Open Source for (Data) Science

Andreas Müller
Associate Research Scientist
Columbia University
Scikit-learn Technical Committee

Hi, I'm Andreas Mueller. I'm grateful for the opportunity to talk to you today about my work here at the DSI, and about Open Source for Science and Data Science.

I'm an Associate Research Scientist, and I mostly work on open source software for machine learning, in particular the scikit-learn library.

My talk will have two parts: first, I want to talk a bit about the importance of software for science, and the scientific python ecosystem.

Then I'll talk more about the project I'm working on, scikit-learn, and my group here at Columbia.

Very briefly some background on myself. I did my PhD in Machine learning at the University of Bonn in Germany. Then I worked for the Amazon Machine Learning Research lab in Germany in Berlin. During my PhD I had started contributing to open source, and I couldn't really do that well while at Amazon.

I was then lucky to get a position at NYU as a Research Engineer as part of the moore-sloan data science environment, where I continued my open source work and also co-authored this book about machine learning with python.

After two years at NYU I moved to the DSI, first as lecturer and now as Associate Research Scientist on a soft-money position.

Why Software Matters

> “An article about computational science in a scientific publication is not the scholarship itself, it is merely advertising of the scholarship.
>
> The actual scholarship is the complete software development environment and the complete set of instructions which generated the figures.”
>
> - Buckheit and Donoho (1995)

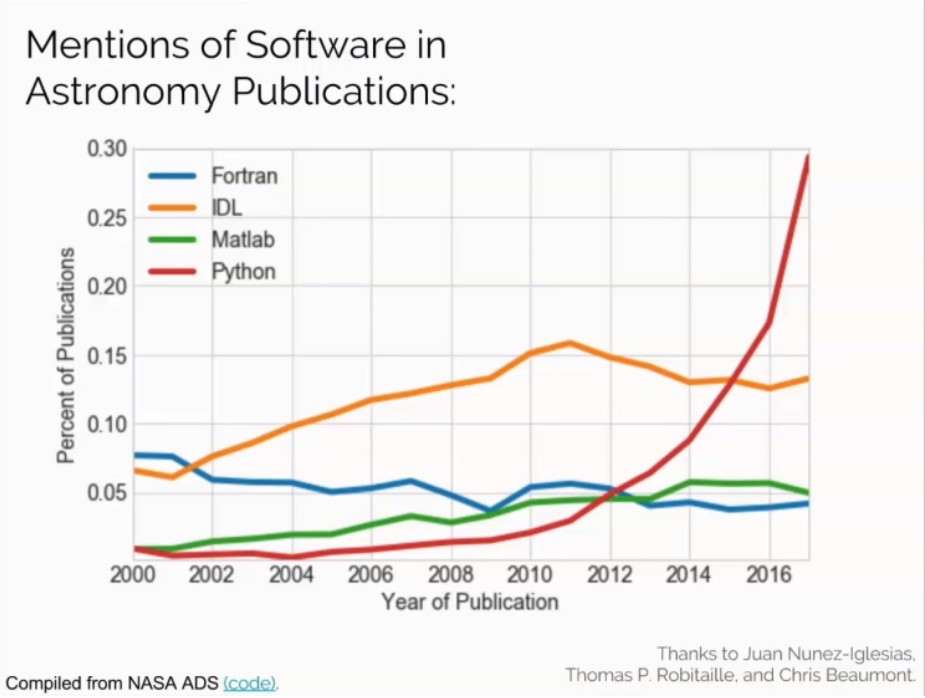
I care a lot about software, because I think software is an important tool for science.

Here is a well-known quote on reproducible research that I really believe in. It says (…)

The actual scholarship is the software, and the data. And so how this software is written is very important to how good the science is. In particular, bad software leads to bad science, while good, and easy to use tools enable good science and reproducible science.

In particular, we want open software, that’s available for everybody, and that can be reviewed and inspected. This is why I and others push for the adoption of open source principles in scientific software – and are quite successful.

I see myself very much as a tool-builder that provides tools for others to do science. I don’t do a lot of research myself, but the software I build is used by many researchers across disciplines.



The software I'm working on is part of what's known as PyData or the scientific python ecosystem. This ecosystem has found tremendous adoption in science. Here is an example from Astronomy, where there has been spectacular adoption.

People move away from low-level languages like Fortran and C++ towards python because it is much easier to use and integrates better into a scientific workflow. People also move away from Matlab, because Python is more versatile, and because it's free and open and available to everybody. The tools for science shouldn't be locked away behind some license agreement.

You could plot similar graphs for other disciplines, like biology or geology or neuroscience and would see similar trends. Engineering is still locked in with matlab a bit, but they are also moving towards python.

The other big player in this field is the R programming language, which is a domain specific language for statistical analysis. Some fields that have more of a history in statistics prefer using R over python, though Python is more broadly applicable.

This is not an either-or. Both are important tools for scientists.

R is also open source and has many great tools for statistical analysis and visualization available.

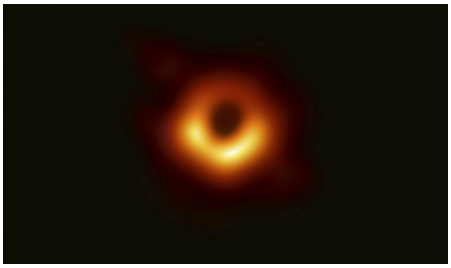
Impact

"The scientific Python ecosystem is critical infrastructure for the research done at LIGO."

David Shoemaker, LIGO Scientific Collaboration

These [SciPy/PyData] are fundamental to training biologists to have the computational skills they need to be effective in today's research realm.

