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

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

Anaconda on your own machine

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

- Link: Download Anaconda (https://www.anaconda.com/download/#linux) and run installer
    - if no browser available
        - save link, e.g., https://repo.continuum.io/archive/Anaconda3-2018 Linux-x86_64.sh (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-x86_64.sh
```

        - 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`

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 (http://www.grant-mckinnon.com/)
    - Setting up AWS for Kaggle (http://www.grant-mckinnon.com/?p=6%E2%80%8B)

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

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