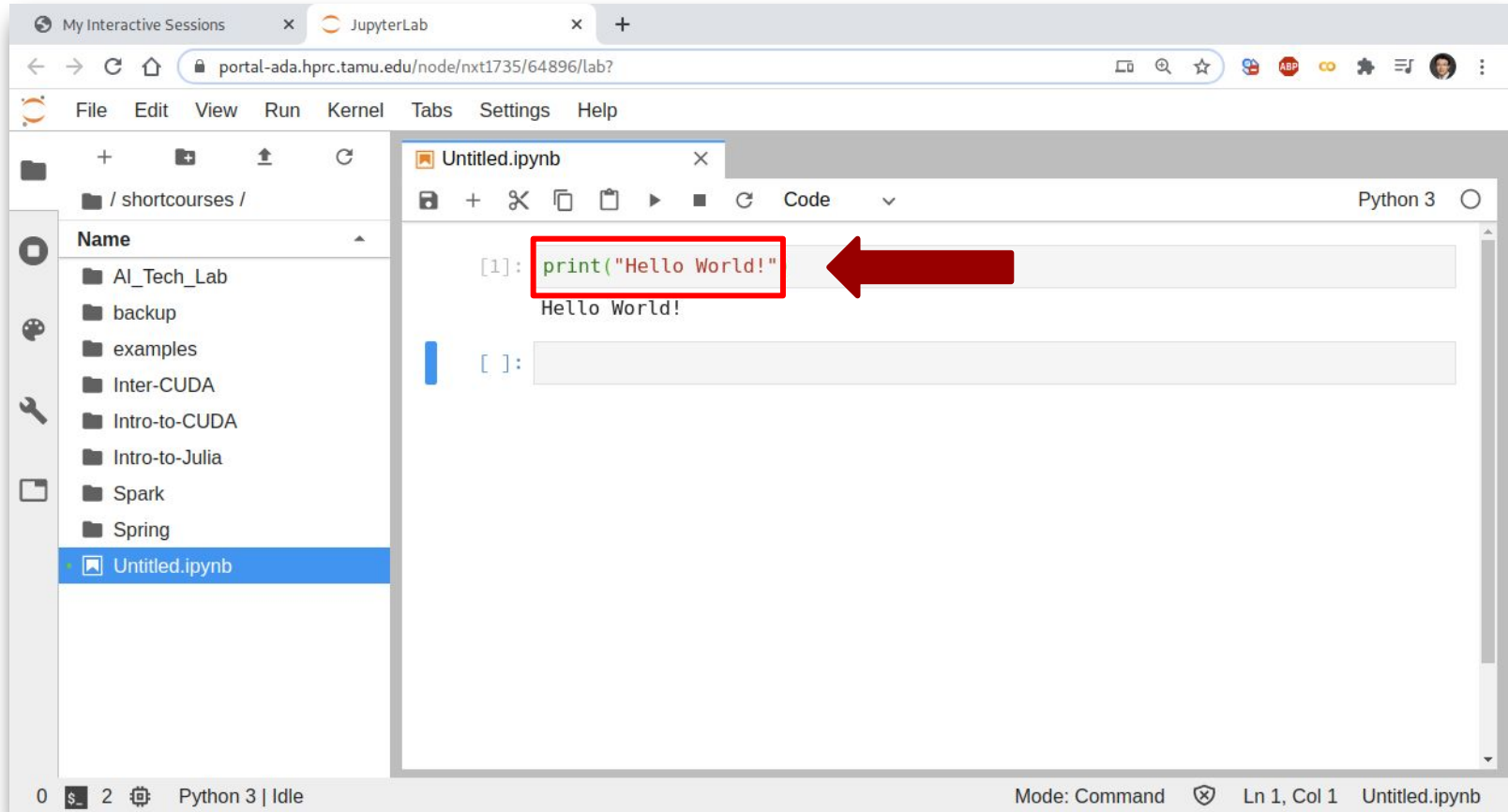
- Host: `nxt1735`
- Created at: 2020-09-19 19:10:05 CDT
- Time Used: 5 minutes
- Session ID: `e18d47a1-3d4f-4f15-b46d-bff5fa09174a`

A red "Delete" button is located to the right of the session details. At the bottom of the session card, a blue button labeled "Connect to JupyterLab" is highlighted with a red rectangular box. A large red arrow points from the right towards this button.

L1 - Create a Jupyter Notebook

The image shows a web browser window displaying the JupyterLab interface. The browser's address bar shows the URL `portal-ada.hprc.tamu.edu/node/nxt1735/64896/lab?`. The JupyterLab interface includes a top menu bar with options: File, Edit, View, Run, Kernel, Tabs, Settings, and Help. On the left side, there is a file browser showing a directory structure under `/shortcourses/` with folders: AI_Tech_Lab, backup, examples, Inter-CUDA, Intro-to-CUDA, Intro-to-Julia, Spark, and Spring. The main area is titled "Launcher" and displays a "shortcourses" directory. Inside this directory, there are three items: a "Notebook" icon, a "Python 3" kernel icon (which is highlighted with a red rectangular box and a red arrow pointing to it from the right), and a "Console" icon. At the bottom of the interface, there is a status bar showing a terminal icon, the number "1", and a gear icon, with the word "Launcher" on the far right.

