



Data Mining and Text Analytics 2025-2026

Dr Alessandro Bruno

IULM University

A.A. 2025 – 2026

Artificial Intelligence for Business and Society

# Python Project

---

Towards Data Mining and Text Analysis Exam  
Sessions:

Tips on how to deliver the Python project



# Outline

---

- Choice of topic
- Task
- Ways to Implementation
- An example:
  - Text Sentiment Analysis
- With a little help of ChatGPT
- Running and configuring the Project on your laptop
- Wrap-up
- Project Delivery
  - Uploading project + documentation to GitHub

# Step 1

Pick up a topic of your interest that you can tackle with data mining techniques (note that machine learning and deep learning can also be used for data and text mining)



Example: Sentiment Analysis

## Step 2

---

- Implement a technique to work out a specific problem.
  - For instance, you might be interested in running sentiment analysis on 20 textual reviews of the same item from different e-commerce websites and check whether customers exhibit the same opinion.





## Step 3

---

- Decide how to implement the solution:
  - You can code it from scratch (generally it takes a while as you need to design the whole project and know each step involved in the technique)
  - You can use an off-the-shelf solution by downloading from the Internet and testing it on your laptop (it usually saves you some time as you do not necessarily need to know every single step of the technique).
  - You can also use Virtual Chatbot based on LLMs such as ChatGPT and seek practical solutions (to be tested).
  - A compelling example for sentiment analysis is also given below:
  - <https://thecleverprogrammer.com/2020/05/25/movie-reviews-sentiment-analysis-binary-classification-with-machine-learning/>

# Step 4

---



REGARDLESS OF WHETHER YOU FIND A PROJECT OVER THE INTERNET OR SIMPLY ASK CHATGPT, BARD AND OTHER LLM-BASED VIRTUAL CHATBOTS, THERE ARE SOME RECOMMENDATIONS YOU NEED TO STICK TO.

YOU NEED TO SET UP A NEW VIRTUAL ENVIRONMENT USING ANACONDA (AS WE DID IN LAB SESSIONS)

THE VERY FIRST GOAL IS TO MAKE THE CODE WORK ON YOUR LAPTOP.

NO PANIC IF YOU STUMBLE INTO ERRORS (IT HAPPENS ALL THE TIME).

YOU CAN USE SEARCH ENGINES AND GENERATIVE-AI TOOLS (CHATGPT, BARD, LLAMA, ETC.) TO WORK OUT ISSUES WITH YOUR CODE.

# Step 4 (Example)

- I wanted to run Sentiment Analysis on user typed text from the terminal.
- I asked ChatGPT to make an example with the following requirements:



You

please write down a code file in python for sentiment analysis. The program has to be input text typed by users in the terminal and return the 2 highest detected sentiments.



ChatGPT

Sure, you can use the `transformers` library along with the `pipeline` module from the Hugging Face library to achieve sentiment analysis. First, make sure to install the required libraries:

```
bash
```

Copy code

```
pip install transformers  
pip install torch
```



# Step 4 (Example)

Going on with ChatGPT

Now, you can create a Python script for sentiment analysis as follows:



 Copy code

```
python

from transformers import pipeline

def get_sentiments(text):
    # Load the sentiment analysis pipeline
    sentiment_pipeline = pipeline("sentiment-analysis")

    # Perform sentiment analysis on the input text
    results = sentiment_pipeline(text)

    # Sort the results by the highest score
    sorted_results = sorted(results, key=lambda x: x['score'], reverse=True)
```

