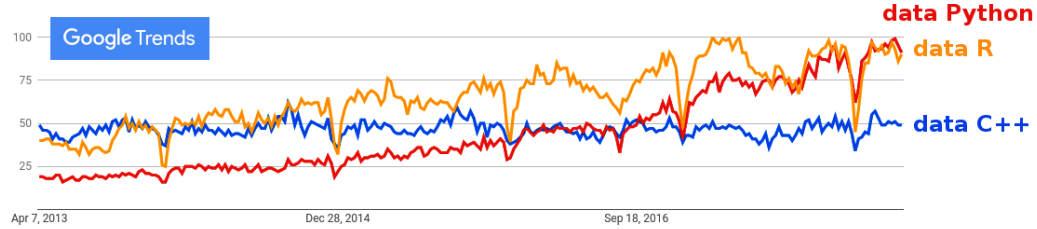
Scientific Python Ecosystem HATS

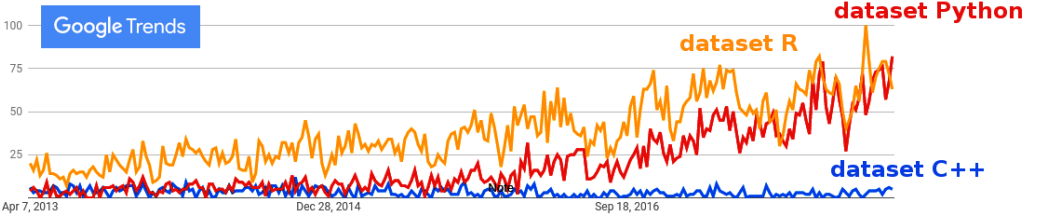
Jim Pivarski

Princeton University – DIANA-HEP

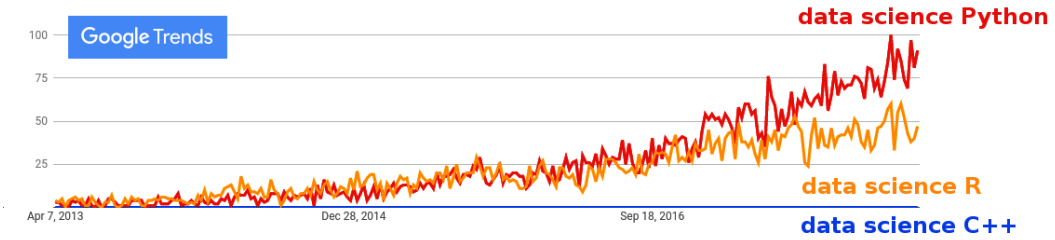
June 4, 2018

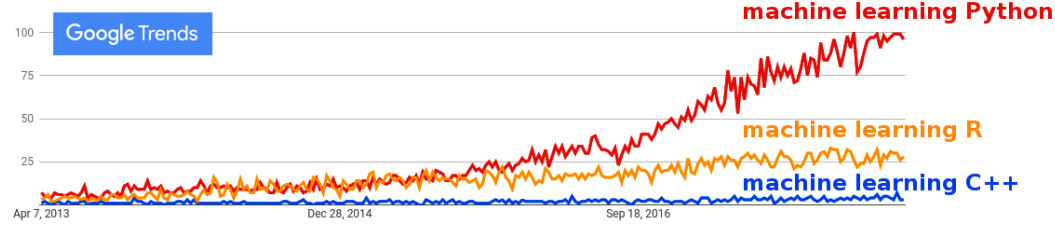
A lot of data analysis is in Python these days





A lot of data analysis is in Python these days





The “scientific Python ecosystem”

The word “ecosystem” is deliberately vague: no clear boundaries, but a few key components and many projects that share conventions.

simple, clean language
good C/C++ interface

The “scientific Python ecosystem”

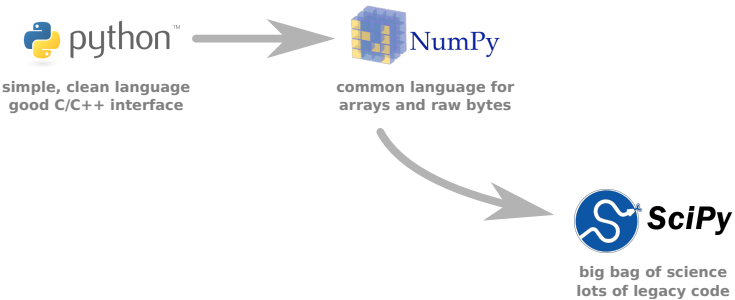
The word “ecosystem” is deliberately vague: no clear boundaries, but a few key components and many projects that share conventions.