L1 - Test JupyterLab



The screenshot displays the JupyterLab web interface. The browser address bar shows the URL `portal-ada.hprc.tamu.edu/node/nxt1735/64896/lab?`. The interface includes a top menu bar with options like File, Edit, View, Run, Kernel, Tabs, Settings, and Help. On the left, a file browser shows a directory structure under `/shortcourses/` with folders such as `AI_Tech_Lab`, `backup`, `examples`, `Inter-CUDA`, `Intro-to-CUDA`, `Intro-to-Julia`, `Spark`, and `Spring`. The `Untitled.ipynb` file is selected. The main workspace shows a code cell with the following content:

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

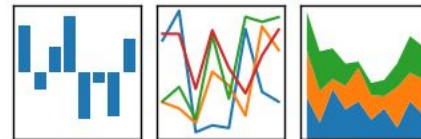
The code `print("Hello World!")` is enclosed in a red rectangular box, and a large red arrow points from the right towards this box. Below the first cell is an empty code cell with the prompt `[]:`. The status bar at the bottom indicates `Python 3 | Idle`, `Mode: Command`, and `Ln 1, Col 1 Untitled.ipynb`.

Lab II. Data Exploration

matplotlib

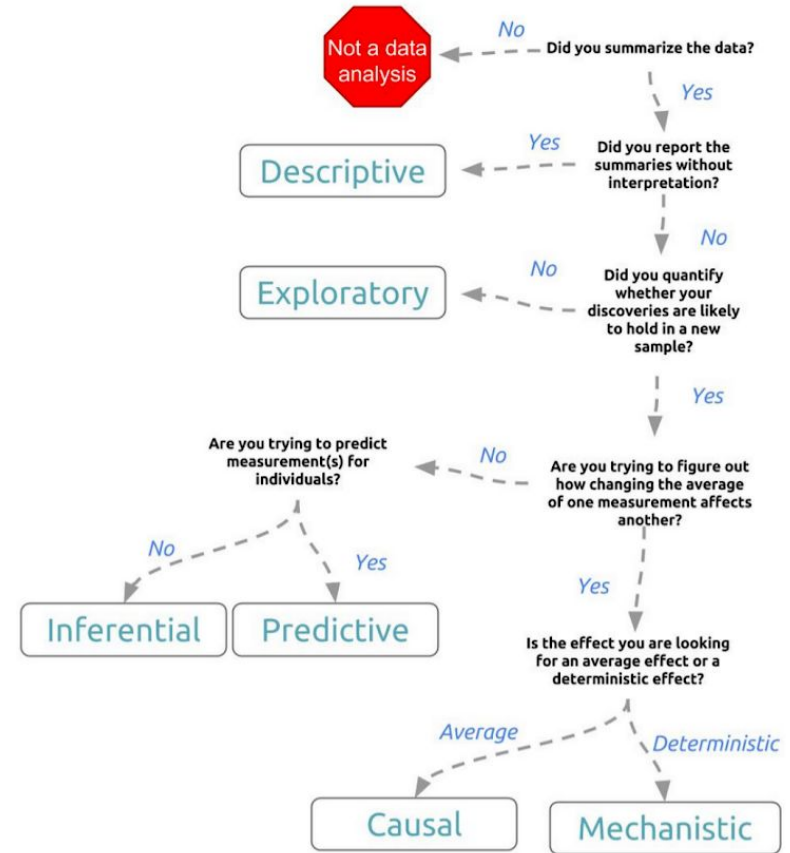
pandas

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



Types of Data Science Problems

- **Descriptive** (summaries, e.g., census)
- **Exploratory** (search for unknowns, e.g., SETI@home, Einstein@home)
- **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)



Matplotlib Cheat Sheet

Python For Data Science Cheat Sheet

Matplotlib

Learn Python interactively at [www.DataCamp.com](https://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.

1 Prepare The Data

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])
>>> T, X = np.meshgrid(np.linspace(0, 1, 10), np.linspace(0, 1, 10))
>>> 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/irisdata_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=run
>>> ax2 = fig.add_subplot(222)
>>> fig3, axes = plt.subplots(nrows=2, ncols=2)
>>> fig4, axes2 = plt.subplots(ncols=2)
```

3 Plotting Routines

1D Data

```
>>> lines = ax.plot(x, y)
>>> ax.scatter(x, y)
>>> axes[0,0].bar([1,2,3],[3,4,5])
>>> axes[1,0].barh([0.5,1,2],[1,0.5,1])
>>> axes[1,1].axhline(0.45)
>>> axes[0,1].axvline(0.45)
>>> ax.fill(x, y, label='blue')
>>> ax.fill_between(x, y, color='yellow')
```

2D Data or Images

```
>>> fig, ax = plt.subplots()
>>> im = ax.imshow(img, cmap='gist_warth', interpolation='nearest', vmin=2, vmax=2)
```

Colormapped or RGB arrays

```
>>> axes[0,1].pcolormesh(data2)
>>> axes[1,0].pcolormesh(data)
>>> axes[2,1].contour(x, z)
>>> axes[2,2].contour(data1)
>>> axes[2,1] = ax.clabel(C2)
```

Pseudocolor plot of 2D array
Plot contour
Plot filled contours
Label a contour plot

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

4 Customize Plot

Colors, Color Bars & Color Maps

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

Markers

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

Linestyles

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

Text & Annotations

```
>>> ax.text(-2, 1, 'Example Graph', style='italic')
>>> ax.annotate('time', xy=(8, 0), xycoords='data', ytext=10.5, dt, textcoords='data', arrowprops=dict(arrowstyle='->', connectionstyle='arc3'))
```

Mattext

```
>>> plt.title('Sigma_i=100', fontsize=20)
>>> ax.axis('equal')
>>> ax.set_xlim(0,10.5), ylim=(-1.5,1.5)
>>> ax.set_xlim(0,10.5)
```

Limits, Legends & Layouts

Limits & Autoscaling

