# An Introduction to Using Python with Data

Patrick Hall

jpatrickhall@gmail.com

# Course Goals

- To communicate a general understanding of software languages and their uses in processing and analyzing data.

- To convey specific knowledge regarding the Python language through examples related to processing and analyzing data.

# Course Overview

- Software languages and programming for processing and analyzing data
- Python
  - Introduction
  - Basic operations and strings
  - Controlling the flow of a program
  - Reading and writing files
  - Data structures
  - Defining your own functions
  - Scraping data from the web
  - Numerical Python (NumPy) and data analysis
  - Plotting Results and IPython

# Introductions

- About me …
  - Research Statistician Developer for SAS Enterprise Miner
    http://www.sas.com/en_us/software/analytics/enterprise-miner.html
  - Cloudera Certified Data Scientist
    http://www.cloudera.com/content/cloudera/en/training/certification/ccp-ds.html
  - Follow me on Quora and Github.


# Introductions

- About you …
  - Your education and experience in processing and analyzing data.
  - Your education and experience with programming and Python.
  - Your goals for this class.

# Preliminary Course Instructions

As a group …

1. Download course materials

   https://github.com/jphall663/bellarmine_py_intro/archive/master.zip

2. Download Anaconda Python version 2.0.1

   http://repo.continuum.io/archive/index.html

3. Install Anaconda Python version 2.0.1
4. Set working directory in Spyder IDE

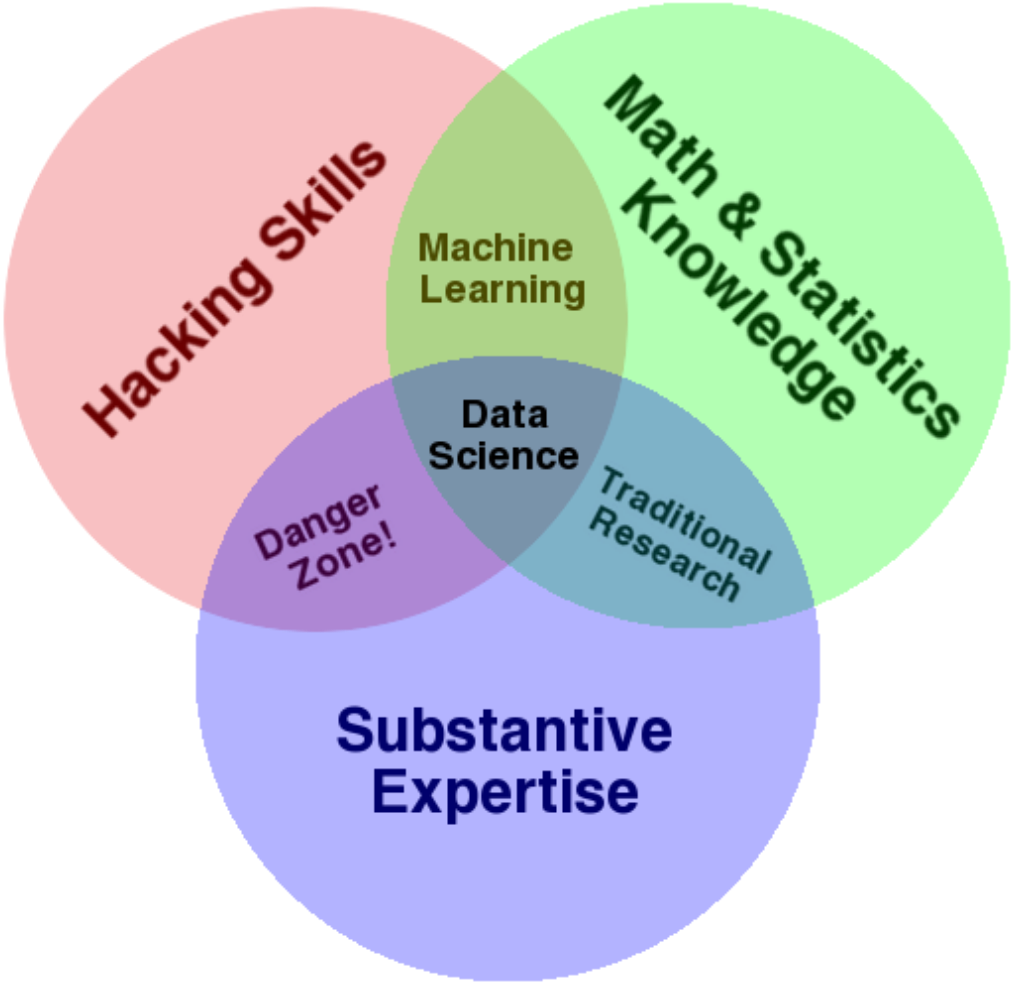
# Course Logistics

- Schedule
- Course Materials
- Python Documentation

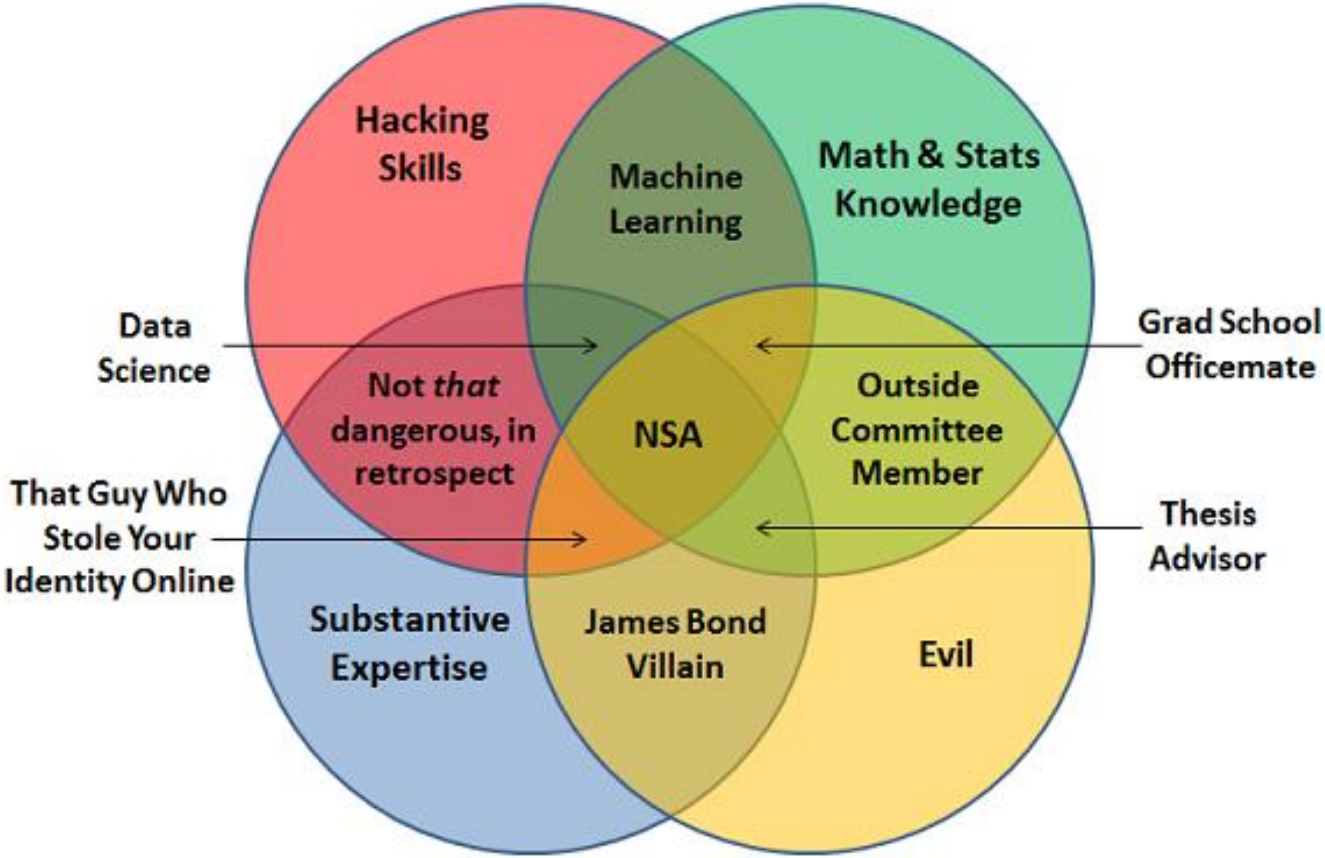
  https://docs.python.org/2/tutorial/
- Questions and Discussions
- Hands-on Examples

Break time.

