Installing Python packages[1]

Python is available for all three major operating systems — Microsoft Windows, macOS, and Linux — and the installer, as well as the documentation, can be downloaded from the official Python website: https://www.python.org.

It is recommended you use the most recent version of Python 3 that is currently available, although most of the code examples may also be compatible with older versions of Python 3.

Note

You can check your current default version of Python by executing

```
$ python -V
```

In my case, it returns

```
Python 3.6.1 :: Continuum Analytics, Inc.
```

Pip

The additional packages that we will be using throughout this book can be installed via the `pip` installer program, which has been part of the Python standard library since Python 3.3. More information about pip can be found at https://docs.python.org/3/installing/index.html.

After we have successfully installed Python, we can execute pip from the command line terminal to install additional Python packages:

```
pip install SomePackage
```

(where `SomePackage` is a placeholder for numpy, pandas, matplotlib, scikit-learn, and so forth).

Already installed packages can be updated via the `--upgrade` flag:

```
pip install SomePackage -upgrade
```

Anaconda

A highly recommended alternative Python distribution for scientific computing is Anaconda by Continuum Analytics. Anaconda is a free—including commercial use—enterprise-ready Python distribution that bundles all the essential Python packages for data science, math, and engineering in one user-friendly cross-platform distribution. The Anaconda installer can be downloaded at http://continuum.io/downloads#py34, and an Anaconda quick start-guide is available at https://store.continuum.io/static/img/Anaconda-Quickstart.pdf.

After successfully installing Anaconda, we can install new Python packages using the following command:

```
conda install SomePackage
```

Existing packages can be updated using the following command:

```
conda update SomePackage
```

[1] Adapted from https://github.com/rasbt/python-machine-learning-book-2nd-edition/blob/master/code/ch01/README.md

Throughout this book, we will mainly use NumPy's multi-dimensional arrays to store and manipulate data. Occasionally, we will make use of pandas, which is a library built on top of NumPy that provides additional higher-level data manipulation tools that make working with tabular data even more convenient. To augment our learning experience and visualize quantitative data, which is often extremely useful to intuitively make sense of it, we will use the very customizable matplotlib library.

