Recitation 1: Python for Machine Learning

Rohan Choudhury
rchoudhury@caltech.edu

Caltech

Winter 2018
Audience

- Are you very experienced with Python?
- Have you used numpy/matplotlib for your work before?
- Did you take 156a and do well?

If you answered yes you probably don’t need to be here
Goals

This recitation will be pretty short and easy

- Get situated with coding expectations
- Installation
- Useful packages + Examples
- General tips
Assumptions + Expectations

- You have to write (usually) significantly more code (in Python) in this class than in 156a.

- We assume you already know Python.
- Basic competency with UNIX/Linux (can use a terminal)
- Write clean, efficient, readable code!
Installing Python

- Python 2 and Python 3 are fine.
- Let us know if you use Python 3 (with a comment in your code)
- https://www.python.org/downloads/
Packages

- You can use a package manager:
  https://www.anaconda.com/download
- Or you can use pip: pip install numpy

You need:
- numpy
- scikit-learn
- matplotlib

Using git and \LaTeX{} is recommended :v)
numpy

- Used for numerical computing/matrix operations
- Your data is going to be in a matrix, so manipulate it with numpy
scikit-learn

- Used for basic ML algorithms, tools and techniques
- You’ll get to use this sometimes
- Can’t do neural nets
matplotlib

- Use it to plot stuff
- You will need to make plots on every set
Jupyter Notebooks

- You can install with pip or use anaconda: http://jupyter.readthedocs.io/en/latest/install.html
- It comes with anaconda
- Excellent for writing code incrementally/testing as you go; used in the homework assignments
Review: The Supervised Learning Recipe

the recipe:

▶ get training data
▶ pick a model class
▶ pick a loss function
▶ pick a learning objective to optimize
Review: $K$-fold cross validation

- How to pick a model set?
- Approximate generalization error
- Idea: validation sets
Algorithm: $k$ fold cross validation

- For 1, 2, ... $k$
  - Use the first $k$th of data as validation, and train on the remaining data points.
  - Evaluate error on the validation set, and store it.
- Average the validation errors and return this as the $k$-fold cross validation error.

How to implement this? Not hard, but it requires work :(
Debugging Tips

- Google it
- Print it
- Try using dummy data
- Ask for help!
- Take a nap
Coding Resources

- Remember: Stack Overflow is your best friend!
- Numpy tutorial:
  https://cs231n.github.io/python-numpy-tutorial/
- Numpy polyfit, polyval (for the set!)
- sklearn’s kfold method