```
>>> ax.margins(x=0.5, y=0.1)
>>> ax.axis('equal')
>>> ax.set_xlim(0,10.5), ylim=(-1.5,1.5)
>>> ax.set_xlim(0,10.5)
```

Legends

```
>>> ax.set(title='An Example Axes', ylabel='Y-Axis', xlabel='X-Axis')
>>> ax.legend(loc='best')
```

Ticks

```
>>> ax.xaxis.set(ticks=range(1,5), ticklabels=[3,100,-12,'foo'])
>>> ax.ticks_params(label='y', direction='inout', length=10)
```

Subplot Spacing

```
>>> fig.subplots_adjust(wspace=0.5, hspace=0.3, left=0.125, right=0.9, top=0.9, bottom=0.1)
```

```
>>> fig.tight_layout()
>>> ax.spines['top'].set_visible(False)
>>> ax.spines['bottom'].set_position(('outward', 10))
```

Add padding to a plot
Set the aspect ratio of the plot to 1
Set limits for x-and y-axis
Set limits for x-axis

Set a title and x-and y-axis labels

No overlapping plot elements
Manually set x-ticks
Make y-ticks longer and go in and out

Adjust the spacing between subplots

Fit subplot(s) in to the figure area
Make the top axis line for a plot invisible
Move the bottom axis line outward

5 Save Plot

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

6 Show Plot

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

Close & Clear

```
>>> plt.close()
>>> plt.clf()
>>> plt.close()
```

Clear an axis
Clear the entire figure
Close a window

DataCamp
Learn Python for Data Science interactively

Key Plotting Concepts in Matplotlib

- **Matplotlib: Figure**

Figure is the object that keeps the whole image output.

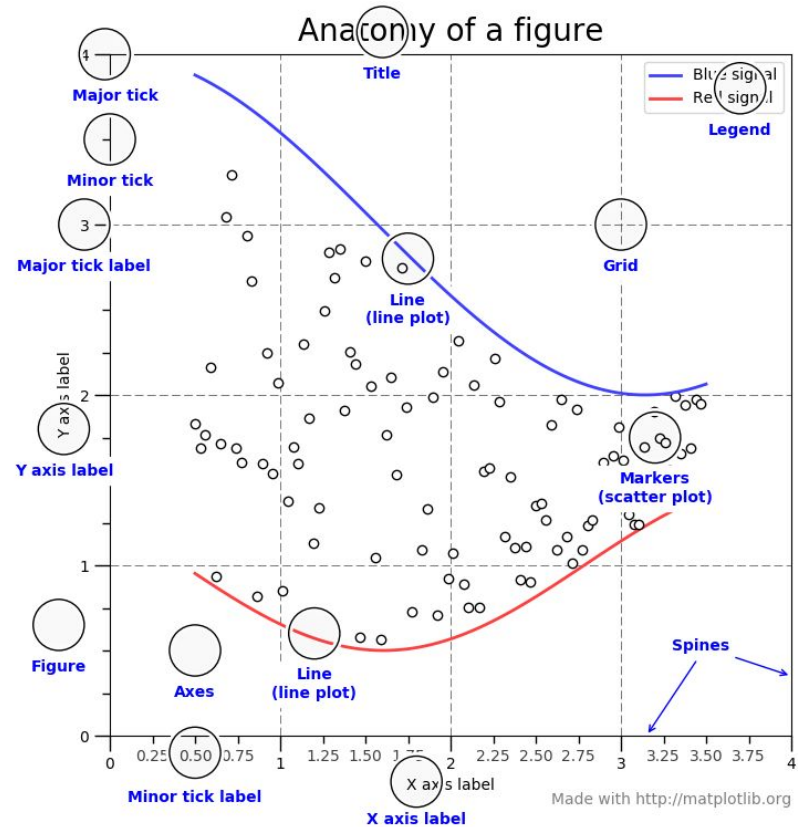
Adjustable parameters include:

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)

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'])
```

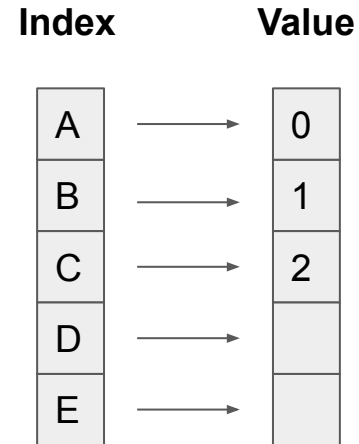
```
In [5]: s.index
```

```
In [6]: pd.Series(np.random.randn(5))
```

```
In [7]: d = {'b': 1, 'a': 0, 'c': 2}
```

```
In [8]: pd.Series(d)
```

```
In [12]: pd.Series(5., index=['a', 'b', 'c', 'd', 'e'])
```



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 |
| 2 | 5 | 8 |
| 3 | 6 | 9 |
| 4 | 7 | 10 |

```
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 |
|---|---|----|
| 1 | 4 | 7 |
| 2 | 5 | 8 |
| 3 | 6 | 9 |
| 4 | 7 | 10 |

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

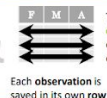
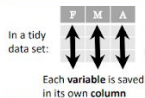
Create DataFrame with a MultiIndex

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'))
```

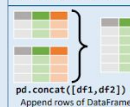
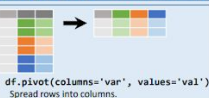
Tidy Data – A foundation for wrangling in pandas