Bonnie Hurwitz, Hurwitz lab, iPlant

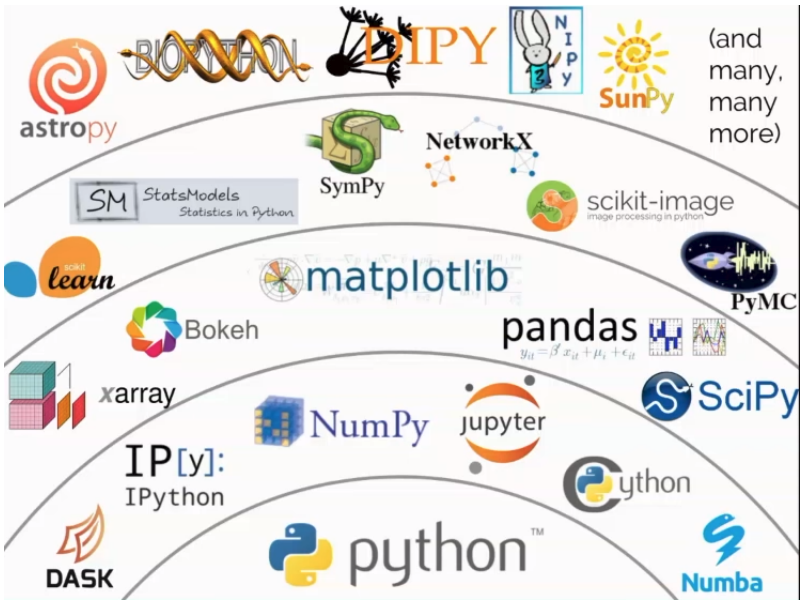


Here are some more data points on the use of the scientific python ecosystem. IT's critical for LIGO, and the discovery of gravitational waves was made using these tools.
It's indespensable in biology and training biologists.
You might have seen this picture of a black hole recently, it, of course, has also been made using the scientific python ecosystem.

The Scientific Computing Center (NERSC) found that about half of the users of their supercomputer use
Python, and that NumPy, SciPy, Matplotlib, Pandas and scikit-learn are the five most frequently
used Python libraries among NERSC users.

Basically, you can say, a large fraction of computational science that's done in any discipline, anywhere in the world, builds on the scientific python ecosystem.

The Scientific Python Ecosystem



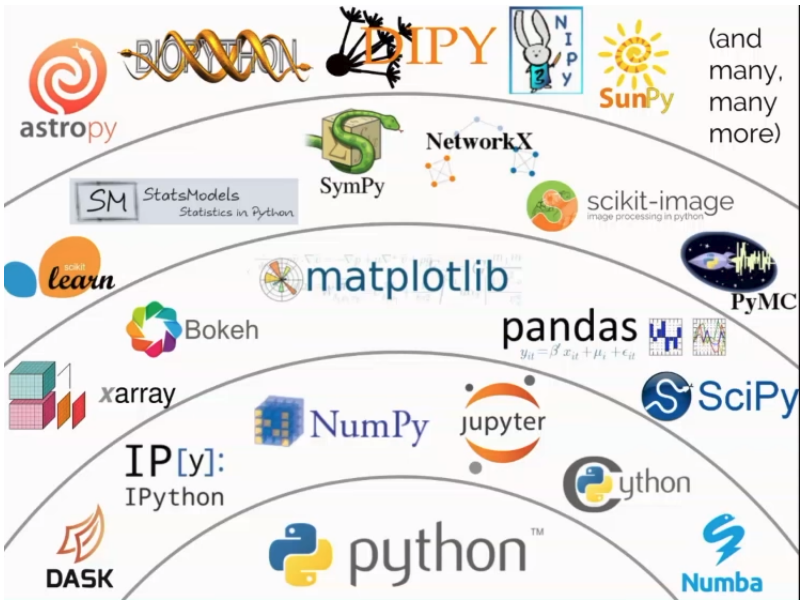
Let me explain a bit more what I mean by the scientifc python ecosystem. Sometimes I and others just say “Python”. Python is the programming language that is used to build all these tools. But the real strength of the ecosystem are all the tools that build on top of it.

So at the bottom we have the language Python. Above that we have tools for efficient numeric computation, in particular numpy and cython, and tools for interacting with the computation in an intuitive way, jupyter and ipython. These build on the language, of course.

The layer above builds on these in turn, and consists of librarys for working with n-dimensional data, doing data visualization, doing optimization and doing ETL.

Everybody that uses Python in science uses these tools.

