

INRAE



université  
PARIS-SACLAY

# ➤ Organizing repositories

**Alberto Tonda, Ph.D., Senior permanent researcher (DR)**

*UMR 518 Mathématiques et Informatique Appliquées - PS, INRAE, U. Paris-Saclay*

*UAR 3611 Institut des Systèmes Complexes Paris Ile-de-France, CNRS*

[alberto.tonda@inrae.fr](mailto:alberto.tonda@inrae.fr)

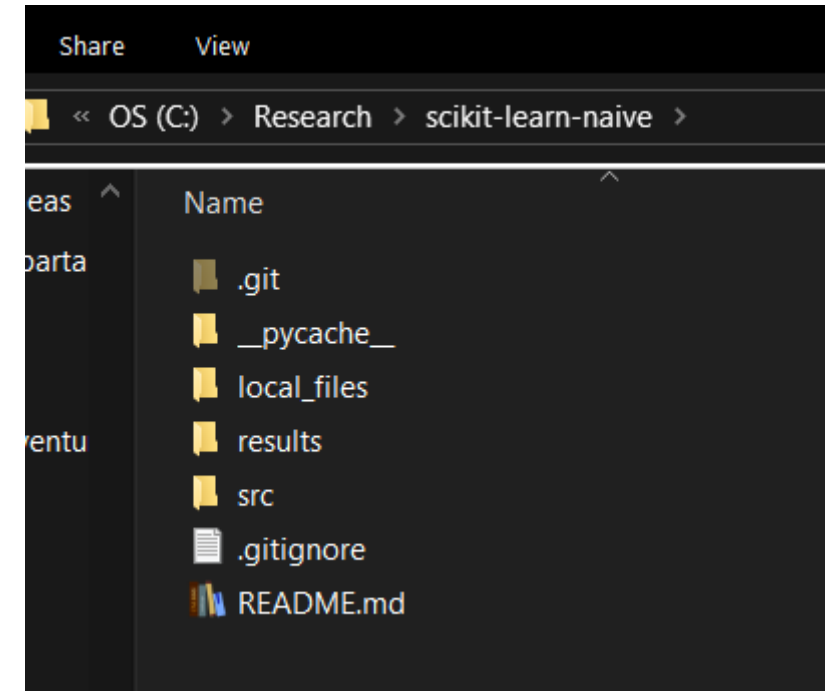
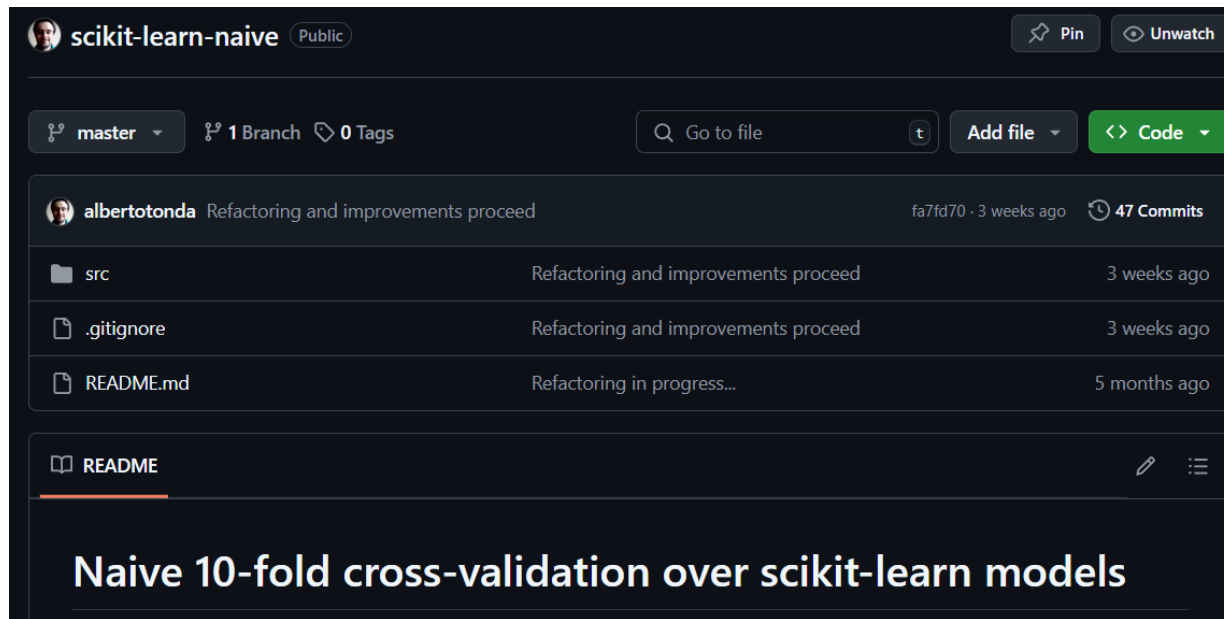
# ➤ Another example of title for a slide

