**CSC386 Text Analysis - Finalize Scraping and Sentiment Analysis**

# **Purpose**:

This assignment is meant to enable you to use the data you have scraped to learn how to process it and extract a general sentiment analysis from it.

# **Background**:

In order to use Scrapy on your computer you will need the following installed on your device:

* Python
* pip
* Microsoft Visual Studio Code
* Internet Browser

You can install all these packages on your Berea laptop on your windows 10 operating system and your machine should be adequate.

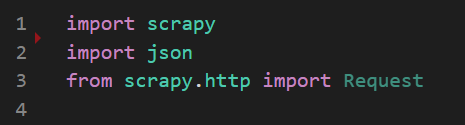
In order to do the Google Colaboratory part of this assignment you will need a google account.

**Assignment**

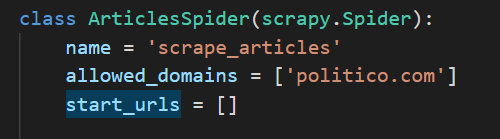
# Task 1: Complete scraping your news website

In the previous assignment you scrapped the title, link, intro, category and date and time of articles relating to one topic from a particular news site. In this task you will create another spider to go through each of those articles (which you’ve saved into a json in the previous assignment).

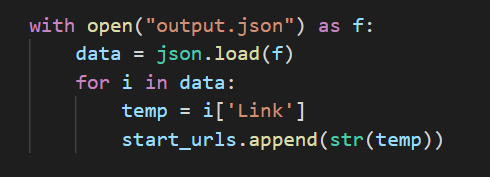
1. Move your output json file into your spiders folder.
2. Create a new spider. In your new spider python file import the following:



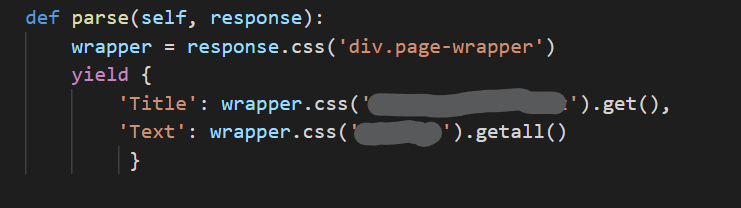
1. In your new class leave your start\_urls as an empty list.



1. To get the articles you must scrape from the websites found in the links from your json file. You can do this by appending all the links in your json file into your start\_urls list which you’ll use later. Your code might look something like the following:



1. Since you’ve appended each url into start\_url, scrapy will automatically iterate through them. The next step is to go to one of the articles and inspect the page to tell scrapy what information to parse (you can use xpath or css). It could look something like the following:



You should inspect the page to find the identifiers and class names you need. If you want to access just the text you could use ::text inside the quote like ‘a::text’ .

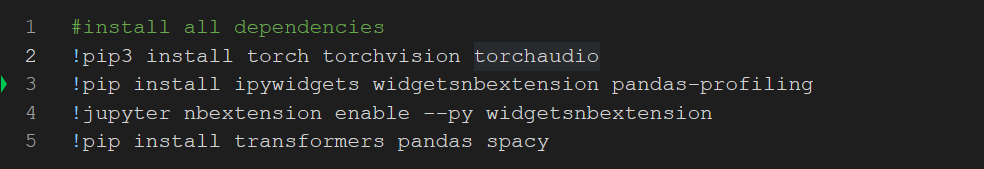
1. After writing your spider, go to your command prompt and from your project folder run scrapy crawl new\_spider\_name -o filename.csv (export a csv file!)

# Task 2: Transition to Colab

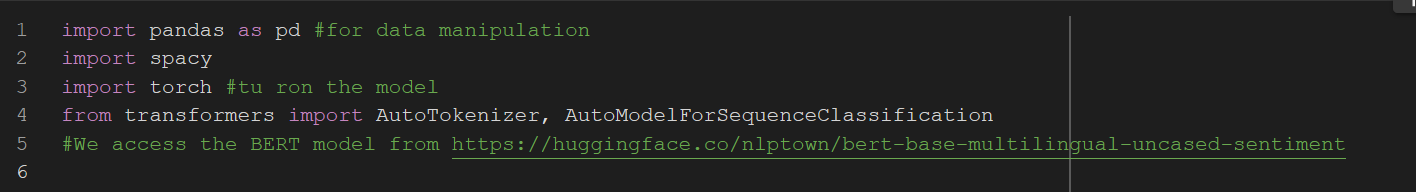
1. Go to your google drive
   1. if you have a folder that you use for this class create a new folder inside that
   2. If not, create a new folder anyway because you will need a separate space for your files.
   3. Name your folder “sentiment\_analysis” (no spaces and all lower case).
2. Upload your output csv file from the previous task into your new folder. You can just drag and drop the file or you can right click and select upload.
3. If you right click in your google folder, find Google Colaboratory under More and click it to create a new colab project. Change the name of the colab file to “C”.