The Scientific Python Ecosystem



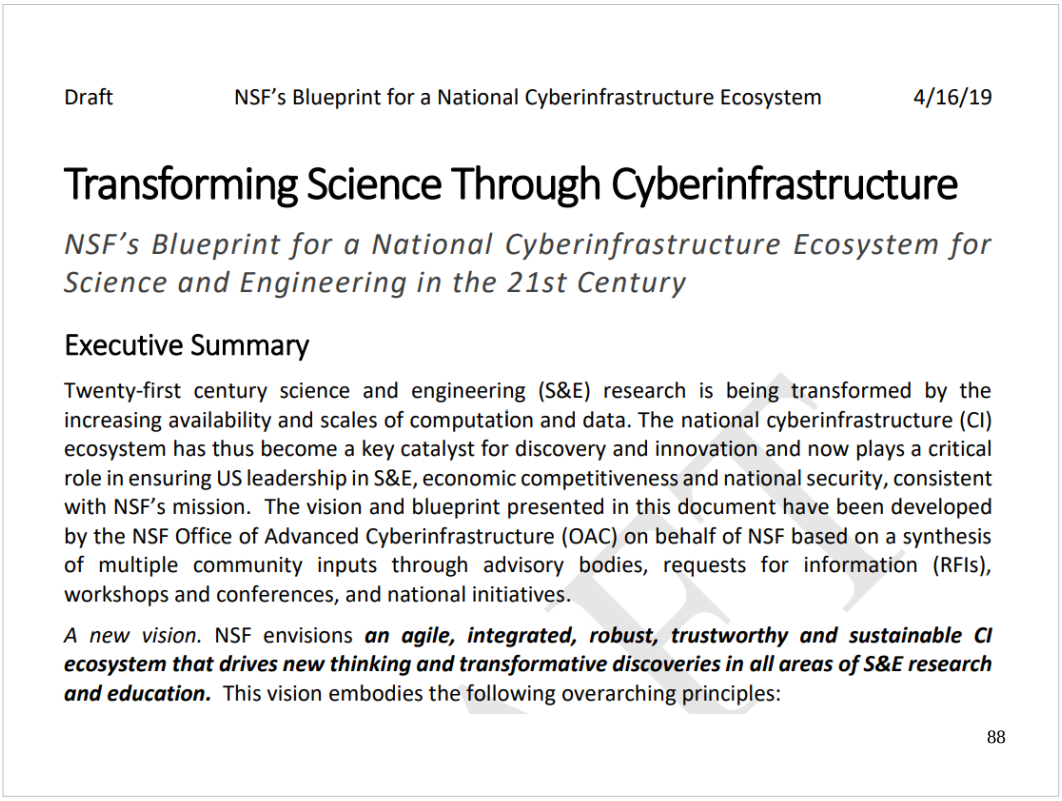
77

Building on these are more specialized libraries, like scikit-learn for machine learning, which is what I'm primarily working on. There's also statsmodels, and networkx for graph computation, probabilistic modelling, image processing and others, which are specialized but applicable across domains.

Finally on top of the stack, we have domain specific packages like astropy, biopython, dipy and nipy for neuro imaging, sunpy for solar physics and many more.

So when someone says Python in the context of science or data science, what is meant is the collection of all of these projects working together.

So far, that all sounds great, right? And actually both the python community and adoption in the scientific community have made strides and I'm very happy about this. But not all is good.



Draft NSF's Blueprint for a National Cyberinfrastructure Ecosystem 4/16/19

Transforming Science Through Cyberinfrastructure

NSF's Blueprint for a National Cyberinfrastructure Ecosystem for Science and Engineering in the 21st Century

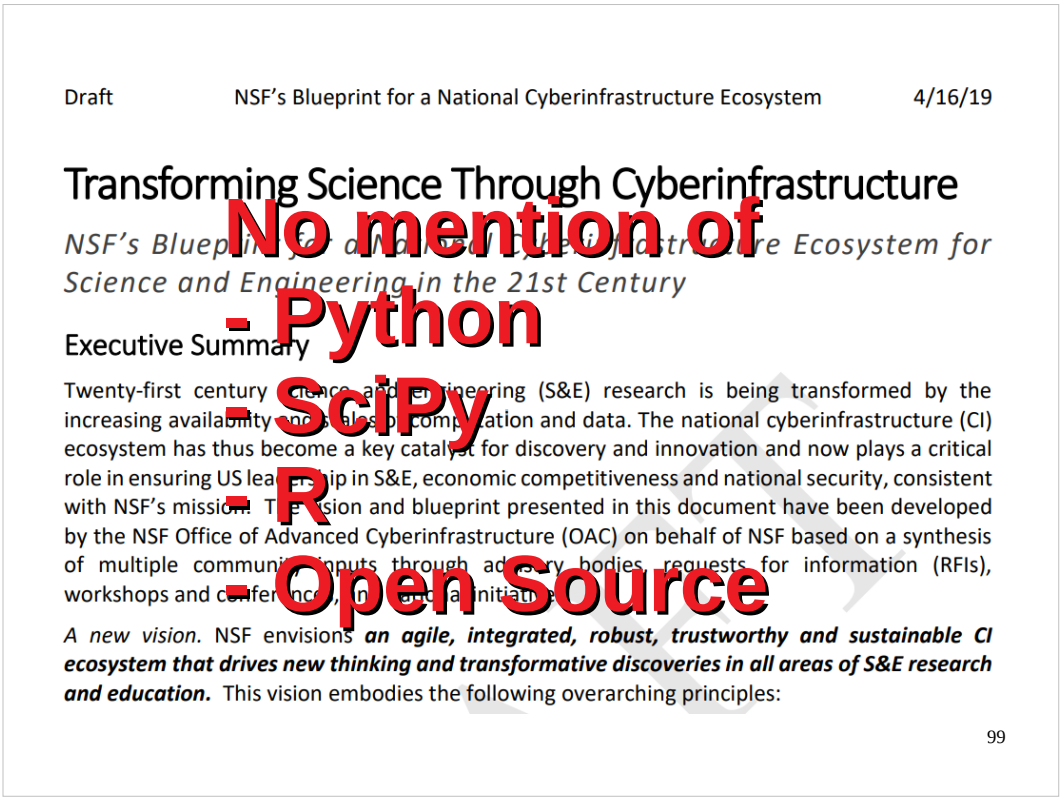
Executive Summary

Twenty-first century science and engineering (S&E) research is being transformed by the increasing availability and scales of computation and data. The national cyberinfrastructure (CI) ecosystem has thus become a key catalyst for discovery and innovation and now plays a critical role in ensuring US leadership in S&E, economic competitiveness and national security, consistent with NSF's mission. The vision and blueprint presented in this document have been developed by the NSF Office of Advanced Cyberinfrastructure (OAC) on behalf of NSF based on a synthesis of multiple community inputs through advisory bodies, requests for information (RFIs), workshops and conferences, and national initiatives.

A new vision. NSF envisions ***an agile, integrated, robust, trustworthy and sustainable CI ecosystem that drives new thinking and transformative discoveries in all areas of S&E research and education.*** This vision embodies the following overarching principles:

88

Let's look at this roadmap document from the NSF. This is the current draft from April about how to transform science through cyberinfrastructure – which mostly means software.