The whole snippet  
can be copied for  
your project

## Step 4 (Example)

The second code chunk is reported here.

```
# Return the two highest detected sentiments
return sorted_results[:2]

def main():
    # Get user input
    user_input = input("Enter text for sentiment analysis: ")

    # Perform sentiment analysis
    sentiments = get_sentiments(user_input)

    # Display the results
    print("\nTop 2 detected sentiments:")
    for sentiment in sentiments:
        print(f"{sentiment['label']}: {sentiment['score']:.4f}")

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



Save this code in a file, for example, `sentiment\_analysis.py`, and run it in the terminal. The program will prompt you to enter text for sentiment analysis, and it will then return the top two detected sentiments along with their scores.

Keep in mind that the performance of sentiment analysis models may vary, and you can explore different models or fine-tune them based on your specific requirements.

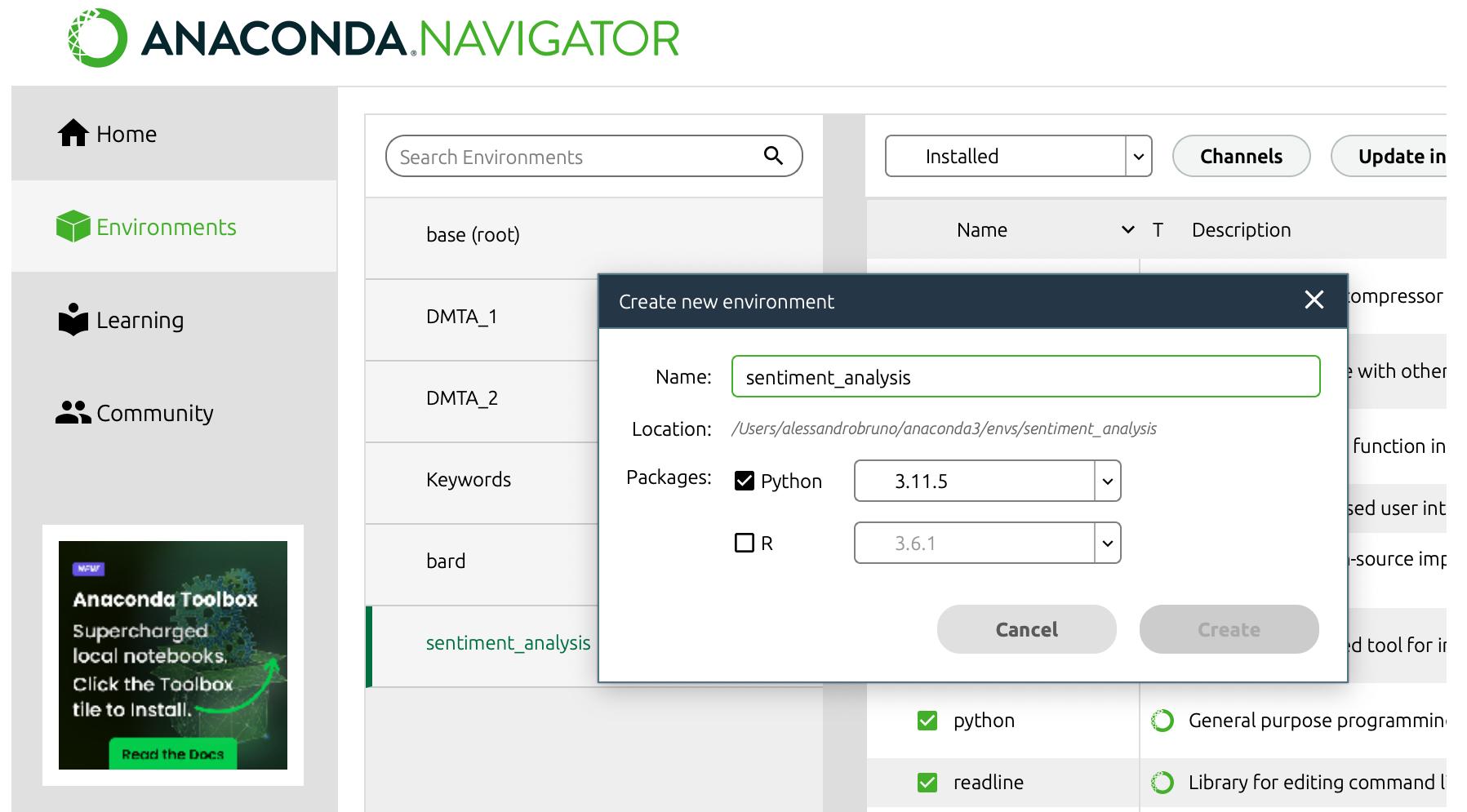




Step 5 (Running it on your  
laptop)

# Step 5 (Configuring & Running it on your laptop)

From Anaconda Navigator you set up a new environment 'sentiment\_analysis' as previously studied in the lab sessions.



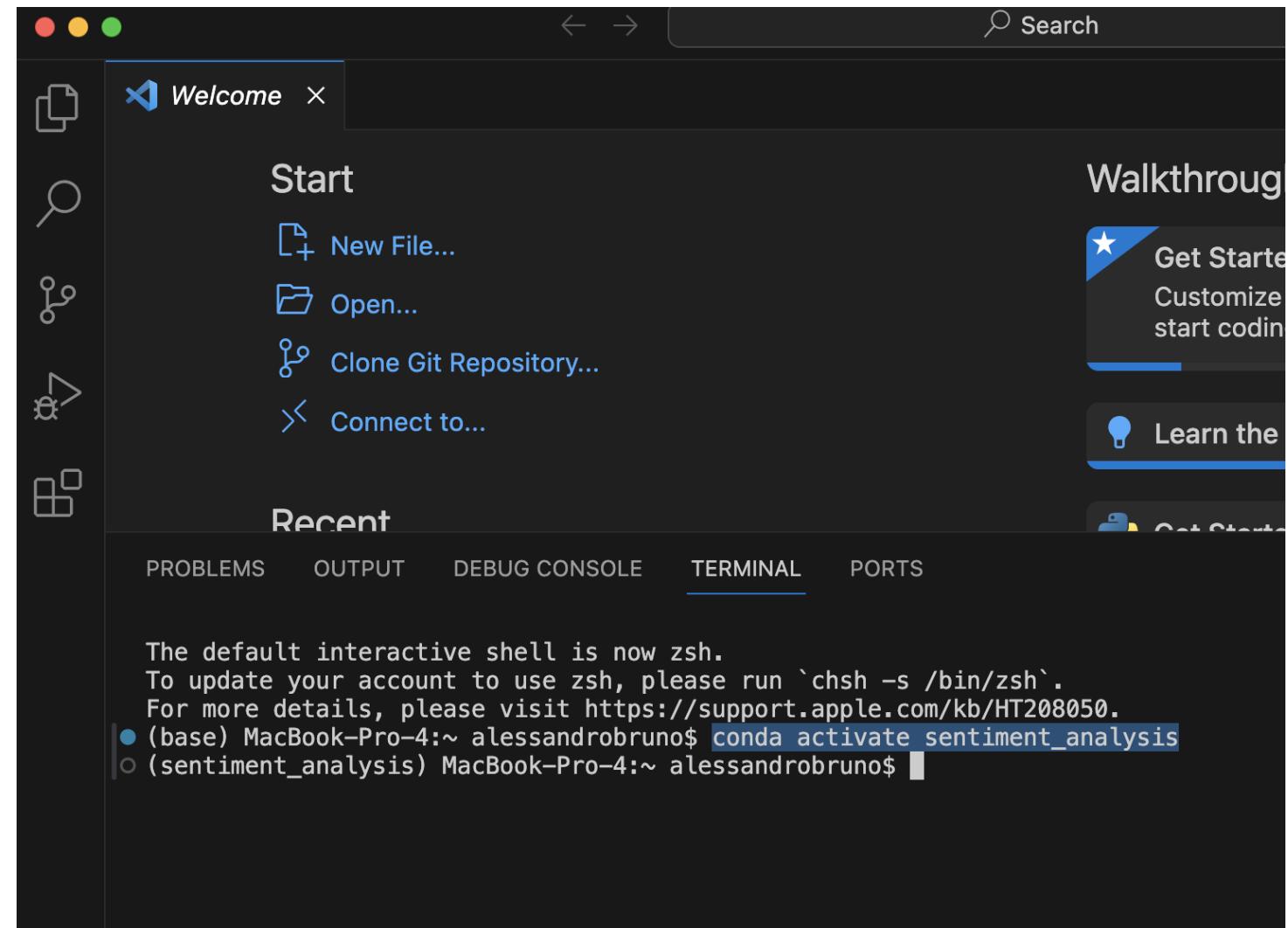
## Step 4 (Running on your laptop)

Open Visual Studio Code and go to the Terminal.

Activate your new environment using conda just like in the example by typing  
`conda activate sentiment_analysis`

Now, you need to create a project folder containing all the project files.

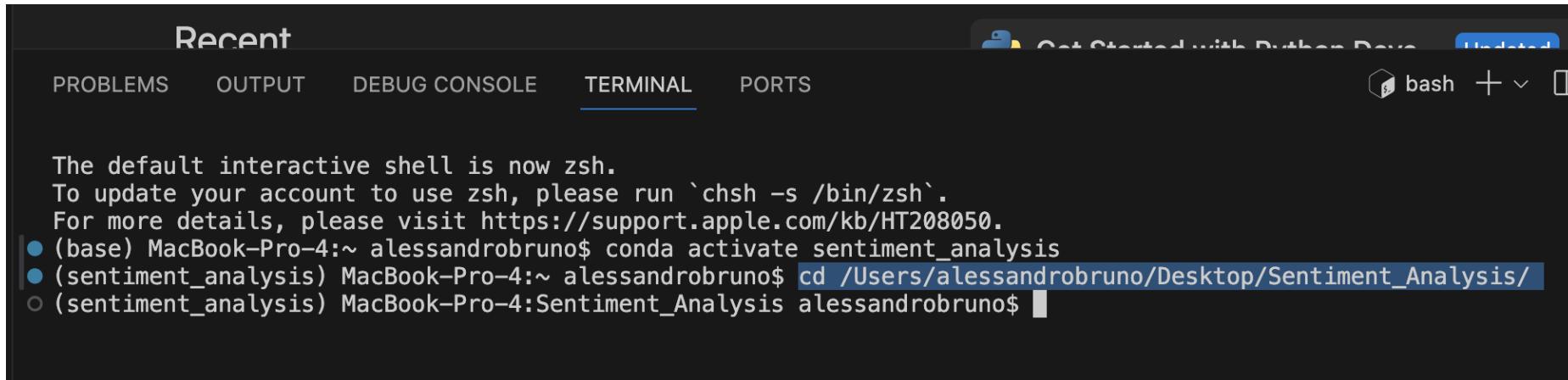
For simplicity's sake, I set up a directory named `Sentiment_Analysis` on my desktop.



The screenshot shows the Visual Studio Code interface. On the left, there's a sidebar with icons for file operations (New File..., Open...), cloning a Git repository, and connecting to a remote host. The main area is titled "Welcome" and has sections for "Start" (with options like New File..., Open..., Clone Git Repository..., Connect to...) and "Recent". Below the main area, there are tabs for PROBLEMS, OUTPUT, DEBUG CONSOLE, TERMINAL, and PORTS. The TERMINAL tab is active, showing a terminal session. The terminal output includes a message about the default interactive shell being zsh, instructions to update the account using `chsh -s /bin/zsh`, and a link to Apple support. It also shows two entries: a blue circle indicating the current session is in a base environment (MacBook-Pro-4:~ alessandrobruno\$) and a white circle indicating another session in a "sentiment\_analysis" environment (MacBook-Pro-4:~ alessandrobruno\$). The command `conda activate sentiment_analysis` is visible in the terminal.

# Step 5 (Running it on your laptop)

- Enter the project folder with your terminal



The screenshot shows a terminal window in the Visual Studio Code interface. The window title is "bash". The terminal tab is selected in the top navigation bar. The terminal output shows the following text:

