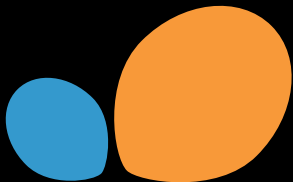


Scikit Learn

Développement communautaire

Gaël Varoquaux — *Inria*



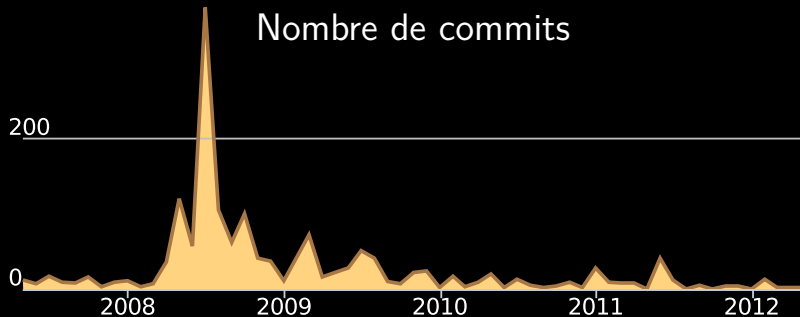
Vlad Niculae
Lars Buitinck
Peter Prettenhofer
Robert Layton
Jake VanderPlas
Gilles Louppe
Brian Holt
Olivier Grisel
Gael Varoquaux
Fabian Pedregosa
Alexandre Gramfort
Mathieu Blondel
Andreas Mueller

Package XXX

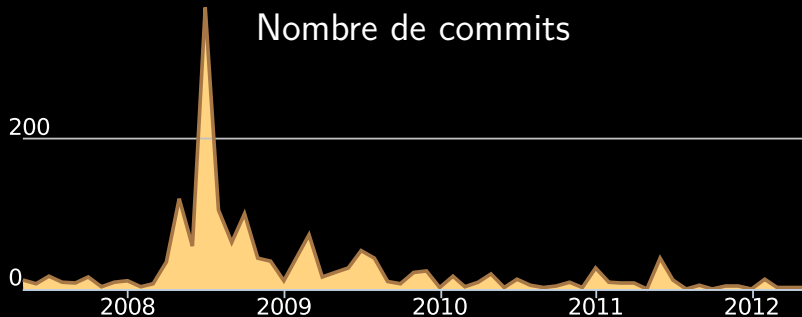
- 2007: inclut dans Debian
- 2008: .1% des utilisateurs l'installent
- 2012: .5% des utilisateurs (scilab = 1.2%)

Package XXX

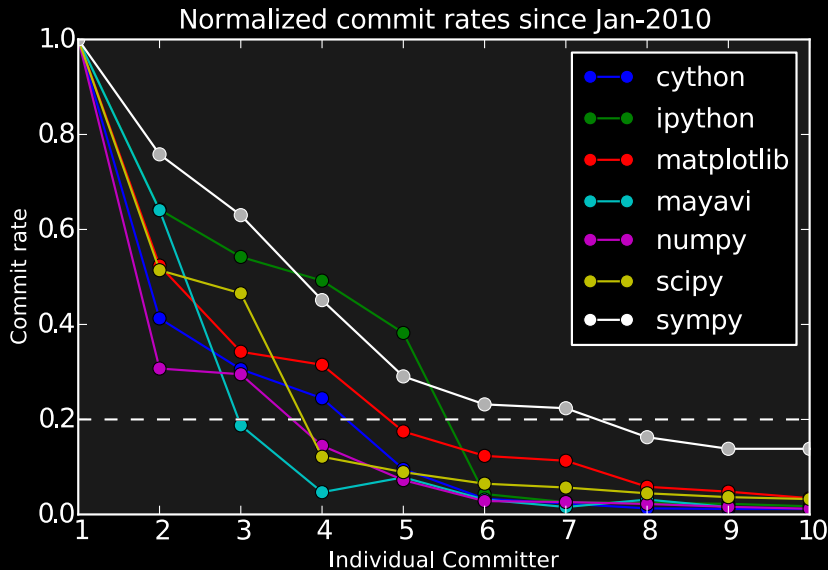
- 2007: inclut dans Debian
- 2008: .1% des utilisateurs l'installent
- 2012: .5% des utilisateurs (scilab = 1.2%)



“““Hey Gael, I take it you're too busy. That's okay, I spent a day trying to install XXX and I think I'll succeed myself. Next time though please don't ignore my emails, I really don't like it. You can say, 'sorry, I have no time to help you.' Just don't ignore.”””