Draft NSF's Blueprint for a National Cyberinfrastructure Ecosystem 4/16/19

Transforming Science Through Cyberinfrastructure

NSF's Blueprint for a National Cyberinfrastructure Ecosystem for Science and Engineering in the 21st Century

Executive Summary

Twenty-first century science and engineering (S&E) research is being transformed by the increasing availability and sophistication of computation and data. The national cyberinfrastructure (CI) ecosystem has thus become a key catalyst for discovery and innovation and now plays a critical role in ensuring US leadership in S&E, economic competitiveness and national security, consistent with NSF's mission. The vision and blueprint presented in this document have been developed by the NSF Office of Advanced Cyberinfrastructure (OAC) on behalf of NSF based on a synthesis of multiple community inputs through advisory bodies, requests for information (RFIs), workshops and conferences, and related initiatives.

A new vision. NSF envisions ***an agile, integrated, robust, trustworthy and sustainable CI ecosystem that drives new thinking and transformative discoveries in all areas of S&E research and education.*** This vision embodies the following overarching principles:

99

No mention of
- Python
- SciPy
- R
- Open Source

Shockingly, this document doesn't mention Python or the scientific python ecosystem. It also doesn't mention R. In fact, it doesn't mention even the concept of open source.

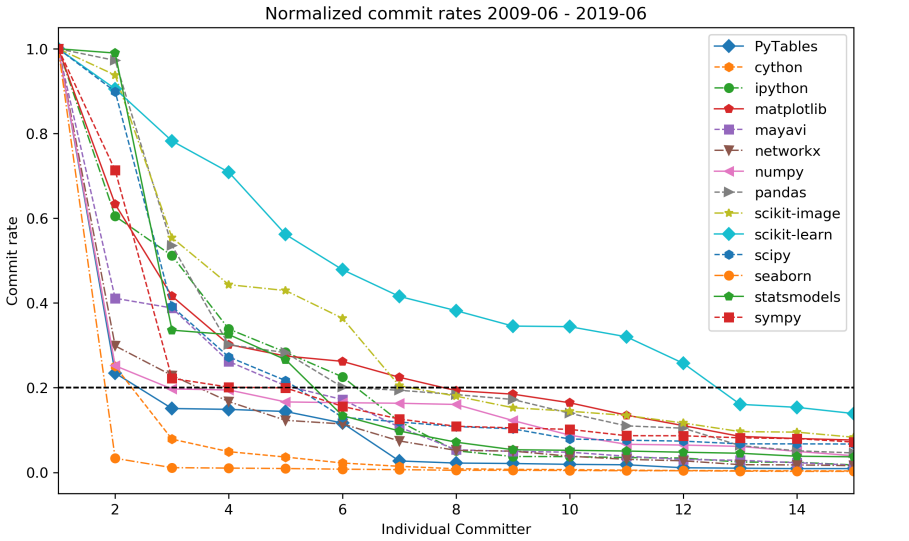
I just laid out to you that the scientific python stack is the computational basis of most of the science that's funded by the NSF. Yet the vision completely ignores it.

I think one of the reasons for this is that all these projects are grass-root projects. They were created by grad students to solve problems. They weren't some grant proposals. Good software never gets written as a consequence of some proposal to a funding agency. Anything that's thought up by a PI will only gather dust. Good software is written because there's a need for it!

I think developing software bottom-up is the right way to go. But unfortunately it means that there's no institutional support, either from industry or academia.

I focused on research here, but all of these tools are also widely used across all industries, anywhere there's computation.

Support and Bus Factor



1010

That lack of support has some dire consequences.
Here is a plot of contributions to the major packages of the scipy ecosystem, normalized across contributors. On the right is the most prolific contributor, then the second most prolific and so on. This is aggregated over the last 10 years.

What you can see here is that for many projects, the majority of the work might have been done by only one or two developers. These are mostly people working in their free time out of idealism. If any of these top contributors would leave the project, this would significantly harm the ecosystem. Yet this work is not usually even part of their job, they fit it in on the side.

There is the somewhat macabre notion of bus factor, which says “how many people have to be run over by a bus for this project to die”. In reality, recently it’s more the “google bus” that hires devs away from academia to industry jobs where they have less free time to work on open source.

This issue that’s pervasive in the ecosystem leads to what I call the scholz factor.

Scholz Factor

1111

There is this conference every year in Austin where many of the developers from the scipy ecosystem gather. And there's a social event at this nice place here, Scholz Beer Garden.

We are right now in a situation, where if someone dropped a bomb on this beer garden on the right day in July, basically the whole computational infrastructure for science, globally, would collapse.

I know it sounds like I'm exaggerating. The sad part is, I am not. There are maybe 50 people in the world that in their free time develop and maintain the whole infrastructure that computational science relies on, and most of them don't get anything for it.

I will come back to this later, but now let me switch gears and talk about scikit-learn, the project I'm working on.

Scikit-learn, as I said, is part of the scientific python ecosystem. And it's the main tool to do machine learning within this ecosystem.

It's a tool mainly for supervised learning, but also unsupervised learning, preprocessing, model selection, model evaluation and other tools around machine learning.

Our Mission:

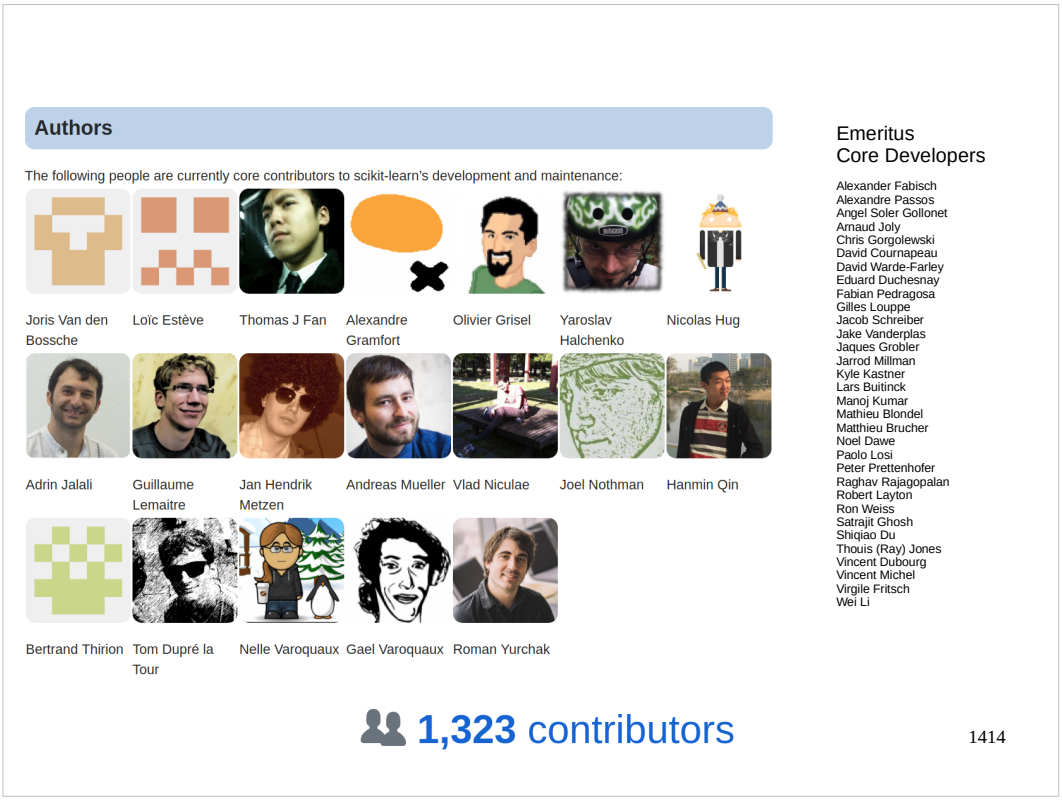
Commoditize and Democratize Machine Learning

1313

The mission of scikit-learn is to commoditize and democratize machine learning.
We want to enable everybody to use machine learning and data science. We want to make it easy, and we want to make it easy to do it right.

We are not at the cutting edge of research, we are building tools that actually work, and work robustly, and are well documented and easy to use.

And I would say we've been quite successful.

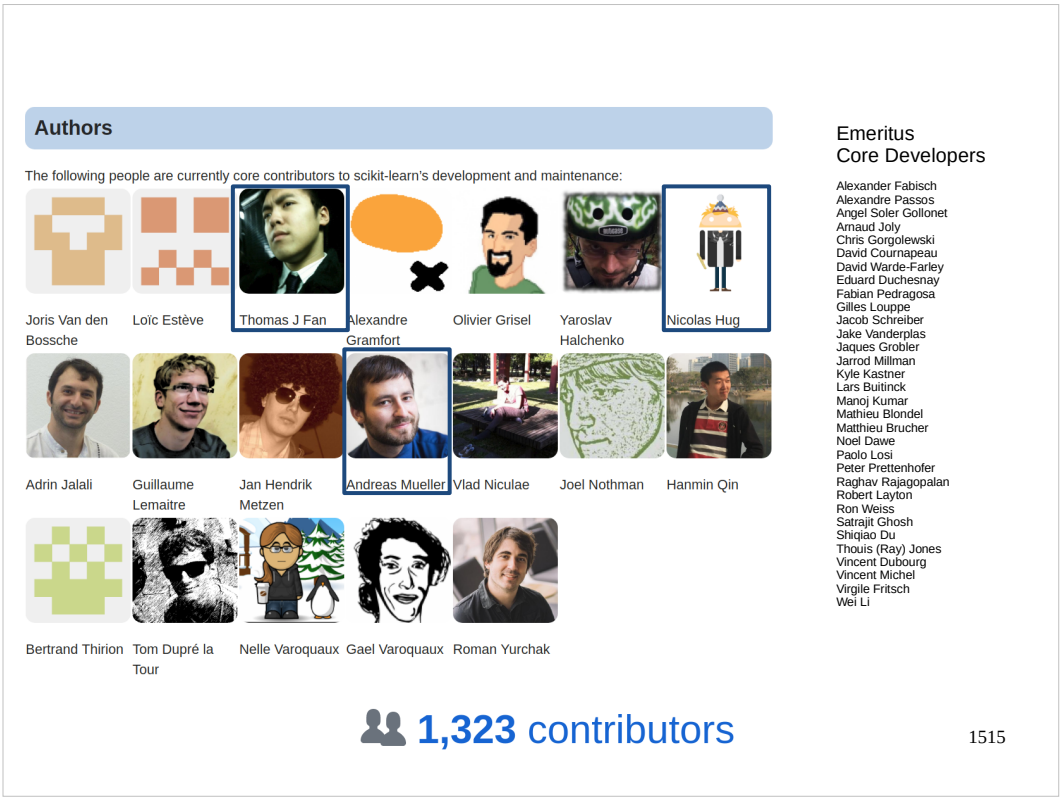


Here is a list of the current active core developers, 19 in total. However, there have been many previous core developers involved in the project, and in total the project has had contributions from over 1,300 developers, including researchers at at least 34 US universities.

The project was started around 2010 by a team working on Neuroscience at INRIA in Paris, and that team is still very involved. For example Alex Gramfort, Olivier Grisel, Bertrand Thiron and Gael Varoquaux are among the original authors.