```
The default interactive shell is now zsh.  
To update your account to use zsh, please run `chsh -s /bin/zsh`.  
For more details, please visit https://support.apple.com/kb/HT208050.  
● (base) MacBook-Pro-4:~ alessandrobruno$ conda activate sentiment_analysis  
● (sentiment_analysis) MacBook-Pro-4:~ alessandrobruno$ cd /Users/alessandrobruno/Desktop/Sentiment_Analysis/  
○ (sentiment_analysis) MacBook-Pro-4:Sentiment_Analysis alessandrobruno$ █
```

- Note that I use the path of the folder I set up on my laptop as an example. Double-check you are in the correct project folder!

# Step 5 (Running it on your laptop)

- Now, let's check out ChatGPT's instructions
- We need to configure the environment with two libraries: transformers and torch.
- We can do that on the terminal by using 'pip install name\_of\_library'
- In this case:
- **pip install transformers**
- **pip install torch**

```
The default interactive shell is now zsh.  
To update your account to use zsh, please run `chsh -s /bin/zsh`.  
For more details, please visit https://support.apple.com/kb/HT208050.  
● (base) MacBook-Pro-4:~ alessandrobruno$ conda activate sentiment_analysis  
● (sentiment_analysis) MacBook-Pro-4:~ alessandrobruno$ cd /Users/alessandrobruno/Desktop/Sentiment_Analysis/  
○ (sentiment_analysis) MacBook-Pro-4:Sentiment_Analysis alessandrobruno$ pip install transformers█
```

## Step 5 (Running it on your laptop)

- If the environment is correctly configured you should see some lines rolling down on your terminal and some ending statements on the successful installation of the package. The procedure needs to be accomplished for both packages (transformers and torch).

```
Requirement already satisfied: urllib3<1.27,>=1.21.1 in /usr/local/lib/python3.9/site-packages (1.26.3)
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.9/site-packages (2020.12.5)
Requirement already satisfied: idna<3,>=2.5 in /usr/local/lib/python3.9/site-packages (from)
Collecting safetensors>=0.3.1
  Downloading safetensors-0.4.1-cp39-cp39-macosx_10_7_x86_64.whl (441 kB)
    |██████████| 441 kB 11.2 MB/s
Collecting tokenizers<0.19,>=0.14
  Downloading tokenizers-0.15.0-cp39-cp39-macosx_10_7_x86_64.whl (2.6 MB)
    |██████████| 2.6 MB 9.0 MB/s
Installing collected packages: pyyaml, fsspec, filelock, huggingface-hub, tokenizers, safe
Successfully installed filelock-3.13.1 fsspec-2023.12.2 huggingface-hub-0.19.4 pyyaml-6.0.5.0 transformers-4.36.0
(sentiment_analysis) MacBook-Pro-4:Sentiment_Analysis alessandrobruno$
```

# Step 5 (Running it on your laptop)

- Everything is set and done with your environment:
  - Packages installed
  - Environment correctly configured on Anaconda Navigator
- Now, it is all about testing the example provided by ChatGPT.
- Copy and paste it in a new Python file in the [Project Folder](#)
- I set up a project folder named "**Sentiment\_Analysis**" on my Desktop

```
mirror_mod = modifier_obj.modifiers.new("MIRROR")
mirror_mod.mirror_object = active_object
if mirror_mod.mirror_object == None:
    print("please select exactly one object")
else:
    if operation == "MIRROR_X":
        mirror_mod.use_x = True
        mirror_mod.use_y = False
        mirror_mod.use_z = False
    elif operation == "MIRROR_Y":
        mirror_mod.use_x = False
        mirror_mod.use_y = True
        mirror_mod.use_z = False
    elif operation == "MIRROR_Z":
        mirror_mod.use_x = False
        mirror_mod.use_y = False
        mirror_mod.use_z = True

selection_at_the_end = False
if mirror_mod.select == 1:
    mirror_mod.select = 0
    bpy.context.scene.objects.active = context.scene.objects.active
    print("Selected" + str(modifier))
    mirror_mod.select = 1
    bpy.context.selected_objects.append(mirror_mod)
    data.objects[one.name].select = selection_at_the_end
    print("Selected" + str(modifier))

print("please select exactly one object")
print("Selected" + str(modifier))

- OPERATOR CLASSES ----

class MirrorOperator(bpy.types.Operator):
    bl_idname = "object.mirror"
    bl_label = "X mirror to the selected object.mirror_mirr...or X"
    bl_options = {'REGISTER', 'UNDO'}
```

The screenshot shows a Jupyter Notebook interface with the following details:

- Title Bar:** Welcome > sentiment\_analysis.py X
- File Path:** Users > alessandrobruno > Desktop > Sentiment\_Analysis > sentiment\_analysis.py
- Code Content:**

