

# Data Science Professional Practicum (DSCI 560)

## Laboratory Assignment 2

**Team Name:** Guardians of the Algorithm

**Team members:**

Jaival Chintankumar Upadhyay – 8278442205

Pratham Solanki - 3242692358

Mayank Patil – 9101437684

**Github link:** <https://github.com/jaivalupadhyay/dsci-560/tree/main>

For our group project, we have selected three key domains: **NLP Forums**, **Health and Lifestyle**, and **Educational Course Materials**. These domains provide diverse and rich datasets that align with our objectives.

### 1. NLP Forums

- **Dataset Source:** [Hugging Face Discussions](#)
- **Description:** Contains threads on fine-tuning large language models with PDF documents.
- **Sample Excerpt:** *Screenshots attached below*

### 2. Health and Lifestyle

- **Dataset Source:** [Smokers Health Data on Kaggle](#)
- **Description:** Includes data on individuals' smoking habits and various health indicators.
- **Sample Excerpt:** *Screenshots attached below*

### 3. Educational Course Materials

- **Dataset Source:** [CMU Course Lectures](#)
- **Description:** Provides information on machine learning course lectures, topics, and resources.
- **Sample Excerpt:** *Screenshots attached below*

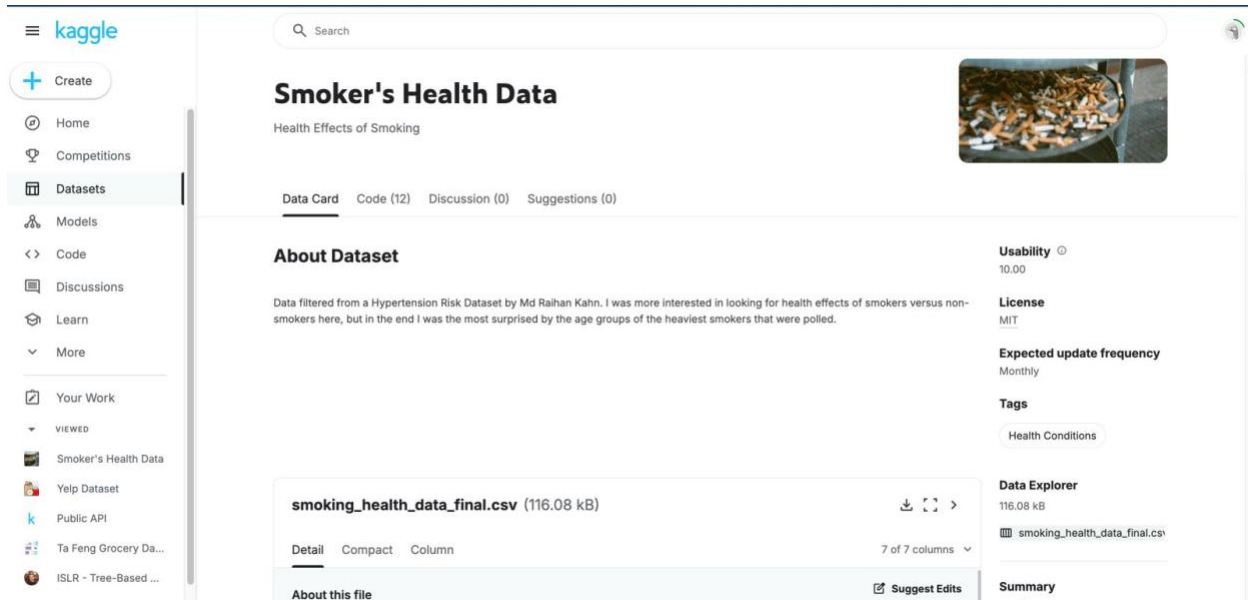
**Reasoning Behind Topic Choice:** These domains offer a comprehensive mix of technical discussions, real-world health data, and academic resources. This combination allows us to explore machine learning applications, analyze health-related patterns, and understand educational methodologies, providing a well-rounded foundation for our project.

i) CSV or Excel

### Snapshot of Code:

```
216     def Extract_CSV_data():
217
218         output_path = 'extracted_csv.csv'
219
220         kaggle_username = 'prathamsolanki1202'
221         kaggle_key = 'b8fb55cfc4e620e3fd0e33da4b374d10'
222
223
224         kaggle_json_path = 'kaggle.json'
225         with open(kaggle_json_path, 'w') as f:
226             f.write(f'{{"username":"{kaggle_username}","key":"{kaggle_key}"}}')
227
228
229         os.environ['KAGGLE_CONFIG_DIR'] = os.getcwd()
230
231
232         dataset_identifier = 'jaceprater/smokers-health-data'
233
234         download_dir = 'temp_download'
235
236         os.makedirs(download_dir, exist_ok=True)
237
238         os.system(f'kaggle datasets download -d {dataset_identifier} -p {download_dir}')
239
240         zip_file = [f for f in os.listdir(download_dir) if f.endswith('.zip')][0]
241         zip_file_path = os.path.join(download_dir, zip_file)
242
243         with zipfile.ZipFile(zip_file_path, 'r') as zip_ref:
244             zip_ref.extractall(download_dir)
245
246         csv_file = [f for f in os.listdir(download_dir) if f.endswith('.csv')][0]
247         csv_file_path = os.path.join(download_dir, csv_file)
248
249         os.rename(csv_file_path, output_path)
250
251         os.remove(zip_file_path)
252         os.rmdir(download_dir)
253         os.remove(kaggle_json_path)
```

## SOURCE PAGE



**Smoker's Health Data**  
Health Effects of Smoking

**About Dataset**

Data filtered from a Hypertension Risk Dataset by Md Raihan Kahn. I was more interested in looking for health effects of smokers versus non-smokers here, but in the end I was the most surprised by the age groups of the heaviest smokers that were polled.