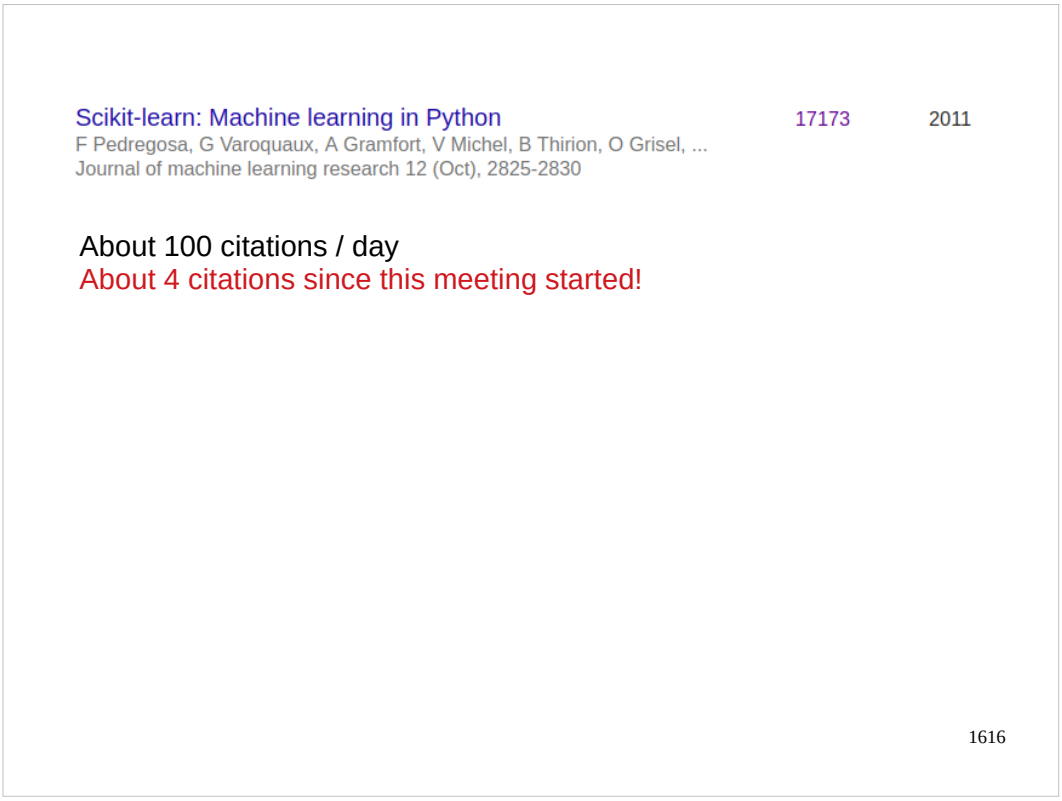
I have been contributing to the project for about 8 years now and I was the release manager for most of that time.

Both I and INRIA have people working on the project on the clock, but a lot of the contributions are done by people in their free time. In particular by Joel Nothman, who has been the most prolific contributor in the last several years.



Here I highlighted my current team, Thomas Fan, Nicolas Hug and myself.

We have nearly 3 FTEs on the project, about the same amount as Paris. This makes us some of the biggest contributors to the project and my team has been making many important improvements.

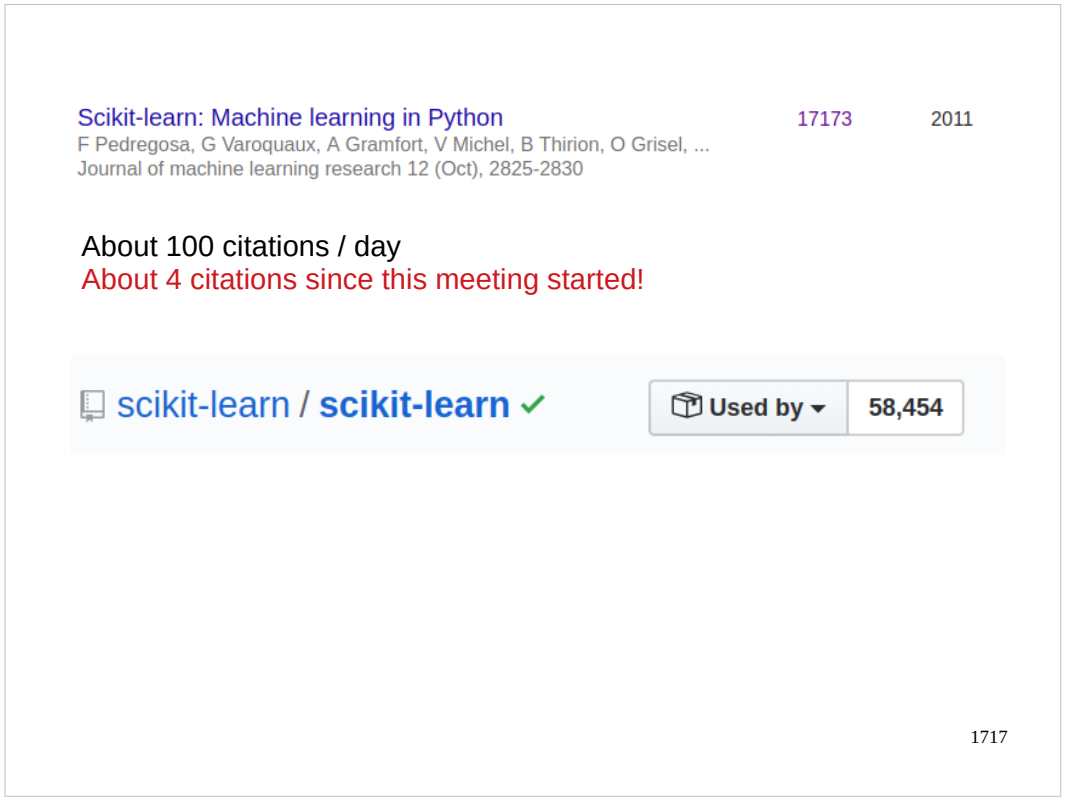


Here are a couple of measures of our success. The paper describing scikit-learn has ober 17,000 citations, and it's growing at an amazing rate of about 100 citations a day.

That means since this meeting started, about 4 scientific papers have been published using our software.

Actually, these numbers are likely to underestimate the usage, as software is rarely cited. Personally I think multiplying these numbers by 10 seems more realistic.