simple, clean language
good C/C++ interface

common language for
arrays and raw bytes

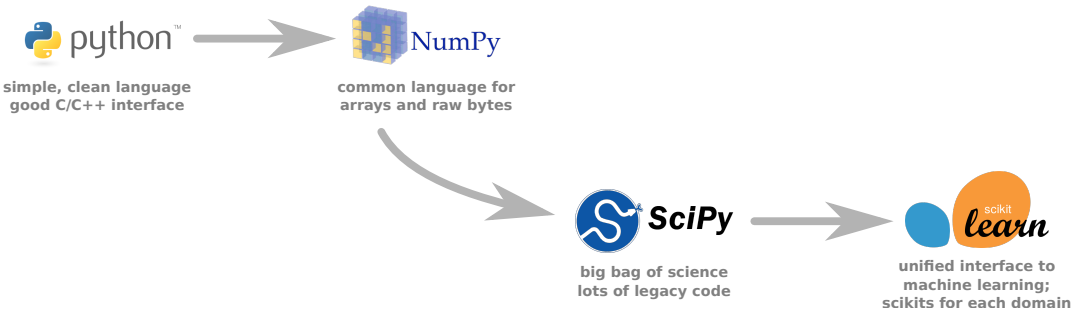
The “scientific Python ecosystem”

The word “ecosystem” is deliberately vague: no clear boundaries, but a few key components and many projects that share conventions.



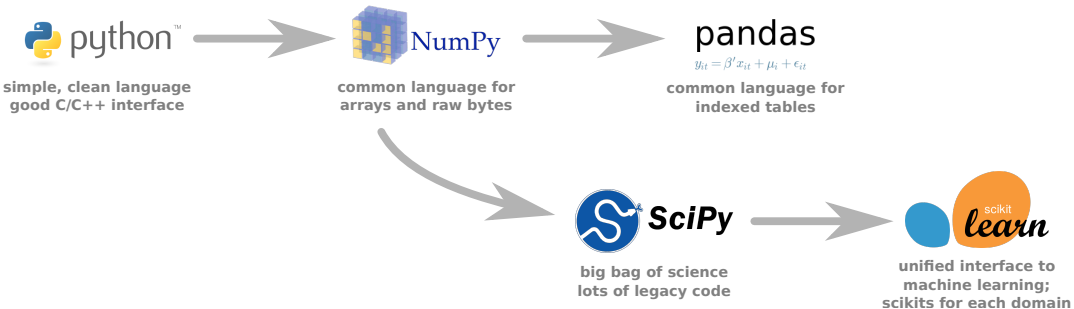
The “scientific Python ecosystem”

The word “ecosystem” is deliberately vague: no clear boundaries, but a few key components and many projects that share conventions.



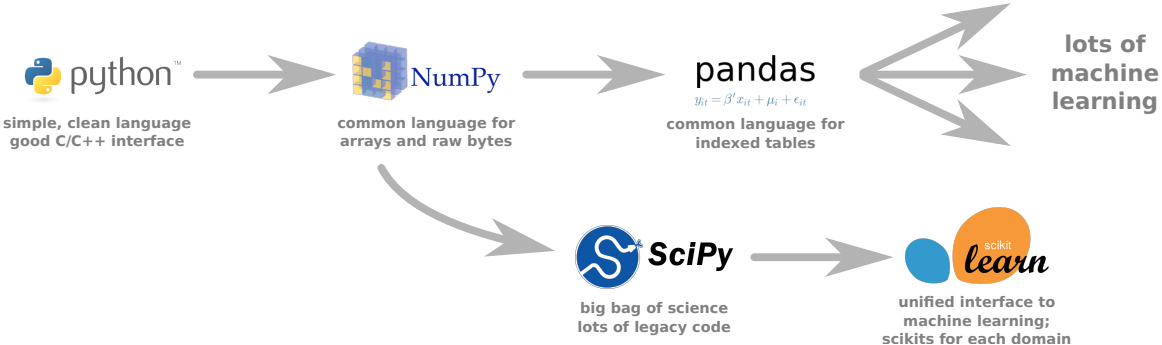
The "scientific Python ecosystem"

The word "ecosystem" is deliberately vague: no clear boundaries, but a few key components and many projects that share conventions.