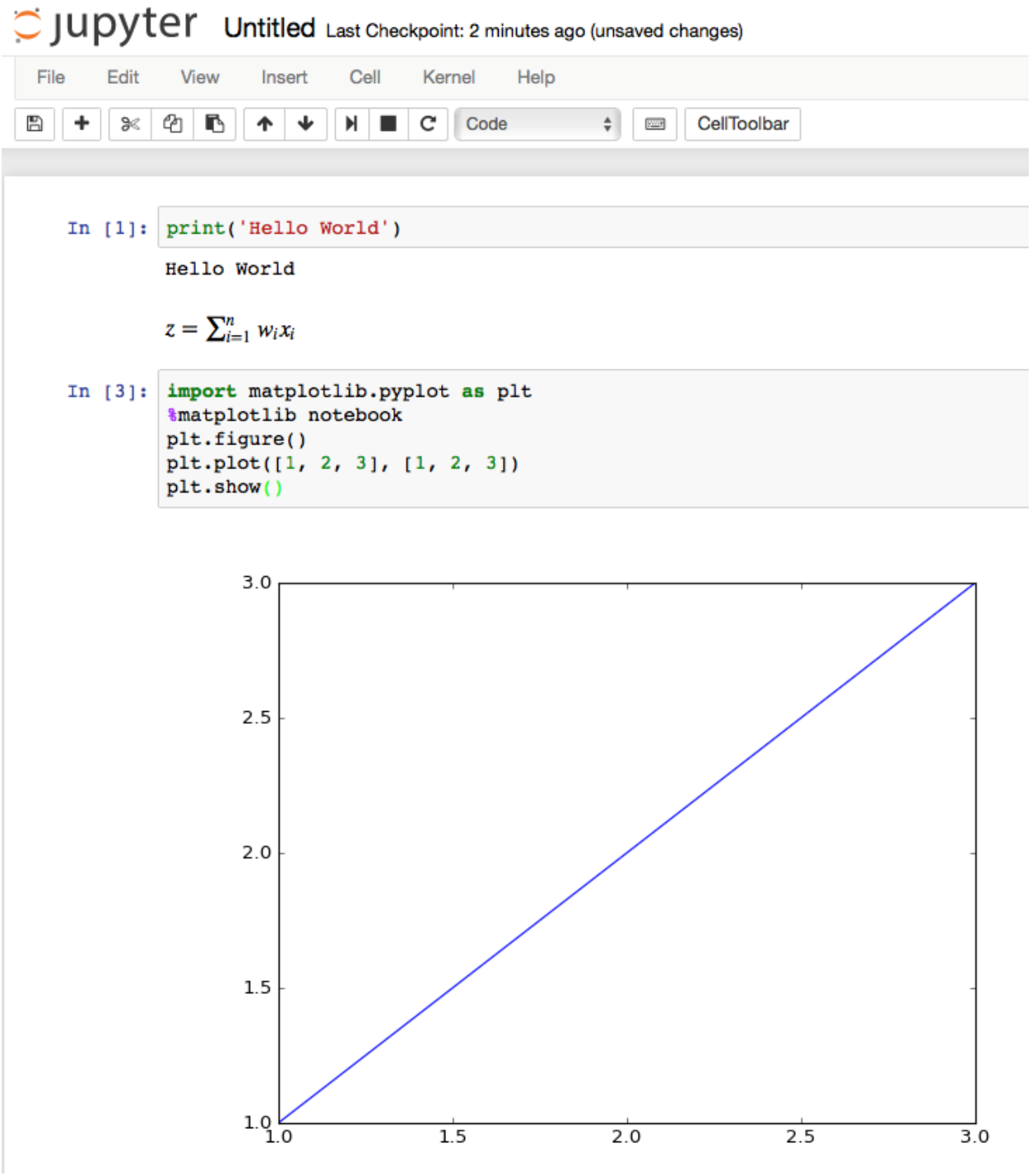
Core packages

The version numbers of the major Python packages that were used for writing this book are listed below. Please make sure that the version numbers of your installed packages are equal to, or greater than, those version numbers to ensure the code examples run correctly:

- NumPy >= 1.12.1
- SciPy >= 0.19.0
- scikit-learn >= 0.18.1
- matplotlib >= 2.0.2
- pandas >= 0.20.1

Python/Jupyter Notebook

Some readers were wondering about the `.ipynb` of the code files -- these files are IPython notebooks. I chose IPython notebooks over plain Python `.py` scripts, because I think that they are just great for data analysis projects! IPython notebooks allow us to have everything in one place: Our code, the results from executing the code, plots of our data, and documentation that supports the handy Markdown and powerful LaTeX syntax!



Side Note: "IPython Notebook" recently became the "Jupyter Notebook"; Jupyter is an umbrella project that aims to support other languages in addition to Python including Julia, R, and many more. Don't worry, though, for a Python user, there's only a difference in terminology (we say "Jupyter Notebook" now instead of "IPython Notebook").

The Jupyter notebook can be installed as usually via pip.

```
$ pip install jupyter notebook
```

Alternatively, we can use the Conda installer if we have Anaconda or Miniconda installed:

```
$ conda install jupyter notebook
```

To open a Jupyter notebook, we `cd` to the directory that contains your code examples, e.g,.

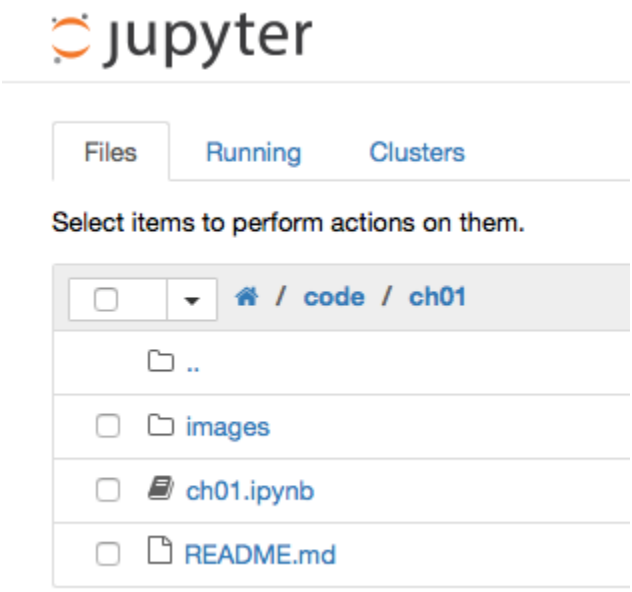
```
$ cd ~/code/python-machine-learning-book
```

and launch `jupyter notebook` by executing

```
$ jupyter notebook
```

Note that if you use a virtual environment (venv) version of the Python interpreter, activate it first by using, e.g.:
```
$ activate py38_venv
```

Jupyter will start in our default browser (typically running at http://localhost:8888/). Now, we can simply select the notebook you wish to open from the Jupyter menu.



