

M2DS data-camp : classification supervisée de type cellulaire

Arthur Leroy - arthur.leroy@inrae.fr

M2 data science - Université Évry Paris Saclay

Challenge du 3 novembre 2025 au 12 janvier 2026

Programme du jour

Organisation

Présentation rapide des données et du challenge

Github classroom

La plateforme RAMP

Quelques outils utiles pour le développement Python

À vous !

Organisation

3 sessions + soutenance

- **Horaires:** Lundi 8h30-11h45
- **Lieu:** Salle 113, IBGBI (cette salle)

- Session 1 : lundi 03/11/2025 (aujourd'hui)
- Session 2 : lundi 17/11/2025
- Session 3 : lundi 01/12/2025
- Soutenance : lundi 12/01/2026

Bien envoyer son rapport avant la soutenance

Règles du challenge

- ▶ **Collaboration** : équipes de 2 minimum et 3 maximum
- ▶ Pas d'utilisation de données auxiliaires
- ▶ Créer un compte **Github** (si pas d'existant)
- ▶ Créer un compte **RAMP.studio** + demander l'accès à l'évènement *Challenge on single-cell type classification (2025)*
- ▶ Le suivi se fera via le fichier `README.md` de votre repository d'équipe.

Notation globale

1. **5 points** de gestion du projet et de travail en équipe via le suivi README du github de l'équipe : bilans hebdomadaire, répartition du travail équitable, etc.
2. **8 points** pour le rapport écrit avec prise en compte du classement RAMP
3. **7 points** pour la soutenance orale (points individuels)

Attention vous êtes évalués sur la qualité de l'ensemble votre démarche.

Sont valorisés

- une revue de littérature rapide sur les données singe-cell RNAseq
- l'analyse de données : statistiques descriptives, nettoyage, pre-processing, etc.
- l'interprétation des sorties de votre solution : points forts, points faibles, idées d'amélioration possibles, etc.
- La qualité du code

Ne sont pas valorisés : les solutions qui performant bien sans aucune tentative d'analyse.

Pour aujourd'hui

A la fin de cette journée, tout le monde devra avoir

- ▶ installé **Git**
- ▶ créé un environnement Python nommé `data-camp` avec `python>=3.10` installé `conda create -n data-camp python=3.11` (ou **uv**, **env**)
- ▶ rejoins une équipe sur l'assignement Github Classroom
- ▶ cloné son repository avec la commande `git clone <le-repo-de-son-équipe>` (ou via une interface graphique comme **GitHub Desktop**)
- ▶ exécuté `pip install -r requirements.txt`
- ▶ exécuté `ramp-test` sans erreur

Présentation rapide des données et du challenge

Pour 1 cellule unique $\rightarrow$ comptage de niveau d'expression des gènes

(Hyp) : $\neq$ gènes exprimés $\implies \neq$ types cellulaire

But : prédire le type cellulaire à partir des niveaux d'expressions de gènes

Formalisation statistique : classification supervisée

- n cellules et p gènes (variables).
- Observations : $\mathbf{X} \in \mathbb{N}^{n \times p}$
- Label : types cellulaire $\mathbf{y} \in \{1, \dots, K\}^n$

On souhaite apprendre un classifieur : $f(\mathbf{X}) = \mathbf{y}$

Le jeu de données scMARK

Jeu de données *benchmark* créé dans le but d'avoir un équivalent de MNIST pour les données single-cell

- Meta dataset : plusieurs dataset compilé en un seul
- 100,000 cellules et $p \approx 14,000$ gènes
- Les types cellulaires (labels) sont harmonisé (28 au total)
- <https://www.biorxiv.org/content/10.1101/2021.12.08.471773v1.full.pdf>

Extraction pour le challenge

- ▶ on travaillera avec $K = 4$ types cellulaires
 1. Cancer cells
 2. NKcells
 3. T cells CD4+
 4. T cells CD8+
- ▶ dataset public : $n = 1500$ séparées en $\frac{2}{3}$ train / $\frac{1}{3}$ test.

Github classroom

Deux inscriptions à faire (mêmes équipes) :

1. **Suivi-challenge** <https://classroom.github.com/a/04E5Md7M>
2. Challenge : <https://classroom.github.com/a/UIWDfGNN>

The screenshot shows a web browser window with the URL <https://classroom.github.com/assignment-invitations/c9285595dd0fba3e149dc20da1c8a156/team>. The page header includes the GitHub Classroom logo and navigation links for GitHub Education, chat, settings, and a profile icon. The main content area displays the assignment title "DataCampM2DS2023-classroom" and the instruction "Accept the group assignment — Suivi data camp". Below this, a message states: "Before you can accept this assignment, you must create or join a team. Be sure to select the correct team as you won't be able to change this later." The "Join an existing team" section shows a team named "test_team1" with 1 student and a "Join" button. The "OR Create a new team" section features a text input field "Create a new team" and a green "+ Create team" button.

