Concordia University
Engineering and Computer Science

COMP 479/6791
Information Retrieval and Web Search

Assignment 1

Demo File

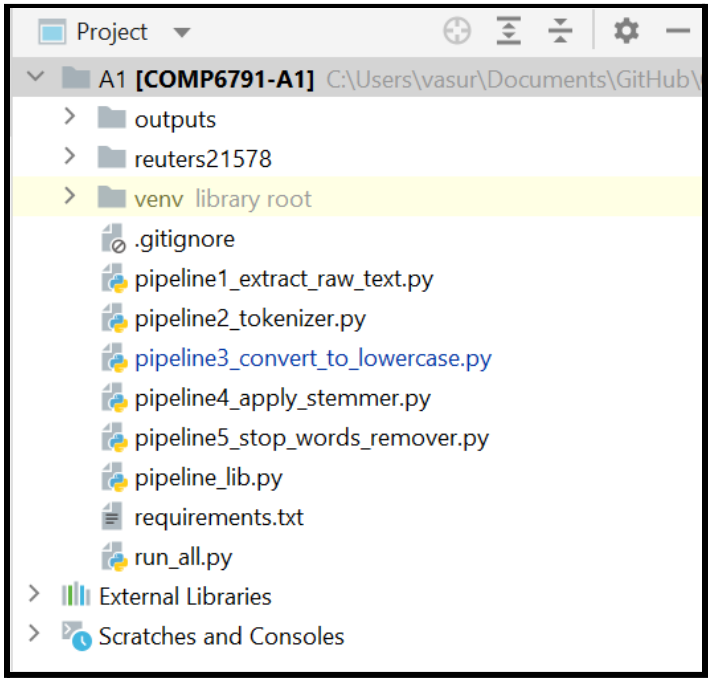
Under the guidance of Dr. Sabine Bergler

Vasu Ratanpara 40135264

October 2021

Project Files

These are the files that are included in my project. The work of each of the files is explained below. The working process of each file is mentioned in detail under the **Project Report**. This Document is more about how to use this project and which files contain what and why.



[The Project Structure]

Note: The Reuter's Corpus collection folder (reuters21578) is assumed to be in the same directory as all scripts which are mentioned below. The Output will be generated under the "outputs" folder automatically when you will run each pipeline one by one.

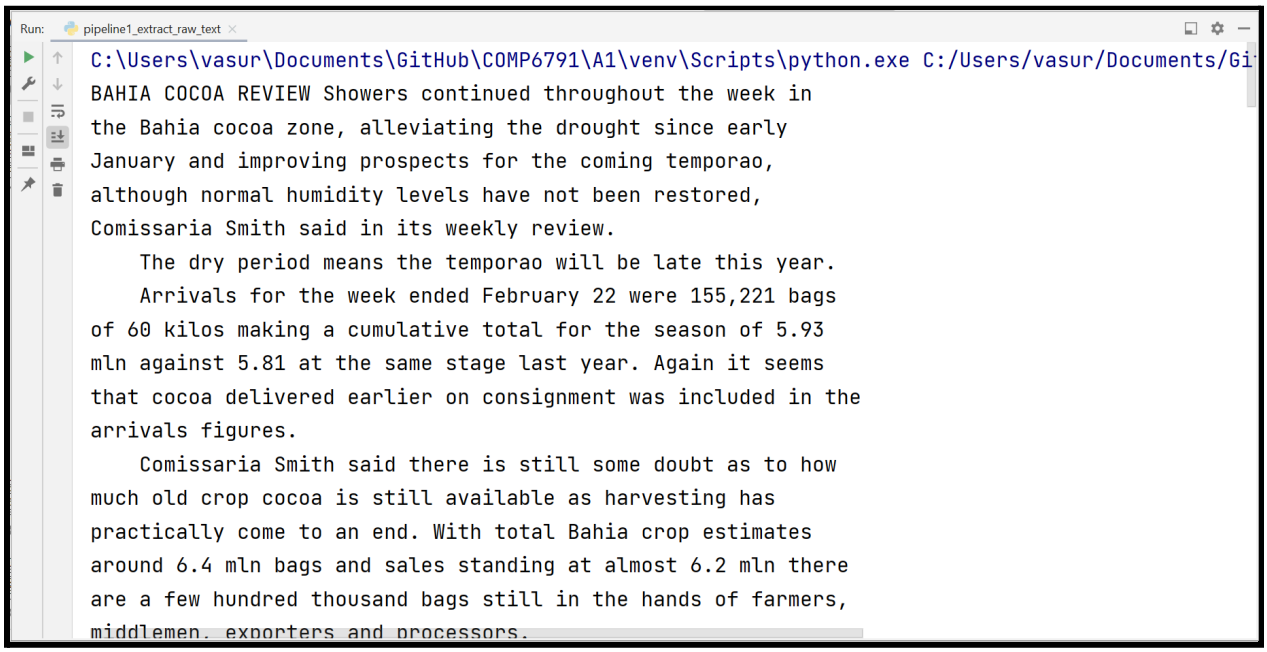
1. Pipeline_lib.py

This file has some common utility tools such as **load_folders**(), **create_folder**(path), **create_file**(file_path, text), **save**(id, name, content), **remove_old_outputs**().