La tragédie des logiciels libres



Source: Fernando Perez

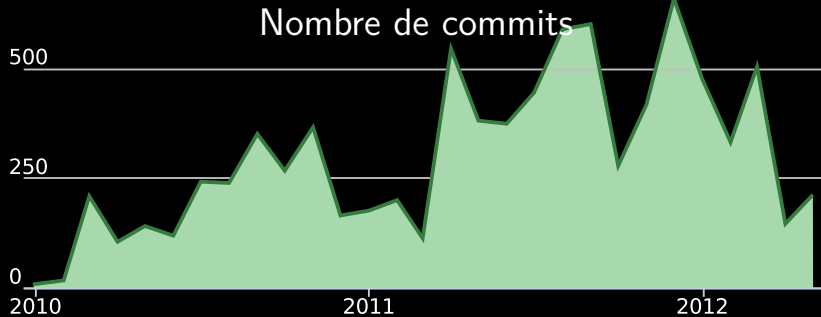
Mon gros succès logiciel libre – 2^{eme} essaie

Notre gros succès logiciel libre

Notre gros succès logiciel libre

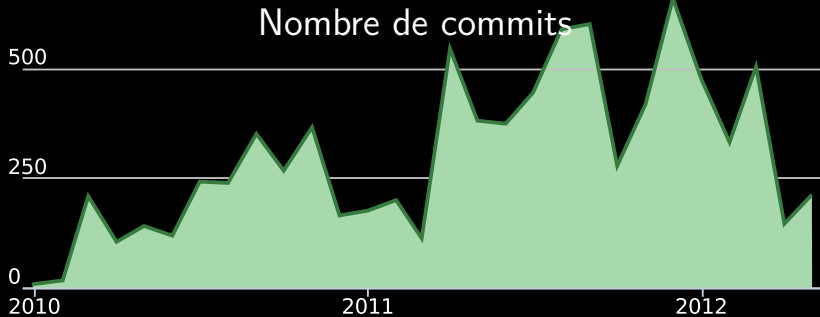
Scikit-learn

- 2006-2009: créé par D. Cournapeau, dépérit
- 2010: reprise, release 0.1
- 2012: release 0.11 avec 31 contributeurs



Notre gros succès logiciel libre

- 130 email/semaine sur la mailing list
- 28 000 lignes de code Python ($\frac{1}{4}$ Django)
- 246 forks et 574 followers sur github
- 15 000 lignes de documentation
- 6 000 visiteurs uniques/mois sur la page web



10 leçons durement appries



1 Connaître son marché

- A marché niche, petit projet
- Ne pas viser de public qui ne peut se transformer en contributeur
- Connaître ses concurrents

Savoir ne pas créer de projet

Pourra-t-on avoir un impact?

2 Utiliser Github

- Github = outil à la mode
- Mener toutes les discussions en ligne
- Le système de *fork/pull request* met tous le monde à égalité
- Critique de code sur les 'pull requests'

The screenshot shows a GitHub pull request discussion. At the top, it says "agramfort started a discussion in the diff 7 months ago". Below this is a code diff for the file `sklearn/cluster/_inertia.pyx`. The diff shows a change in the `compute_ward_dist` function. Line 24 is highlighted in red, showing the original code: `n = (m_1[row] * m_1[col]) / (m_1[row] + m_1[col])`. Line 39 is highlighted in green, showing the proposed change: `n = 1.0 / (1.0 / m_1[row] + 1.0 / m_1[col])`. Below the code, there are two comments. The first comment is from "agramfort" (repo collab) and says "i am afraid this is numerically less stable. it is justified by speed". The second comment is from "jmetzen" and says "you are right, I reverted it to the old implementation". At the bottom, there is a comment from "jmetzen" (added some commits) with the commit hash `8e00db8` and the message "ENH: CompleteLinkage supports more effic".