Rejoindre les dépôts de l'équipe

GitHub Classroom

GitHub Education



You're ready to go — test-team

You accepted the assignment, **compte-rendu**.

Your team's assignment repository has been created:

 <https://github.com/DataCampDS/compte-rendu-test-team>

We've configured the repository associated with this assignment.

Note: You may receive an email invitation to join [DataCampDS](#) on your behalf. No further action is necessary.



Join the GitHub Student Developer Pack

Verified students receive free GitHub Pro plus thousands of dollars worth of the best real-world tools and training from GitHub Education partners — for free. For more information, visit [GitHub Student Developer Pack](#).

Apply

Tableau de bord d'équipe



[Pull requests](#) [Issues](#) [Codespaces](#) [Marketplace](#) [Explore](#)



DataCampM2DS

Follow

[Overview](#) [Repositories 2](#) [Projects](#) [Packages](#) [Teams 1](#) [People 2](#)

Repositories

Type

Language

Sort

New

[suivi-data-camp-2023-test_team1](#) Private

suivi-data-camp-2023-test_team1 created by GitHub Classroom

 0  0  0  0 Updated 2 minutes ago

[data-camp-cell-type-classification-for-scrna-seq-data-test_team1](#) Private

data-camp-cell-type-classification-for-scrna-seq-data-test_team1 created by GitHub Classroom

 Jupyter Notebook  0  0  0  0 Updated yesterday

 View as: **Public**

You are viewing the README and pinned repositories as a public user.

You can [create a README file](#) visible to anyone.

People

Top languages

 Jupyter Notebook

Suivi via README

 DataCampDS / compte-rendu-test-team

Q Type  to search

<> Code Issues Pull requests Actions Projects Wiki Security Insights Settings

 **compte-rendu-test-team** Public

Edit Pins Watch 0 Fork 0 Star 0

forked from [DataCampDS/suivi-data-camp-compte-rendu-suivi-datacamp](#)

 main  1 Branch  0 Tags

This branch is 2 commits ahead of [DataCampDS/suivi-data-camp-compte-rendu-suivi-datacamp:main](#).

 Contribute  Sync fork

 **ArthurLeroy** Update README.md bd7023b · now 3 Commits

 README.md Update README.md now

 **README** 

suivi-datacamp

Les points hebdomadaires se feront sur ce README.md

About 

suivi-data-camp-compte-rendu-suivi-datacamp created by GitHub Classroom

-  Readme
-  Activity
-  Custom properties
-  0 stars
-  0 watching
-  0 forks

[Report repository](#)

Releases

No releases published
[Create a new release](#)

Packages

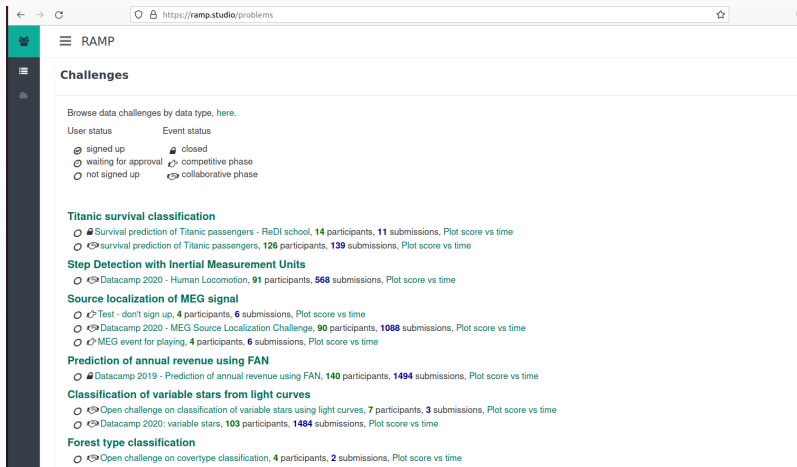
No packages published
[Publish your first package](#)

 © 2025 GitHub, Inc. [Terms](#) [Privacy](#) [Security](#) [Status](#) [Community](#) [Docs](#) [Contact](#) [Manage cookies](#) [Do not share my personal information](#)

La plateforme RAMP

Inscription à l'évènement

Inscriptions : https://ramp.studio/sign_up



The screenshot shows a web browser at the URL <https://ramp.studio/problems>. The page has a dark sidebar on the left with a menu icon and a 'RAMP' header. The main content area is titled 'Challenges' and includes a sub-header 'Browse data challenges by data type, here.' Below this, there are two columns: 'User status' and 'Event status'. The 'User status' column has three radio buttons: 'signed up', 'waiting for approval', and 'not signed up'. The 'Event status' column has three radio buttons: 'closed', 'competitive phase', and 'collaborative phase'. The main content area lists several challenges with their respective details:

- Titanic survival classification**
 - ☐ Survival prediction of Titanic passengers - ReDL school, **14** participants, **11** submissions, Plot score vs time
 - ☐ survival prediction of Titanic passengers, **126** participants, **139** submissions, Plot score vs time
- Step Detection with Inertial Measurement Units**
 - ☐ Datacamp 2020 - Human Locomotion, **91** participants, **568** submissions, Plot score vs time
- Source localization of MEG signal**
 - ☐ Test - don't sign up, **4** participants, **6** submissions, Plot score vs time
 - ☐ Datacamp 2020 - MEG Source Localization Challenge, **90** participants, **1088** submissions, Plot score vs time
 - ☐ MEG event for playing, **4** participants, **6** submissions, Plot score vs time
- Prediction of annual revenue using FAN**
 - ☐ Datacamp 2019 - Prediction of annual revenue using FAN, **140** participants, **1494** submissions, Plot score vs time
- Classification of variable stars from light curves**
 - ☐ Open challenge on classification of variable stars using light curves, **7** participants, **3** submissions, Plot score vs time
 - ☐ Datacamp 2020: variable stars, **103** participants, **1484** submissions, Plot score vs time
- Forest type classification**
 - ☐ Open challenge on covertype classification, **4** participants, **2** submissions, Plot score vs time

En local la commande `ramp-test --submission ma_soumission`

- `ma_soumission` contient au moins un fichier `classifier.py` (nom obligatoire)
- `classifier.py` définit obligatoirement une classe Python `Classifier` avec méthodes
 - `.fit(X, y)` : ne retourne rien
 - `.predict_proba(X)` : retourne une matrice avec les probas d'appartenance à chaque classe.
 - vous pouvez définir des fonctions/classe supplémentaires qui seront utilisées par `fit()` et `predict_proba()`

Sur le serveur

- ▶ **Soumission** : on soumet `ma_soumission` via la sandbox
- ▶ **Exécution** : le code tourne sur un serveur distant
- ▶ **Classement** : score de classement calculé sur une jeu de donnée test privé (non-accessible). Seul le score sur le test public est accessible.

Demo en live

Format d'une soumission :

- toute soumission commence **forcément** par le nom de l'équipe.
- vous pouvez ajouter une courte description de votre soumission, par exemple :
`team_XXX_PCA50_SVC`.

Quelques outils utiles pour le développement Python

Sur le code que vous produisez

Au fur et à mesure de vos expériences, vous allez produire beaucoup de code Python : analyse de données, petits scripts d'essais, etc.

On peut distinguer 2 niveaux

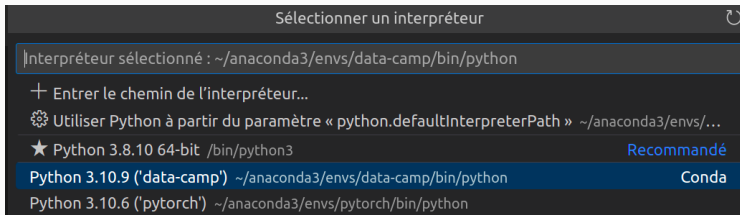
- ▶ **Local (votre machine)** : vous êtes totalement libre sur l'organisation.
- ▶ **Repository d'équipe** : il faut qu'il soit organisé de manière claire avec du code propre.

Attention à ne pas stocker de trop gros fichiers sur github (« 1Mo) : utilisez le `.gitignore` !

J'ai accès au code de vos soumissions sur RAMP

Visual Studio Code (VSCode)

- ▶ **Python Extension** : syntax color, autocomplete, line-by-line execution, debugger, etc.
- ▶ Gestion environnements : `ctrl + maj + P >python: select interpreter`



- ▶ Auto-formatage du code à la sauvegarde avec Black: `$pip install black`
- ▶ Terminal intégré (ou conda prompt pour Windows)
- ▶ Intègre Git

```
conda activate data-camp  
jupyter-lab
```

Conseils :

- **(Pros)** très utiles pour communiquer vos résultats de manière reproductible à l'équipe.
- **(Cons)** **Attention** ce n'est pas un bon environnement de développement ! Séparer les classes/fonctions utiles (.py) et l'expérience (.ipynb) !

Flake8 et black pour *linter* son code

Linter = "analyse statique du code"

- variable non déclarée
- variable inutilisée
- **non respect des bonnes pratiques du code** → flake8 + black

Exemple : `flake8 ugly_script.py`

```
ugly_script.py:1:80: E501 line too long (121 > 79 characters)
ugly_script.py:5:1: E302 expected 2 blank lines, found 1
ugly_script.py:6:1: W191 indentation contains tabs
ugly_script.py:7:1: W191 indentation contains tabs
```

Pour gagner du temps : VSCode + Black pour auto-formatage à la sauvegarde.

Deux prochaines séances

- ▶ Retours sur votre progression et conseils/feedback
- ▶ Conseils généraux pour développer

À vous !