**smoking\_health\_data\_final.csv** (116.08 kB)

**Detail** Compact Column 7 of 7 columns

**Usability** 10.00

**License** MIT

**Expected update frequency** Monthly

**Tags** Health Conditions

**Data Explorer** 116.08 kB

**Summary**

**Description:** The `Extract_CSV_data` function automates the process of downloading the "Smokers Health Data" dataset from Kaggle by using provided API credentials. It begins by creating a `kaggle.json` file with the necessary authentication details and sets the environment for the Kaggle API. The function then downloads the specified dataset into a temporary directory, extracts the CSV file from the downloaded ZIP archive, and renames it to `extracted_csv.csv` for easier access. After successfully saving the CSV file, the function cleans up by removing the temporary files and directories. Finally, it performs basic data operations by loading the CSV into a pandas DataFrame and printing out key information such as column names, dataset shape, null values, and the last few rows, providing an initial overview of the dataset for further analysis.

## OUTPUT:

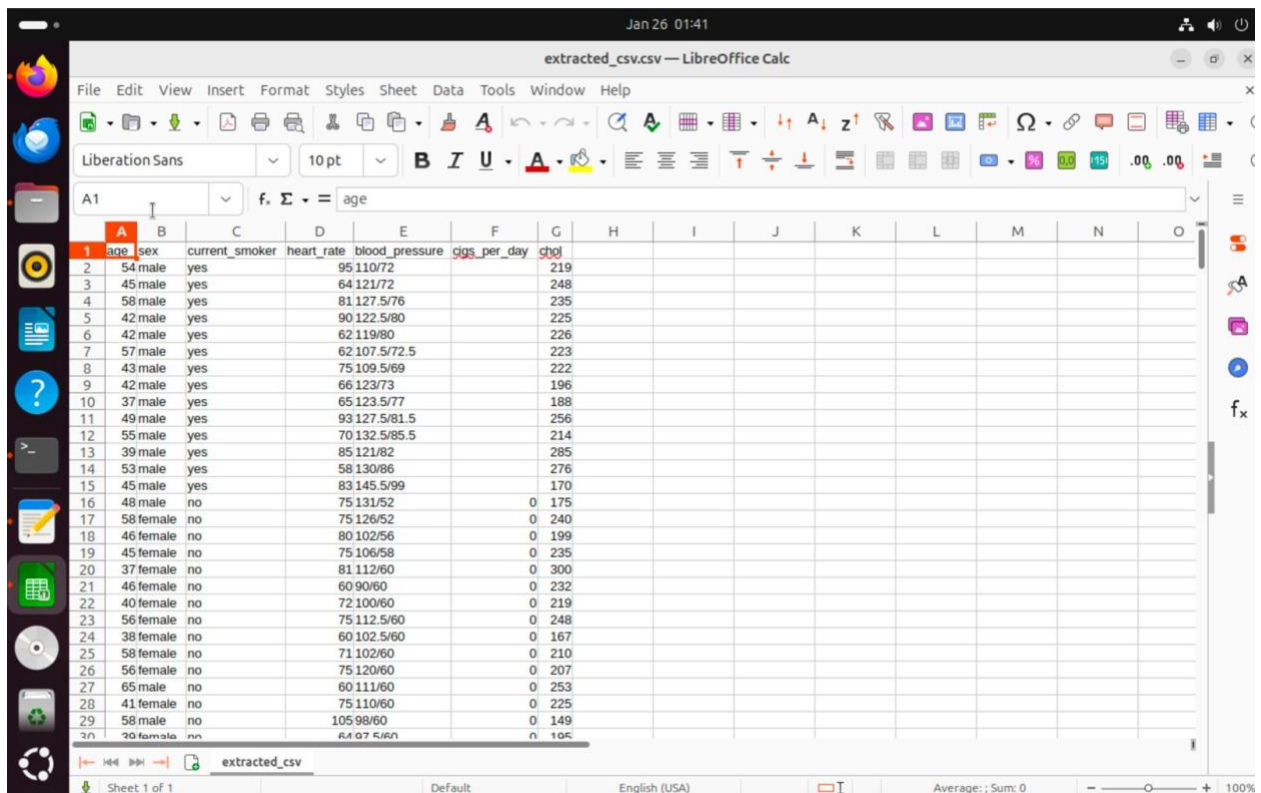
```
Dataset downloaded and saved to: /home/prathamuser/Desktop/prathamsolanki_324269
2358/data/processed_data/extracted_csv.csv
Columns Index(['age', 'sex', 'current_smoker', 'heart_rate', 'blood_pressure',
               'cigs_per_day', 'chol'],
              dtype='object')
Shape of dataset (3900, 7)
Null values:
  age      0
  sex      0
  current_smoker  0
  heart_rate  0
  blood_pressure  0
  cigs_per_day  14
  chol        7
dtype: int64
```

```

age sex current_smoker heart_rate blood_pressure cigs_per_day chol
3895 37 male yes 88 122.5/82.5 60.0 254.0
3896 49 male yes 70 123/75 60.0 213.0
3897 56 male yes 70 125/79 60.0 246.0
3898 50 male yes 85 134/95 60.0 340.0
3899 40 male yes 98 132/86 70.0 210.0
/home/prathamuser/Desktop/prathamsolanki_3242692358/scripts/data_exploration.py:

```

## OUTPUT CSV FILE



	age	sex	current_smoker	heart_rate	blood_pressure	cigs_per_day	chol
1	54	male	yes	95	110/72		219
2	45	male	yes	64	121/72		248
3	58	male	yes	81	127.5/76		235
4	42	male	yes	90	122.5/80		225
5	42	male	yes	62	119/80		226
6	57	male	yes	62	107.5/72.5		223
7	43	male	yes	75	109.5/69		222
8	42	male	yes	66	123/73		196
9	37	male	yes	65	123.5/77		188
10	49	male	yes	93	127.5/81.5		256
11	55	male	yes	70	132.5/85.5		214
12	39	male	yes	85	121/82		285
13	53	male	yes	58	130/86		276
14	45	male	yes	83	145.5/99		170
15	48	male	no	75	131/52	0	175
16	58	female	no	75	126/52	0	240
17	46	female	no	80	102/56	0	199
18	45	female	no	75	106/58	0	235
19	37	female	no	81	112/60	0	300
20	46	female	no	60	90/60	0	232
21	40	female	no	72	100/60	0	219
22	56	female	no	75	112.5/60	0	248
23	38	female	no	60	102.5/60	0	167
24	58	female	no	71	102/60	0	210
25	56	female	no	75	120/60	0	207
26	65	male	no	60	111/60	0	253
27	41	female	no	75	110/60	0	225
28	58	male	no	105	98/60	0	149
29	70	female	no	64	07.5/60	0	195