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



```
df.sort_values('mpg')
df.sort_values('mpg', ascending=False)
df.rename(columns = {'y': 'year'})
df.sort_index()
df.reset_index()
df.drop(columns=['Length', 'Height'])
```

Subset Observations (Rows)



```
df[df.Length > 7]
df.sample(frac=0.5)
df.sample(n=10)
df.iloc[10:20]
df.nlargest(n, 'value')
df.nsmallest(n, 'value')
```

Subset Variables (Columns)



```
df[['width', 'length', 'species']]
df['width']
df.filter(regex='regex')
```

regex (Regular Expressions) Examples

| regex | Matches |
|------------------|--|
| '\.' | Matches strings containing a period '.' |
| 'Length\$' | Matches strings ending with word 'Length' |
| '^Sepal' | Matches strings beginning with the word 'Sepal' |
| '*[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']
df.iloc[:, 1:2, 5]
```

| Logic in Python (and pandas) | | |
|------------------------------|------------------------|-----------------------------|
| < | Less than | != |
| > | Greater than | df.column.isin(values) |
| == | Equals | pd.isnull(obj) |
| <= | Less than or equals | pd.notnull(obj) |
| >= | Greater than or equals | ![], ~, ~df.any(), df.all() |

Summarize Data

```
df['v'].value_counts()
len(df)
```

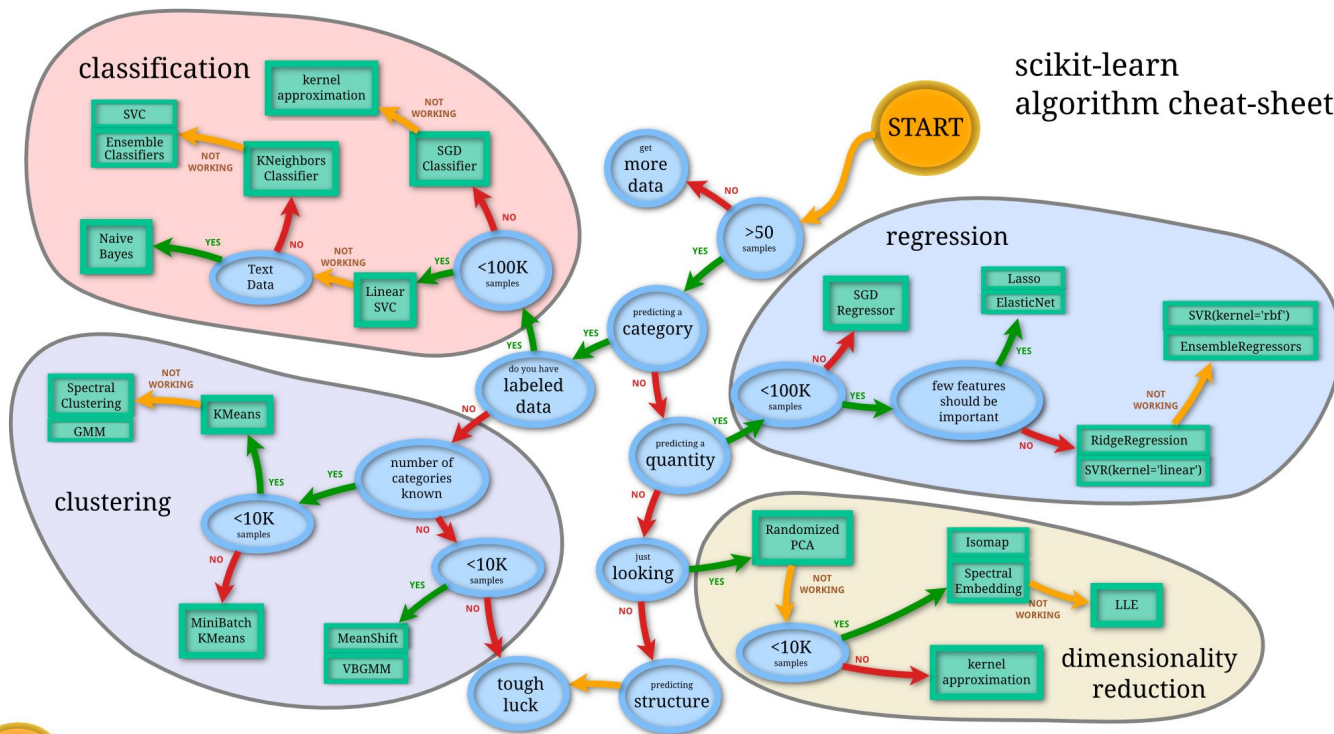
Handling Missing Data

