Contributing code

=================

This guide is adapted from [scikit-learn](https://github.com/scikit-learn/scikit-learn/blob/master/CONTRIBUTING.md).

How to contribute

-----------------

The preferred way to contribute to imbalanced-learn is to fork the

[main repository](https://github.com/scikit-learn-contrib/imbalanced-learn) on

GitHub:

1. Fork the [project repository](https://github.com/scikit-learn-contrib/imbalanced-learn):

click on the 'Fork' button near the top of the page. This creates

a copy of the code under your account on the GitHub server.

2. Clone this copy to your local disk:

$ git clone git@github.com:YourLogin/imbalanced-learn.git

$ cd imblearn

3. Create a branch to hold your changes:

$ git checkout -b my-feature

and start making changes. Never work in the ``master`` branch!

4. Work on this copy on your computer using Git to do the version

control. When you're done editing, do:

$ git add modified\_files

$ git commit

to record your changes in Git, then push them to GitHub with:

$ git push -u origin my-feature

Finally, go to the web page of your fork of the imbalanced-learn repo,

and click 'Pull request' to send your changes to the maintainers for

review. This will send an email to the committers.

(If any of the above seems like magic to you, then look up the

[Git documentation](https://git-scm.com/documentation) on the web.)

Contributing Pull Requests

--------------------------

It is recommended to check that your contribution complies with the

following rules before submitting a pull request:

- Follow the

[coding-guidelines](http://scikit-learn.org/dev/developers/contributing.html#coding-guidelines)

as for scikit-learn.

- When applicable, use the validation tools and other code in the

`sklearn.utils` submodule. A list of utility routines available

for developers can be found in the

[Utilities for Developers](http://scikit-learn.org/dev/developers/utilities.html#developers-utils)

page.

- If your pull request addresses an issue, please use the title to describe

the issue and mention the issue number in the pull request description to

ensure a link is created to the original issue.

- All public methods should have informative docstrings with sample

usage presented as doctests when appropriate.

- Please prefix the title of your pull request with `[MRG]` if the

contribution is complete and should be subjected to a detailed review.

Incomplete contributions should be prefixed `[WIP]` to indicate a work

in progress (and changed to `[MRG]` when it matures). WIPs may be useful

to: indicate you are working on something to avoid duplicated work,

request broad review of functionality or API, or seek collaborators.

WIPs often benefit from the inclusion of a

[task list](https://github.com/blog/1375-task-lists-in-gfm-issues-pulls-comments)

in the PR description.

- All other tests pass when everything is rebuilt from scratch. On

Unix-like systems, check with (from the toplevel source folder):

$ make

- When adding additional functionality, provide at least one

example script in the ``examples/`` folder. Have a look at other

examples for reference. Examples should demonstrate why the new

functionality is useful in practice and, if possible, compare it

to other methods available in scikit-learn.

- Documentation and high-coverage tests are necessary for enhancements

to be accepted.

- At least one paragraph of narrative documentation with links to

references in the literature (with PDF links when possible) and

the example.

You can also check for common programming errors with the following

tools:

- Code with good unittest coverage (at least 80%), check with:

$ pip install pytest pytest-cov

$ pytest --cov=imblearn imblearn

- No pyflakes warnings, check with:

$ pip install pyflakes

$ pyflakes path/to/module.py

- No PEP8 warnings, check with:

$ pip install pycodestyle

$ pycodestyle path/to/module.py

- AutoPEP8 can help you fix some of the easy redundant errors:

$ pip install autopep8

$ autopep8 path/to/pep8.py

Filing bugs

-----------

We use Github issues to track all bugs and feature requests; feel free to

open an issue if you have found a bug or wish to see a feature implemented.

It is recommended to check that your issue complies with the

following rules before submitting:

- Verify that your issue is not being currently addressed by other

[issues](https://github.com/scikit-learn-contrib/imbalanced-learn/issues)

or [pull requests](https://github.com/scikit-learn-contrib/imbalanced-learn/pulls).

- Please ensure all code snippets and error messages are formatted in

appropriate code blocks.

See [Creating and highlighting code blocks](https://help.github.com/articles/creating-and-highlighting-code-blocks).

- Please include your operating system type and version number, as well

as your Python, scikit-learn, numpy, and scipy versions. This information

can be found by runnning the following code snippet:

```python

import platform; print(platform.platform())

import sys; print("Python", sys.version)

import numpy; print("NumPy", numpy.\_\_version\_\_)

import scipy; print("SciPy", scipy.\_\_version\_\_)

import sklearn; print("Scikit-Learn", sklearn.\_\_version\_\_)

import imblearn; print("Imbalanced-Learn", imblearn.\_\_version\_\_)

```

- Please be specific about what estimators and/or functions are involved

and the shape of the data, as appropriate; please include a

[reproducible](https://stackoverflow.com/help/mcve) code snippet

or link to a [gist](https://gist.github.com). If an exception is raised,

please provide the traceback.

Documentation

-------------

We are glad to accept any sort of documentation: function docstrings,

reStructuredText documents (like this one), tutorials, etc.

reStructuredText documents live in the source code repository under the

doc/ directory.

You can edit the documentation using any text editor and then generate

the HTML output by typing ``make html`` from the doc/ directory.

Alternatively, ``make`` can be used to quickly generate the

documentation without the example gallery. The resulting HTML files will

be placed in \_build/html/ and are viewable in a web browser. See the

README file in the doc/ directory for more information.

For building the documentation, you will need

[sphinx](http://sphinx-doc.org),

[matplotlib](https://matplotlib.org), and

[pillow](https://pillow.readthedocs.io).

When you are writing documentation, it is important to keep a good

compromise between mathematical and algorithmic details, and give

intuition to the reader on what the algorithm does. It is best to always

start with a small paragraph with a hand-waving explanation of what the

method does to the data and a figure (coming from an example)

illustrating it.