Setting up a working environment

Machine Learning in Finance for Python (ECON5130)

Richard Foltyn

University of Glasgow

15th November 2022

Setup guide

This guide gives you some hints on how to set up a working Python environment.

Multiple options:

1. Run in the cloud
    1. Launch in binder
    2. Launch in Google Colab
2. Install locally (advanced)

    Exact procedure depends on your operating system: hints for Microsoft Windows and Linux are provided below.

Running in the cloud

Running notebooks in the cloud

The course itself is based on interactive notebooks which you can run directly in your browser – no need to set up anything!

- **Launch in binder**
  1. Go to https://github.com/richardfoltyn/MLFP-ECON5130
  2. Click on the button launch binder (shown on the next slide)
  3. Wait. Starting up the environment can take a while.

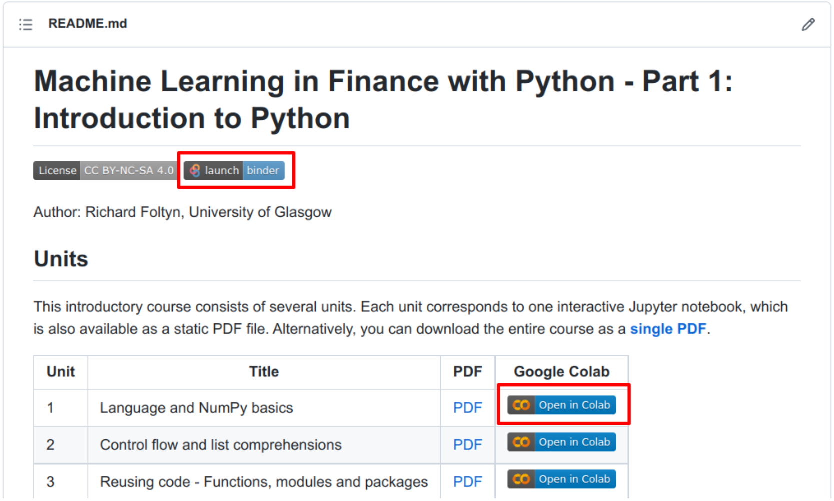
- **Launch in Google Colab**
  1. Go to https://github.com/richardfoltyn/MLFP-ECON5130
  2. Click on the button Open in Colab (shown on the next slide)

Binder vs. Google Colab:

- Google Colab starts instantly
- The environment is somewhat restricted compared to binder (not trivial to import custom modules or local data files)
- Might require Google login to run anything

Running notebooks in the cloud

Launching notebooks from GitHub repository



README.md

Machine Learning in Finance with Python - Part 1: Introduction to Python

Author: Richard Foltyn, University of Glasgow

Units

This introductory course consists of several units. Each unit corresponds to one interactive Jupyter notebook, which is also available as a static PDF file. Alternatively, you can download the entire course as a **single PDF**.

Unit	Title	PDF	Google Colab
1	Language and NumPy basics	PDF	Open in Colab
2	Control flow and list comprehensions	PDF	Open in Colab
3	Reusing code - Functions, modules and packages	PDF	Open in Colab

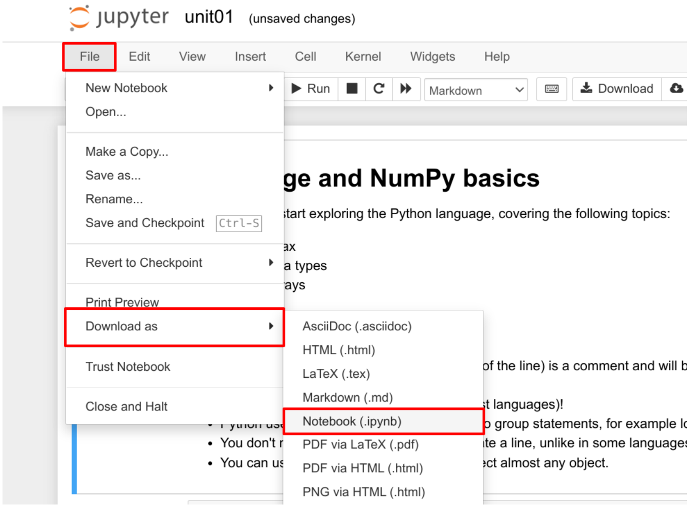
Saving and opening notebooks

- **Important:** Notebooks launched in binder are not saved and will eventually disappear
- You need to manually save and restore notebooks:
    1. Before you stop working on a notebook, make sure to download it to your computer
    2. When you want to continue working on an existing notebook, upload it to binder first and then open it

See following slides for instructions!

