

Git Practical Session / Workflow

Instructors

Battista Biggio and **Luca Didaci**

M.Sc. in Computer Engineering, Cybersecurity and Artificial Intelligence

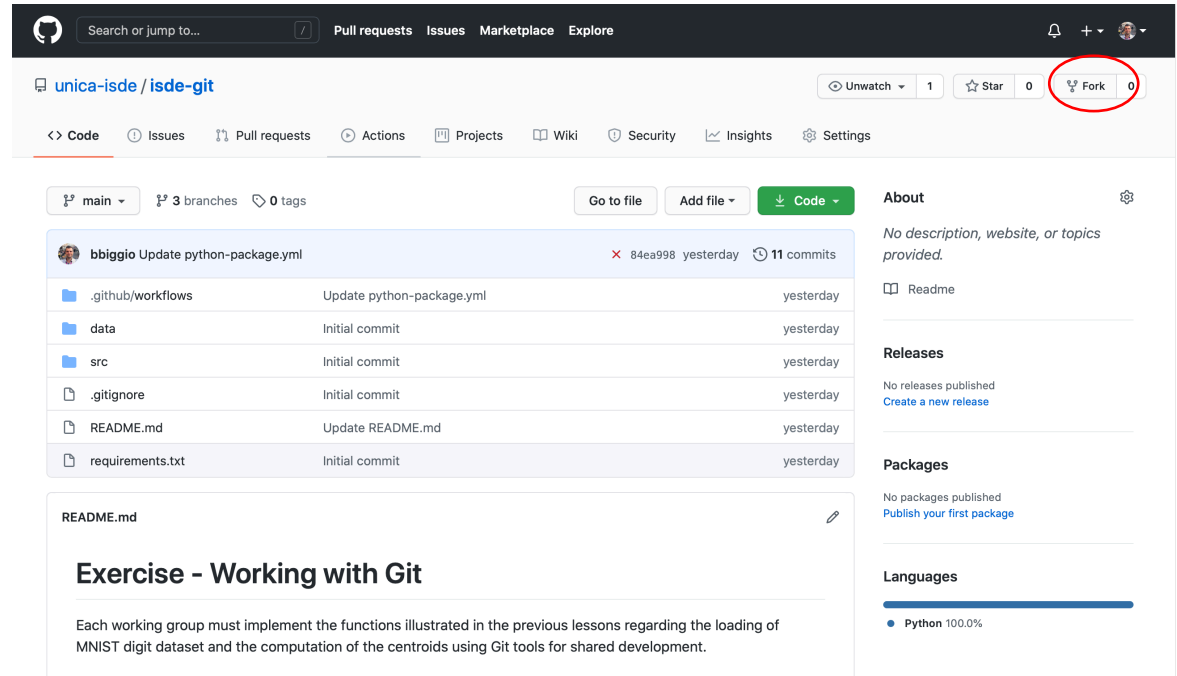
University of Cagliari, Italy

Git Workflow

- We will solve the Git exercise proposed in <https://github.com/unica-isde/isde-git>
- We will use a simplified Git workflow and we will see its integration with PyCharm
- The basic workflow for **Git** is as follows
 1. Fork the project (create a new master project)
 2. Create an issue in Git
 - Each issue has a separate identifier (e.g., #1)
 3. Create a branch associated to the issue to address the required changes
 4. Initial Implementation / testing, commit and push
 5. Create a merge/pull request associated to the aforementioned issue/branch
 6. Refine implementation / testing until completed
 7. Merge the branch to the master and close the corresponding issue and merge/pull request

Fork the Project

- Go to <https://github.com/unica-isde/isde-git>
- Click the *Fork* button
- This will create a copy of the project into your user home in GitHub
- Go to your forked project and create a new issue (next slide)



The screenshot shows the GitHub repository page for `unica-isde/isde-git`. The repository has 1 branch, 0 tags, 1 star, and 0 forks. The `Fork` button is circled in red. The repository contains the following files and folders:

File/Folder	Commit Message	Commit Date
<code>.github/workflows</code>	Update python-package.yml	yesterday
<code>data</code>	Initial commit	yesterday
<code>src</code>	Initial commit	yesterday
<code>.gitignore</code>	Initial commit	yesterday
<code>README.md</code>	Update README.md	yesterday
<code>requirements.txt</code>	Initial commit	yesterday

The `README.md` file contains the following text:

Exercise - Working with Git

Each working group must implement the functions illustrated in the previous lessons regarding the loading of MNIST digit dataset and the computation of the centroids using Git tools for shared development.