- This is some text
  - And some smaller text

## Caveat

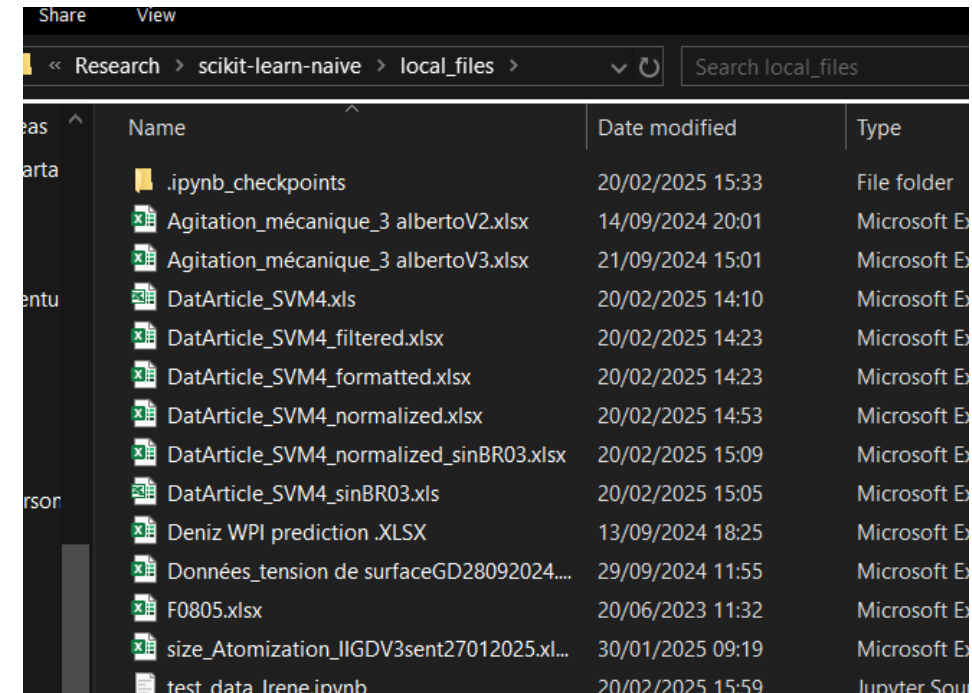
- This is mostly a mix of recommendations + experience