# Task 3: Install necessary packages

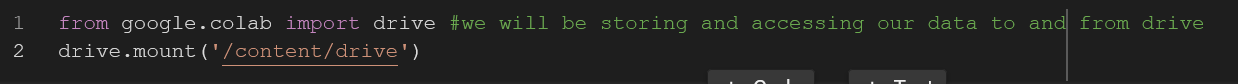
1. Now that you are in your colab file, install all the following dependencies. It might take a few minutes.



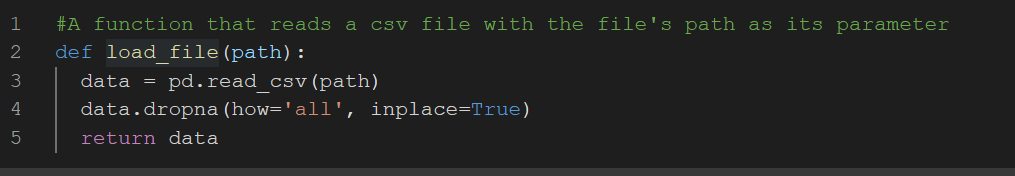
1. Import all of the following dependencies.



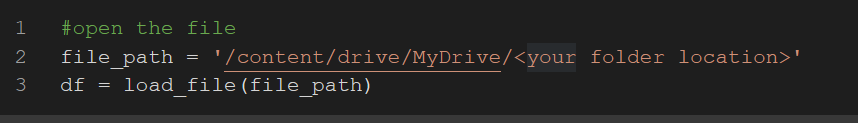
1. Mount your google drive. You will be asked for your gmail credentials in a pop, select allow and login if necessary.



1. Create a function that can read a csv file.

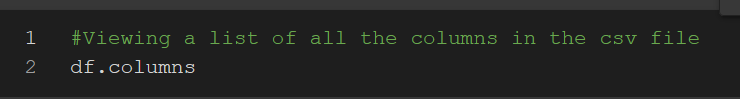


1. Use your function to load your csv file

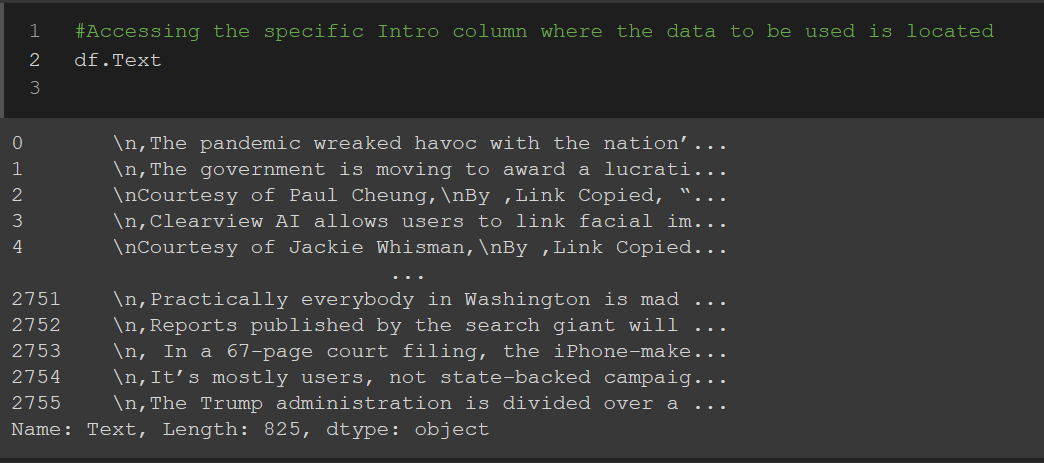


Replace the <your folder location> with the location of your file in your drive folder. Include folders and subfolders like you would in a command prompt.

1. To view the list of columns in your csv file, use the following command.



1. The column you will be using in this assignment is the one that contains the article text. You could run the following to get a list of text from the article returned.