Saving (downloading) notebooks

Select **File ▹ Download as ▹ Notebook (.ipynb)**



ge and NumPy basics

;tart exploring the Python language, covering the following topics:

ax

a types

ays

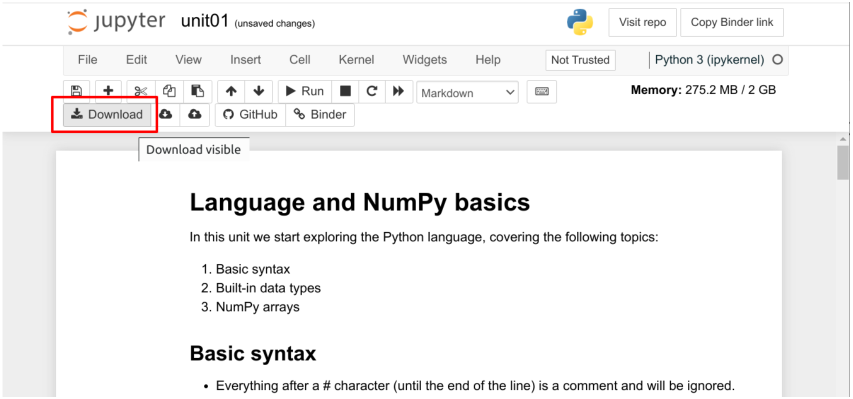
of the line) is a comment and will b

t languages)!

- Python us o group statements, for example lo
- You don't r ite a line, unlike in some language
- You can us ect almost any object.

Saving (downloading) notebooks

Alternatively, you can click on **Download** in the tool bar.



Language and NumPy basics

In this unit we start exploring the Python language, covering the following topics:

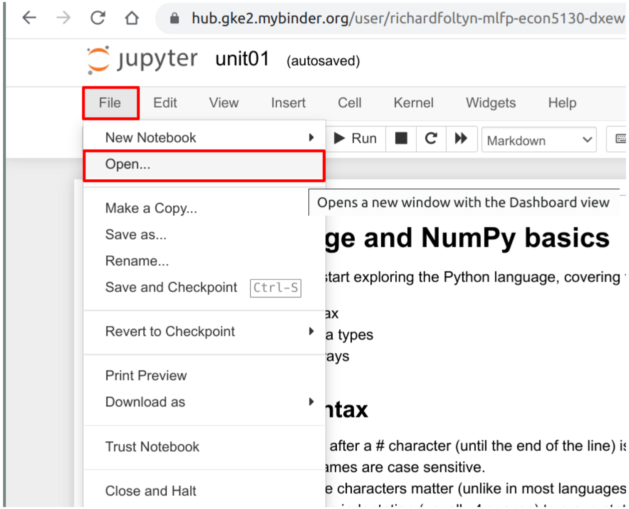
1. Basic syntax
2. Built-in data types
3. NumPy arrays

Basic syntax

- Everything after a # character (until the end of the line) is a comment and will be ignored.

