Setup

January 28, 2020

[1]:
```
from IPython.core.interactiveshell import InteractiveShell
InteractiveShell.ast_node_interactivity = "all"

# Reload all modules imported with %aimport
%load_ext autoreload
%autoreload 1

%matplotlib inline
```

- We will use the Anaconda Distribution
  - simple to install
  - comprehensive
  - package and environment management
    - No more `pip install` followed by dependency chasing

0.1 Set up a ML environment

Several choices

- Anaconda on your own machine
- Anaconda on AWS (or Azure, Google)
- Turn-key, cloud solution: Floydhub
- Turn-key, cloud solution: Paperspace

0.1.1 Anaconda on your own machine

- Pro: cheapest
- Con: potential limited by memory and power of your machine

- Link: Download Anaconda and run installer
  - if no browser available
    - save link, e.g., https://repo.continuum.io/archive/Anaconda3-2018.12-Linux-x86_64.sh
    - use wget on the link: >`wget https://repo.continuum.io/archive/Anaconda3-2018.12-Linux-`
    - run the downloaded file: `bash Anaconda3-2018.12-Linux-x86_64.sh`
  - accept defaults
    - allow your .bashrc to be updated:
      - or can do it later yourself: `source .bashrc`

0.1.2 Anaconda on AWS

- Same setup as Anaconda on your own machine **ONCE** you have knowledge of how to create machines on AWS
- Pro:
  - high potential: you can rent machines with increased power, memory and GPU !
  - knowing how to use a cloud services (AWS, Azure, Google) is a valuable skill !
- Con: Free-tier machine good to start but need to rent resouces (i.e., money)

Links

- Grant McKinnon
  - Setting up AWS for Kaggle

0.1.3 Floydhub

- Pro:
  - **Turn-key** and cloud-based. No installation to start
- Con:
  - Best as a Jupyter notebook server, not as a full-service machine
    - You WILL want a text editor at some point, particularly as you develop Python Classes/Modules

0.1.4 Paperspace

- Pro:
  - Notebook server is **Turn-key** and cloud-based. No installation to start
  - "ML in a box": Full featured machine (like AWS)
    - BUT only advanced frameworks (e.g., TensorFlow) not sklearn !
- Con:
  - Cost to rent is much higher than AWS
    - n.b., some providers use AWS as a back-end; you pay for convenience