GitLab WorkFlow

1. Fork Project
2. Open Issue
3. Create Branch and Merge Request
4. Work & Discuss
5. Merge

The screenshot shows the GitLab web interface for a project named 'SecML'. The left sidebar contains navigation links: Project overview, Repository, Issues (18), List, Boards, Labels, Service Desk, Milestones, Iterations, Merge Requests (0), Requirements, CI / CD, Security & Compliance, Operations, Analytics, Wiki, and Members.

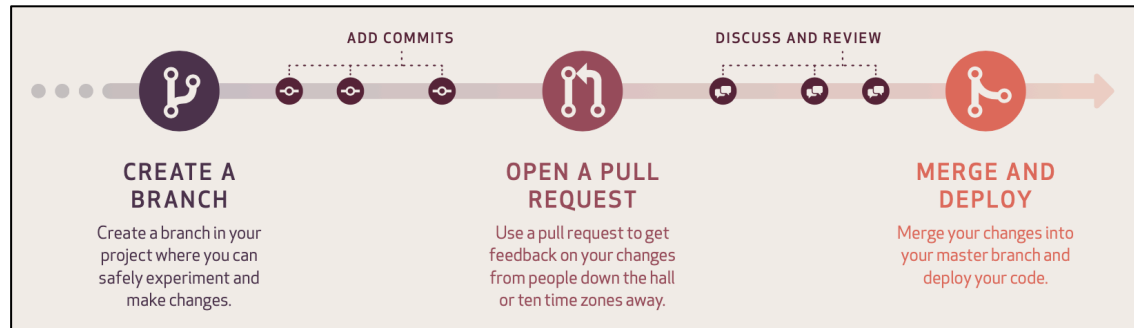
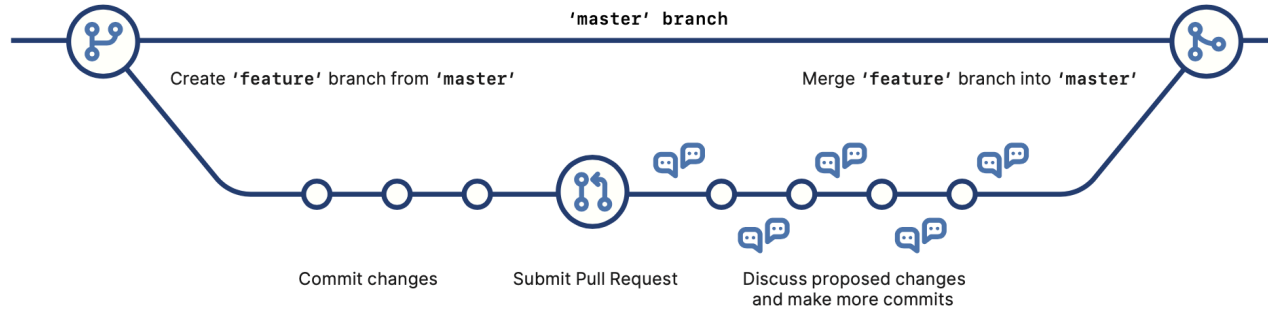
The main content area displays an issue titled 'Invalid format specifier when normalize=True in plot_confusion_matrix'. The issue is reported by 'SecML-Bot' and has a description: 'When `normalize=True` in `plot_confusion_matrix`, the function returns `ValueError: Invalid format specifier`, as the format specifier is wrongly set to `%2f`. This is the old print format, and should be abandoned. It should be set to `%.2f` instead.' The issue is marked as 'Open' and was opened 1 month ago by 'SecML-Bot'.

The right sidebar contains a 'To Do' section with links to 'Add a to do' and 'Edit'. Below this are sections for 'Assignee' (Marco Melis), 'Epic' (This feature is locked), 'Milestone' (None), 'Time tracking' (No estimate or time spent), 'Due date' (None), 'Labels' (bug, confirmed, package: figure, v0.14), 'Weight' (None), 'Health status' (None), 'Confidentiality' (Not confidential), and 'Lock issue' (Unlocked).

A red box highlights the 'Create merge request' button in the right sidebar. The button is located in the 'Create merge request' section of the sidebar, which also includes a 'Create branch' link and a 'Branch name' field (containing '858-invalid-format-specifier-when'). The 'Source (branch or tag)' field is set to 'master'.

