Jupiter Lab, a Python Programming Environment

Jupyter Lab is a programming environment and a *Jupyter Notebook* is a corresponding file that includes, in a single "`.ipynb`" file:

- cells of **Python code** for programming
- cells of **text/Markdown for English**, optionally including **Latex for math notation**
- the textual and graphical outputs of code and Markdown cells

We can render it to `.html` or `.pdf` or other formats. Jupyter Lab simplifies the otherwise tedious and error-prone process of writing, debugging, and assembling a data analysis.

To get started

- *Anaconda* is a free Python distribution for data analysis. Get it from https://www.anaconda.com/products/individual#Downloads and install it.
- Make a `Desktop/451` folder (use Finder in macOS or File Explorer on Windows).
- Create a first tiny `.ipynb` file:
  1. Start "Anaconda Navigator".
  2. Under "Jupyter Lab", click "Launch".
  3. In the "JuptyerLab" tab in your browser, click `Desktop` and then `451`.
  4. "File > New > Notebook > Python 3 > Select".
  5. "Right-click > Rename" on "Untitled" and rename the notebook to "tinyNotebook.ipynb".
  6. Type `3 + 4` in the first cell (which is labeled by "`In []:`").
  7. Click "Run" to see that $3 + 4 = 7$.
  8. Notice the new `tinyNotebook.ipynb` file.
- To see most of the Notebook features we need:
  1. Download (via Right-click, Save link as) JupyterExample.ipynb to `Desktop/451` (optional: `.html`).
  2. In the "Jupyter" tab in your browser, click the "refresh" symbol (upper right).
  3. Click `JupyterExample.ipynb` to open it in a Jupyter Lab tab.

Now (in lecture–these are notes and a reference sheet) we explore the elements of this example Notebook: **Python code**, **text/Markdown for English**, and **Latex for math notation**.

Python code

Python is a programming language. Here we have a tiny sample; we will study Python soon.

text/Markdown for English

Markdown is human-readable plain text that can be rendered to HTML including styled text, headings, bulleted and numbered lists, tables, links, etc. To see its basics, use the "JupyterExample" tab's menu choice "Help > Markdown Reference" and glance through the page; return as needed. Also see "Basic writing and formatting syntax."

Latex for math notation

Latex is plain text that can be rendered to math notation. In a Markdown cell, delimit Latex math by `$ ... $` to render inline, or by `$$ ... $$` to render as a separate paragraph. Here are basics:

Latex	Result
`x^y`	x^y
`x_y`	x_y
`\alpha, \mu, \sigma`	α, μ, σ
`\bar{x}`	$\bar{x}$
`\hat{x}`	$\hat{x}$
`\sqrt{x}`	$\sqrt{x}$
`\sum`	$\sum$
`\frac{x}{y}`	$\frac{x}{y}$
`\text{words}`	words (i.e. non-italicised text in math expression)

e.g.

- `$Z = \frac{\bar{x} - \mu_0}{\sigma / \sqrt{n}}$` in a Markdown cell gives $Z = \frac{\bar{x}-\mu_0}{\sigma/\sqrt{n}}$.
- `$$\bar{X} = \frac{1}{n} \sum_{i=1}^n X_i$$` gives

$$\bar{X} = \frac{1}{n} \sum_{i=1}^{n} X_i$$

- `$$\text{rate} = \frac{\text{distance}}{\text{time}}$$` gives

$$\text{rate} = \frac{\text{distance}}{\text{time}}$$

To learn more, see https://en.wikibooks.org/wiki/LaTeX/Mathematics.

Using Jupyter Lab

A Notebook has two modes, Edit and Command, each with its own keyboard shortcuts (here "`C-`" refers to "`Command-`" on a Mac or "`Ctrl-`" on Windows):

- Press Enter to enable *Edit* mode in a cell (green). Then:
  - `Crtl-Enter`, `Shift-Enter`: run cell, run and select next
  - `C-]` (or `TAB`), `C-[`: indent, dedent
  - `C-/`: toggle comment (with "`# `")
  - `C-Z`: undo
  - `Ctrl-Shift-Minus`: split cell
- Press Esc to enable *Command* mode to manipulate cells (blue). Then:
  - `a`, `b`: add cell above, below
  - `c`, `v`, `x`: copy cell, paste, cut
  - `k`, `j`: select cell above, below
  - `dd`: delete cell
  - `z`, `Shift-z`: undo cell deletion, redo
  - `Shift-m`: merge selected cells
  - `C-a`: select all cells
  - `C-f`: find; triangle toggles replace
  - `m`, `y`, `r`: change cell to Markdown, code, raw
  - `C-Shift-`↓, `C-Shift-`↑: move cell down, up
  - `C-s`, `C-Shift-s`: save, save as

In the "Kernel" menu:

- Interrupt: stops running code
- Restart & Clear Outputs: all variables and output are cleared
- Restart & Run All
- Restart & Run up to Selected Cell

To turn in homework:

1. Run cleanly via “Kernel > Restart & Run All”.
2. Save via `C-s`.
3. Create `.html` output via “File > Save and Export Notebook As > HTML”
4. Upload your `.ipynb` and `.html` files to Canvas.

To learn more, in the “JupyterExample” tab’s “Help” menu, choose:

- Show Keyboard Shortcuts
- JupyterLab Reference
- Markdown Reference

Notice that the “Help” menu includes Python and these packages:

- *NumPy* for array data via its `ndarray` object
- *pandas* for tabular data (like a spreadsheet) via its `DataFrame` object
- *SciPi* for scientific algorithms like clustering, FFT, integration, interpolation, linear algebra, optimization and root-finding
- *Matplotlib* for static, animated, and interactive data visualization.

Updating Software

- `import sklearn; sklearn.__version__` gives current version of `scikit-learn`
- `import sklearn; sklearn.show_versions()` gives version information on Python, libraries

In the “Jupyter” browser tab, make the “New > Terminal” drop-down menu choice. Here are examples of commands I’ve used in the Terminal to update software. Some are slow.

- `conda update scikit-learn` (or update `numpy`, `scipy`, `pandas`, `matplotlib`)
- `conda update --all` (update everything, possibly getting ahead of stable version)
- `conda update -n base -c defaults conda` (update `conda` itself)
- uninstall and reinstall Anaconda
- `conda install -c conda-forge imbalanced-learn` installs a package we need in §08