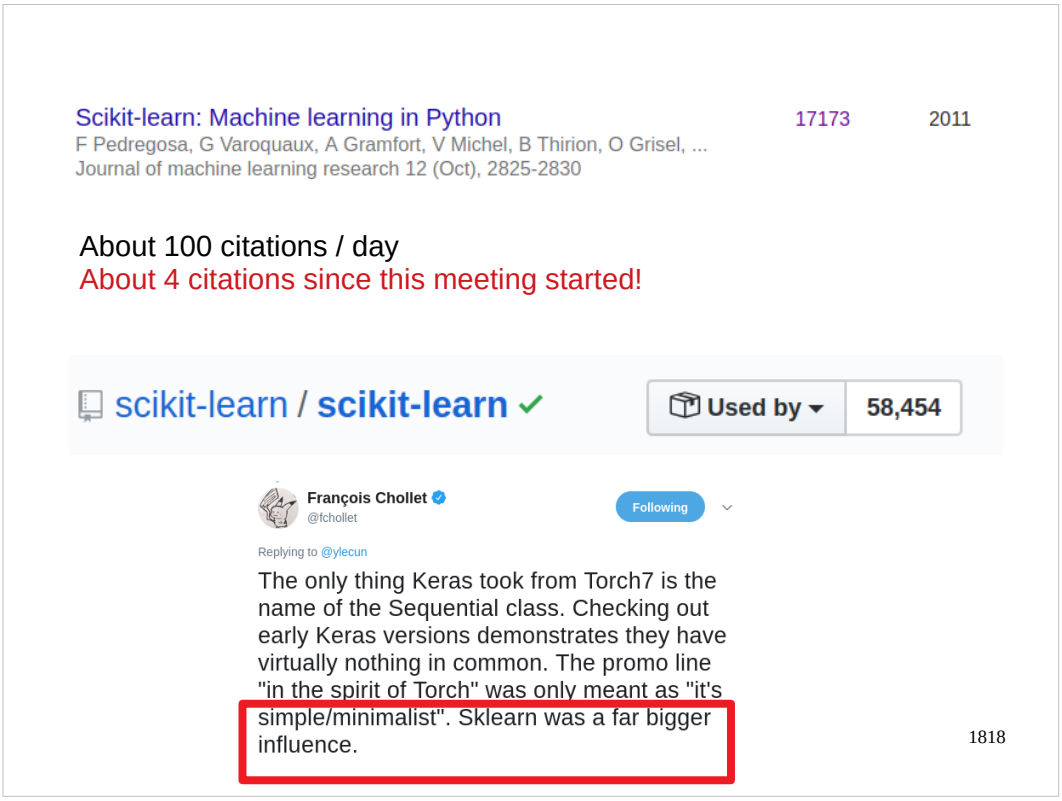
So I know 4 people have published research using these tools since the meeting started, but I estimate it was actually 40.



It's really hard to measure downloads for a widely used project like this with many distribution channels, but github tells us that there are nearly 60,000 projects on github that specify scikit-learn as a dependency.

I would say that a large fraction of machine learning projects that are not doing deep learning are driven by scikit-learn. That's true both for academia and industry.

There was a time when the spotify artist radios were done by scikit-learn. Mayor credit card companies use it for fraud detection. Microsoft told me about half of their data scientists use scikit-learn, which is the same amount as use their internal tool.



While scikit-learn doesn't do deep learning, and is pretty conservative in it's scope, it has inspired other packages that mimic it.

The user interface we created was the blueprint for deep learning packages like keras, which is now developed by google.

Here's a tweet by the original keras author, citing us as his influence.

Activities

- Extreme Gradient Boosting (Nicolas Hug)
- Preprocessing, usability
- Maintenance, bug fixes, community management and support

Let me get a bit more concrete about what my team does. A lot of the work we do is pretty technical, and I don’t want to go into details too much.

One of the recent additions by my team is an implementation of extreme gradient boosting by Nicolas Hug. This is much, much faster than what was in scikit-learn before and it will have a huge impact on the ML community.

A lot of our work is among preprocessing, in particular in improving usability with dirty data.

And last but not least, we spend a lot of time on maintenance activities, small improvements, bug fixes and community management and support.

Future Work

- Model understanding and inspection
- Visualization
- Automatic Machine Learning (AutoML) & dabl

2020

I also wanted to list some of our future directions, which we have already started.

We're improving model inspection and understanding in scikit-learn, which will allow users to better interpret what their models are doing and why they might not be working, or whether they are trustworthy.

In the same vein we want to assist users more in creating informative visualizations. We want to make it easier to test out ideas and quickly visualize the results. There's never a good reason not to look at your data.

Finally, we started developing a new library, which we call dabl, which automatically shows interesting and useful visualizations, and does automatic tuning of models.

The goal of dabl is to automate as much as possible, while also keeping the user in close contact with the data and the algorithms, and making the process as transparent as possible.

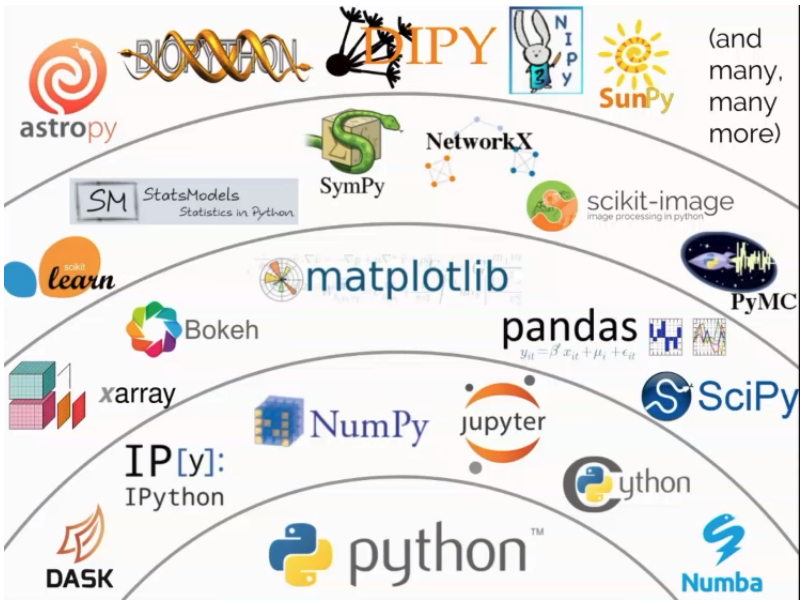
There are several possible routes to improving the funding situation for the scientific python stack.