Opening (uploading) notebooks: step 1

Select **File** ▸ **Open**. This brings up a file explorer page (see next slide)



ge and NumPy basics

tart exploring the Python language, covering

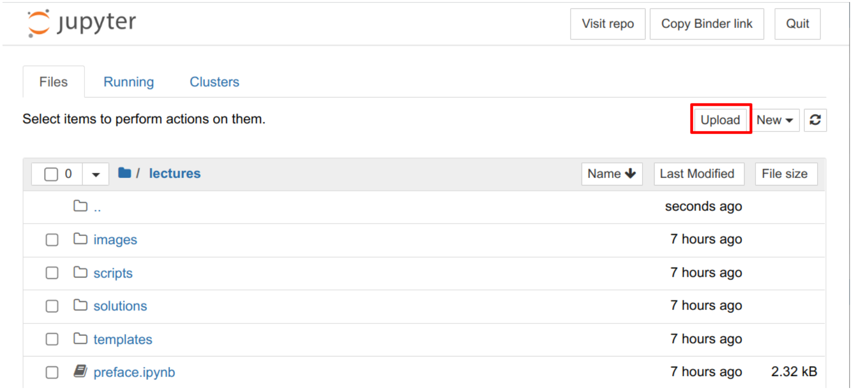
ax
a types
ays

ntax

after a # character (until the end of the line) is
ames are case sensitive.
e characters matter (unlike in most languages

Opening (uploading) notebooks: step 2

Click on **Upload**. You will be prompted to select a file on your computer.



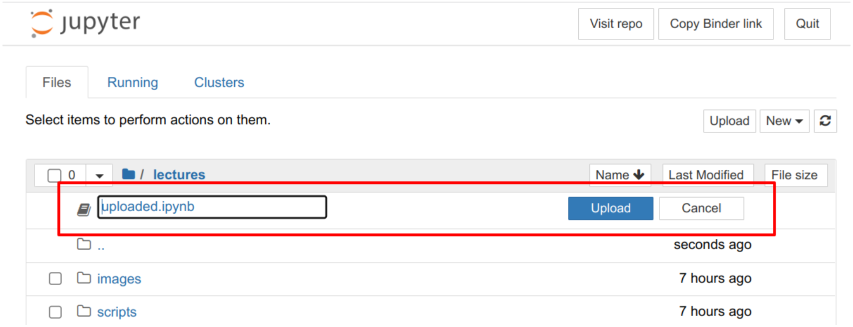
Select items to perform actions on them.

0 / lectures

..	seconds ago	
images	7 hours ago	
scripts	7 hours ago	
solutions	7 hours ago	
templates	7 hours ago	
preface.ipynb	7 hours ago	2.32 kB

Opening (uploading) notebooks: step 3

The selected file is displayed on top. Select **Upload** one more time.



Select items to perform actions on them.

/ lectures

uploaded.ipynb

.. seconds ago

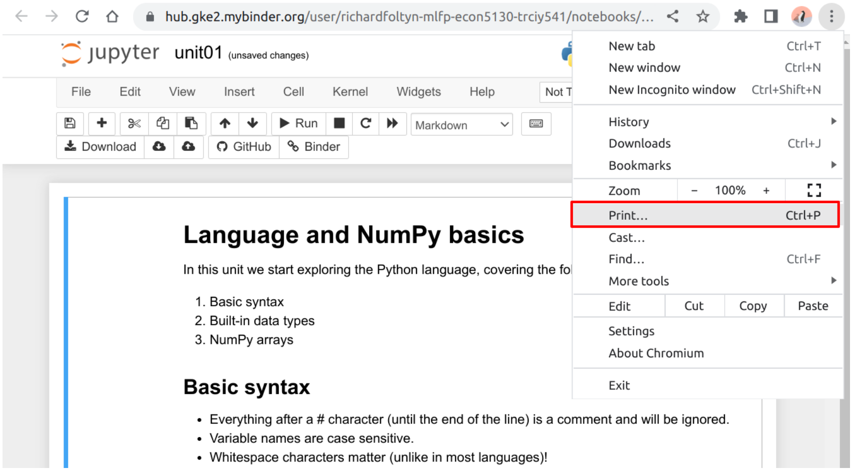
images 7 hours ago

scripts 7 hours ago

The uploaded file should now be visible in the list. Click on it to open the notebook.

Saving as PDF: step 1

Downloading the notebook as PDF will most likely not work.
Instead, use your browser's print function.