```
df.dropna()
df.fillna(value)
```

Combine Data Sets

```
adf + bdf
```

Lab III. Machine Learning



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

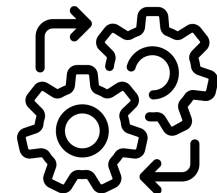
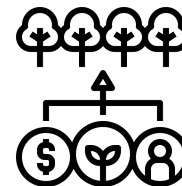
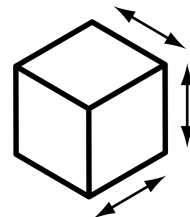
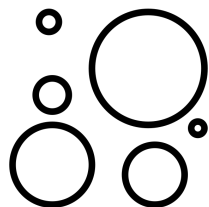
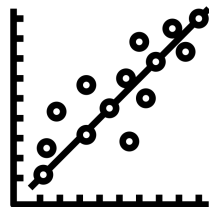
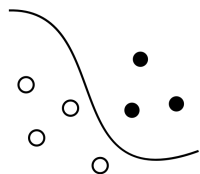
Selecting models with parameter search

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...



Lab IV. Deep Learning

Deep Learning

by Ian Goodfellow, Yoshua Bengio, and Aaron Courville

<http://www.deeplearningbook.org/>

Animation of Neutron Networks

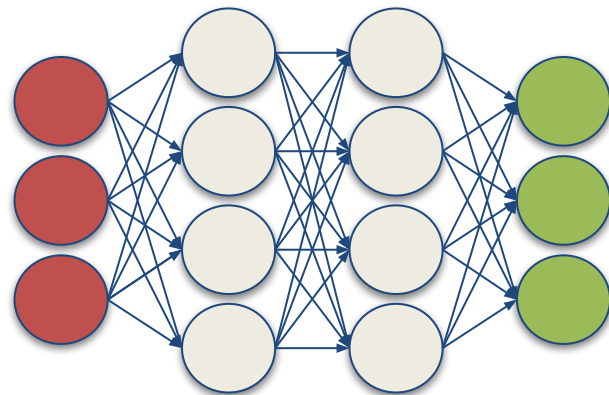
by Grant Sanderson

<https://www.3blue1brown.com/>

Visualization of CNN

by Adam Harley

<https://www.cs.ryerson.ca/~aharley/vis/conv/>

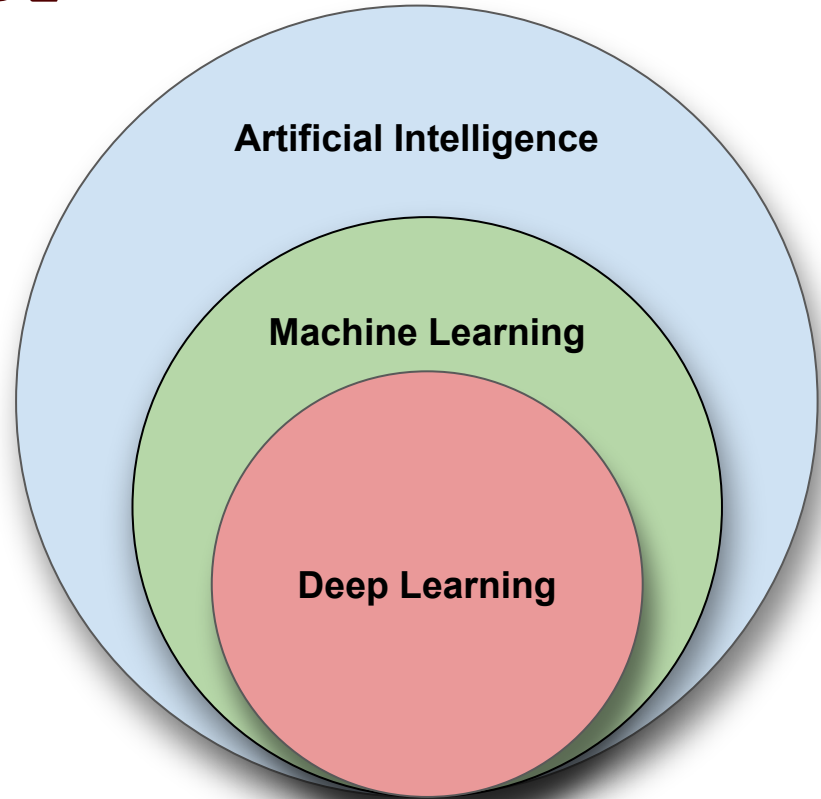


 TensorFlow
2.0

 Keras

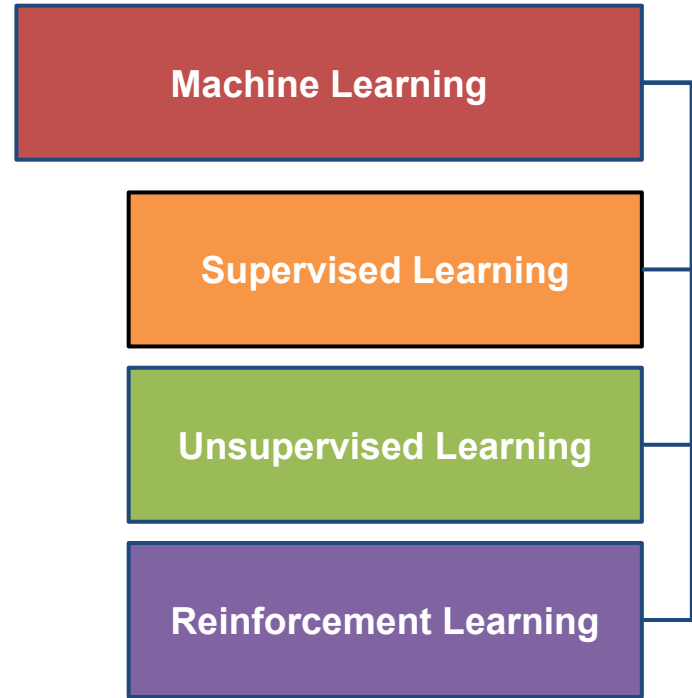
Relationship of AI, ML, and DL

- **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**.



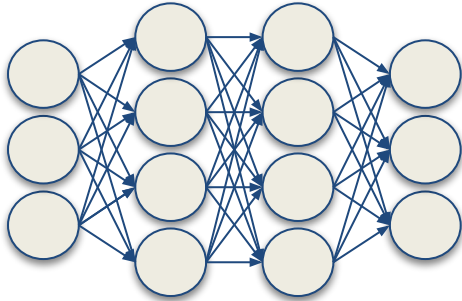
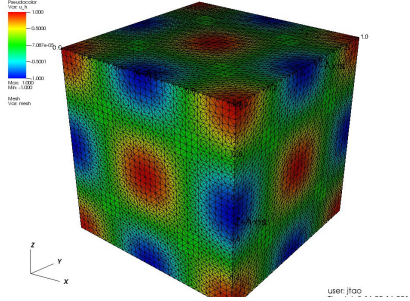
Types of ML Algorithms

- **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.

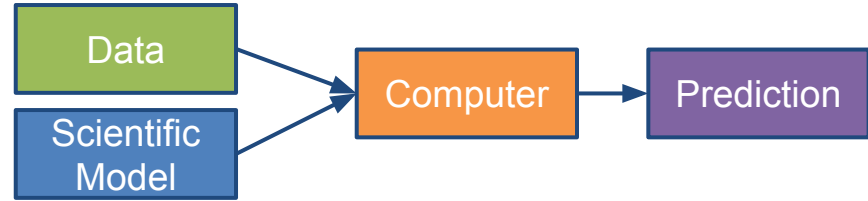