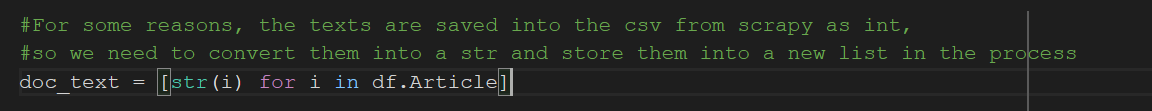


Do you notice something? There are new lines (/n) all over the data and not all the words are uniform, some are uppercase and others are lower case. We need to clean the data! You might remember doing this in NLTK, in this assignment we will use Spacy which is smarter.

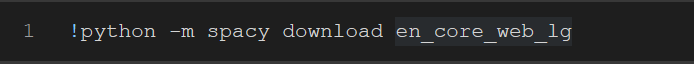
# Task 4: Clean your data using Spacy

The advantage of Spacy is that it has a much bigger list of tokens therefore if you take u.s. It is a token on spacy while it might not be with NTLK or other tools of text processing. With spacy the integrity of the text will most likely be maintained. In short it is the best tool for text processing at the moment.

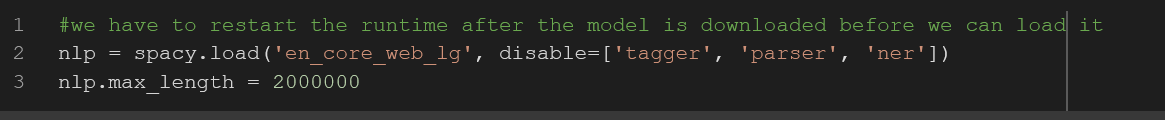
1. By default the imported csv file saves the data as a datatype that’s different from a string. Before we can process the text with spacey we have to convert them into a string as follows.



1. Download the spacy model that contains the list of tokens. There is a small version (en\_core\_web\_sm) but the larger one has more tokens and is more efficient because of that.

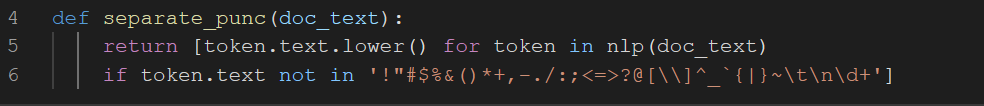


1. **Comment out the code from b (make sure you do this first)** andthen,go to the Runtime Menu select Restart and run all.
2. Load spacey with the model you just downloaded.



Since we only use it for tokenization, we disable the features such as parser, tagger and ner(named entity recognition) because these are RAM hungry features i.e. they may slow down our machine while not being used.

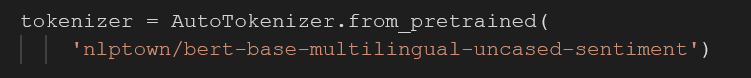
1. Next we will temporarily use spacy's tokenizer to clean the text and remove all non-required characters. Since we will be using [BERT](https://huggingface.co/nlptown/bert-base-multilingual-uncased-sentiment)  for sentiment analysis which has its own tokenizer, a conversion of the list of tokens into texts is necessary. We can do this as the following :



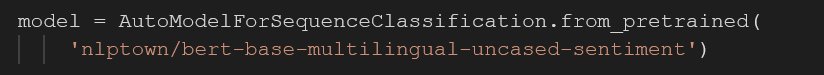
1. Next we convert the tokens into a full string for each of the text articles.



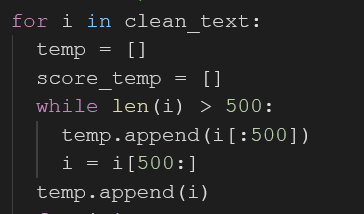
1. We need to load the tokenizer of this specific model.It should handle everything from tokenization to vectorization.



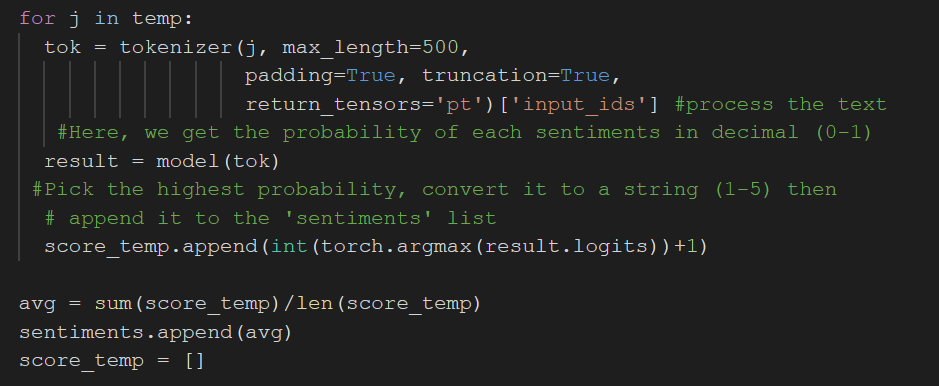
1. Then we can load the actual model itself.