For more information about the Jupyter notebook, I recommend the Jupyter Beginner Guide.

A good brief intro in Jupyter Notebooks is here.

IDEs for Jupyter Notebook Development

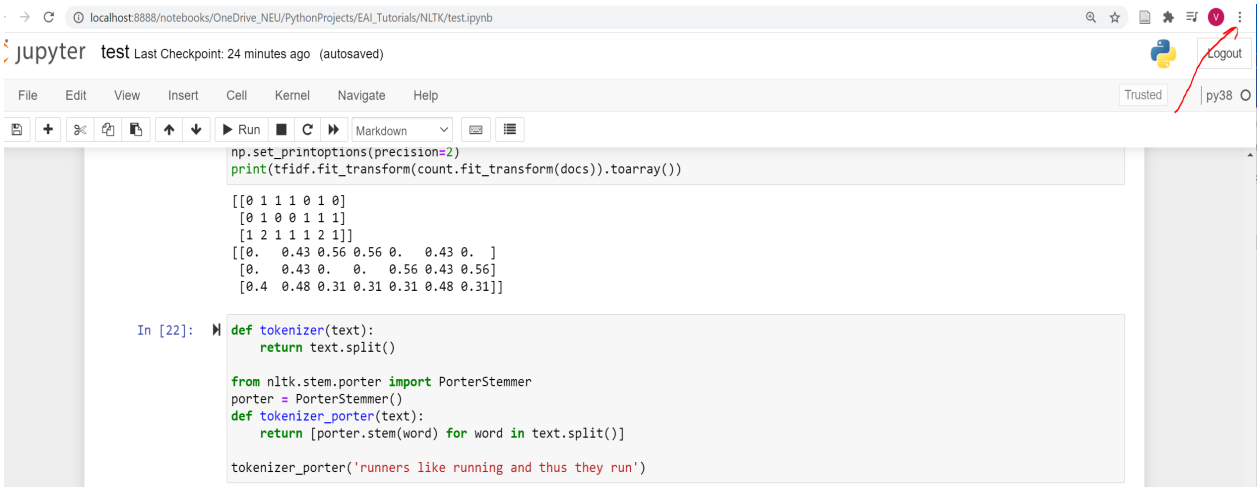
Integrated Development Environment (IDE) is a source code editor combined with the code execution and debugging tool. It allegedly helps increasing code development productivity. Among the popular IDEs there are:

- JetBrains DataSpell (student version is available)
- Visual Studio Code (free, open source)
- JupyterLab (free, open source)

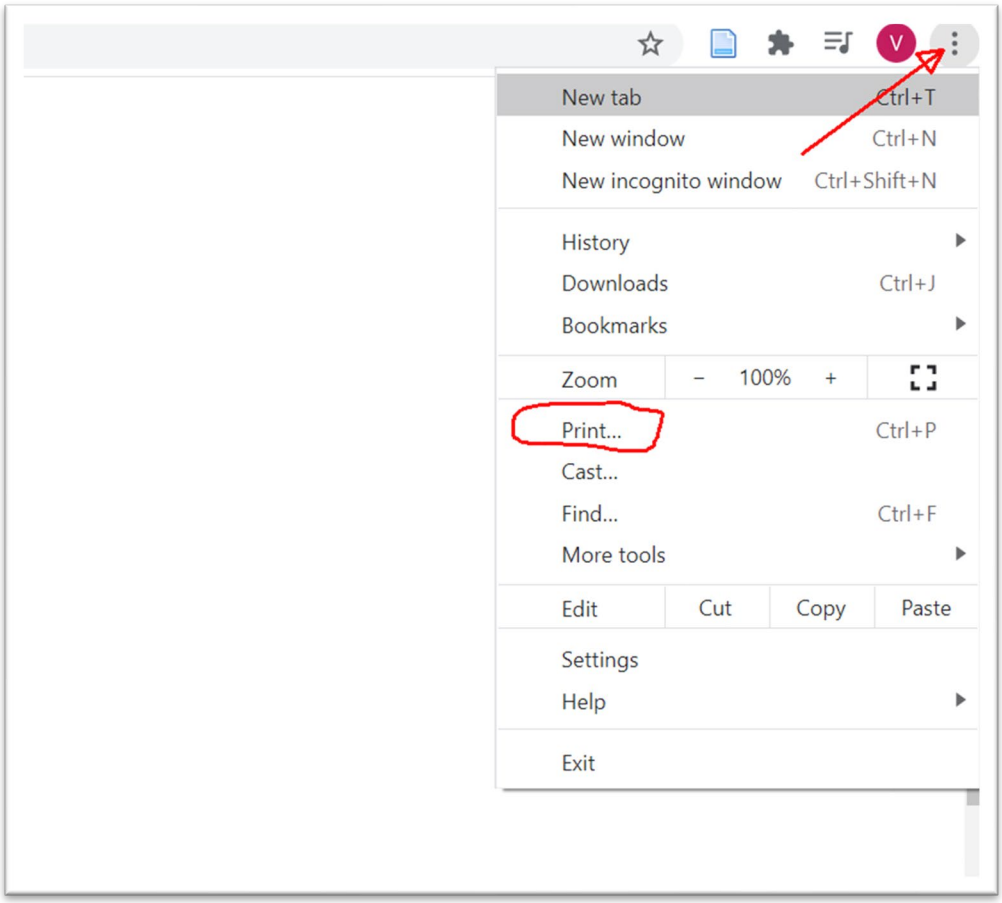
Create PDF output off of the Jupyter Notebook