# ➤ Exploit functionalities of git



# ➤ Exploit functionalities of git

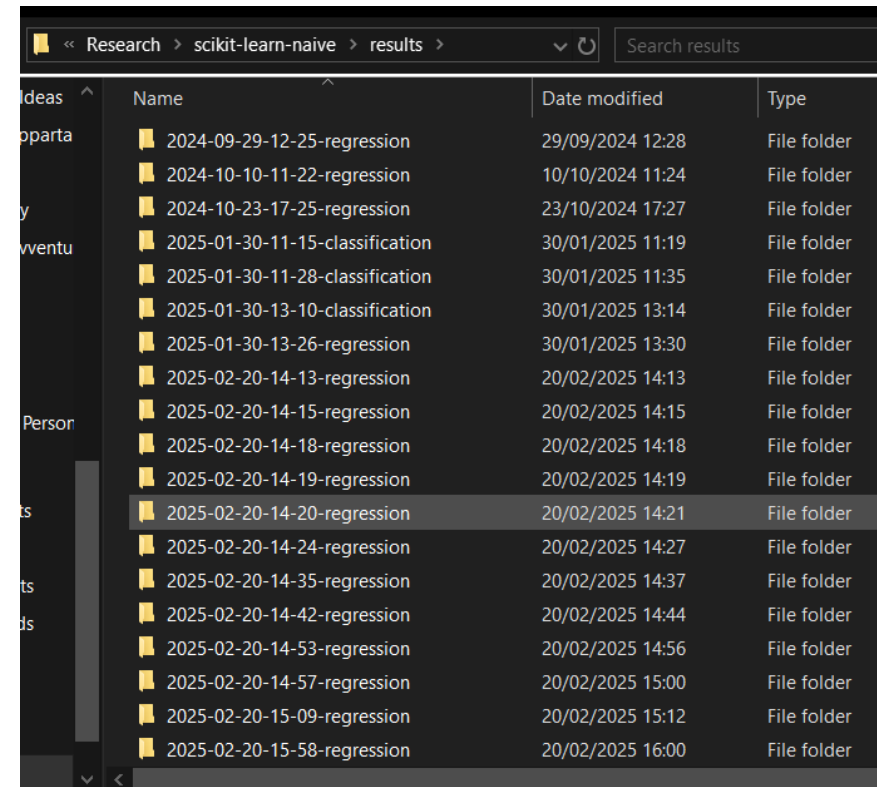
- You can ask git to ignore entire folders!
  - Good for files you don't want to put under version control
  - E.g. sensitive data from your collaborators



Name	Date modified	Type
.ipynb_checkpoints	20/02/2025 15:33	File folder
Agitation_mécanique_3 albertoV2.xlsx	14/09/2024 20:01	Microsoft Excel spreadsheet
Agitation_mécanique_3 albertoV3.xlsx	21/09/2024 15:01	Microsoft Excel spreadsheet
DatArticle_SVM4.xls	20/02/2025 14:10	Microsoft Excel spreadsheet
DatArticle_SVM4_filtered.xlsx	20/02/2025 14:23	Microsoft Excel spreadsheet
DatArticle_SVM4_formatted.xlsx	20/02/2025 14:23	Microsoft Excel spreadsheet
DatArticle_SVM4_normalized.xlsx	20/02/2025 14:53	Microsoft Excel spreadsheet
DatArticle_SVM4_normalized_sinBR03.xlsx	20/02/2025 15:09	Microsoft Excel spreadsheet
DatArticle_SVM4_sinBR03.xls	20/02/2025 15:05	Microsoft Excel spreadsheet
Deniz WPI prediction .XLSX	13/09/2024 18:25	Microsoft Excel spreadsheet
Données_tension de surfaceGD28092024....	29/09/2024 11:55	Microsoft Excel spreadsheet
F0805.xlsx	20/06/2023 11:32	Microsoft Excel spreadsheet
size_Atomization_IIGDV3sent27012025.xl...	30/01/2025 09:19	Microsoft Excel spreadsheet
test_data Irene.ipynb	20/02/2025 15:59	Jupyter Notebook

# ➤ Exploit functionalities of git

- But also useful to not clutter folder with results!
  - Put all results of your scripts (text files, CSV, etc.) in another folder