# Software Languages and Programming for Data Processing and Analysis.




http://drewconway.com/zia/2013/3/26/the-data-science-venn-diagram




http://joelgrus.com/2013/06/09/post-prism-data-science-venn-diagram/

# Computer Languages

- TIOBE Index

http://www.tiobe.com/index.php/content/paperinfo/tpci/index.html

- Speed Benchmark

http://benchmarksgame.alioth.debian.org/

# Computer Languages

Why are they different?

- Compiled vs. Interpreted
- Compiled languages are turned into (“compiled”) 1’s and 0’s and then run.
- Compiled languages are usually faster, but harder to develop.
- http://en.wikipedia.org/wiki/Compiled_language

# Computer Languages

Why are they different?

- Compiled vs. Interpreted
- Interpreted languages are turned into 1's and 0's and run at the same time.
- Interpreted languages are usually slower, but easier to develop.
- http://en.wikipedia.org/wiki/Interpreted_language
- Python is an interpreted language.

# Computer Languages

Why are they different?

- General Purpose vs. Domain specific
- General purpose languages can be used to build almost any kind of application.
- General purpose languages usually have a steep learning curve and are more difficult to develop.
- Python is a general purpose language.

# Computer Languages

Why are they different?

- General Purpose vs. Domain specific
- Domain specific languages can be used only for specific purposes, like data analysis.
- Domain specific languages are usually easier to learn and develop (within their domain).
- Python has many large libraries that make it feel like a domain specific language.

# Computer Languages

Why are they different?

- Procedural vs. Object Oriented (Paradigms)
- Object oriented (OO) code is usually easier to understand and maintain over many years, but more difficult to develop.
- Python is a multi-paradigm language.

# Computer Languages

Why are they different?

- Procedural vs. Object Oriented (Paradigms)
- Procedural code is usually harder to understand and maintain over many years, but easier develop.
- Obfuscated C Code Contest:

  http://www.ioccc.org/years-spoiler.html
- Python is a multi-paradigm language.

# Computer Languages

Why are they different?

- Procedural vs. Object Oriented vs. Functional (Paradigms)
- Functional programming is a special type of programming paradigm that is theoretically well-suited for analyzing large data sets.
- It is usually simpler to develop than OO code and easier to maintain than procedural code.
- Python is not really a functional language.

# Computer Languages

Why are they different?

- Procedural vs. Object Oriented vs. Functional (Paradigm)
- *Hadoop* is a very popular framework for processing and analyzing big data. It was inspired by the functional programming paradigm:
  - Google MapReduce paper:

    http://static.googleusercontent.com/media/research.google.com/en/us/archive/mapreduce-osdi04.pdf
  - Apache Hadoop homepage:

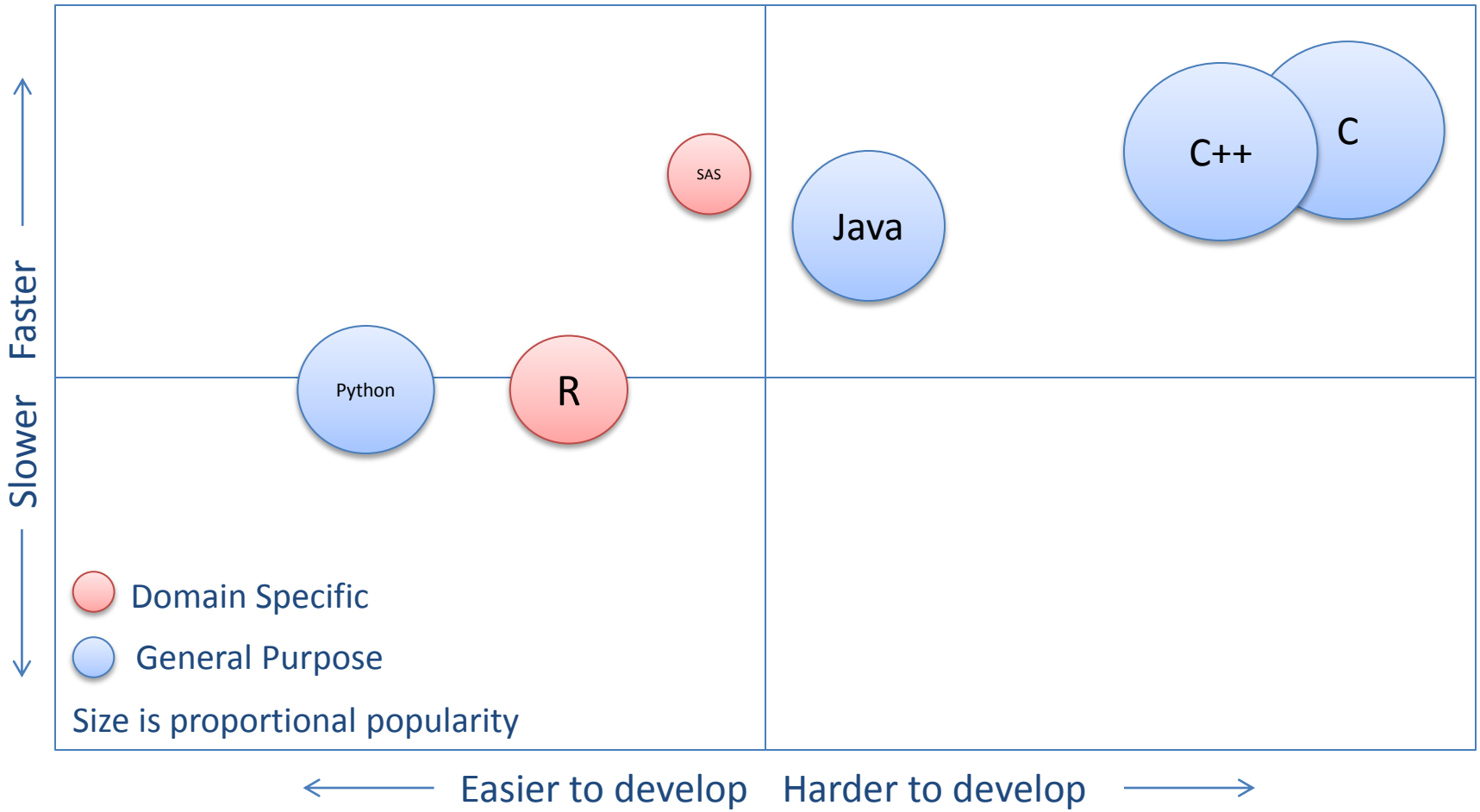
    http://hadoop.apache.org/

# Computer Languages

Why are they different?

- Procedural vs. Object Oriented vs. Functional (Paradigm)
- Languages often associated with the functional paradigm:
  - Scala
  - Clojure
  - Haskell
  - Lisp

http://xkcd.com/224/


Faster
Slower
SAS
Java
C++
C
Python
R
Domain Specific
General Purpose
Size is proportional popularity
Easier to develop
Harder to develop

# A Brief History of Python

- Python was created by the Dutch computer scientist Guido van Rossum (BDFL) in the late 1980s.
- It is rumored to be named after Monty Python.
- Python 2 was released in 2000. (We are using Python 2.7.)
- Python 3 was released in 2008. Python 3 is not completely backward compatible with Python 2.

# Why Python?

- Python is a general purpose, object oriented language that is easy to develop.

# Why Python?

- Python has so many libraries and modules that it usually feels more like a domain specific language.
  - NLTK: http://www.nltk.org/
  - SciPy: http://www.scipy.org/
  - Bokeh: http://bokeh.pydata.org/
  - 195+ packages included in Anaconda: http://docs.continuum.io/anaconda/pkg-docs.html

# Why Python?

- Python is not that slow … for an interpreted language.
- Python can interface with faster, compiled languages for doing computationally intensive tasks.

# Why Python?

- Python has a very expressive (“Pythonic”) syntax.

Exercise 0

http://legacy.python.org/dev/peps/pep-0020/

>>> import this

Break time.

# Python: Basic Operations and Strings.

# Section Goals

https://docs.python.org/2/tutorial/introduction.html

- The interactive shell
- Operations for assignment and comparison
- Strings
- Escape Characters
- Slicing

https://docs.python.org/2/library/stdtypes.html#string-methods

- String functions

# Exercise 1

- Working With Strings

# Controlling the Flow of Your Python Program.

# Section Goals

https://docs.python.org/2/tutorial/controlflow.html

- `if` statements
- `for` statements
- `break` and `continue` statements
- `pass` statements

- `enumerate` statements

# Reading and Writing Files with Python.

# Section Goals

https://docs.python.org/2/tutorial/inputoutput.html#reading-and-writing-files

- Opening and closing files
- File modes

https://docs.python.org/2/reference/compound_stmts.html

- `with` statements

- Combining `for` loops, `if` statements and file operations to read and write files.