Method A:

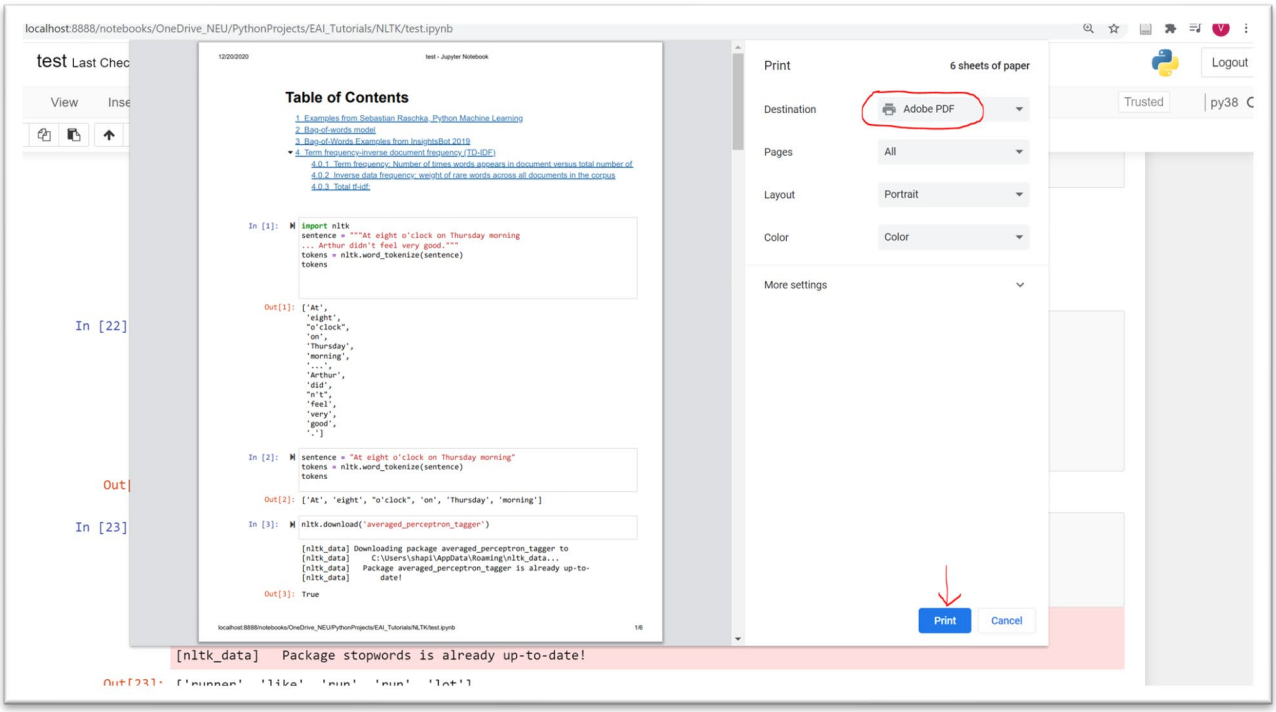
1. While on the Jupyter Notebook browser tab, select the browser settings. See below in Chrome shown by the red arrow:



2. Select "**Print**" in the drop-down menu



3. Select “**Adobe PDF**” option, followed by “**Print**”



Method B:

1. For conversion of Jupyter Notebook to PDF follow this link

Google Colaboratory

1. Read https://colab.research.google.com/notebooks/intro.ipynb
2. Navigate to and open a Jupyter notebook on your Google Drive
3. Run as if you would on your local machine