Language and NumPy basics

In this unit we start exploring the Python language, covering the fol

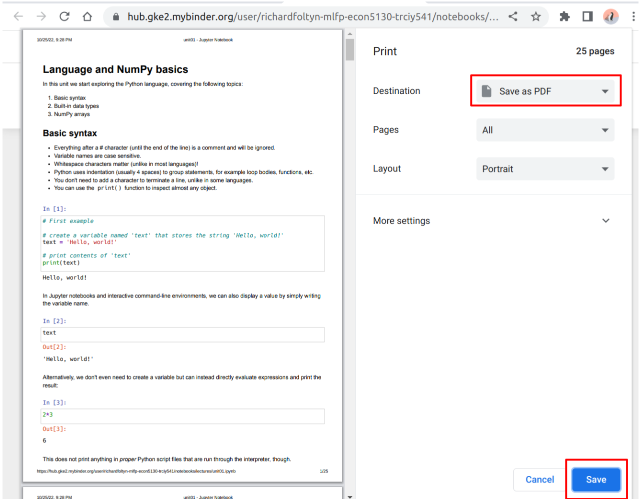
1. Basic syntax
2. Built-in data types
3. NumPy arrays

Basic syntax

- Everything after a # character (until the end of the line) is a comment and will be ignored.
- Variable names are case sensitive.
- Whitespace characters matter (unlike in most languages)!

Saving as PDF: step 2

Select **Save as PDF**.



10/25/22, 9:28 PM unit01 - Jupyter Notebook

Language and NumPy basics

In this unit we start exploring the Python language, covering the following topics:

1. Basic syntax
2. Built-in data types
3. NumPy arrays

Basic syntax

- Everything after a # character (until the end of the line) is a comment and will be ignored.
- Variable names are case sensitive.
- Whitespace characters matter (unlike in most languages)!
- Python uses indentation (usually 4 spaces) to group statements, for example loop bodies, functions, etc.
- You don't need to add a character to terminate a line, unlike in some languages.
- You can use the `print()` function to inspect almost any object.

```
In [1]:
# First example

# create a variable named 'text' that stores the string 'Hello, world!'
text = 'Hello, world!'

# print contents of 'text'
print(text)

Hello, world!
```

In Jupyter notebooks and interactive command-line environments, we can also display a value by simply writing the variable name.

```
In [2]:
text
Out[2]:
'Hello, world!'
```

Alternatively, we don't even need to create a variable but can instead directly evaluate expressions and print the result:

```
In [3]:
2*3
Out[3]:
6
```

This does not print anything in *proper* Python script files that are run through the interpreter, though.

https://hub.gke2.mybinder.org/user/richardfoltyn-mlfp-econ5130-trciy541/notebooks/lectures/unit01.ipynb 1/25

10/25/22, 9:28 PM unit01 - Jupyter Notebook

Running locally on your computer

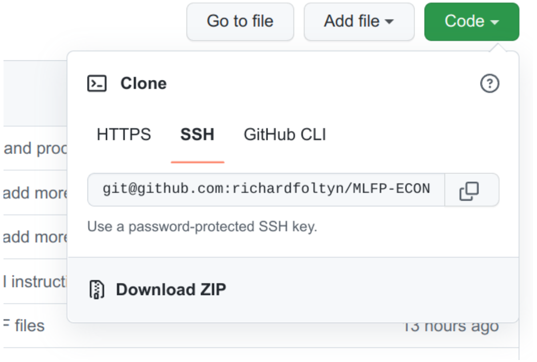
Advanced method: install locally

- **Python versions**
  - Current version of Python is 3.10, but earlier version such as 3.9 and 3.8 probably work as well
  - Do **not** use Python 2.7, it's no longer supported!
- **Python distribution**
  - The core Python language / runtime directly from the Python project (https://www.python.org/) is not particularly useful for statistics / data analysis
  - Instead use distribution that allows you to easily install required packages
  - Most popular distribution for scientific computing is Anaconda (https://www.anaconda.com/products/distribution)
    - For Windows, download 64-bit variant
    - For Linux, download 64-bit (x86) variant
    - For Mac there is only one option