Machine Learning

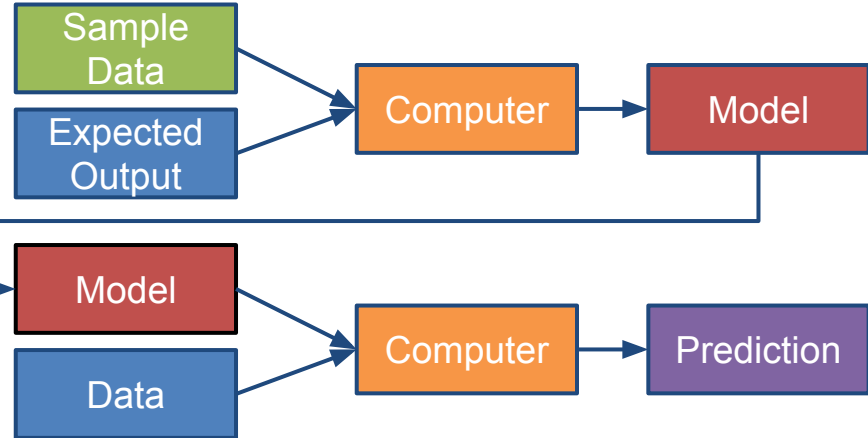
DB: simplest.vtk



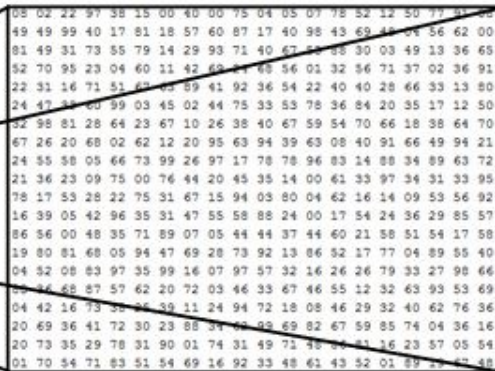
Traditional Modeling



Machine Learning (Supervised Learning)



Inputs and Outputs



What the computer sees

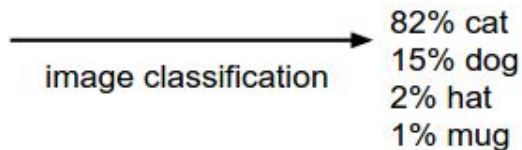
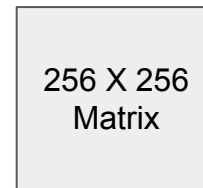
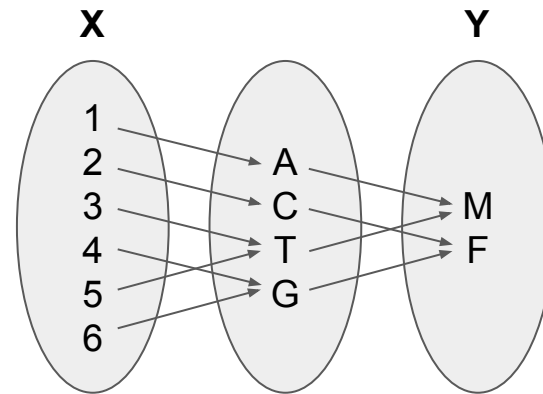


Image from the [Stanford CS231 Course](#)



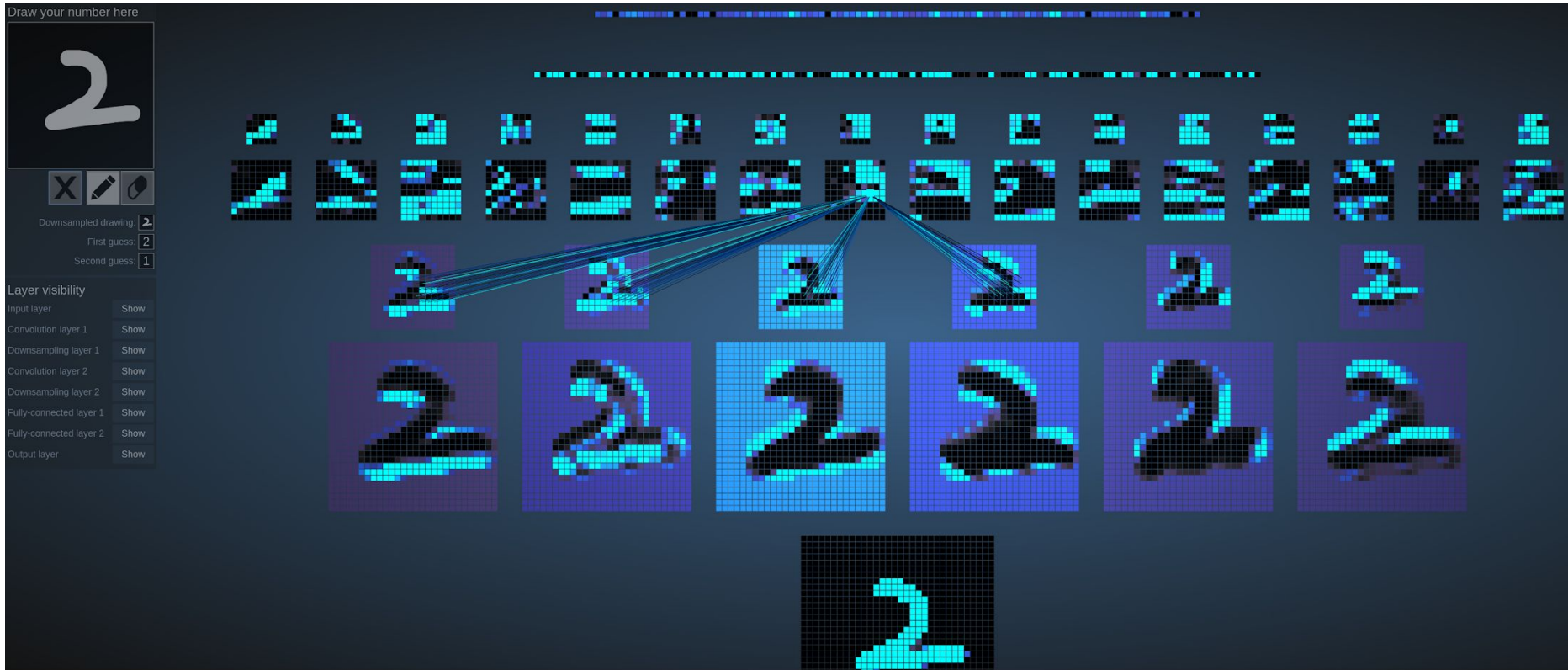
DL model

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



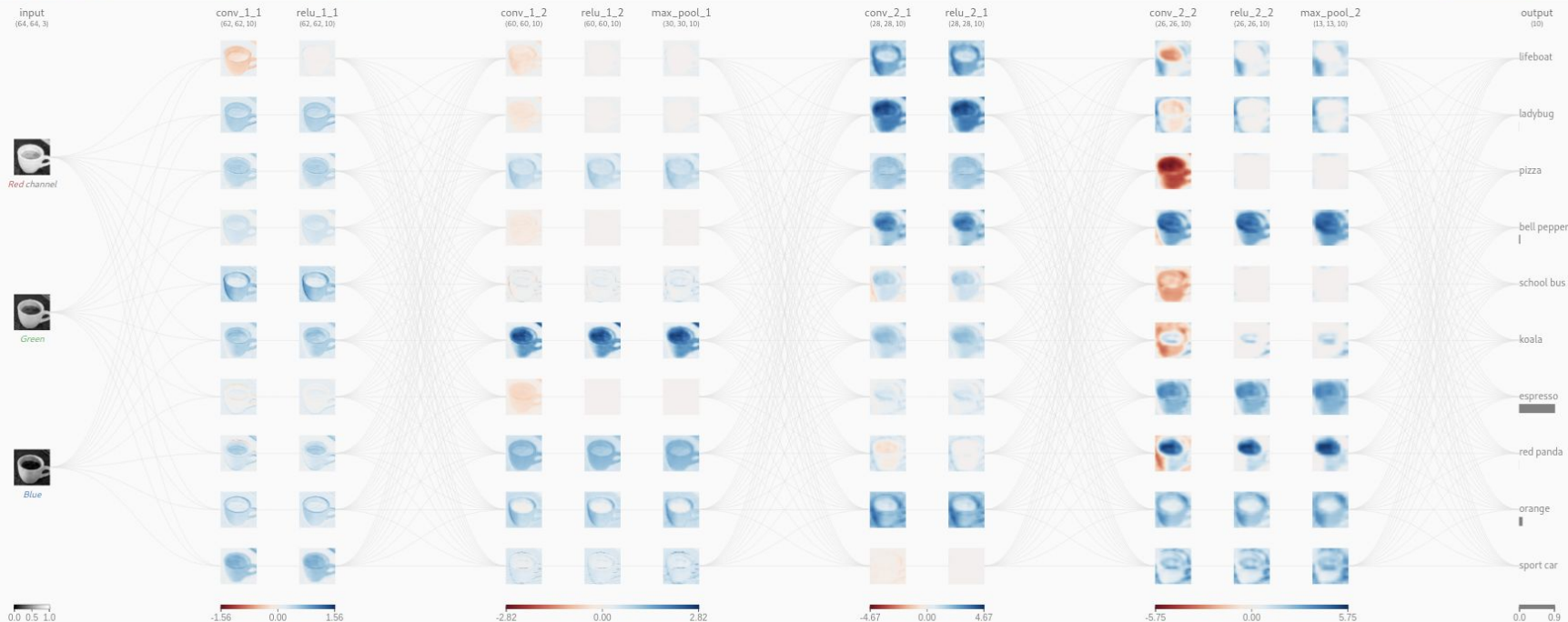
(Image Credit: <http://scs.ryerson.ca/~aharley/vis/>)