# Exercise 2

- Loops and File I/O

# Basic Data Structures in Python.

# Section Goals

https://docs.python.org/2/tutorial/datastructures.html

- Lists
- List Comprehensions
- Sets
- Dictionaries
- Looping Techniques
- Conditions

# Section Goals

https://docs.python.org/2/library/collections.html

- Counters

# Exercise 3

- Lists, Dictionaries and Sets

# Defining Your Own functions.

# Section Goals

https://docs.python.org/2/tutorial/controlflow.html#defining-functions

- Defining functions

# Scraping Data from the Web.

# Section Goals

https://docs.python.org/2/howto/urllib2.html

- `urllib2` fetches HTML and other data from websites
- Fetching URLs using the `urlopen` function
- Reading information from an URL using the `read` function

http://www.crummy.com/software/BeautifulSoup/bs4/doc/

- `BeautifulSoup` parses HTML into more meaningful data
- `prettify` function
- `get_text` function
- `find_all` function

# Section Goals

- Data Sources on the Web

# Exercise 4

- Scraping Data from the Web

# Numerical Python (NumPy) and Data Analysis.

# Section Goals

http://wiki.scipy.org/Tentative_NumPy_Tutorial

- What is NumPy?
- The Basics:
  - NumPy Arrays
  - Basic Array Operations
  - Indexing, Slicing and Iterating
- Iteration vs. vector operations

# Section Goals

https://docs.python.org/2/library/csv.html

- CSV and delimited data
- Reading CSV data using the `csv` module
- Potential problems with CSV data

# Section Goals

http://www.kaggle.com/c/titanic-gettingStarted

- What is Kaggle?
- What is predictive modeling?
- The famous Titanic data set

# Section Goals

- Numpy data types
- Masking arrays

# Exercise 5

- Numpy: Kaggle Titanic Competition

# Plotting Results and IPython.

# Section Goals

http://matplotlib.org/

- solution_6.py
- Adding values to a plot
- Decorating a plot
- Magic Numbers
- `matplotlib` examples

# Section Goals

- Starting an IPython session
- Creating an IPython notebook
- Sharing an IPython notebook using GitHub
  - https://gist.github.com/
  - http://nbviewer.ipython.org/
  - http://nbviewer.ipython.org/github/jphall663/bellarmine_py_intro/blob/master/Titanic.ipynb

# Exercise 6

- IPython: Graphing Results

# Additional Resources

- SAS University Edition (FREE!)

http://www.sas.com/en_us/software/university-edition.html

- SAS on Demand for Academics

http://www.sas.com/en_us/industry/higher-education/on-demand-for-academics.html

- SAS Data Mining Community

https://communities.sas.com/community/support-communities/sas_data_mining_and_text_mining/

- "Overview of Machine Learning with SAS Enterprise Miner"

http://support.sas.com/resources/papers/proceedings14/SAS313-2014.pdf
http://support.sas.com/rnd/papers/sasgf14/313_2014.zip

- Sparse Data

https://communities.sas.com/docs/DOC-5323
http://support.sas.com/resources/papers/proceedings14/SAS195-2014.pdf

- Certifications, documentation, training, videos, and more

http://support.sas.com

- Products Page

http://www.sas.com/en_us/insights/analytics/machine-learning.html

# Additional Resources

- **“Big Data, Data Mining, and Machine Learning”**

http://www.sas.com/store/prodBK_66081_en.html

- **Cloudera data science study materials**

http://www.cloudera.com/content/dev-center/en/home/developer-admin-resources/new-to-data-science.htmlhttp://cloudera.com/content/cloudera/en/training/certification/ccp-ds/essentials/prep.html

- **Kaggle data mining competitions**

http://www.kaggle.com/

- **Python machine learning packages**

**OpenCV:** http://opencv.org/
**Pandas:** http://pandas.pydata.org/
**Scikit-Learn:** http://scikit-learn.org/stable/
**Theano:** http://deeplearning.net/software/theano/

# Additional Resources

- **R machine learning task view**

http://cran.r-project.org/web/views/MachineLearning.html

- **Quora: list of large public data sets**

http://www.quora.com/Where-can-I-find-large-datasets-open-to-the-public

- **Quora: list of data mining and machine learning papers**

http://www.quora.com/Data-Mining/What-are-the-must-read-papers-on-data-mining-and-machine-learning

The end.