Advanced method: install locally

Getting the course material

- You need to download all the content from https://github.com/richardfoltyn/MLFP-ECON5130
- For example, to download as a ZIP file:



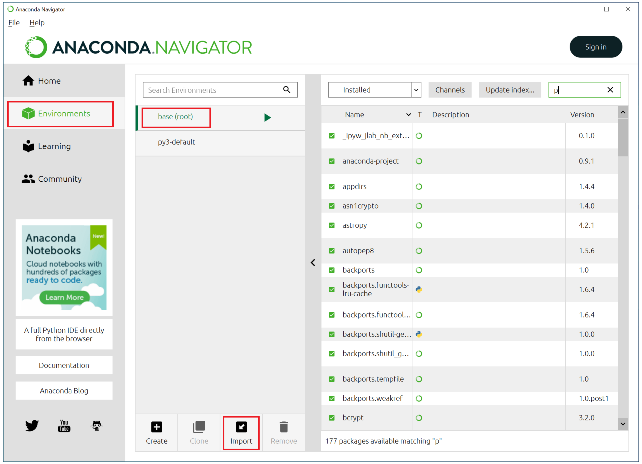
```
git@github.com:richardfoltyn/MLFP-ECON
```

Use a password-protected SSH key.

- Alternatively, you can clone the repository if you are familiar with git.

Setting up Anaconda: Microsoft Windows

Once you installed Anaconda, open the Anaconda Navigator application:



Anaconda Navigator

File Help

ANACONDA.NAVIGATOR

Sign in

Home

Environments

Learning

Community

Anaconda Notebooks New! Cloud notebooks with hundreds of packages ready to code. Learn More

A full Python IDE directly from the browser

Documentation

Anaconda Blog

Search Environments

base (root)

py3-default

Create Clone Import Remove

Installed Channels Update index... p

	Name	T	Description	Version
✓	_ipyw_jlab_nb_ext...	○		0.1.0
✓	anaconda-project	○		0.9.1
✓	appdirs	○		1.4.4
✓	asn1crypto	○		1.4.0
✓	astropy	○		4.2.1
✓	autopep8	○		1.5.6
✓	backports	○		1.0
✓	backports.functools-lru-cache	●		1.6.4
✓	backports.functool...	○		1.6.4
✓	backports.shutil-ge...	●		1.0.0
✓	backports.shutil_g...	○		1.0.0
✓	backports.tempfile	○		1.0
✓	backports.weakref	○		1.0.post1
✓	bcrypt	○		3.2.0

177 packages available matching "p"

Initially, you'll have a single Python environment called **base (root)**

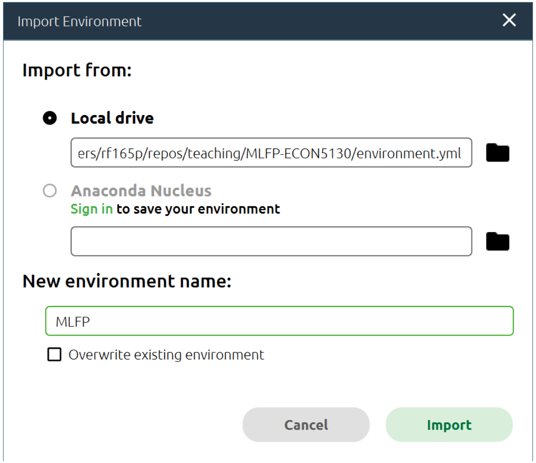
Setting up Anaconda: Microsoft Windows

To make sure you have compatible versions of Python and of various packages, it is best to create a new environment.

1. Download the environment definition file environment.yml from the GitHub repository and save it locally.
   This file defines the packages and their exact versions required for this course.
2. In Anacoda Navigator, select **Import** (see previous slide) and enter the path to the environment file you just downloaded.
3. Call the new environment **MLFP** (for Machine Learning in Finance with Python), see screenshot on next slide.

This will create a Python environment with all packages required for this course.

