



/ Congratulations,

You made it to your first build week!



/ Your boss is still not
convinced

Meet your boss 🔪🔥



Meet your boss 🔥

- No knowledge about Data Science.
- Was a designer & likes pretty slides. Else 🔥
- Will hire only the best team. Else, 🔥
- If someone doesn't speak, 🔥
- 5 minutes presentation + QA. Overtime, 🔥
- Meeting start time: 5pm GMT+1. Late? 🔥



Meet your ally, Sr Data Sci.



Meet your ally

- Cares a lot about Data Science.
- Has answers to all your questions
- Wants you to succeed and get the job
- Is extremely busy. Has a gap 2-3pm GMT+1
- Will give you honest feedback
- Will get upset if you don't improve later



How is your scraping?



How are your pandas?



How is your math?





/ Your boss trusts you a bit
this time, you have 1 week



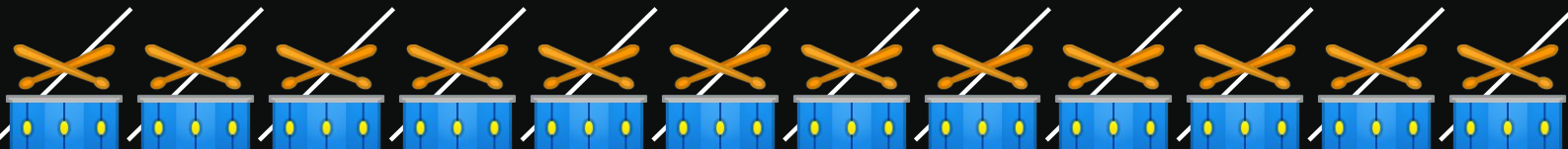
/ Challenge

The week will be divided into **three main goals**:

- **D1: Web Scraping & Analysis**
- **D2: Data Visualization**
- **D3: Complexity**
- **D4: Publishing**
- **D5: Presenting**



/ Challenge

and it will be about... 

Books! 

You will have to scrape [goodreads.com](https://www.goodreads.com), to create your own database with the scraped information, to process and analyse them, also with a bit of Data Visualization and finally make the result public!



/ Challenge

Each student group will be assigned to a list of [GoodReads Popular Lists](#) and scrape the information of the first 5000 books of the list.

Note: Iterate up to max number in case it has less.



/ Challenge

The pieces of information to be stored in a `.csv` file are the following:

Name	Description	Python Type
url	The Goodreads url of the book (e.g. https://www.goodreads.com/book/show/3.Harry_Potter_and_the_Sorcerer_s_Stone)	str
title	The title of the book e.g. "Harry Potter and the Sorcerer's Stone"	str
author	The author/authors of the book (e.g. J.K. Rowling, Mary GrandPré , in the format reported by Goodreads)	str
num_reviews	The number of user reviews (e.g. 115181, usually much less then num_ratings)	int
num_ratings	The number of ratings by user (e.g. 7270558, usually much more then num_reviews)	int



Name	Description	Python Type
avg_rating	The average rating (1 - 5)	int, from 1 to 5
num_pages	The total number of pages of the visualized edition (e.g. 309)	int
original_publish_year	The number of user reviews (e.g. 115181, usually much less then num_ratings)	int
series	A boolean indicating if this book belongs to a series or not (1 if yes, 0 if not)	bool
genres	A list of the first three genres of a book separated by commas (https://imgur.com/a/Ff2fefE e.g. Fantasy, Fiction, Young Adult)	str
awards	A list of awards (if any) won by this novel separated by commas (e.g. Mythopoeic Fantasy Award for Children's Literature (2008), British Book Award for Children's Book of the Year (1998), Prijs van de Nederlandse Kinderjury for 6-9 jaar en 10-12 jaar (2002), American Booksellers Book Of The Year Award for Children (1999)...)	str
places	A list of places (locations) that occur in this novel separated by commas (e.g. London, Hogwarts School of Witchcraft and Wizardry)	str



/ Challenge

In the table above, the first column entries represent the name of the columns of your database; the second column is a short description of the type of data you need to scrape; the third column is the Python type of the corresponding column.

Be aware that not all the books have all the information available directly on the website! Use `None` for those cases!



