Machine learning: prediction, classification and clustering

UBB Faculty of Sociology

Course Agenda

#1 Intro, Simple Linear Regression

#2 Python recap, Git, Handling data, EDA

#3 Regression, Decision Trees

#4 Bias Variance, Overfitting, Classification, Metrics

#5 Random Forest Classifier, Clustering

#6 Neural Networks

#7 Help Final Project

#8 Help Final Project

2. git, Python recap, Data Handling

1.1 Catch up

Share one thing that stood out to you:

one thing you found surprising / interesting / useful etc.

1.2 Git

What is Git?

Git is a distributed version control system that allows developers to track changes in their code, collaborate seamlessly, and manage project history. It ensures efficient collaboration and provides tools for branching, merging, and resolving conflicts.

Key Features

- Version Control: Track and revert changes.
- Collaboration: Work with multiple contributors.
- Branching: Experiment without affecting the main code.

What is Version Control?

Version control is a tool for managing changes to a set of files.
There are many different version control systems:

- Git
- Mercurial (hg)
- CVS
- Subversion (svn)

Why use version control?

Version control is a tool for

- Better kind of backup.
- Review history ("When did I introduce this bug?").
- Restore older code versions. Ability to undo mistakes.
- Maintain several versions of the code at a time.
- "How can I share my code?"
- "How can I submit a change to someone else's code?"

Git != GitHub

Git: version control system tool to manage source code history. GitHub: hosting service for Git repositories.

```
>> git config --global user.name "YOUR NAME HERE"

>> git config --global user.email "yourname@example.com"
```

Git != GitHub

Git: version control system tool to manage source code history.

try these commands:

```
>> ls
>> tree
```

```
>> git status
>> git branch
```

Telling Git about the File

So, let’s tell Git that test.md is a file which is important, and we would like to keep track of its history:

```
>> git add test.md
```

Don’t forget: Any files in repositories which you want to “track” need to be added with git add after you create them.

Our first commit

Now, we need to tell Git to record the first version of this file in the history of changes:

```
>> git commit -m "First commit of the class"

>> git log
```

Git now has one change in its history:

You can see the commit message, author, and date...

Sharing your work

So far, all our work has been on our own computer. But a big part of the point of version control is keeping your work safe, on remote servers. Another part is making it easy to share your work with the world.

In this example, we’ll be using the GitHub cloud repository to store and publish

our work.

If you have not done so already, you should create an account on GitHub: go to

https://github.com/, fill in a username and password, and click on “sign up for free”.

GitHub private repositories

For this course, you should use public repositories in your personal account for your example work: it’s good to share! GitHub is free for open source, but in general, charges a fee if you want to keep your work private.
In the future, you might want to keep your work on GitHub private.

Students can get free private repositories on GitHub, by going to GitHub Education and filling in a form (look for the Student Developer Pack).

How it will work

Clone the repository. You, the collaborator need to go to the URL of the repository and **note the URL on the top of the screen**.

Create a feature branch: It’s a good practice to create a new branch that’ll contain the changes we want. Just think of this as a separate area where our changes will be kept not to interfere with other people’s work.

!Be sure to first navigate to the cloned directory, or git commands will not use this repository

```
>> git clone https://github.com/zahariesergiu/ubb-sociology-ml
>> cd ubb-sociology-ml
>> git checkout -b my-feature-branch-name
```

How it will work

Make, commit and push changes to new branch

For example, let’s create a new file in Google Colab called: `python_exercise.ipynb`

and edit it to add this:

```
>> print(‘Hello, Sociology + ML is fun !’)
```

Note:
There is also github desktop.
Check it!

Save it, and push this changes to your clone new branch:

```
>> git add python_exercise.ipynb
>> git commit -m "Implementing python exercises"
>> git push origin my-feature-branch-name
```

Push code during the work and when finished.
git push will push your whole history onto the server, and now you’ll be able to see it on the internet!
Refresh your web browser, and you’ll see your branch in the repository!

How much of python do we need to know before we get started with ML?

1.3 Python Recap

Let’s Go on Colab for a python recap!

Pull the code and run it:

```
>>git pull origin python-recap
```

1.01: Containers

1.02: Dictionaries

1.03: Data structures

1.04: Fields and records

1.05: Numpy

1.4 Python Exercises

Exercises for today, try to solve on your own:

`1.06: Exercises`

1.6 Data Handling with Python: Numpy, Pandas

Let’s go on Colab again.

Duplicate the notebook and run it in your laptop.

2.2.1: Pandas intro

2.2.2: Data wrangling: Penguins dataset

2.2.3: Consistency: modifying columns

2.2.4: Feature engineering

Handling data

Data is critical to data science. Two important considerations to keep in mind when faced with a dataset are

- **how** the **data was created**
  - **Understanding Data Sources**: Knowing how the data was generated, collected, or recorded helps you assess its quality
  - **Assessing Data Reliability**: Data collected under poor conditions (e.g., using faulty equipment or through unreliable respondents) might lead to errors or inaccuracies.
  - **Ethical Considerations**: How the data was created might raise ethical concerns (e.g., informed consent for survey participants or data privacy issues), affecting whether the data can or should be used.

- **how representative** the data **is.**
  - **Generalizability**: If the dataset isn't representative of the population or scenario you're analyzing, the conclusions drawn might not be applicable to other groups or contexts.
  - **Preventing Overfitting**: Non-representative data may lead to models that overfit specific patterns in the dataset but fail when applied to new data.
  - **Fairness and Equity**: A lack of representativeness can lead to unfair outcomes, especially in sensitive domains like healthcare, criminal justice, or hiring.

1.7 Final Project Task 1 - Census Data Preprocess

1.7 Further Reading & Questions

#1 Intro to version control: https://alan-turing-institute.github.io/rse-course/html/module04_version_control_with_git/index.html

#2 Research data in Python: https://alan-turing-institute.github.io/rse-course/html/module03_research_data_in_python/index.html

#3 Data feminism: https://data-feminism.mitpress.mit.edu/

#4 Living with Machines project:
https://blogs.bl.uk/thenewsroom/2019/07/moving-from-a-newspaper-collection-to-a-news-collection.html

#5 Project Odysseus:
https://www.turing.ac.uk/research/research-projects/project-odysseus-understanding-london-busyness-and-exiting-lockdown

#6 Data sources and formats: https://alan-turing-institute.github.io/rds-course/modules/m2/2.1.4-DataSourcesAndFormats.html

#7 GDPR, privacy and anonymisation:
https://alan-turing-institute.github.io/rds-course/modules/m2/2.2.5-PrivacyAndAnonymisation.html

#8 Data visualization: https://alan-turing-institute.github.io/rds-course/modules/m3/overview.html

#9 Dealing with an imbalanced dataset: https://towardsdatascience.com/how-to-deal-with-imbalanced-data-34ab7db9b100

Thank you !!

Machine Learning Engineer / Data Scientist
zahariesergiu@gmail.com
https://www.linkedin.com/in/zahariesergiu/
https://github.com/zahariesergiu/ubb-sociology-ml