Lab 8: File processing

Labs are graded for participation rather than correctness. Keep all your lab code in your course GitHub repo to receive credit for your work. We'll be looking to see that you have at least partially completed all problems in each lab.

If you finish the lab assignment early, you may get started on the homework.

Professional skills

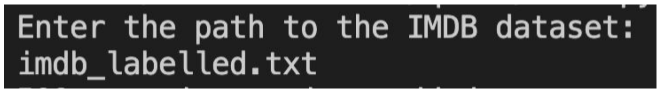
If you did not participate in lab 7's collaborative coding, consider collaborating for this lab. We'll set up some breakout rooms for you to self-select to work with other students. See lab 7 for setup instructions. You can also earn professional skills points by volunteering to walk through your lab solution toward the end of class.

The Assignment

You will write a program to process the IMDB dataset—a benchmark dataset widely used to develop and evaluate sentiment classifiers. The text file you will work with is a smaller version of the original dataset, containing 500 negative and 500 positive movie reviews. Download the two text files in the lecture-code repo > Starter code > Lab 8 folder.

Part 1 - reading the dataset

Prompt the user to enter the file path for the IMDB dataset on their machine then use that file path to read in the dataset. Here's an example of what that looks like in the terminal:



```
Enter the path to the IMDB dataset:
imdb_labelled.txt
```

Each line in the file has the following format, where "`\t`" represents a tab character and "`\n`" represents a new line character:
`<review text>\t<review label>\n`

E.g. line 4:
`Very little music or anything to speak of.\t0\n`
This line is labeled 0, meaning it is a negative review.

As you read the file, create two lists: one for negative reviews (labeled 0) and one for positive reviews (labeled 1). If you have processed the file correctly, each list will contain 500 reviews.

This part of your code must use try/except to handle the following error conditions:

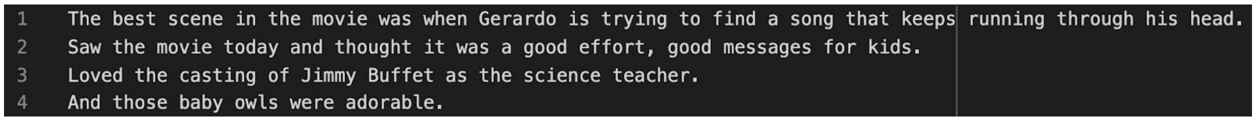
- If the path the user provides is invalid, print a message to inform the user e.g.“`File not found!`” Then, prompt them to keep trying until the file can be opened and processed.
- If the user provides a valid file path but it’s contents are not in the expected format (e.g. bad_file.txt), print the following message: “`Error processing the file.`” Then, prompt them to keep trying until the file can be opened and processed.

Part 2 - write new files

As soon as file processing (part 1) is complete, your program should write two files: `positive.txt` and `negative.txt`. `positive.txt` should contain all positive reviews and `negative.txt` should contain all negative reviews.

Each file should contain only review text (no review labels) and each review should be on a separate line.

The following screenshot shows the expected first few lines of `positive.txt`:



```
The best scene in the movie was when Gerardo is trying to find a song that keeps running through his head.
Saw the movie today and thought it was a good effort, good messages for kids.
Loved the casting of Jimmy Buffet as the science teacher.
And those baby owls were adorable.
```

Part 3 - display selected reviews

Allow the user to view individual reviews by entering commands in the following format:

- `p review_index`: display the positive review at the given index.
- `n review_index`: display the negative review at the given index.
- `q`: quit.

For example, if the user enters “`p 3`”, your program should print “`And those baby owls were adorable`”. Keep prompting for input until the user enters “`q`”, even if there is a problem with the input, as described below.

Your program should use try/except to handle the following error conditions and inform the user what went wrong:

- The user doesn’t provide a valid integer when requesting a review e.g. “`n stuff`”.
- The user provides a valid integer but it is out of range e.g. “`p 5000`”.

A note on testing: you are not expected to write unit tests for functions that read/write files.