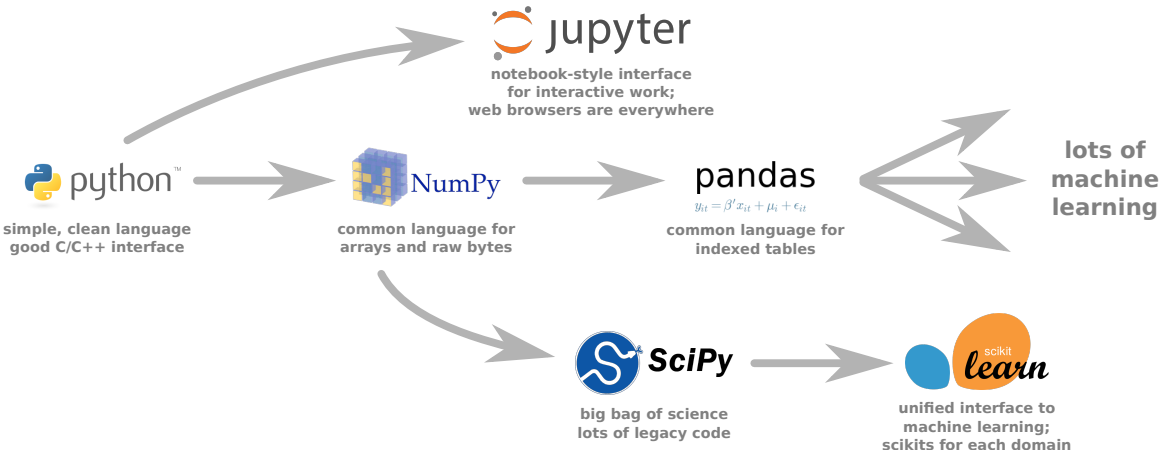
The "scientific Python ecosystem"

The word "ecosystem" is deliberately vague: no clear boundaries, but a few key components and many projects that share conventions.



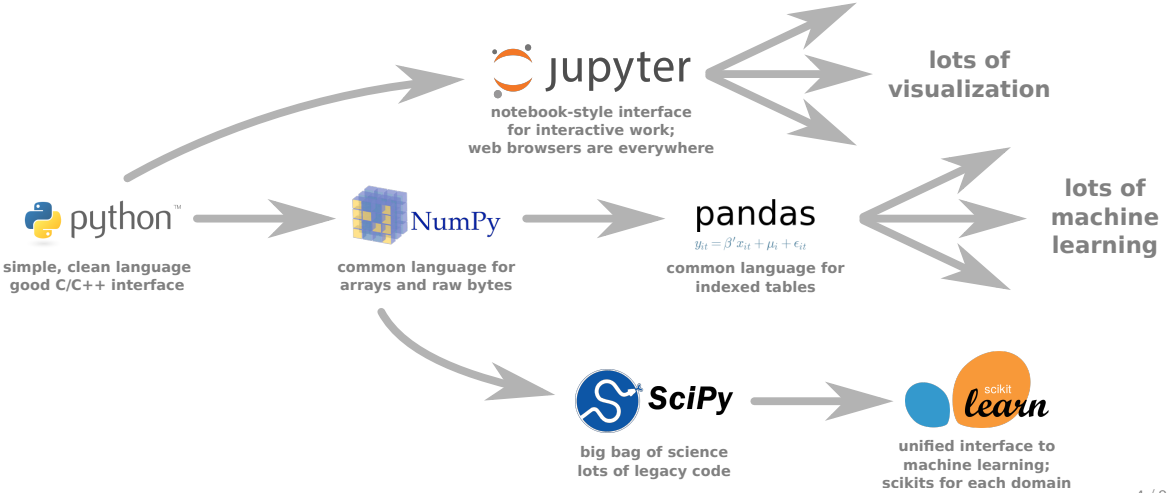
The "scientific Python ecosystem"

The word "ecosystem" is deliberately vague: no clear boundaries, but a few key components and many projects that share conventions.



The "scientific Python ecosystem"

The word "ecosystem" is deliberately vague: no clear boundaries, but a few key components and many projects that share conventions.



Some of what we need is readily available in this ecosystem

What we can find off-the-shelf	What we must develop in-house
fast calculations	reading/writing ROOT files
fast data sharing with C/C++	integration with collaboration frameworks
libraries of special functions, matrix math	advanced histogramming
fitting/minimization, integration, differentiation, interpolation	efficient variable-length lists (e.g. number of electrons per event)
symbolic algebra	HEP functions (e.g. boost 4-vectors)
advanced statistics	HEP-style limit setting
machine learning	
graphics, advanced plotting	
graphical interfaces, user workflows	
parallel and distributed processing	

Tutorials

1. numpy: common language for bulk data exchange. Introduction to vectorization and fancy indexing.
2. uproot*: implementation of ROOT I/O in Python and Numpy. Quick way to get your data into Numpy and everything else. Includes PyTorch example.
3. histbook*: HEP-style histogramming in Numpy, Pandas, and Vega (web graphics).
4. numba and oamap*: Numba is a just-in-time Python compiler for number crunching. OAMap ("object-array mapping") extends it for HEP-style data.
5. and more: SymPy, iminuit, GooFit, NumPythia...

*part of my work to integrate HEP and the scientific Python ecosystem