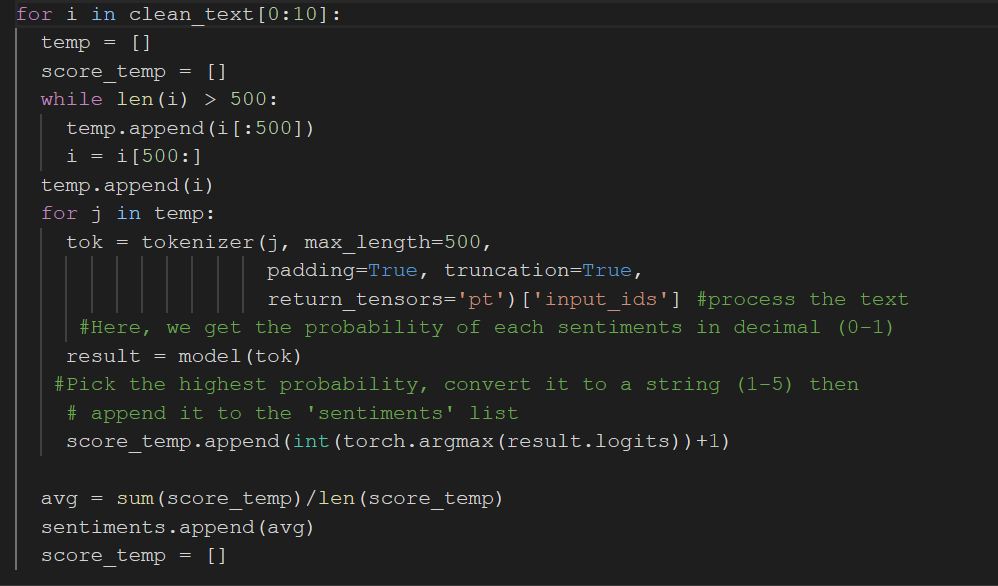
1. Next we need to create an empty list to save the sentiment analysis scores for each article. Sentiment analysis is used to classify the polarity of a given text which helps with identifying if a given text is positive, neutral or negative. In the model we are using here 5 is the negative extreme, and 1 is negative extreme. This is how it is in the Bert model but it could be different in other models.
2. Now we will feed each of the articles to the model to do the sentiment analysis. Since BERT has a maximum input length of 500, we have to slice our articles into different pieces of 500 characters or less.



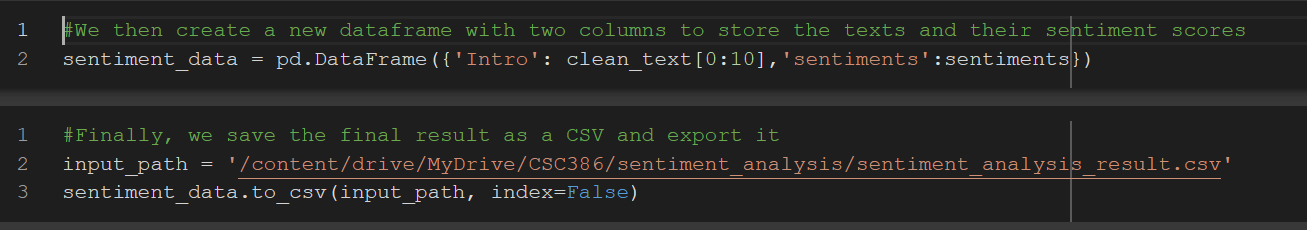
1. Then we feed each slice of a single article to the model and keep its score. In the end, we calculate the average of the score of all the slices to get the score of that specific article. We then store the final score to the sentiments list then move on to the next article.



1. Since the model has an average runtime of a few seconds per input, running our thousands of articles at once will take hours to complete. So, for now, only run a few articles so that you can view the result.



1. If you would like to process all the articles you can do so however it might take a very long time. In this assignment you are expected to do at least 50 articles, however you can do as many as you would like.
2. Once the processing is done, you can save the file as a new csv as follows. Please make sure to specify the range of text based on how many you have processed. For example if you processed articles, you should only include those articles in the new csv file.



# Task 5: Submission Instruction

To submit your work, submit the link to your Google Colab and the csv file you’ve exported to Moodle by the deadline.