GitHub Workflow

- **Main difference w/ GitLab:** no direct/one-to-one connection between issues and branches



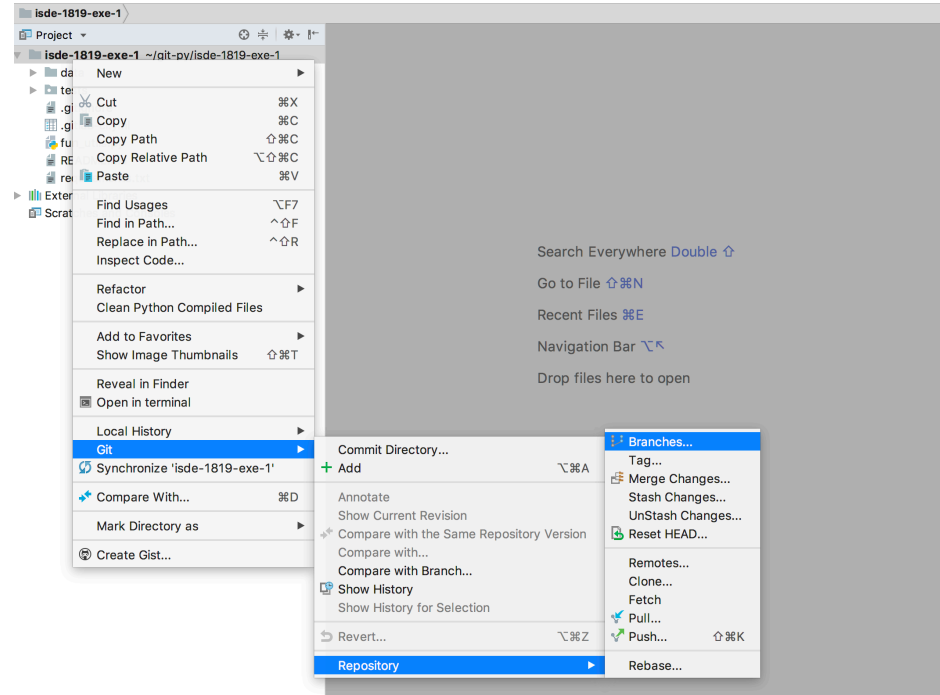
Create a New Issue

- Create a new issue and a corresponding branch; recall however that the exercise required creating a different issue for each function to be implemented (3 in total)

The image displays two screenshots of web interfaces for issue management. The left screenshot shows the GitLab interface for a repository named 'ISDe 1819 - Exercise 1'. It features a sidebar with navigation options like Project, Repository, Issues, Boards, Labels, Service Desk, and Milestones. The main content area shows an issue titled 'Implementation of split_data, count_digits, and fit.' created by Battista Biggio. The right screenshot shows the GitHub interface for a repository named 'unica-isde / isde-git'. It displays an issue titled 'add-feat1 #1' created by bbiggio. The interface includes a sidebar with navigation options like Code, Issues, Pull requests, Actions, Projects, Wiki, Security, Insights, and Settings. The main content area shows the issue details, including a comment by bbiggio and a section for writing a new comment.

Create the Corresponding Branch

- Open Pycharm, and create a new project by checking out your forked project from Git
 - Use *checkout from version control*, select *Git*, and copy-paste the URL from the project page
 - The URL is something like:
<https://github.com/your-username/isde-git.git>
- Create a new branch from the Git menu (or from the Git webpage)
 - Name it using the convention *issue_id-branch_name*
 - e.g., 1-split_and_fit
- Push the branch to the remote repository



Implement, Test, Commit & Push

- Implement the required functions
- Check if they pass the tests, otherwise keep working on them to fix the problems
- Commit (and keep coding if necessary)
 - Recall that *commit* only affects your local repository
- Push when finished
 - Pushes the changes to the remote branch

Create the Merge/Pull Request

Active branches

1-split_and_fit
4c432370 · New branch to address issue #1 · 41 seconds ago

0 1 Merge request Compare

master default protected
7ae9a64f · Update README.md · 1 month ago

New Merge Request

From 1-split_and_fit into master [Change branches](#)

Title WIP: New branch to address issue #1

Remove the **WIP:** prefix from the title to allow this **Work In Progress** merge request to be merged when it's ready.
Add [description templates](#) to help your contributors communicate effectively!

Description

Write Preview

Closes #1

Markdown and quick actions are supported [Attach a file](#)

Merging the Request and Closing the Issue

- Remove the «WIP» prefix from the merge request (GitLab) or comment that you finished on the pull request (GitHub)
 - This means: we are now done with this issue and ready to merge
- Merge the branch *1-split_and_fit* to the master/main
- Close the corresponding issue/merge request

That's it! This was a simple exercise to demonstrate how to use Git and a very basic Git workflow. You should follow this workflow in your home assignments, also using the Git Board to prioritize and track issues.

Homework: open another issue to document the code using docstrings, and then merge that to the master. Generate the docs locally on your machine (no need to upload them to the repository)

Recap: GitHub and Gitlab Cheat Sheets

- GitHub Cheat Sheet (basic commands, workflow)
 - <https://education.github.com/git-cheat-sheet-education.pdf>
- GitLab Cheat Sheet (basic commands, workflow)
 - <https://about.gitlab.com/images/press/git-cheat-sheet.pdf>