2 The tool: a Python library
Scientific computing needs a high-level interactive environment.
Scientific computing needs a high-level interactive environment.

Memory management
⇒ a virtual machine

Python’s virtual machine is rudimentary
Enables low-level computation and coupling to numerical libraries
numpy: array computing in Python

numpy =
- memory & data specification
- reshaping with minimal copies
- semantics of operations

Represented any regular data in a structured way

G Varoquaux
numpy: array computing in Python

- memory & data specification
- reshaping with minimal copies
- semantics of operations

Matches the memory model of numerical libraries

⇒ Enables copyless interactions

Numpy is really a memory model
The scientific-computing ecosystem

- **scipy**: numerical algorithms
  - linear algebra
  - special function
  - statistical functions
  - optimization
  - interpolation
  - FFT

- **matplotlib**: plotting
  *matlab-like plotting and beyond*

- **pandas**: columnar data
  *programatic excel*

- **scikit-image**: image processing

- **statsmodels**: statistical models

...
The scientific-computing ecosystem

- **scipy**: numerical algorithms
  - linear algebra
  - special function
  - statistical functions
  - optimization
  - interpolation
  - FFT

- **matplotlib**: plotting
  [http://scipy-lectures.org](http://scipy-lectures.org)

- **pandas**: columnar data
  
  *programatic excel*

- **scikit-image**: image processing

- **statsmodels**: statistical models

...
A library, not a program

- More expressive and flexible
- Easy to include in an ecosystem
API:
The greybox model

Building bricks
to combine with domain-specific knowledge interchangeable (mostly)
The greybox model

```python
from sklearn import svm
classifier = svm.SVC()
classifier.fit(X_train, Y_train)
Y_test = classifier.predict(X_test)
```

Access to the model’s inner parameters

```python
coef = classifier.coef_
```
2 Vectorizing

From raw data to a sample matrix $X$

- For text data: counting word occurrences
  - Input data: list of documents (string)
  - Output data: numerical matrix
Vectorizing

From raw data to a sample matrix \( X \)

- For text data: counting word occurrences
  - Input data: list of documents (string)
  - Output data: numerical matrix

```python
from sklearn.feature_extraction.text import HashingVectorizer
hasher = HashingVectorizer()
X = hasher.fit_transform(documents)
```
Very rich feature set: 160 estimators

Supervised learning
- Decision trees (Random-Forest, Boosted Tree)
- Linear models
- Gaussian processes
- SVM
- ...

Unsupervised Learning
- Clustering
- Dictionary learning
- Outlier detection
- Mixture models
- ICA
- ...

Model selection
- Cross-validation
- Parameter optimization
I prefer C++ to C

C without malloc, free, and pointer arithmetics

Cython

- typed Python syntax
- generates C code running in the Python virtual machine
- native support for numpy arrays

Staying high level is crucial in the long term
SAG:

```python
linear_model.LogisticRegression(solver='sag')
```

Fast linear model on biggish data
Some gems in scikit-learn

SAG:

```
linear_model.LogisticRegression(solver='sag')
```
Fast linear model on biggish data

PCA == RandomizedPCA:  
Heuristic to switch PCA to random linear algebra
Huge speed gains for biggish data
Fights global warming
Some gems in scikit-learn

Outlier detection and isolation forests (0.18)
Some gems in scikit-learn

Outlier detection and isolation forests (0.18)

1. Isolation Forest (errors: 6)
2. One-Class SVM (errors: 14)
3. Robust covariance (errors: 14)

Distributed computing (soon)
dask & hadoop backends to joblib
The project: a community
Community-based development in scikit-learn

Huge feature set:
benefits of a large team

Project growth:

- More than 200 contributors
- ~12 core contributors
- 1 full-time INRIA programmer
  from the start

Estimated cost of development: $6 millions
COCOMO model,
http://www.ohloh.net/p/scikit-learn
Many eyes makes code fast


G Varoquaux
1. Focus on **quality**

2. Build great **docs and examples**

3. Use **github**

4. Limit the technicality of your codebase

5. Releasing and packaging matter

6. Focus on your contributors, give them credit, decision power

http://www.slideshare.net/GaelVaroquaux/scikit-learn-developpement-communautaire

G Varoquaux
Unit testing

- Everything is tested
  If it’s not tested, it’s broken

- Test API
  Test as grey box

- Test numerics
  Check mathematical properties
  (e.g. decrease of energy)

- Tests should run fast

- Perfect control of randomness
Quality assurance

**Code review**: pull requests

- Can include newcomers
- We read each other's code
- Everything is discussed:
  - Should the algorithm go in?
  - Are there good defaults?
  - Are names meaningful?
  - Are the numerics stable?
  - Could it be faster?
Success: scikit-learn user base

350,000 returning users

- Windows: 50%
- Mac: 30%
- Linux: 20%

5,000 citations

- Industry: 63%
- Academia: 34%
- Other: 3%
Open source is infrastructure
Everybody uses it everyday
Open source is infrastructure
Everybody uses it everyday

In scientific research
- R: paper published in 2000, 67,248 citations
- scikit-learn: paper published in 2011, 5,525 citations

In the industry
“Roads and Bridge”: Ford foundation report
Talk by Heather Miller www.youtube.com/watch?v=17yy5BwliTw
Open source is infrastructure
Everybody uses it everyday

It needs maintainance
Recipe for good software:

Make it work, make it right, make it boring
The tragedy of the commons

Individuals, acting independently and rationally according to each one’s self-interest, behave contrary to the whole group’s long-term best interests by depleting some common resource.

Make it work, make it right, make it boring

Core projects (boring) taken for granted
⇒ Hard to fund, less excitement

They need citation, in papers & on corporate web pages
The tragedy of the commons

Individuals, acting independently and rationally according to each one’s self-interest, behave contrary to the whole group’s long-term best interests by depleting some common resource.

It’s so hard to scale
User support
Growing codebase

Make it work, make it right, make it boring

Core projects (boring) taken for granted
  ⇒ Hard to fund, less excitement

They need citation, in papers & on corporate web pages
The tragedy of the commons

Individuals, acting independently and rationally according to each one’s self-interest, behave contrary to the whole group’s long-term best interests by depleting some common resource.

- It’s so hard to scale
  - User support
  - Growing codebase

Make it work, make it right, make it boring

Core projects (boring) taken for granted
  ⇒ Hard to fund, less excitement

They need citation, in papers & on corporate web pages

G Varoquaux
The tragedy of the commons

Individuals, acting independently and rationally according to each one's self-interest, behave contrary to the whole group's long-term best interests by depleting some common resource. Wikipedia

Make it work, make it right, make it boring

Core projects (boring) taken for granted

⇒ Hard to fund, less excitement

They need citation, in papers & on corporate web pages

+ It's so hard to scale

User support

Growing codebase

G Varoquaux
Scikit-learn

The vision

Make machine learning accessible
Versatile library: the "right" level of abstraction
Close to research, but seeking different tradeoffs
Scikit-learn

The vision
Make machine learning accessible

The tool
Scientific Python ecosystem
Simple API uniform across learners
The vision
Make machine learning accessible

The tool
Scientific Python ecosystem

The project
Many people working together
Tests and discussions for quality

Sustainability? A consortium?

@GaelVaroquaux