The standard way to get money for software is to bundle it in with some research and we could try doing that.

For example, take the person that maintains the library that created all the images for the black hole and that's used in over 12% of all scientific papers on the arxiv website. He is allowed to spend a small percentage of his time working on this. But not because basically every scientist in every discipline around the world use it, no, because his lab also uses it.

However, I think for this to be sustainable we need to fund the software directly, and long-term. It's really hard to keep people on these projects and to hire good people on these projects because grants are short. For example I have another year of funding, my team has less

The Scientific Python Ecosystem



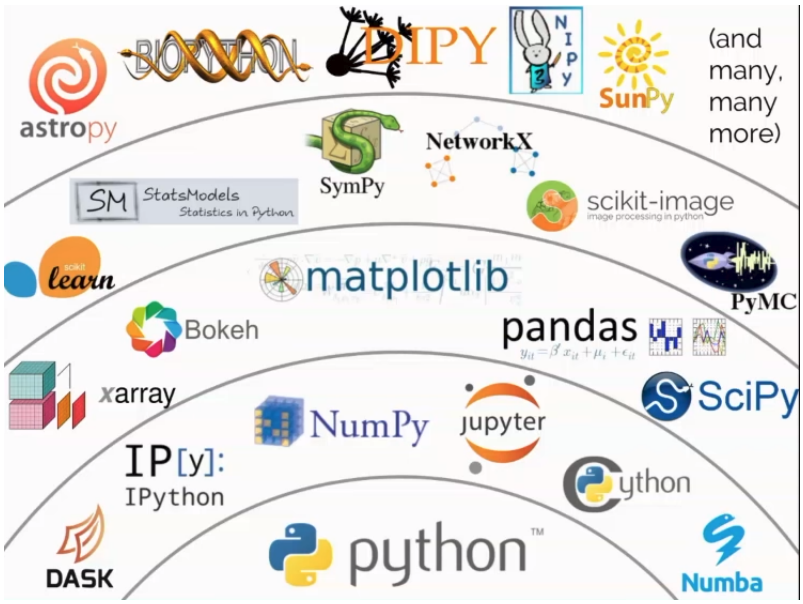
2222

Going back to the stack, ironically the higher up you are, the easier it is to find funding, because you are closer to science – but the further down, the more essential you are.

I am the only one at this level or below that has received public funding, as far as I know.
Numpy has received short-term funding from the Sloan foundation, Jupyter has received some private funding.

The NSF has made some changes, but there is basically still no way to directly and sustainably fund existing infrastructure.

The Scientific Python Ecosystem



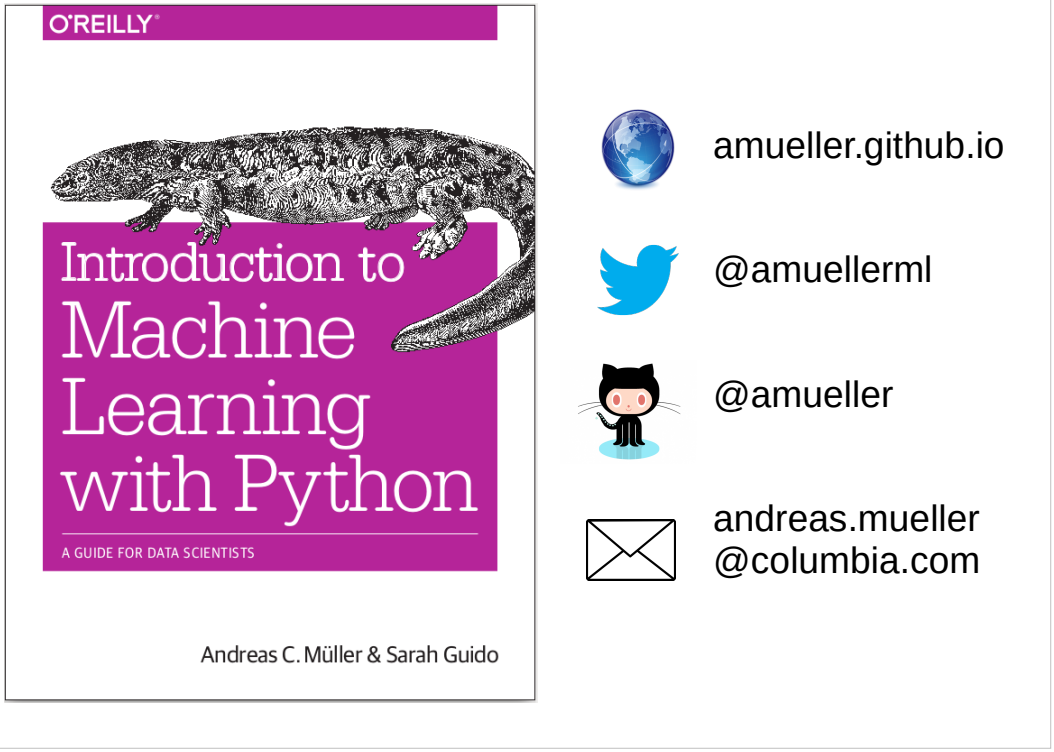
2323

Luckily Jeannette has been very supportive, which has helped me to obtain funding and build a team.

Right now with my team, the DSI is at the center of open source development for data science. But to stay there, we need to have sustainable funding and career paths. With DSI's help I think I was able to create relatively attractive positions for my team, but they are definitely not the norm in academia.

Ideally I would like to see more people working on building tools. Investing in this infrastructure is hugely productive and the contributions my team could make have already been tremendously impactful already.

Nicolas has started last summer and Thomas has started this spring. I'm quite excited for what we'll achieve in the next year, and hope we can continue beyond that.



Thank you for your attention and I'm happy to take any questions!