In the above images, we see that the data has successfully been extracted and is stored on our desktop

## ii) ASCII Texts like Forum Postings and HTML

### Snapshot of Code:

```
31
32 > def fetch_html(url,driver_path,output_path,service,driver): ...
50     return None
51
52 > def read_html(output_path): ...
60     return html_parsed
61
62 > def extract_data(html_parsed): ...
164     return question_dict,all_responses_list
165
166 > def write_to_csv(question_dict,all_responses_list): ...
178     return None
179
180 > def basic_operations(csv_output_path): ...
187     print(df.head())
188
189
190
191     #Get html
192     fetch_html(url,driver_path,output_file_path,service,driver)
193     print("HTML fetched successfully")
194
195     #parse
196     html_parsed = read_html(output_file_path)
197     print("Data Parsed successfully")
198
199     #Extract elements
200     question_dict,all_responses_list = extract_data(html_parsed)
201     print("Data extracted successfully")
202
203
204     #write to csv
205     write_to_csv(question_dict,all_responses_list)
206     print("Data written to CSV successfully")
207
208     #basic operations
209     basic_operations(csv_output_path)
```

**Description:**

First, we are fetching the HTML source code using Selenium from the specified URL: <https://discuss.huggingface.co/t/fine-tune-llms-on-pdf-documents/71374>. This allows us to gather the complete HTML structure of the webpage for further processing.

After fetching and parsing, the HTML content is saved into a file named `html_parsed.html` for record-keeping and easier access.

Next, we read the `html_parsed.html` file to extract data. Using BeautifulSoup, we extract the following:

- Title of the forum post.
- Page statistics, such as the number of 'views', 'likes', 'links', and 'users'.
- For each post in the forum:
  - `time_stamp`
  - name/author of the post
  - post content
  - likes received on the post

All the extracted data is then written into a structured CSV file for further use and analysis.

Finally, we read the generated CSV file using Pandas and perform some basic operations, such as inspecting the data, print the column names, shape, null value counts, and the first few rows of the dataset.

**Source Website:** "<https://discuss.huggingface.co/t/fine-tune-llms-on-pdf-documents/71374>"





## OUTPUT

```
(my_virtual_env) prathamuser@prathamserver:~/Desktop/prathamsolanki_3242692358/scripts$ python3 data_exploration.py
HTML fetched successfully
Data Parsed successfully
Data extracted successfully
Data written to CSV successfully
Columns Index(['time_stamp', 'title', 'name', 'post', 'views', 'likes', 'links',
              'users'],
              dtype='object')
Shape of dataset (20, 8)
Null values:
time_stamp    0
title         0
name          0
post          0
views         0
likes         0
links        19
users        19
dtype: int64
```

	time_stamp	title	name	\
0	Jan 31, 2024 5:59 pm	Fine tune LLMs on PDF Documents	imvbhuvan	
1	Feb 2, 2024 7:29 pm	Fine tune LLMs on PDF Documents	juhoinkinen	
2	Feb 3, 2024 4:37 pm	Fine tune LLMs on PDF Documents	imvbhuvan	
3	Apr 3, 2024 2:27 pm	Fine tune LLMs on PDF Documents	sabber	
4	Apr 4, 2024 4:57 pm	Fine tune LLMs on PDF Documents	sabber	

	post	views	likes	links	\
0	['We are currently seeking assistance in fine-...	20.1k	16	13.0	
1	['I assume you want to extract raw text from t...	20.1k	3	NaN	
2	['Thank you for your response.', 'We aim to cu...	20.1k	0	NaN	
3	['Hello there@imvbhuvan, Were you able to fine...	20.1k	0	NaN	
4	['Thank you very much for the reply. I will em...	20.1k	0	NaN	



**Jan 24 01:33**

**forum\_extracted\_data.csv — LibreOffice Calc**

File Edit View Insert Format Styles Sheet Data Tools Window Help

Liberation Sans 10 pt B I U A Z Ω ↻ 🔍 📄 🗑️ 💬 ⌨️

A1	B	C	D
f. Σ = time_stamp			
A	B	C	D
time_stamp	title	name	post
Jan 31, 2024 5:59 pm	Fine tune LLMs on PDF Documents	mrbuhyan	[We are currently seeking assistance in fine-tuning the Mistral model using approximately 48 PDF documents
Feb 2, 2024 7:29 pm	Fine tune LLMs on PDF Documents	mrbuhyan	[I assume you want to extract raw text from the pdfs? In what kind of form you want the data for fine-tune t
Feb 23, 2024 4:37 am	Fine tune LLMs on PDF Documents	mrbuhyan	[Thank you for your response," We aim to customize the LLMs for a specific domain by fine-tuning them usin
Apr 3, 2024 2:27 pm	Fine tune LLMs on PDF Documents	sabber	[Hello there@mrbuhyan, Were you able to fine tune model using pdf (I assume unstructured data) ? I am als
Apr 4, 2024 4:57 pm	Fine tune LLMs on PDF Documents	sabber	[Thank you very much for the reply. I will email you shortly!]
Apr 10, 2024 5:37 pm	Fine tune LLMs on PDF Documents	sagekhan	[Hi, Im also trying to fine tune mistral on some documents. Actually its text file extracted from 1-5 page pdf wh
May 1, 2024 8:46 am	Fine tune LLMs on PDF Documents	omar8	[did you find a way to do it"]
May 1, 2024 1:02 pm	Fine tune LLMs on PDF Documents	njes	[There are 2 options here I'd say,' either you fine-tune a text-only LLM (like Mistral, LLama, etc.) on the OCR
May 1, 2024 3:16 pm	Fine tune LLMs on PDF Documents	sagekhan	[Make your own dataset and train on it] facing some issues with The excessive replies and stuff. It tends i
May 29, 2024 1:21 pm	Fine tune LLMs on PDF Documents	AnasA	[Thank you for this information!', In general, when you talk about PDF to JSON, in the notebook we find ima
Sep 10, 2024 2:23 pm	Fine tune LLMs on PDF Documents	Pendrakatara12	[Hi @mrbuhyan were able to succeed and perform well inferences on the same task. I am also researching for
Oct 23, 2024 8:06 am	Fine tune LLMs on PDF Documents	Triptigarg2711	[Hi Sabber, I also face similar problem, I have some 100s of pdfs on which I want to train/fine tune lm. The thi
Oct 23, 2024 9:29 am	Fine tune LLMs on PDF Documents	Triptigarg2711	[Hi, I also face similar problem, I have some 100s of pdfs on which I want to train/fine tune lm. The thing is, w
Nov 5, 2024 9:16 am	Fine tune LLMs on PDF Documents	mrbuhyan	[You can use continued pre-training]
Nov 5, 2024 12:00 pm	Fine tune LLMs on PDF Documents	triprpt27	[Thanks. But can u please elaborate the process of how to use pdf data. The problem is, I have tables in pdfs
Nov 17, 2024 8:06 am	Fine tune LLMs on PDF Documents	kobg	[Sounds to me like there is a misunderstanding going on." When you say "fine tune on PDF documents", it sc
Nov 13, 2024 8:31 am	Fine tune LLMs on PDF Documents	mrbuhyan	[Yup, thank you for the points Can I connect with you to discuss further on an interesting problem we're solvi
Nov 15, 2024 8:47 am	Fine tune LLMs on PDF Documents	triprpt27	[, Sounds to me like there is a misunderstanding going on." When you say "fine tune on PDF documents", it s
Nov 15, 2024 8:48 am	Fine tune LLMs on PDF Documents	triprpt27	[Hi, Can you please elaborate the solution.' Thanks in advance.]
Nov 17, 2024 11:13 am	Fine tune LLMs on PDF Documents	Chandrashekar	[Hi @mrbuhyan have fine tuned pdf to text with close to 98% accuracy.]