```
1 from transformers import pipeline
2
3 def get_sentiments(text):
4     # Load the sentiment analysis pipeline
5     sentiment_pipeline = pipeline("sentiment-analysis")
6
7     # Perform sentiment analysis on the input text
8     results = sentiment_pipeline(text)
9
10    # Sort the results by the highest score
11    sorted_results = sorted(results, key=lambda x: x['score'], reverse=True)
12
13    # Return the two highest detected sentiments
14    return sorted_results[:2]
15
16 def main():
17     # Get user input
18     user_input = input("Enter text for sentiment analysis: ")
```
- Status Bar:** Ln 2, Col 1 Spaces: 4 UTF-8 LF Python 3.11.5 ('sentiment\_analysis': conda)

## Step 5 (Running it on your laptop)

- Here is the Python file saved as `sentiment_analysis.py` in the project folder
- Now, let's run it!

# Step 5 (Running it on your laptop)

A screenshot of the Visual Studio Code (VS Code) interface. The title bar shows the file name "sentiment\_analysis.py". The code editor displays Python code for sentiment analysis:

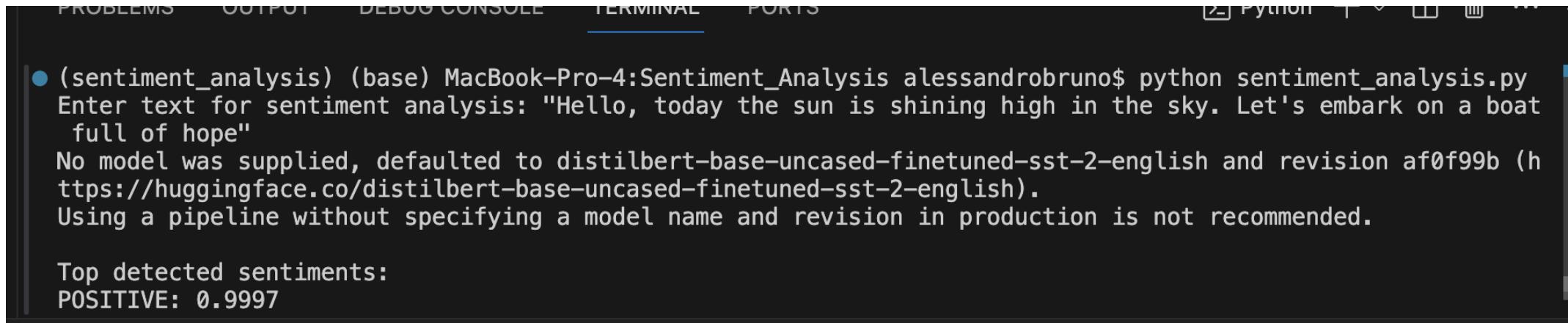
```
1 from transformers import pipeline
2
3 def get_sentiments(text):
4     # Load the sentiment analysis pipeline
5     sentiment_pipeline = pipeline("sentiment-analysis")
6
7     # Perform sentiment analysis on the input text
```

The bottom right corner shows the terminal window with the following output:

```
Downloading safetensors-0.4.1-cp39-cp39-macosx_10_7_x86_64.whl (441 kB)
|██████████| 441 kB 11.2 MB/s
Collecting tokenizers<0.19,>=0.14
  Downloading tokenizers-0.15.0-cp39-cp39-macosx_10_7_x86_64.whl (2.6 MB)
  |██████████| 2.6 MB 9.0 MB/s
Installing collected packages: pyyaml, fsspec, filelock, huggingface-hub, tokenizers, safetensors, transformers
Successfully installed filelock-3.13.1 fsspec-2023.12.2 huggingface-hub-0.19.4 pyyaml-6.0.1 safetensors-0.4.1
tokenizers-0.15.0 transformers-4.36.0
(sentiment_analysis) MacBook-Pro-4:Sentiment_Analysis alessandrobruno$ python sentiment_analysis.py
```

Option 1:  
Type  
**python sentiment\_analysis.py**  
on your terminal and press  
enter

# Running it on your laptop

A screenshot of a terminal window titled "Python". The window has tabs for PROBLEMS, OUTPUT, DEBUG CONSOLE, TERMINAL (which is selected), and PORTS. The terminal content shows a command-line session: "(sentiment\_analysis) (base) MacBook-Pro-4:Sentiment\_Analysis alessandrobruno\$ python sentiment\_analysis.py". The user enters the text "Hello, today the sun is shining high in the sky. Let's embark on a boat full of hope". A message follows: "No model was supplied, defaulted to distilbert-base-uncased-finetuned-sst-2-english and revision af0f99b (https://huggingface.co/distilbert-base-uncased-finetuned-sst-2-english)". A warning is given: "Using a pipeline without specifying a model name and revision in production is not recommended." The output then lists "Top detected sentiments: POSITIVE: 0.9997".

```
(sentiment_analysis) (base) MacBook-Pro-4:Sentiment_Analysis alessandrobruno$ python sentiment_analysis.py
Enter text for sentiment analysis: "Hello, today the sun is shining high in the sky. Let's embark on a boat
full of hope"
No model was supplied, defaulted to distilbert-base-uncased-finetuned-sst-2-english and revision af0f99b (h
ttps://huggingface.co/distilbert-base-uncased-finetuned-sst-2-english).
Using a pipeline without specifying a model name and revision in production is not recommended.

Top detected sentiments:
POSITIVE: 0.9997
```

The Python interpreter alerts us that we do not explicitly state a model for sentiment analysis. Therefore, it has defaulted to an available model for sentiment analysis on the statement I typed on the Terminal.

You can tell the program works well as the sentiment detected is 99% positive.

# Step 5 (Running it on your laptop)

The screenshot shows a Visual Studio Code interface with a red border around the main editor area. In the top bar, there are three icons (red, yellow, green) followed by a left arrow, a right arrow, a search bar labeled "Search", and a close button. Below the search bar, the file path "Users > alessandrobruno > Desktop > Sentiment\_Analysis > sentiment\_analysis.py > ..." is displayed. The main editor window contains the following Python code:

```
1 from transformers import pipeline
2
3 def get_sentiments(text):
4     # Load the sentiment analysis pipeline
5     sentiment_pipeline = pipeline("sentiment-analysis")
6
7     # Perform sentiment analysis on the input text
```

Below the editor, there are several icons: PROBLEMS, OUTPUT, DEBUG CONSOLE, TERMINAL (which is underlined), and PORTS. The TERMINAL tab is active, showing command-line output:

```
Downloading safetensors-0.4.1-cp39-cp39-macosx_10_7_x86_64.whl (441 kB)
|██████████| 441 kB 11.2 MB/s
Collecting tokenizers<0.19,>=0.14
  Downloading tokenizers-0.15.0-cp39-cp39-macosx_10_7_x86_64.whl (2.6 MB)
|██████████| 2.6 MB 9.0 MB/s
Installing collected packages: pyyaml, fsspec, filelock, huggingface-hub, tokenizers, safetensors, transformers
Successfully installed filelock-3.13.1 fsspec-2023.12.2 huggingface-hub-0.19.4 pyyaml-6.0.1 safetensors-0.4.1
tokenizers-0.15.0 transformers-4.36.0
(a sentiment_analysis) MacBook-Pro-4:Sentiment_Analysis alessandrobruno$ python sentiment_analysis.py
```

At the bottom of the terminal, status indicators show: Spaces: 4, UTF-8, LF, { Python 3.11.5 ('sentiment\_analysis': conda)}, Go Live, and kite: not installed.

Option 2:  
Click on the "run" icon over here, which will use the interpreter configured in the current session of Visual Studio Code.  
Be advised to DOUBLE-CHECK it to avoid any mismatch between your environment interpreter and interpreters for other projects.  
If the interpreter is not the right one, click on it and pick the one from a dropdown menu.

The background of the slide features a close-up, slightly blurred photograph of a stack of spiral-bound notebooks. The notebooks are bound in the center and come in various colors, including white, red, blue, yellow, and green. Some have floral or abstract designs on the covers. The spiral binding creates a series of concentric circles that wrap around the text area.

# Step 6 (Project Wrap-up)

A close-up photograph of a stack of papers or documents. On the left, a metal fastener or clip is visible. The pages are aged, with some yellowing and brown spotting. A green strip of tape runs diagonally across the stack.

# Project Documentation

- README
- License
- Acknowledgment
- Project Folder (program files)

# README file

- The README.md file is an indispensable asset for any GitHub repository, acting as a user-friendly guide that empowers individuals to effectively understand, utilize, and contribute to the project.
- By carefully crafting and maintaining a comprehensive README file, developers can enhance the project's visibility, adoption, and overall success.



# README file of the project (Example)

- # Sentiment Analysis with Transformers
- This project demonstrates a simple sentiment analysis program using the Hugging Face Transformers library. It allows users to input text in the terminal, and the program returns the top two detected sentiments along with their scores.
- ## Getting Started
- ### Prerequisites
- Make sure you have Python installed on your machine. You can install the required packages using the following command:
  - ``bash
  - pip install transformers torch

# README file of the project (Example)

- Installation

Clone the repository:

```
git clone https://github.com/your-username/sentiment-analysis.git
```

- Navigate to the project directory:

```
cd sentiment-analysis
```

- Usage

Run the script by executing the following command in the terminal:

```
python sentiment_analysis.py
```



# LICENSE

- Concerning the license, that's a matter you should dig in using the straightforward guidelines at the following link:
- <https://choosealicense.com/>
- Most of the time you will need to opt for a license regarding use, reproduction, commercialisation, liability and other legal and ethical aspects.
- The project you are going to deal with has only education purposes, therefore you can pick up the most permissive use license.
- However, reading the content of popular licenses such as MIT, and Apache 2.0.

# Overview of GitHub licenses

- Credits: [Licenses on GitHub](#)

	 Apache	 BSD	 MIT	 GPL Free as in Freedom	 LGPL Free Software Free as in Freedom	 AGPL Free Software Free as in Freedom
Type	Permissive	Permissive	Permissive	Copyleft	Copyleft	Copyleft
Provides copyright protection	✓ TRUE	✓ TRUE	✓ TRUE	✓ TRUE	✓ TRUE	✓ TRUE
Can be used in commercial applications	✓ TRUE	✓ TRUE	✓ TRUE	✓ TRUE	✓ TRUE	✓ TRUE
Provides an explicit patent license	✓ TRUE	✗ FALSE	✗ FALSE	✗ FALSE	✗ FALSE	✗ FALSE
Can be used in proprietary (closed source) projects	✓ TRUE	✓ TRUE	✓ TRUE	✗ FALSE	✗ FALSE partially	✗ FALSE for web
Popular open-source and free projects	Kubernetes Swift Firebase	Django React Flutter	Angular.js JQuery .NET Core .Laravel	Joomla Notepad++ MySQL	Qt SharpDevelop	SugarCRM Launchpad



# Acknowledgement

Generally, you can acknowledge other people's work, projects, sources you find inspiring for your work in this document.

It is a sort of "Credits" section. Here is a an example:

## Acknowledgement

'The authors of the project acknowledge the platform "Kaggle.com". It has been broadly used to gather data for the project.

Regargind the authors' contribution, here is how we developed the project pipeline. (Use initials of full names to indicate each team member followed by the task he or she was involved in the project).

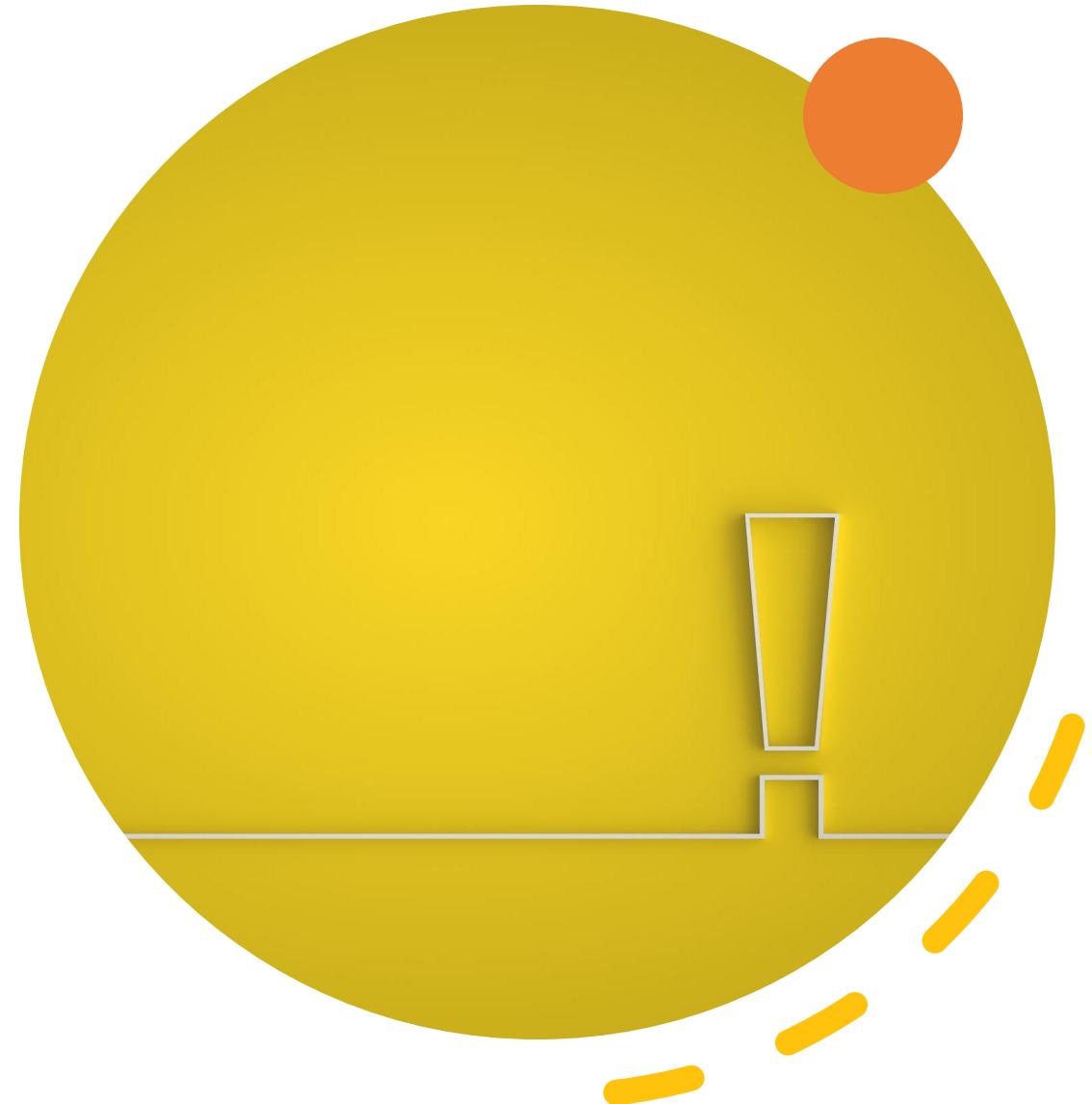
A.B. ran an extensive research over the Internet. M.M. checked out the project steps using Generative AI platforms such as ChatGPT. C.A. dealt with the code debugging using Visual Studio Code and ChatGPT. A.B. and M.M. wrapped up the code. C.A. wrote the documentation. All the team members worked along to upload the final version of the project to GitHub.

# Step 7 (Project Delivery)



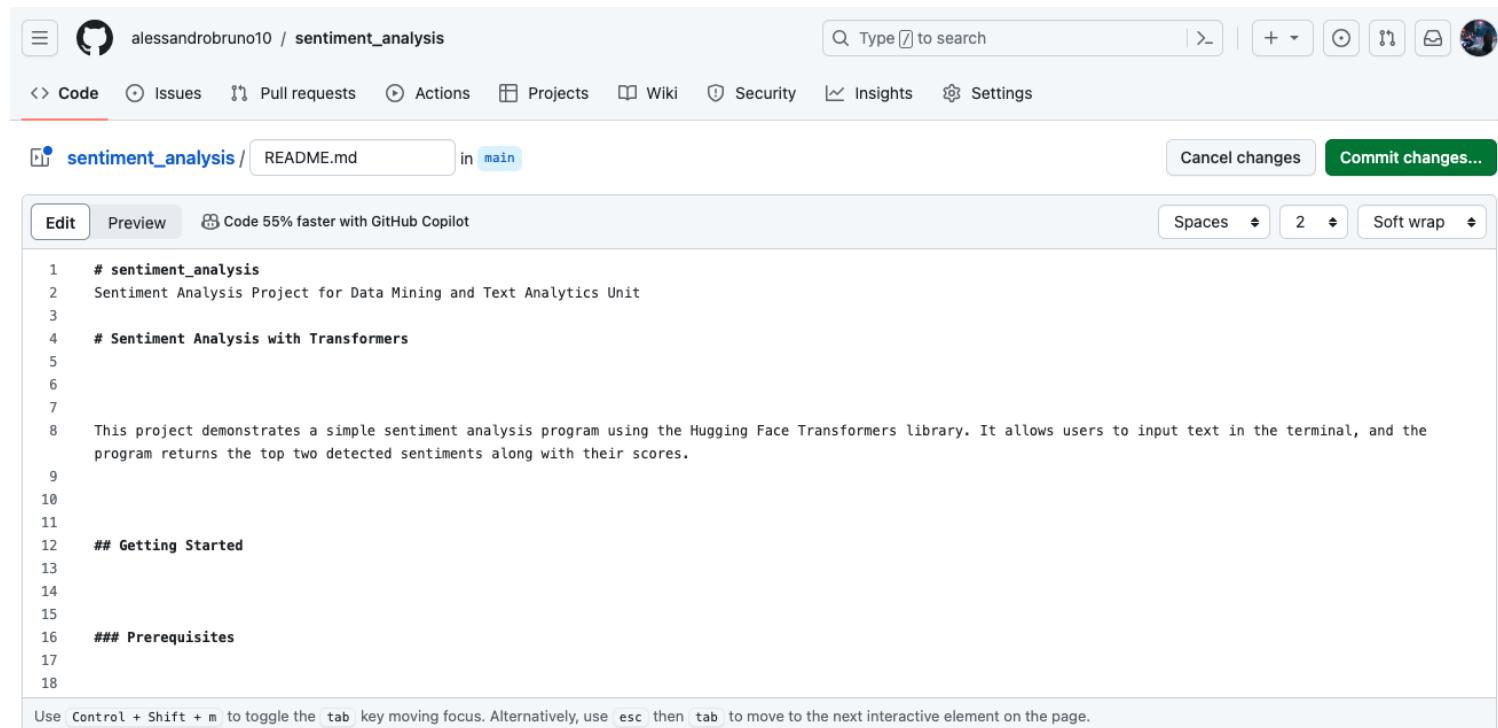
# Upload to GitHub Repository

- A recap on the files to pack in the GitHub Repo:
- Project Files (python, libraries, datasets)
- README file
- LICENSE
- ACKNOWLEDGMENTS



# GitHub Repo

- Set up your Project repo on GitHub
- If you need to edit your README file, you can do that accessing it making changes.
- Note that all changes need to be committed as shown below.



The screenshot shows a GitHub repository interface for a project named "sentiment\_analysis". The user is editing the "README.md" file in the "main" branch. The editor displays the following content:

```
1 # sentiment_analysis
2 Sentiment Analysis Project for Data Mining and Text Analytics Unit
3
4 # Sentiment Analysis with Transformers
5
6
7
8 This project demonstrates a simple sentiment analysis program using the Hugging Face Transformers library. It allows users to input text in the terminal, and the
9 program returns the top two detected sentiments along with their scores.
10
11
12 ## Getting Started
13
14
15
16 ### Prerequisites
17
18
```

At the bottom of the editor, there are "Cancel changes" and "Commit changes..." buttons. A blue arrow points from the right towards the "Commit changes..." button, indicating where the user should click after making changes. The GitHub interface includes a navigation bar with links for Code, Issues, Pull requests, Actions, Projects, Wiki, Security, Insights, and Settings.

Click over  
here after you  
make changes

# README file after committing changes

The screenshot shows a GitHub repository page for a project named "sentiment\_analysis". The page displays the contents of the README.md file. The content includes a title section, a subtitle, and a detailed description of the project's purpose. The GitHub interface shows standard navigation and file statistics.

sentiment\_analysis / README.md

alessandrobrunotD (Create README.md) · 62327s · 62

Preview Code Blame 43 Lines | 123 Loc | 103 Bytes ⚡ Code 66% faster with GitHub Copilot Raw ⌂ ⌄

---

## sentiment\_analysis

Sentiment Analysis Project for Data Mining and Text Analytics Unit

### Sentiment Analysis with Transformers

---

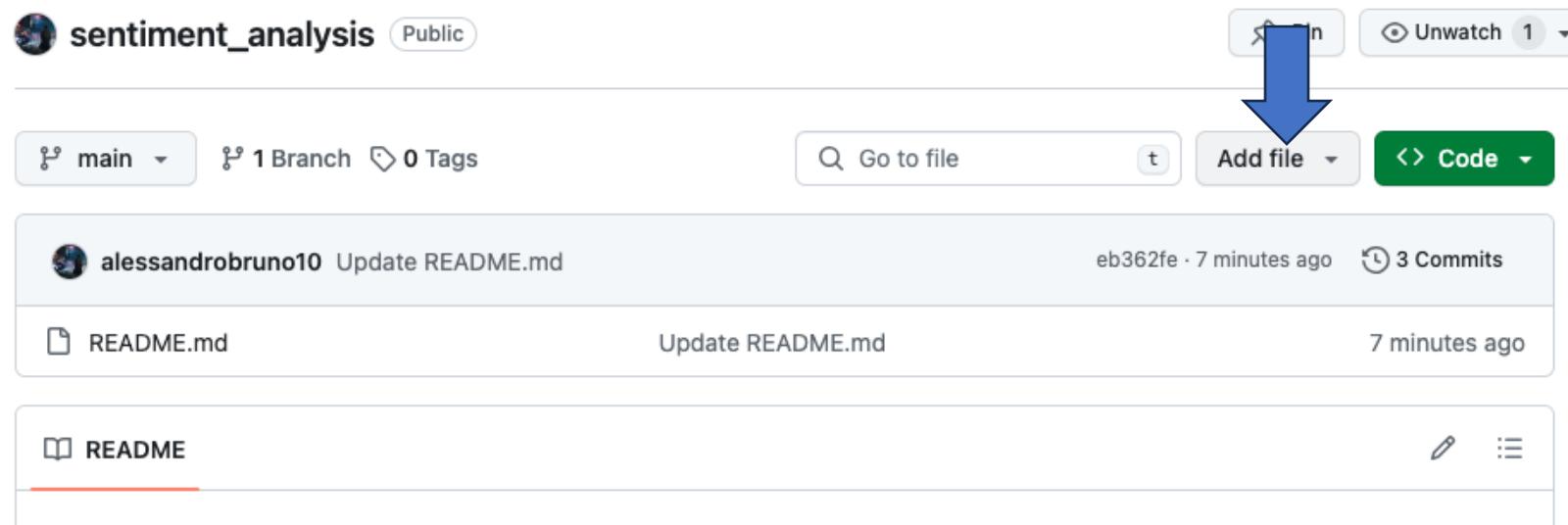
This project demonstrates a simple sentiment analysis program using the Hugging Face Transformers library. It allows users to input text in the terminal, and the program returns the top two detected sentiments along with their scores.

---

#### Getting Started

# One approach for all files

- You can use the same approach for all documentation files.
- Make all edits you need to for your files and commit changes.
- Moreover, If you forget adding the Acknowledgment or other files, you can always upload it straight up to the branch of your GitHub repo.

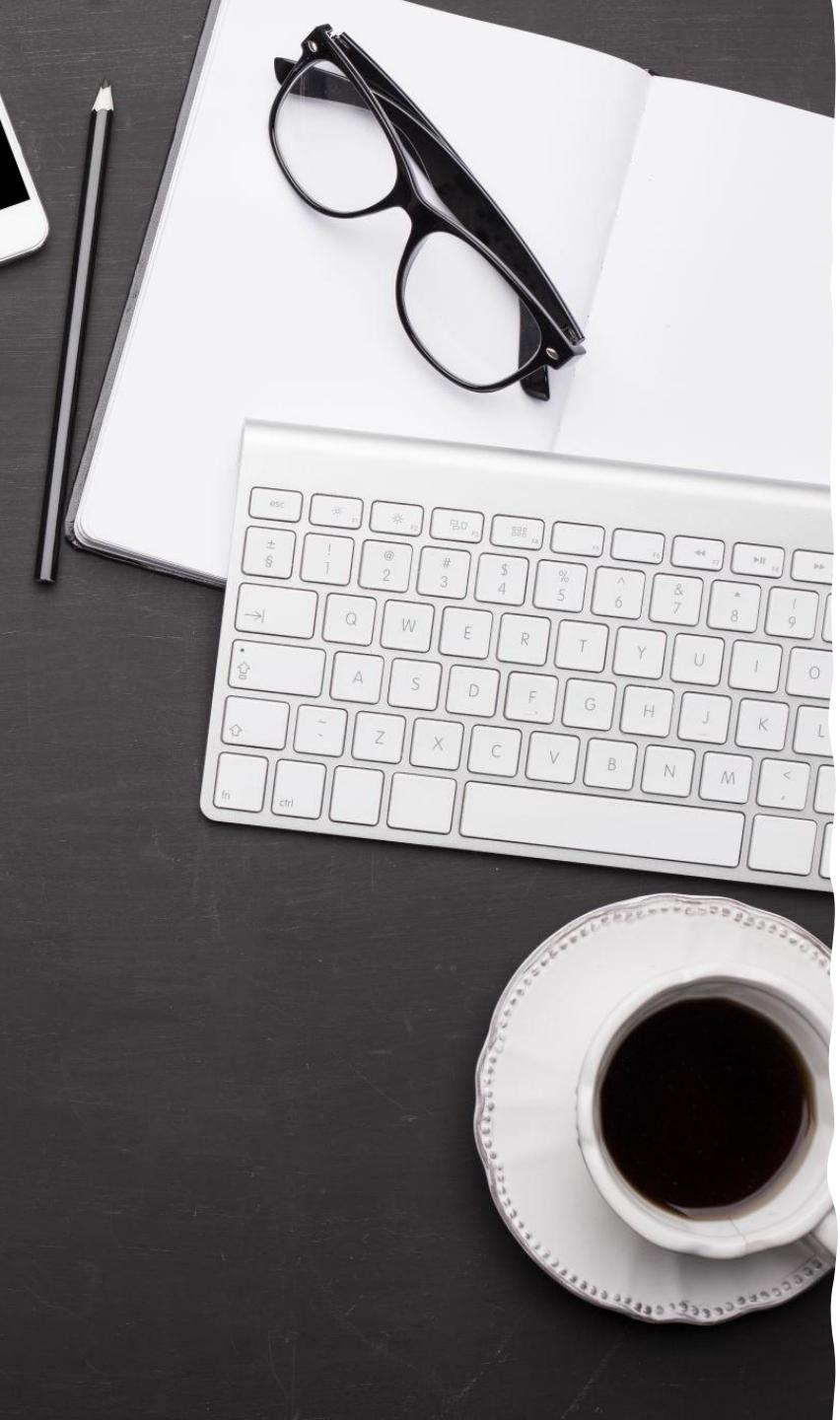


# Final Release

- You can visit the GitHub repo at the following link and check out the final release of the project:
- [https://github.com/alessandrobruno10/sentiment\\_analysis/](https://github.com/alessandrobruno10/sentiment_analysis/)