3 Ne pas s'appropriier l'identité du projet

- Ne pas mettre son nom trop en avant
- Ne pas mettre son employeur trop en avant (attention à l'utilisation des forges)
- Pousser d'autres personnes à communiquer au nom du groupe (conférences)

4 Cibler la qualité

Il faut apporter une valeur ajoutée par rapport au code perso

- Documentation, exemples
- Efficacité / généralité du code
- Test: base de code modifiable en toute confiance
- Conventions de style de code

Théorie de la fenêtre brisée: le bon code amène du bon code

5 Release régulières

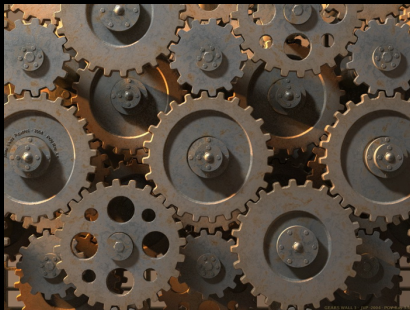
- Les contributeurs donnent leur code pour qu'il arrive chez les utilisateurs
- Effet d'adrénaline de la release (rendre sa copie)
- Créer du buzz

6 Contrôler la technicité

- Choisir les compétences: la population des experts en C++ et en optimisation convexe est petite

Important pour cibler les contours du projet

- Démarche d'ingénieur: équilibrer complexité avec gain en fonctionnalité



7 Entretien une bonne culture de projet

- Attitude positive
- Pragmatisme et impartialité dans la prise de décision
Prendre une solution autre que la sienne peut être bénéfique socialement
- Valoriser les autres
 - Attribuer toujours tous les fichiers
 - Confier des responsabilités (release manager)
 - Donner les droits de commit

8 Organiser des sprints



9 Investir dans le recrutement

- Cibler sa population
- Aider les contributeurs motivés
- Stages / Google summer of code, mais **encadré**

■ Blog, twitter

■ Conférence et article
en étant **inclusif**

Scikit-learn: Machine Learning in Python

Fabian Pedregosa
Gaël Varoquaux
Alexandre Gramfort

Vincent Michel
Bertrand Thirion
*Paris Lodron INRIA Saclay
Neurospin, Bât 145, CEA Saclay
91191 Gif sur Yvette – France*

Olivier Grisel
*Nusoo
20 rue Soleillet
75 020 Paris – France*

Mathieu Blondel
*Kobe University
1-1 Rokkodai, Nada
Kobe 657-8501 – Japan*

Peter Prettenhofer
*Bauhaus-Universität Weimar
Bauhausstr. 11
99421 Weimar – Germany*

Ron Weiss
*Google Inc
76 Ninth Avenue
New York, NY 10011 – USA*

Vincent Dubourg
*Clermont Université, IFMA, EA 3867, LaMI
BP 10448, 63000 Clermont-Ferrand – France*

Jake Vanderplas
*Astronomy Department
University of Washington, Box 351580
Seattle, WA 98195 – USA*

Alexandre Passos
*IESL Lab
UMass Amherst
Amherst MA 01002 – USA*

David Cournapeau
*Enthought
21 J.J. Thompson Avenue
Cambridge, CB3 0FA – UK*

FABIAN.PEDREGOSA@INRIA.FR
GAEL.VAROQUAUX@NORMALSUP.EDU
ALEXANDRE.GRAMFORT@INRIA.FR
VINCENT.MICHEL@LOGILAB.FR
BERTRAND.THIRION@INRIA.FR

OLIVIER.GRISEL@ENSTA.FR

MBLONDEL@ALCS.KOBE-U.AC.JP

PETER.PRETTENHOFER@GMAIL.COM

RONWEISS@GMAIL.COM

VINCENT.DUBOURG@GMAIL.COM

VANDERPLAS@ASTRO.WASHINGTON.EDU

ALEXANDRE.TP@GMAIL.COM

COURNAPEAU@GMAIL.COM

- 1 Connaître son marché
- 2 Utiliser Github
- 3 Ne pas s'approprier l'identité du projet
- 4 Cibler la qualité
- 5 Release régulières
- 6 Contrôler la technicité
- 7 Entretenir une bonne culture de projet
- 8 Organiser des sprints
- 9 Investir dans le recrutement
- 10 Communiquer

 @GaelVaroquaux