### iii. PDF and Word Documents that require conversion and OCR

#### Code Snippet:

```
272 ✓ def extract_course_data():
273     logging.basicConfig(level=logging.INFO)
274
275 > class Path: ...
281     os.makedirs(PDF_FILE_DIR, exist_ok=True)
282
283     class Settings:
284         SITE_URL: str = "https://www.cs.cmu.edu/~ninamf/courses/601sp15/lectures.shtml"
285         BASE_URL: str = "https://www.cs.cmu.edu/~ninamf/courses/601sp15/"
286
287     class Topic(BaseModel):
288         name: str | None
289
290     class ReadingUsefulLinks(BaseModel):
291         name: str
292         link: str | None
293
294 > class CourseItem(BaseModel): ...
301     arbitrary_types_allowed = True
302
303     def is_relative_url(link):
304         parsed_url = urlsplit(link)
305         return not parsed_url.scheme and not parsed_url.netloc
306
307 > def get_lecture_info(columns): ...
311     return lecture, topics_list
312
313 ✓ def get_handouts(columns):
314     slides_video_links = columns[4].find_all("a")
315     handouts = [
316         ReadingUsefulLinks(
317             name=link.text.strip(),
318             link=(urljoin(Settings.BASE_URL, link["href"]) if is_relative_url(link["href"]) else link["href"]),
319         )
320     ]
```

**Description:** The code automates the collection and processing of course-related data from the webpage located at <https://www.cs.cmu.edu/~ninamf/courses/601sp15/lectures.shtml>. The operation begins with setting up the environment and creating essential classes to manage data and file paths. The Path class ensures that directories are created correctly for storing the extracted data and PDFs, while the Settings class defines constants like the base URL and the target page for scraping. By leveraging pydantic, a data structure is established for topics, readings, handouts, and lectures to maintain consistency and clarity.

The script includes utility functions for managing the content of the webpage. First, it detects and resolves relative URLs, ensuring that all links are correctly connected. The scrape\_course\_page function fetches the HTML content from the designated webpage using requests and parses it with BeautifulSoup. It collects relevant information from the schedule table, including lecture titles, topics, readings, and handouts, while disregarding

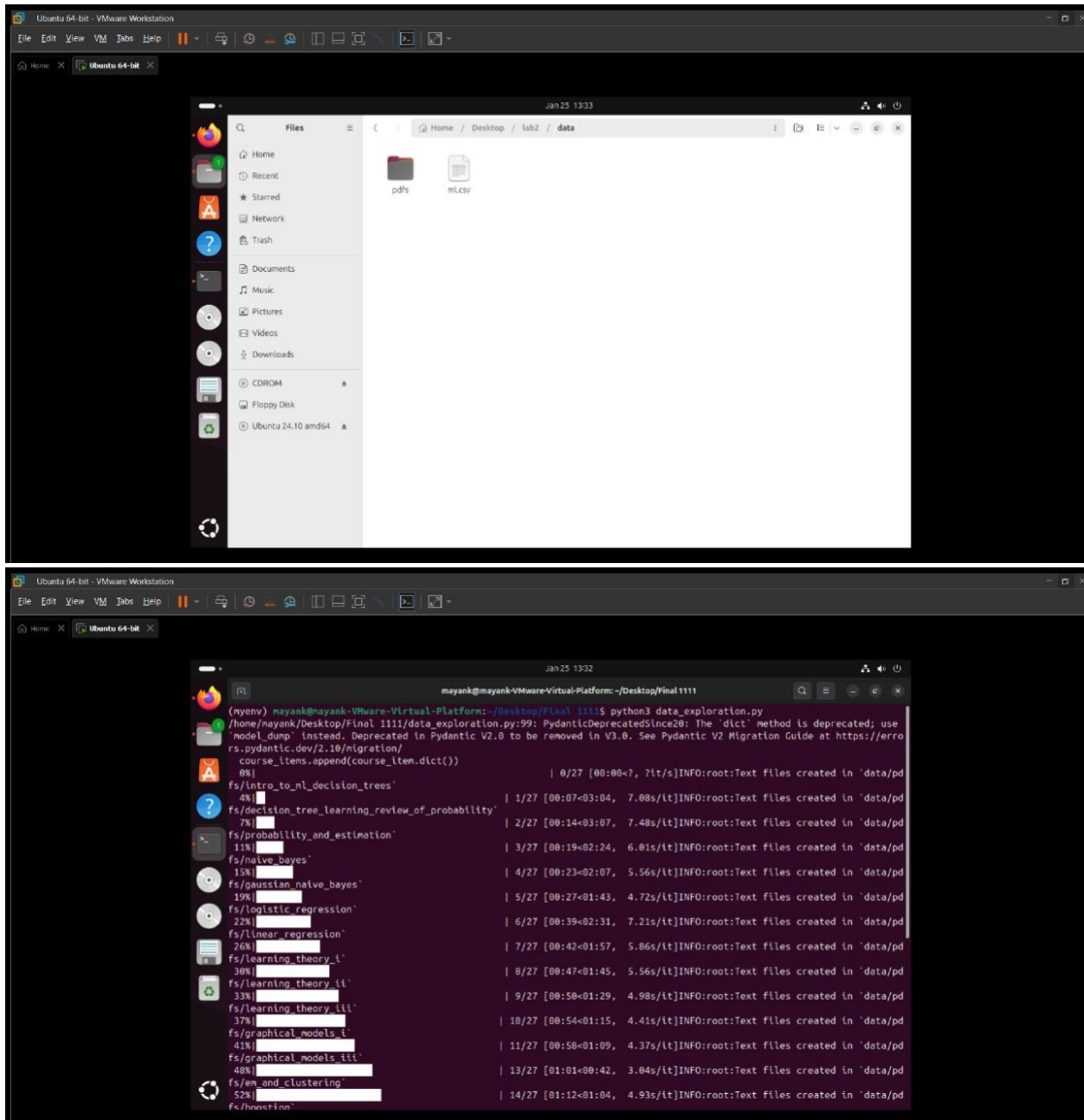
unnecessary rows. For each lecture, structured data is created using the `CourseItem` class, guaranteeing that all fields are well-organized.