/ Scraper

The operations of the scraper will be wrapped into a function called `scraper` (WOW 🤯) that has two main goals:

- Scraping the required info
- Saving them in a `.csv` file



Preprocessing

Once you have built your .csv file you have to build another function called preprocessing, that:

- Loads the scraped data in a `pandas` DataFrame
- Transforms the column `awards` into a numerical column, containing only the numbers of awards (if any) won by the novel.

For example, in [Harry Potter and the Sorcerer's Stone](#) there are 27 awards.



Preprocessing

You know that a lot of reviews have a value between 4 and 5, and it is very hard to spot a difference between the different ratings. For this reason, you have to transform the `avg_ratings` column using the following criteria:

- `Min-Max Normalization`
- `Mean Normalization`



/ Min-Max Normalization

1. min-max normalization:

$$x' = \frac{x - \min(x)}{\max(x) - \min(x)}$$

- e.g.: if ratings range from 1.9 to 4.99, you will update the *BookRating* value as

$$\text{NewRating} = 1 + \frac{\text{BookRating} - 1.9}{4.99 - 1.9} * 9$$

- + :: • Multiplying by 9 and adding to 1 will scale the value from 1 to 10. Another numerical example: if the avg_rating of a book is 4.10 and the avg_rating of another book is 4.7 in the range described above, then we will have:

- $\text{new_rating1} = 1 + (4.1 - 1.9) / 3.09 * 9 = 7.40\text{s}$

- $\text{new_rating2} = 1 + (4.7 - 1.9) / 3.09 * 9 = 9.15$

- + :: • Now the difference is much more meaningful!

- Name the column with the new values `minmax_norm_ratings`

/ Mean Normalization

2. **mean normalization:** $x' = \frac{x - \text{average}(x)}{\max(x) - \min(x)}$

- e.g. if you have only four books, with `avg_rating = [1.9, 4.1, 4.7, 4.99]`, you have to compute the average of them $\frac{1.9+4.1+4.7+4.99}{4} = 3.92$ so the formula becomes:

- $x' = 1 + \frac{x - 3.92}{4.99 - 1.9} * 9$

- where we multiply by 9 and sum to 1 to get the values between 1 and 10.

- Doing the math for 4.1 and 4.7 as above we get:

- $\text{new_rating1} = 1 + (4.1 - 3.92) / 3.09 * 9 = 1.52$

- $\text{new_rating2} = 1 + (4.7 - 3.92) / 3.09 * 9 = 3.27$

Name the column with the new values `mean_norm_ratings`



Analyse

Once you have preprocessed your dataframe, you can play with it!

1. Group the books by `original_publish_year` and get the mean of the `minmax_norm_ratings` of the groups.
2. Create a function that given an author as input it returns her/his book with the highest `minmax_norm_ratings`.

/ Structure of the program

How the program will look like? In [python](#). You can use DeepNote for exploration but this week... it is Github time :)

The skeleton of your [.py](#) file will be similar to the following code:

```
import pandas as pd

def scraper():
    # your code here
def preprocessing(csv_path):
    # your code here
    return dataframe
def analyse(df):
    # your code here

def main():
    scraper()
    df = preprocessing(csv_path)
    analyse(df)

if __name__=="__main__":
    main()
```



/ Teams

You are hired!



