# Project structure - Python

BCAM

October 2020

## Índice



## 1. Directory tree

The directory tree and its files should look like the following:

- example_pkg
  - LICENSE.txt
  - README.md (or README.txt)
  - requeriments.txt
  - tests
    - test_basic.py
    - test_special.py
  - docs

    - images
        - img1.png
        - img2.png
    - paper.pdf
    - usage.*
    - faq.*
- example_pkg
    - main.py
    - scripts
        - load_data.py
        - graphics.py
- data

# 2. Main packpage

## 2.1. README

We should have a readme file that contains information that is commonly required to understand what the project is about, the file is usually README.txt.ºr README.md"
The file should use UTF-8 encoding, typical contents for this file would include an overview of the project, basic usage examples, etc. Generally, including the project changelog in here is not a good idea, although a simple "What's New" section for the most recent version may be appropriate.
Keep in mind that a README file can be too long as we want, and too long is better than too short.
We can take a look at some extra tips on https://github.com/paperswithcode/releasing-research-code, so here we recommend that at least we can find in README file:

- Name
- Badges (optional)
  About License, Software, Social network, Downloads...
- Website (optional)
  Can be a link to the paper
- Version (optional)
- Description
- Visuals
  It could be helpful add screenshots or videos to complete the description and what our code does
- Installation / Requirements
  How the user can install the library and whether there are dependencies

- Training
  How to use the library for training, the necessary parameters and/or inputs, explained in detail to help the end user use it

- Evaluation
  Explain how the model is evaluated, that is, in a classifier would be the "predict"method, and the measures you want to add inside the evaluation section.

- Pre-trained models
  In case of publishing algorithms where we can leave an already trained model, we will indicate it in this section. The aim is to test our model in the easiest possible way, therefore, the user will only load the model and run the "predict"method.

- Results
  Here we will indicate the results we have achieved, we will be able to make a comparison, so that our success in the published project is easily recognizable.

- Support
  where they can go to for help, email address...
- Authors
- License
  Choose a license and indicate it here, it's a good idea add the badge as well, but it's good to make it clear in a separate section
- Citing
  Leave as simple as possible the steps to quote both our code and if we have a paper

#### 2.1.1. Badges

It is very visual way to indicate for example the license, the version of our software, the version of the programming language required (or merely the programming language), the number total of downloads/installations per week/-month (or last week/month), link to a social network account (such as Twitter, Reddit or YouTube), indicate whether we continue working on the repository or it's finished, whether we continue to provide a maintenance to the software we have published, the quality of the code, whether it has any vulnerability detected, whether the publication is open source...
In short, you can create the badge you want to give fast and accurate information to a potential user, the website https://shields.io/ can help.
We can see a repository of badges: https://github.com/Naereen/badges/blob/master/README.md

## 2.2. LICENSE.txt

This tells users who install our package the terms under which they can use our package.
The one that is more permissive corresponds to the MIT license, on the other hand, if we consider that the uploaded software has a potential use, it would be convenient to use the GNU GPLv3 license (it can change with new versions in the future)
So, to create the file "license.txt"we've just copy and paste the template updating some details like the year, fullname... and leave the file in the root directory.
We can add easily the file license.txt with GitHub following this link: https://docs.github.com/en/github/building-a-strong-community/adding-a-license-to-a-repository
A website that helps to choose the license, having templates for each of them is https://choosealicense.com

### 2.2.1. MIT License

Permissions with this license:

- Commercial use: The licensed material and derivatives may be used for commercial purposes
- Distribution: The licensed material may be distributed
- Modification: The licensed material may be modified
- Private use: The licensed material may be used and modified

Conditions with this license:

- License and copyright notice: A copy of the license and copyright notice must be included with the licensed material content...

Limitations with this license:

- Liability: This license includes a limitation of liability
- Warranty: This license explicitly states that it does NOT provide any warranty content...

Here is the MIT license template, so we've just to change the [year] and [fullname]:

Listing 1: MIT license template

```
                MIT License

                Copyright (c) [year] [fullname]

                Permission is hereby granted, free of charge, to any person obtaining a
copy
                of this software and associated documentation files (the "Software"), to
deal
                in the Software without restriction, including without limitation the rights
                to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
                copies of the Software, and to permit persons to whom the Software is
```

furnished to do so, subject to the following conditions:

The above copyright notice **and** this permission notice shall be included **in all** copies **or** substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

### 2.2.2. GNU GPLv3

The GPL is based on four freedoms: the freedom to use the source code for any purpose, the freedom to make modifications, the freedom to share the source code with anyone, and the freedom to share changes.
GPL does not prohibit users from selling derivative works that are based on the original source code, it merely requires source code to be freely available to anyone who wants it. This is the reciprocity obligation."
If our code is published under license X, any user will be obliged to publish his modifications under the same license (it is not possible to publish under license MIT for example)
Permissions with this license:

- Commercial use: The licensed material and derivatives may be used for commercial purposes
- Distribution: The licensed material may be distributed
- Modification: The licensed material may be modified
- Patent use: This license provides an express grant of patent rights from contributors
- Private use: The licensed material may be used and modified

Conditions with this license:

- Disclose source: Source code must be made available when the licensed material is distributed
- License and copyright notice: A copy of the license and copyright notice must be included with the licensed material
- Same license: Modifications must be released under the same license when distributing the licensed material. In some cases a similar or related license may be used

- State changes: Changes made to the licensed material must be documented

Limitations with this license:

- Liability: This license includes a limitation of liability
- Warranty: This license explicitly states that it does NOT provide any warranty

### 2.3. requirements.txt

It should specify the dependencies required to contribute to the project: testing, building, and generating documentation.
We'll install with this command:

Listing 2: requirements.txt

```
pip install -r requirements.txt
```

We can simply leave the name of the library and the latest one will be downloaded, but if there are compatibility problems, we can specify the version with -=". Here is an example:

Listing 3: requirements.txt

```
matplotlib==3.3.0
numpy
scipy==1.5.2
```

## 3. tests

We will store here all the available unit tests of our project.
We do not want to test the code, but the whole library/program, making a practical use that will serve as an example for future users.
We can create subfolders for each example as well.

## 4. docs

We can add additional documentation such as the user guide (user's manual, called üsage"), frequently asked questions (FAQ file, called "faq")
Sometimes, we will be interested in creating folders to improve the organization, such as creating an image folder called ïmages".

### 4.1. images

Here we will put the image files, if we need a structure for better organization, for example to separate by formats or by type of content, we can create subfolders

# 5. data

Here we can put the databases or files of the data source.
It can be at this level or inside the folder with the name of the package.
This directory is not mandatory, since if we do not want to upload the data source, we can omit it.

# 6. Packpage

This folder will have the same name as the root, and will be the name of our software. Inside we will have our software structured in an organized way.
Sometimes, our software will be composed of a main file and other secondary files that contain useful functions, in this case, it is advisable to create a folder called "scripts", to leave in this folder the main file of our software

## 6.1. scripts

Here will be the files containing the functions that are called from the main file.
These are usually functions grouped according to their functionality, such as "data loading", "graphics", "X value calculation", etc.
If we have several functions that we do not know how to group them, a useful resource is to call the file ütils".

# 7. Example

Here we can see examples:

- https://github.com/MachineLearningBCAM