Setting up Anaconda: Microsoft Windows



ers/rf165p/repos/teaching/MLFP-ECON5130/environment.yml

MLFP

Setting up Anaconda: Microsoft Windows

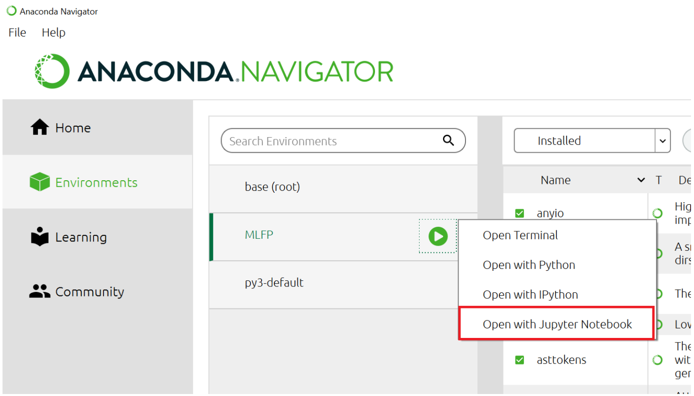
To start a browser-based Jupyter notebook that is running *locally* on your computer, use either of the following methods (see screenshots on next slide):

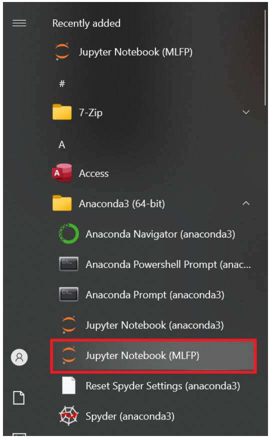
1. In Anaconda Nagivator, select the **MLFP** environment, and from the context menu pick **Open with Jupyter Notebook**
2. In the Windows Start Menu, search for and run the entry **Jupyter Notebook (MLFP)**

- Either alternative will launch your browser and open the Jupyter Notebook file manager.
- Navigate to the folder where you unzipped the git repository contents, and select a notebook from the `lectures` folder, or the overview notebook `index.ipynb`.

Setting up Anaconda: Microsoft Windows

Launching a browser-based Jupyter Notebook





Recently added
Jupyter Notebook (MLFP)
#
7-Zip
A
Access
Anaconda3 (64-bit)
Anaconda Navigator (anaconda3)
Anaconda Powershell Prompt (anac...
Anaconda Prompt (anaconda3)
Jupyter Notebook (anaconda3)
Jupyter Notebook (MLFP)
Reset Spyder Settings (anaconda3)
Spyder (anaconda3)

Setting up Anaconda: Linux

1. Once you have installed Anaconda, you need to set up an environment that contains all the packages required to run the code.
2. Use environment specification in `environment.yml` from the git repository:

   ```
   conda env create -f environment.yml
   ```

3. Active the virtual environment you just created (by default it will be called `MLFP`):

   ```
   conda activate MLFP
   ```

4. Launch a local Jupyter instance:

   ```
   cd /path/to/repository
   jupyter notebook index.ipynb
   ```

Editors

- The course material is provided as interactive notebooks within your browser.
- For more serious programming, use local Python files and an editor!
- Python source files (`*.py`) are plain-text files, so in principle you can use any editor you want.

Editors: Recommendations

- Visual Studio Code (https://code.visualstudio.com/)
    - Flexible code editor with good support for Python.
    - Official tutorial for Python programming with VS Code: https://code.visualstudio.com/docs/python/python-tutorial
    - VS Code also has excellent Jupyter Notebook support: https://code.visualstudio.com/docs/datascience/jupyter-notebooks
    - Data science tutorial with Python and VS Code: https://code.visualstudio.com/docs/datascience/data-science-tutorial
- PyCharm (https://www.jetbrains.com/pycharm/)
    - Most sophisticated integrated development environment (IDE) for Python
    - proprietary, but has free community edition; professional edition is free for educational purposes
    - Use only if you are an experienced programmer, or willing to invest some time