Joby Ingram-Dodd Ibrahim Animashaun
Oluseyi Martin Vilar Karlen





Charles Degraft-Johnson Agnese
Michal Podlaskuk Nimeshsinh Desai





Gyasi Sutton

Tobias Schulz

Aderemi Fayoyiwa

Paramveer Singh

Pawas Awasthi



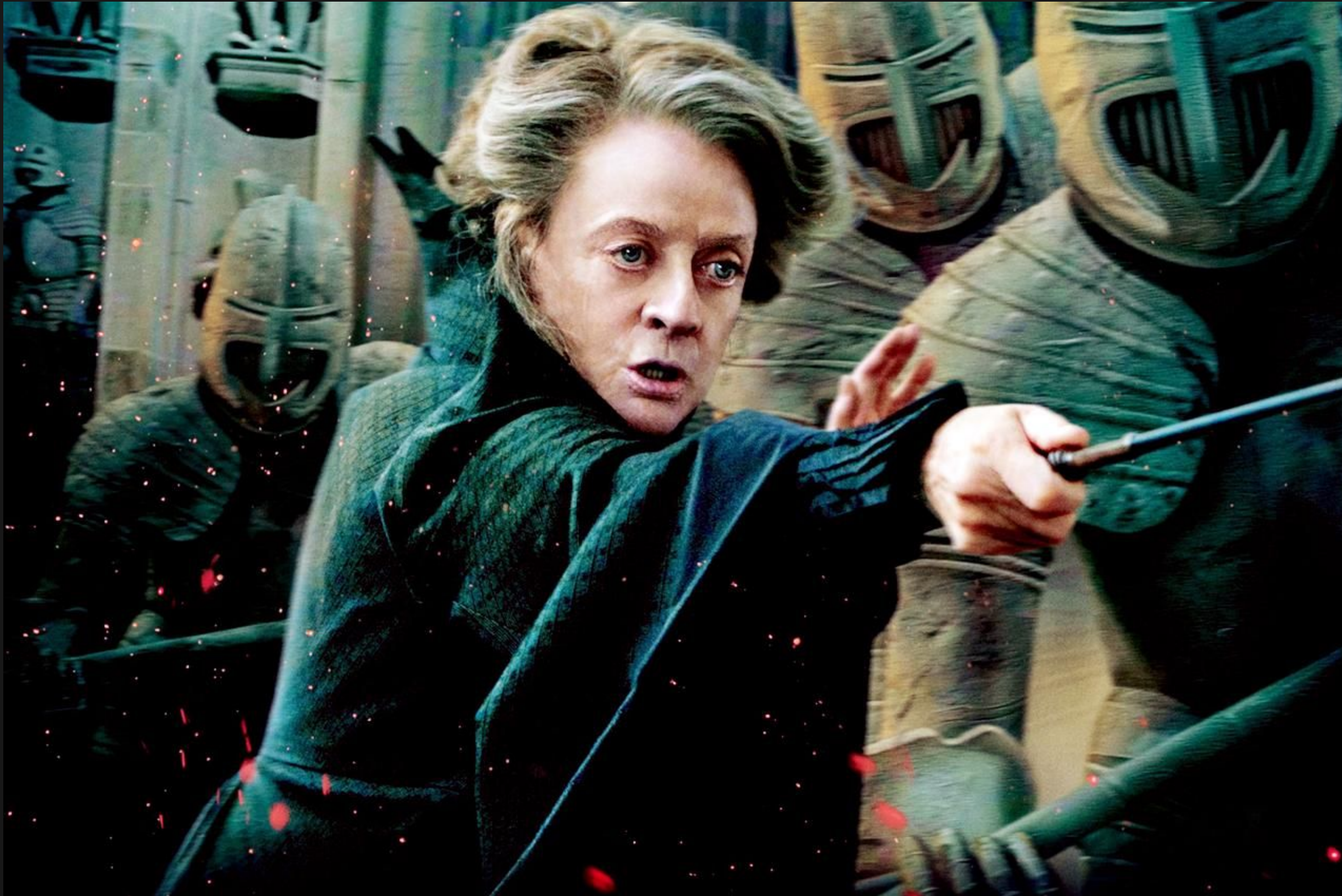
Hufflepuff



Saurabh Sataasia

Deepa Patel

Sai Mohan Reddy Dalli Marcin Szleszynski





Kimberley Taylor
Luca Pianta

Bence Kovacs
Lorenzo Demiri Zakariya M



/ Challenges

- **(Sly)** Best Books Ever
- **(Gry)** Best Dystopian and Post-Apocalyptic Fiction
- **(Huf)** The Best Epic Fantasy (fiction)
- **(McG)** Best Books of the Decade: 2000s
- **(Snape)** Best Books of the 20th Century

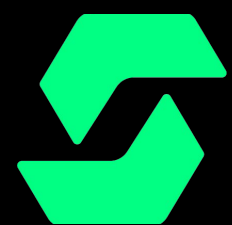
/ Some Ideas

- Organize and distribute the work.
- Have one person responsible for merging Github
- Plan also for tomorrow, you know more or less what is coming ;)



Q&A

What are your doubts?



Good luck!



