This is the page for coordination of the GSoC for scikit-learn.

Scikit-learn is a machine learning module in Python. See http://scikit-learn.org for more details.

Scikit-learn is taking part of the GSoC trough the Python Software Foundation: http://python-gsoc.org/

Instructions to student: achieving a good proposal

Difficulty: Scikit-learn is a technical project. Contributing via a GSoC requires a number of expertise in Python coding as well as numerical and machine learning algorithms.

Important: Read: http://write.flossmanuals.net/gsocstudentguide/what-is-google-summer-of-code/ https://wiki.python.org/moin/SummerOfCode/Expectations https://wiki.python.org/moin/SummerOfCode/FrequentlyAskedQuestions

Application template: https://wiki.python.org/moin/SummerOfCode/ApplicationTemplate2015 Please follow this template.

Also important: A letter from Gaël to former applicants. His suggestions are just as relevant this year.

Hi folks,

The deadline for applications is nearing. I'd like to stress that the scikit-learn will only be accepting high-quality application: it is a challenging, though rewarding, project to work with. To maximize the quality of your application, here are a few advice:

1. First discuss on the mailing list a pre-proposal. Make sure that both the scikit-learn team and yourself are enthusiastic about the idea. Try to have one or two possible mentors that hold a dialog with you.
2. Satisfy the PSF requirements (http://wiki.python.org/moin/SummerOfCode/Expectations) briefly:
    - Demonstrate to your prospective mentor(s) that you are able to complete the project you've proposed
    - Blog for your GSoC project.
    - Contribute at least one patch to the project

I'd add the patch should be somewhat substantial, not just fixing typos.

To contribute patch, please have a look at the [contribution guide] (http://scikit-learn.org/dev/developers/index.html#contributing-code) and the Easy issues in the tracker.

3. In parallel with 2, start a online document (google doc, for instance) to elaborate your final proposal, and if you manage to convince mentors, you

can get feedback on it.

As a final note, I want to stress that GSOC projects are ambitious: we are talking about a few months of full time work. Thus the ideas proposed are idea challenging, and the students are supposed to draw a battle plan, with difficult variants and less difficult variants. The GSOC is a full major set of contributions, not a single pull request.

Good luck, I am looking forward to seeing the proposals. You'll see, the scikit is a big friendly and enthusiastic community,

Gaël

A list of topics for a Google summer of code (GSOC) 2016

Disclaimer: We are planning to take up to 2 students this time for GSoC 2016. The 1st idea is adding fused type support to Cython and the second one is a free idea. Please e-mail the list with any questions.

Adding fused types to Cython files

Possible mentors: Manoj Kumar and Joel Nothman

Possible candidates: Yen-Chen Lin (yenchenlin), Devashish Deshpande (dsquareindia).

Application Link: Yen's proposal, Devashish's proposal

We have removed Cython files from the repo since https://github.com/scikit-learn/scikit-learn/pull/5492 and re-generate it for every build. This provides a good opportunity to NOT blow up the memory usage by refactoring the functions in the ".pyx" files to use Fused Cython types. This will allow `float32` and `int32` dtypes where data is being explicitly cast into `float64` and `int64`.

This is obviously a project that affects the codebase extensively and hence the student must provide a detailed proposal as to which specific parts to touch (SGD, Coordinate descent etc) and the proposed line of attack for the summer. The project requires much less knowledge of machine learning than previous scikit-learn projects; it is an engineering project. The student should expect to make judicious decisions about where fused type support is most urgently needed and where it provides little benefit; the student should anticipate that writing elegant tests for the enhanced functionality is a major part of their contribution.

Related issues:

https://github.com/scikit-learn/scikit-learn/issues/5776

https://github.com/scikit-learn/scikit-learn/issues/5464

A good starting point would be to review the existing Pull Request for KMeans (https://github.com/scikit-learn/scikit-learn/pull/6430) and implement fused-types for sparse functions (https://github.com/scikit-learn/scikit-learn/pull/5932)

Propose your own idea

Possible mentors: Raghav RV and Jacob Schreiber

Possible candidate: Nelson Liu (nelson-liu)

Application Link: [Nelson Liu] GSoC 2016 Project Proposal: Addition of various enhancements to the tree module by completing stalled pull requests. Better missing value handling

You are expected to go through **existing** issues/PRs and **stalled works** in a particular theme of your choice and build a solid work plan and definitive goals that you think is possible for you to finish up within the given time slot.

NOTE:

Please **don't** propose a new feature as we already have a lot of work/PRs in the pipeline.

Please note that Multiple Metric Support and Sample weight support are not up for grabs as I (Raghav R V) intend to continue the work on the same.

An impressive proposal would be one which takes up for instance a moderately sized project (need not be a really difficult project, simple tractable ones are preferred) that has been stalled and to define clear goals/milestones and an **achievable** timeline for the milestones.

Some ideas for this include (Please **don't** restrict yourself to the very much incomplete list of projects, if these do not interest you) -

- Adding post pruning support to decision trees. ()
- Matrix factorization with missing values - (https://github.com/scikit-learn/scikit-learn/pull/4237)

NOTE Brownie points for making a good pass at stalled works and picking the one that you think is most likely to be included (based on previous discussions) and more importantly the one that you think is possible for you within the GSoC time frame.