Name	Date modified	Type
2024-09-29-12-25-regression	29/09/2024 12:28	File folder
2024-10-10-11-22-regression	10/10/2024 11:24	File folder
2024-10-23-17-25-regression	23/10/2024 17:27	File folder
2025-01-30-11-15-classification	30/01/2025 11:19	File folder
2025-01-30-11-28-classification	30/01/2025 11:35	File folder
2025-01-30-13-10-classification	30/01/2025 13:14	File folder
2025-01-30-13-26-regression	30/01/2025 13:30	File folder
2025-02-20-14-13-regression	20/02/2025 14:13	File folder
2025-02-20-14-15-regression	20/02/2025 14:15	File folder
2025-02-20-14-18-regression	20/02/2025 14:18	File folder
2025-02-20-14-19-regression	20/02/2025 14:19	File folder
2025-02-20-14-20-regression	20/02/2025 14:21	File folder
2025-02-20-14-24-regression	20/02/2025 14:27	File folder
2025-02-20-14-35-regression	20/02/2025 14:37	File folder
2025-02-20-14-42-regression	20/02/2025 14:44	File folder
2025-02-20-14-53-regression	20/02/2025 14:56	File folder
2025-02-20-14-57-regression	20/02/2025 15:00	File folder
2025-02-20-15-09-regression	20/02/2025 15:12	File folder
2025-02-20-15-58-regression	20/02/2025 16:00	File folder

# ➤ Create meaningful subfolders for results

- Each time you run your script, create a subfolder
  - How to name subfolder? **datetime**
  - If you go Year-Month-Day-Hour-Minute-Second, your system will sort alphabetically in a descending order by recency!

```
# create uniquely named folder
folder_name = datetime.datetime.now().strftime("%Y-%m-%d-%H-%M") + "-classification"
folder_name = os.path.join(result_folder_name, folder_name)
if not os.path.exists(folder_name) :
    os.makedirs(folder_name)
```

# ➤ How to save results?

- CSV tables are pretty convenient
- The best workflow I found so far is
  - Create **dictionary** of **lists**
  - Keys are the column names, lists are elements in each row
  - Use **pandas** to convert dictionary to a pandas.DataFrame
  - Save dictionary to CSV using built-in **to\_csv()** method

```
# now, here we can write a final report; it's probably a
# probably the best way to go is to first create a dictio
df_dict = dict()
df_dict["classifier"] = []
df_dict["preprocessing"] = []
for metric_name in metrics:
    for t in ["train", "test"] :
        df_dict[metric_name + " " + t + " (mean)"] = []
        df_dict[metric_name + " " + t + " (std)"] = []
df_dict["AUC (mean)"] = []
df_dict["AUC (std)"] = []
```

```
# now that the dictionary is ready, convert it to a DataFrame
df = pd.DataFrame.from_dict(df_dict)

# since we are using a lot of different metrics, we have to pick one that will be used
df.sort_values(reference_metric + " test (mean)", ascending=False, inplace=True)

final_report_file_name = os.path.join(folder_name, final_report_file_name)
logger.info("Final results will be written to file \" + final_report_file_name + "\"..")
df.to_csv(final_report_file_name, index=False)
```

CHAGE -> MASQUE DE DIAPOSITIVES)



# ➤ Don't hard-code values in your code!

- Hard-coding means writing values by hand in the code
  - Try to limit this as much as possible
  - Way better to read values from outside (e.g. CSV, text or JSON file)
  - Why? Don't modify code every time you change something
  - JSON is more convenient to parse if data don't fit a table

# ➤ Logging

- Sometimes the output of your program is LONG
  - Going through the console is too difficult
  - Save output to console AND text file at the same time
  - Some messages just in the text file
- Logging! Using the **logging** module
- Creates a **logger** object (needs to be passed to functions)

## ➤ Scripts or notebooks?

- Always scripts, except for images or data summaries
  - For example, notebooks for plotting images for papers
  - And also generating tables as text files (Latex)
  - If you create images during a run, save them as part of script
- Examples from **symbolic-regression-ode-systems**

INRAE



université  
PARIS-SACLAY



[alberto.tonda@inrae.fr](mailto:alberto.tonda@inrae.fr)