CNN Explainer

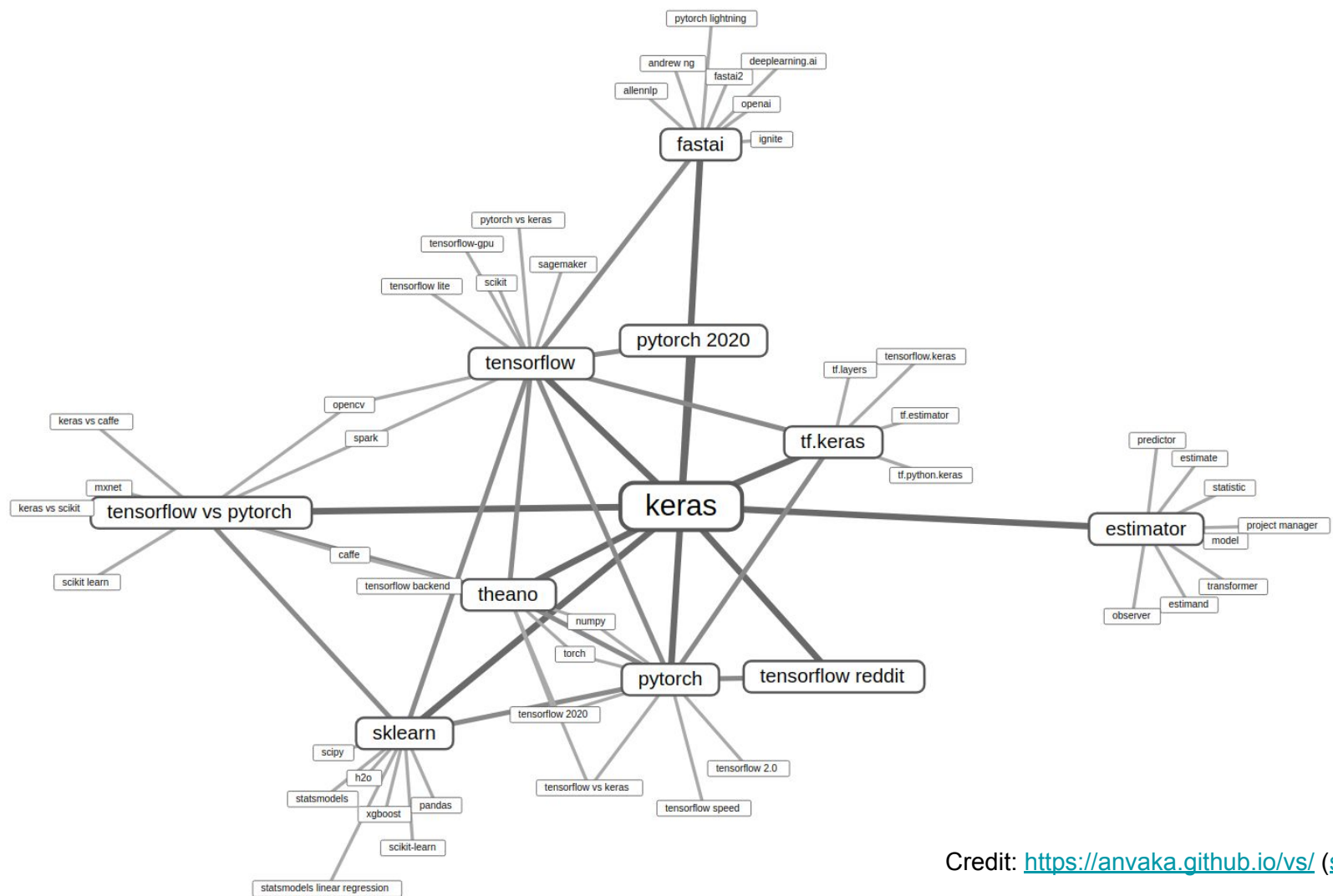
CNN EXPLAINER Learn Convolutional Neural Network (CNN) in your browser!



Show detail Unit

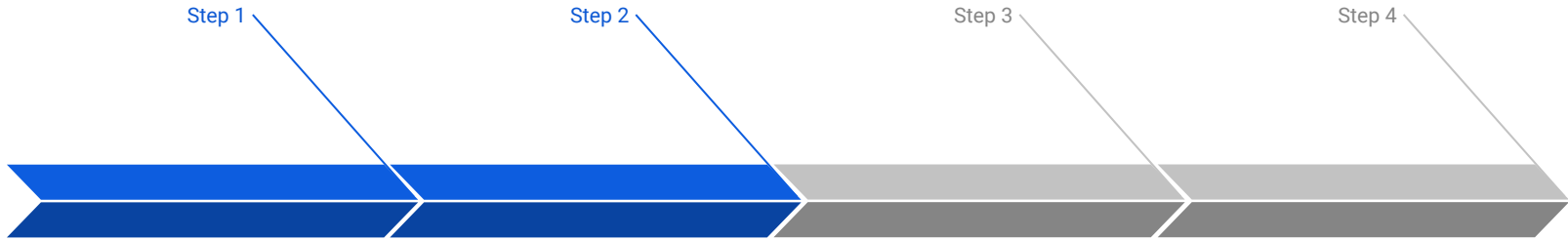


(Image Credit: <https://poloclub.github.io/cnn-explainer/>)



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

Machine Learning Workflow with Keras



Prepare Train Data

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

Define Model

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

Training Configuration

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

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.