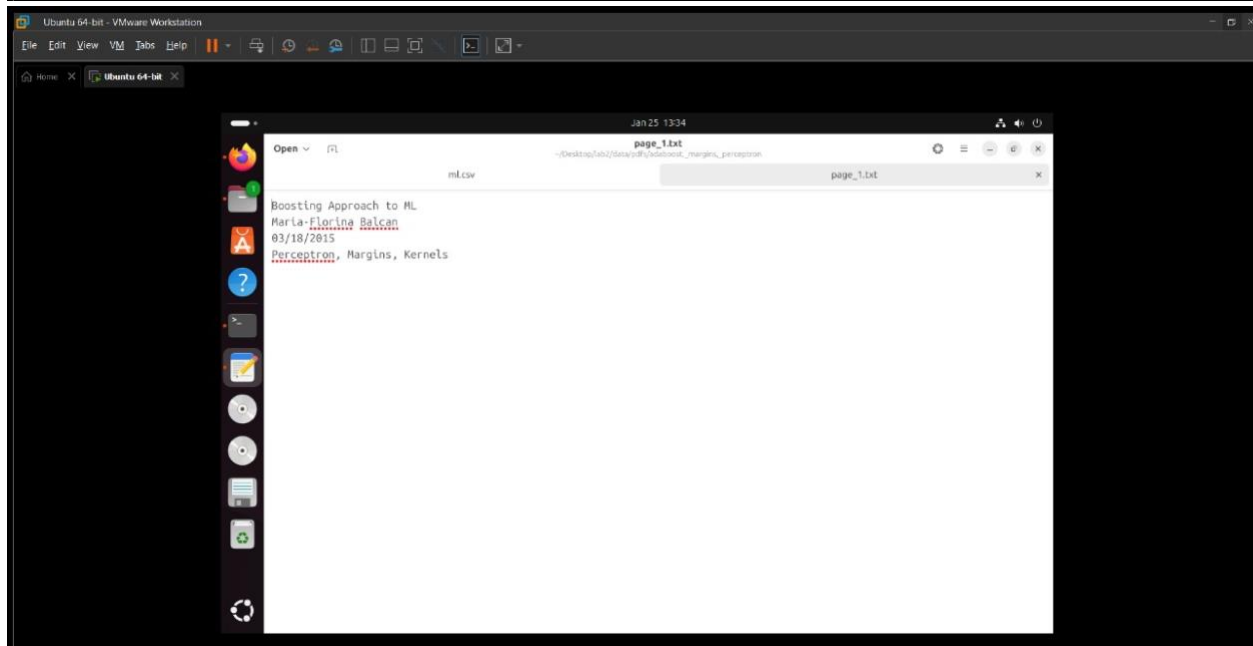
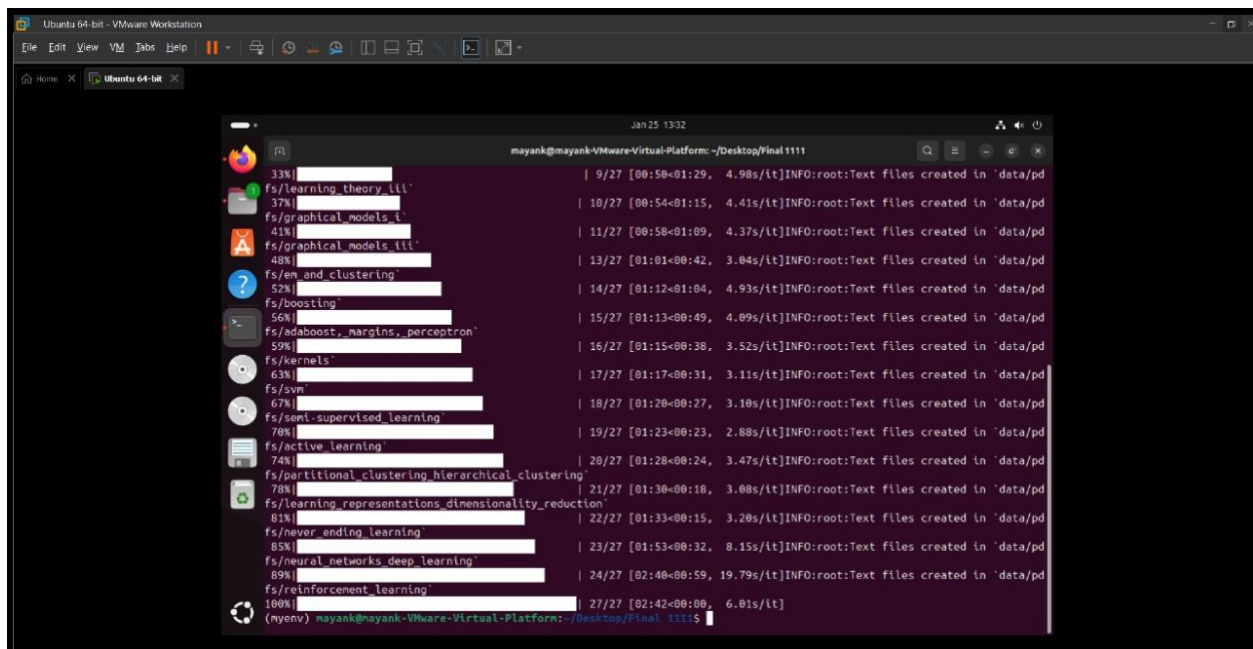
Once the lecture data is gathered, it is stored in a CSV file (`ml.csv`) for future utilization. The script further processes this data to handle lecture handouts, particularly PDFs. The `read_pdf_from_url` function retrieves PDFs from the given URLs, extracts their text with `fitz`, and saves the text as `.txt` files, one for each page. These files are organized in a structured directory hierarchy based on lecture names for easy navigation.

