Overall Goal: An empirical quantitative study of factors that affect the adoption of a package in open-source ecosystems such as NPM, PyPI.

Task: Read and familiarize with the article [1], which qualitatively studies the developer’s perception of software dependency management.

Expected Outcome:

* Development of a tool to extract metadata and metrics from Libraries.io and Github.
* writing a short paper (4 pages ACM Style) in English that describes the approach
* An empirical study using the developed tool to answer the following questions:

1. Would the community support facilitate the adoption of a library? (Observation 2 in [1]) To answer this question, we can find the correlation between community metrics like #stars, #contributors, fixing speed in Github, and the “adoption” metrics like the number of packages that depend on the target package and the number of repositories that depend on the target package (See the definitions at [3]).
2. Would the fixes for security bugs appear fast in a well-supported library? (Observation 2 in [1]) To answer this question, we need to define a “well-supported” library and calculate the fixing speed. Then we do the statistical analysis

[1]<https://securitylab.disi.unitn.it/lib/exe/fetch.php?media=research_activities:experiments:ccs-2020-preprint.pdf>

[2] <https://docs.libraries.io/overview.html>

Overall Goal: An evaluation of the APR techniques under time constraints

Example csv: <https://docs.google.com/spreadsheets/d/1Od3Csu3DJ8fzTz29kRgh2kGLIIWSKqPYxPC6PJoZm9M/edit?usp=sharing>

Hypothesis 1: more supported package -> a lot of users

Source: <https://pypi.org/simple/>

X (factors) ~ Y (the adoption of package)

Package in PyPI (e.g, pandas, ).

1. Package A -> Github repository. find\_repository(package\_name) -> github\_url
   1. Extract features. extract\_features\_from\_github(github\_url) -> Factors( #stars, #contributors, other features), Target variable (#forks). (Github)
   2. extract\_features\_from\_libraries\_io(libraries.io). (Libraries.io). Beautifulsoup, requests
2. Correlation techniques.

**METRICS DESCRIPTION**

GitHub metrics (taken from <https://docs.github.com/en/github/getting-started-with-github/github-glossary>)

* Stars: The number of GitHub users that displayed an appreciation for the repository. Stars are a manual way to rank the popularity of projects.
* Contributors: The number of GitHub users who don’t have collaborator access to a repository but have contributed to a project and had a pull request they opened merged into the repository.
* (Collaborators?: he number of GitHub users with read and write access to a repository who has been invited to contribute by the repository owner)
* Dependent repositories: A GitHub [repository](https://docs.github.com/en/github/getting-started-with-github/github-glossary#repository) which have the current repo as dependency.
* (Dependent packages: A GitHub [project](https://docs.github.com/en/github/managing-your-work-on-github/tracking-the-progress-of-your-work-with-project-boards) (see the [difference](https://stackoverflow.com/questions/40509838/project-vs-repository-in-github)) which have the current repo as dependency)
* (Forks: A fork is a personal copy of another user's repository that lives on your account. Forks allow you to freely make changes to a project without affecting the original upstream repository. You can also open a pull request in the upstream repository and keep your fork synced with the latest changes since both repositories are still connected.)
* Last commit: The date of the last commit
* Commit frequency: The average number of [commits](https://docs.github.com/en/github/getting-started-with-github/github-glossary#commit) per months
* Closed issues: The total number of [issues](https://docs.github.com/en/github/getting-started-with-github/github-glossary#issue) linked to the repository that have been resolved.
* Open issues: The total number of issues linked to the repository that are still open, thus not addressed yet.
* Average time to close an issue: The average time elapsed between the creation of an issue and when it has been resolved.

Libraries.io metrics (taken from <https://libraries.io/data> and <https://docs.libraries.io/overview.html> )

* SourceRank: A score of the repository, from 0 to 30, based on code, community, distribution, documentation and usage.
* (Dependencies: The number of dependencies that the repository have)
* Dependent packages: The number of [packages](https://libraries.io/data#projectFields) that have the repository as dependency
* Dependent repositories: The number of [repositories](https://libraries.io/data#repositoriesFields) that have the repository as dependency
* Stars: taken from GitHub
* (Forks: taken from GitHub)
* Contributors: taken from GitHub

PyPi metrics

* Downloads: the number of users that downloaded the repository from PyPi