The screenshot shows the GitHub repository page for 'sentiment\_analysis'. The repository is public and has 1 branch and 0 tags. The main branch is selected. There are 6 commits from user 'alessandrobruno10' made 1 minute ago. The commits are: 'Create LICENSE', 'LICENSE', 'Update README.md', 'acknowledgment', 'Create acknowledgment', and 'sentiment\_analysis.py' (Add files via upload). The 'Code' button is highlighted.

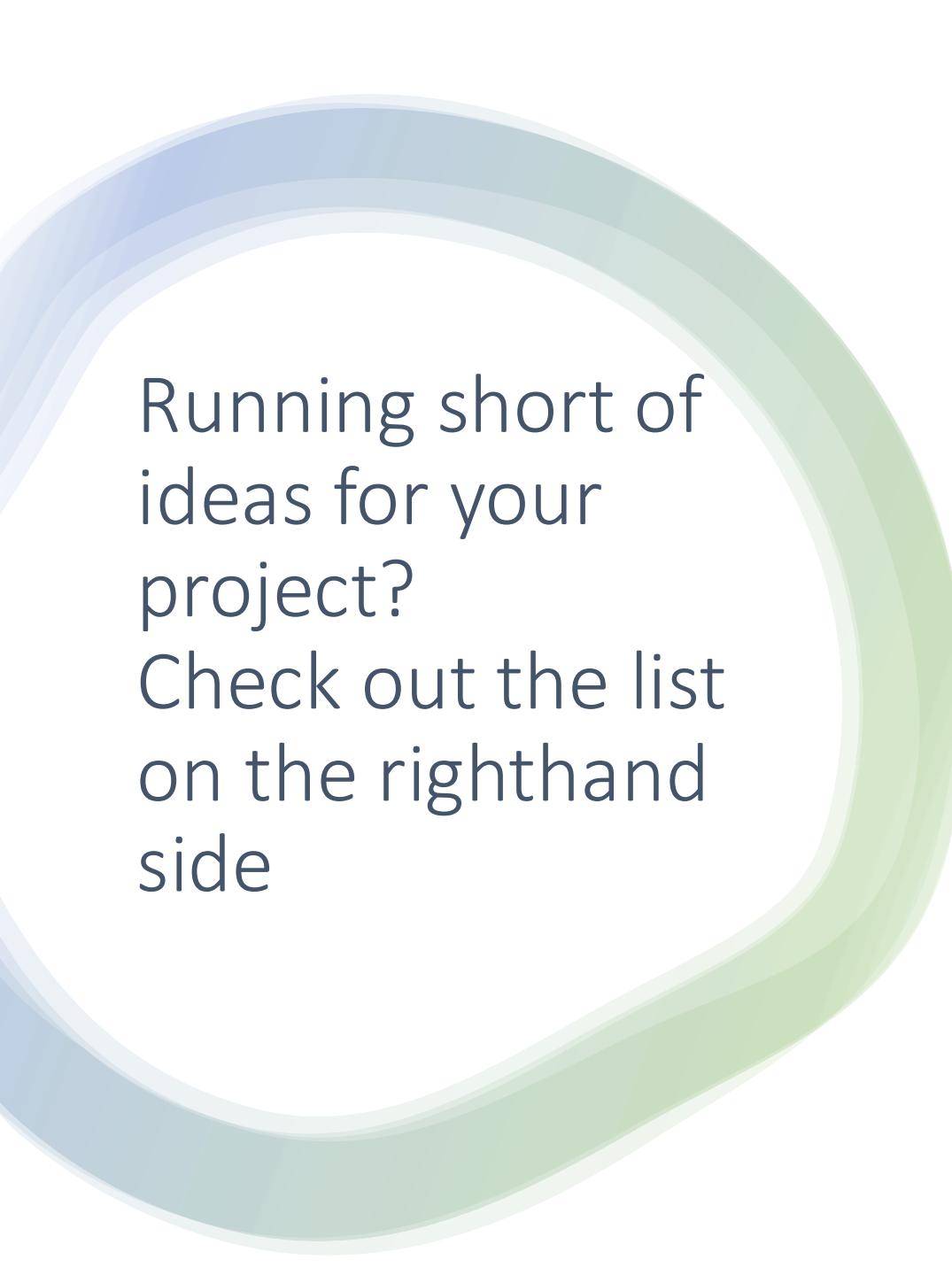
File / Action	Description	Time
LICENSE	Create LICENSE	1 minute ago
README.md	Update README.md	19 minutes ago
acknowledgment	Create acknowledgment	3 minutes ago
sentiment_analysis.py	Add files via upload	3 minutes ago



# What else?

---

- What should I use to present my Python project during the exam session?
  - Just bring your laptop with your project and configurations set and done for a quick demo.
  - Get your GitHub repo up-to-date
  - Each team member will be given 10 minutes to talk about the solution. No need for slide decks or pptx files.
  - You can describe the project by showing how it works locally on your laptop and how you wrapped it up on GitHub,



Running short of ideas for your project?  
Check out the list on the righthand side

- Customer Churn Prediction
- Market Trends Analysis via regression
- Keywords Extraction from Documents
- Sentiment Analysis from Documents
- Topic Modeling
- Opinion Mining
- Recommendation Systems
  - Movies
  - Music genres
  - Items
- Virtual Chatbots
  - Face Detection
  - Object Detection
  - Saliency Detection



# Read it carefully!

- With coding it is easy to overcomplicate a project development especially when we add new interesting tasks to the orginal idea.
- There are way so many brilliant ideas that are worth coding.
- HOWEVER, My advice is: "Don't overdo it!"
- Pick up a project with simple tasks and try following the documentation you find over the Internet (you can also ask ChatGPT) and check whether things go smooth.
- Stick to a bottom-up approach, starting off with simple tasks.
- Then, try building up on what you already achieved.

# Final thoughts

- If you feel like you are not sure about how to proceed or start off with your project, drop me a message at [alessandro.bruno@iulm.it](mailto:alessandro.bruno@iulm.it)
- Brainstorming sessions can help to unleash our creativity or streamline projects and ideas.
- Remember: **we are here to learn!**