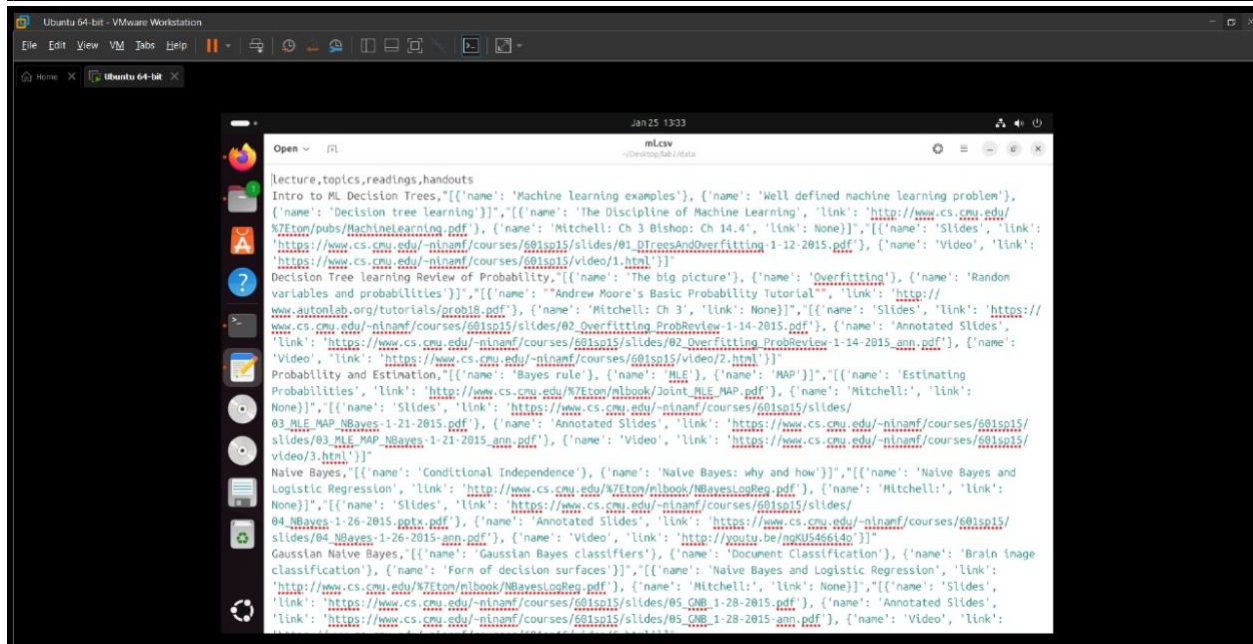
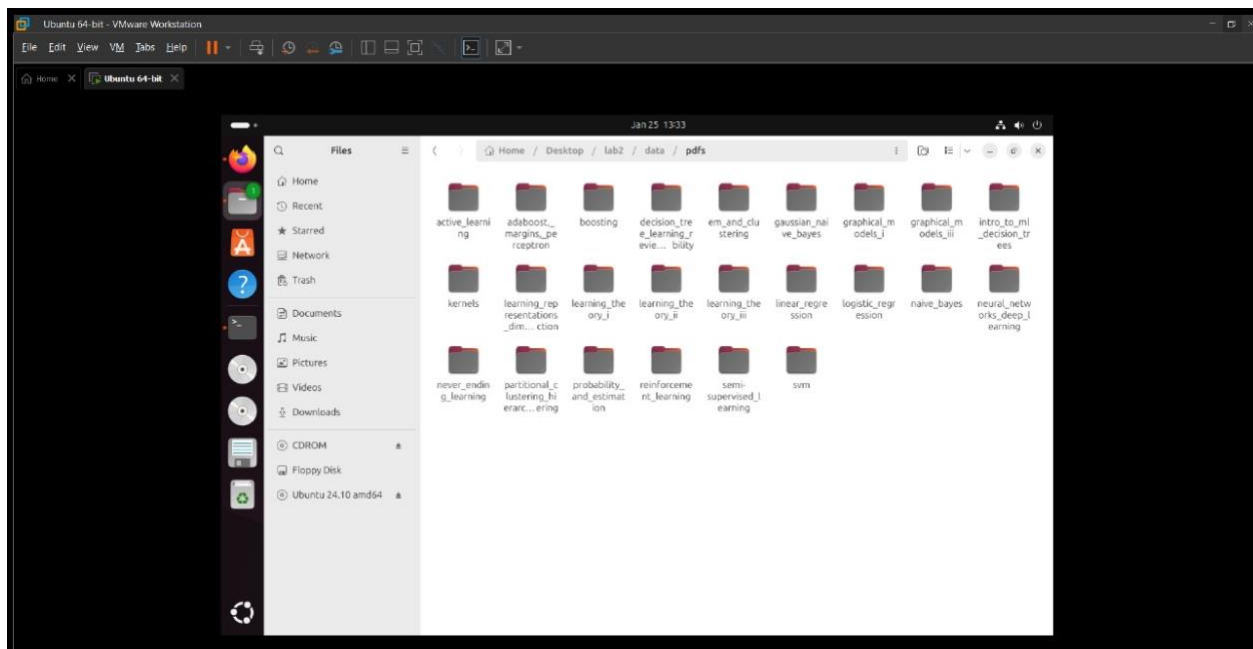
The final step involves reviewing the saved data to process all PDFs labeled as "Slides" in the handouts. A progress bar from `tqdm` provides visual updates throughout this process, enhancing the user experience. Error handling is incorporated into the script to ensure resilience against unexpected problems, such as network disruptions or missing elements on the webpage.

In summary, this code streamlines the process of scraping course information, formatting it as structured CSV files, and extracting text from lecture PDFs for further analysis. The approach guarantees clarity, maintainability, and usability, making it highly suitable for academic data management tasks.

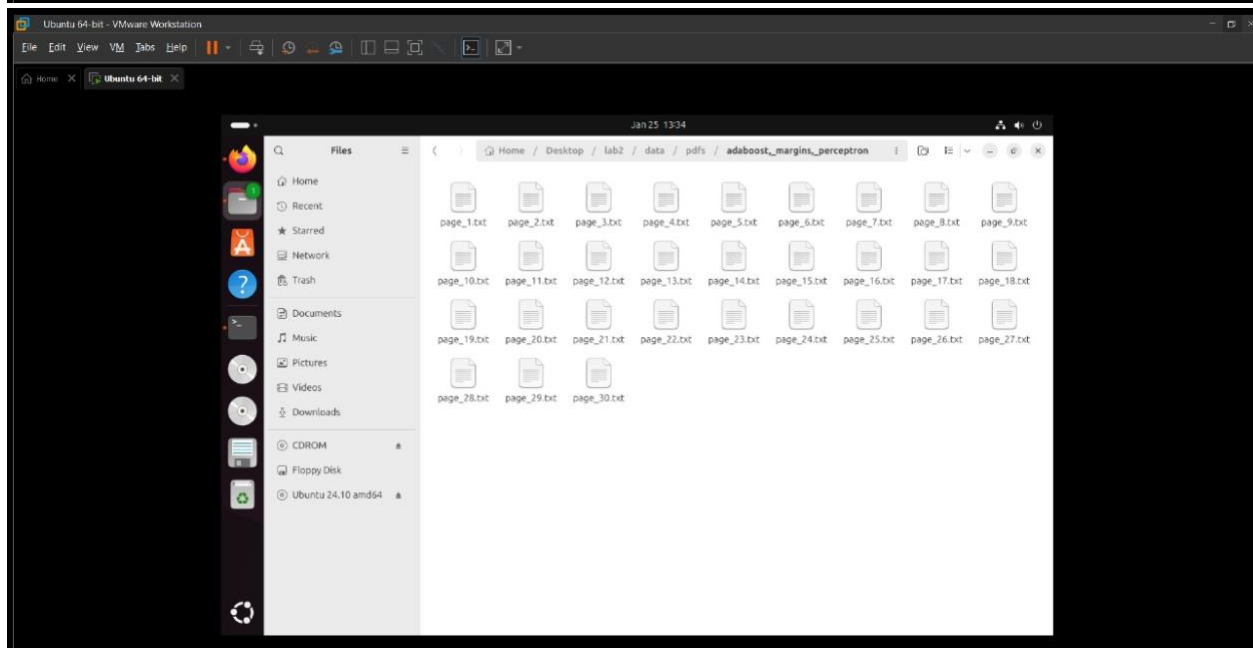
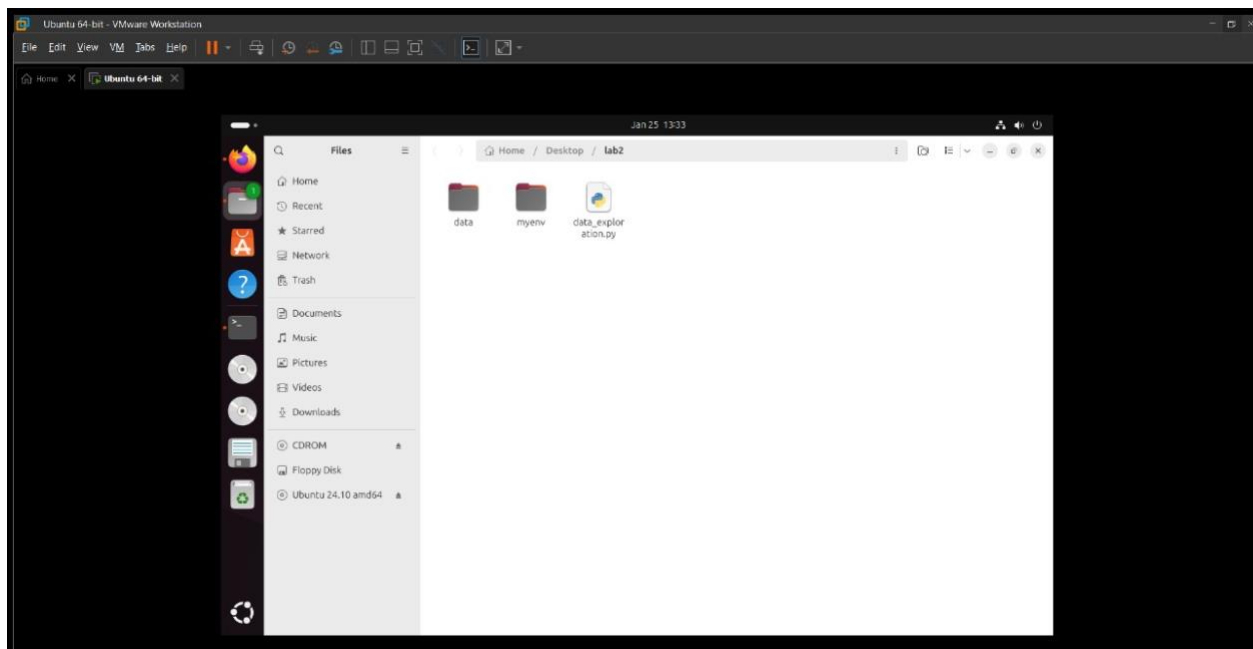
## OUTPUT:



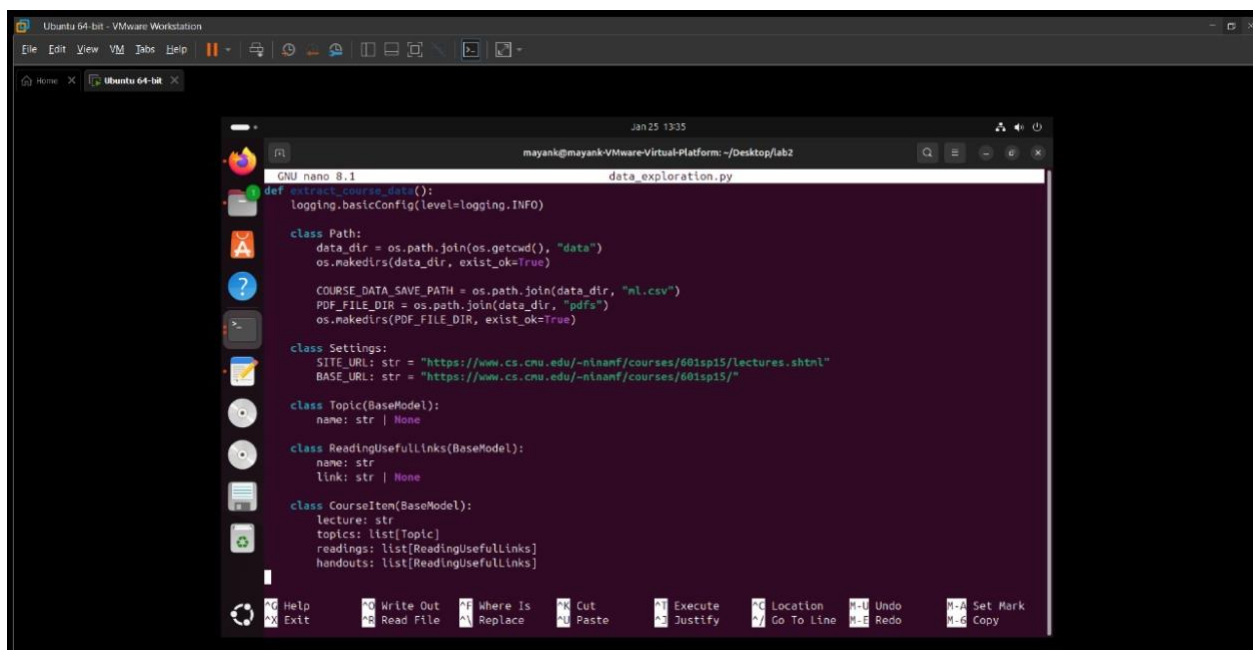
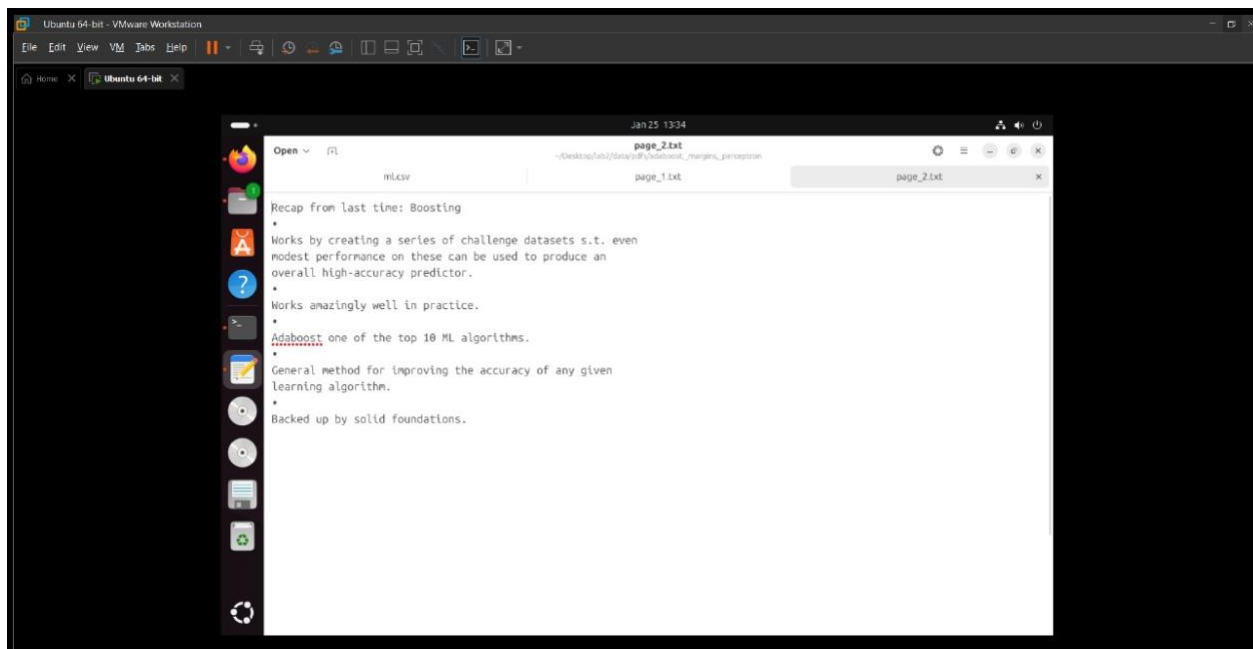












**Question:** In the report, describe what the script does (conversion tasks and tools to keep only the relevant data) to create a clean single dataset. While there are a lot of attempts to build realistic chatbots, most people would rather speak to a real person because their capabilities are very limited. Describe what might be missing in these existing chatbots. Discuss how your dataset might improve the overall performance and correctness.

**Answer:** Many chatbots still miss the mark, making people prefer talking to real humans. They often struggle with complex questions, use outdated information, and give generic answers. Our project fixes these issues by using recent, specialized data focused on Machine Learning applications instead of general knowledge. By integrating up-to-date discussions from NLP forums, detailed health data, and educational materials, our chatbot can better understand and accurately respond to technical queries. This makes our chatbot more reliable and personalized, offering users more relevant and trustworthy interactions in machine learning and healthcare.