The **load_folders**() is responsible for creating the list of all folders which are present in the **output folder**. **create_folder**(path) & **create_file**(file_path, text) are for creating the folders and files respectively. **save**(id, name, content) to write the content & save the file with the given name and directory. Here **id** is the directory name for each article. I am using NEWID from each article to store them separately.

2. Pipeline1_extract_raw_text.py

This is the 1st pipeline of the project which handles the job of extracting the meaningful text from the Reuter's Corpus collection. You can execute this file standalone. It contains 2 functions namely **trace_files**() & **main**(). The main method has 2 arguments **reuters_file** and **number_of_articles (Optional)**. For this project purpose, I processed all Reuter's Corpus files with all articles. But I have submitted the output of only the first 5 articles. If you will run the code then it will process the whole Reuter's Corpus collection **(21578 articles)**. I have commented the code at the bottom of the file to only process just 5 articles from 1st file.



3. Pipeline2_tokenizer.py

This Pipeline has 3 functions, **tokenizer**(text) & **text_cleaner**(string) and **main**(). The **main**() function will read all input files one by one and pass through the cleaning process and tokenizing process respectively.

The **text_cleaner**(string) will take the whole article and it will remove all special characters, symbols and numbers from the text by using the regular expressions from Python 3. Further, we can process the cleaning text and create tokens from that text.

```
Run: pipeline2_tokenizer
C:\Users\vasur\Documents\GitHub\COMP6791\A1\venv\Scripts\python.exe C:/Users/vasur/Documents/Gi
delivered, July, old, now, doubt, thousand, only, Cake, crop, prices, Bean, to, on, end, this,
the, subsidiary, Plc, FINANCIAL, America, said, both, UNIT, BP, interest, joint, Trading, to, b
TCB, the, Currency, Bancshares, bank, PLAN, network, FILES, Texas, said, effort, s, it, deposit
yet, payments, POINT, BankAmerica, offering, now, analysts, major, reaction, when, OFFER, deter
July, rates, owned, Department, average, PRICES, Avge, NATIONAL, reported, and, matured, reserv

Process finished with exit code 0
```

4. Pipeline3_convert_to_lowercase.py

This file only has one function **main**() which will open all files which are generated by the previous pipeline and converts all tokens to lowercase. I have used **set**() from python to remove redundant tokens after converting them to lowercase. The results will be store under the “Step-3.txt”

```
Run: pipeline3_convert_to_lowercase
C:\Users\vasur\Documents\GitHub\COMP6791\A1\venv\Scripts\python.exe C:/Users/vasur/Documents/Gi
late, lower, not, going, showers, areas, trade, also, argentina, oct, doubts, estimated, comiss
unit, north, america, be, called, activities, subsidiary, borrowing, also, companies, and, petr
currency, commerce, filed, would, and, reuter, bank, link, an, to, of, effort, deposits, larges
types, lower, recommended, not, seen, equity, what, hang, improved, vice, arthur, up, way, quar
reflects, follows, not, agriculture, cwt, call, oats, bu, covers, price, matured, sorghum, farm

Process finished with exit code 0
```

5. Pipeline4_apply_stemmer.py

The Pipeline has 2 functions **apply_porter_stemmer**(tokens) and **main**(). The **apply_porter_stemmer**(tokens) using the object of **PorterStemmer**() from the NLTK library to stemming all tokens. The **main**() will read all text from the previous pipeline and pass it through **apply_porter_stemmer**(tokens) and storing the output as “Step-4.txt”

```
Run: pipeline4_apply_stemmer
C:\Users\vasur\Documents\GitHub\COMP6791\A1\venv\Scripts\python.exe C:/Users/vasur/Documents/Gi
go, improv, hundr, sale, earli, bag, end, were, come, went, middlemen, reuter, same, lower, des
money, own, manag, be, trade, of, reuter, srd, both, unit, said, financi, under, manag, co, pct
it, commerc, comptrol, bank, reuter, with, of, file, effort, s, said, largest, link, and, file,
equiti, problem, brazil, news, pressur, reuter, loan, sell, lower, analyst, term, corp, s, merr
farmer, na, x, wheat, report, vi, grain, reuter, juli, rate, natl, reserv, own, corn, us, refle

Process finished with exit code 0
```

6. Pipeline5_stop_words_remover.py

This pipeline has 3 functions. The **read_stop_words**() will take care of reading the stop words from the console which would be separated by space. The main purpose of this pipeline is to remove the given stop words (by the user via console) from the stemmed tokens for each article. For that purpose, I have **remove_stop_words**(tokens) which accepts the list of tokens and returns the filtered token list without any stop words. The **main**() takes care of reading input from files and processing each list by using the above function and generates output files for each input file and save as “Step-5.txt”.

```
Run: pipeline5_stop_words_remover
C:\Users\vasur\Documents\GitHub\COMP6791\A1\venv\Scripts\python.exe C:/Users/vasur/Documents/Gi
Enter the Stop words seprated by Space:go improv hundr money it
sale, earli, bag, end, were, come, went, middlemen, reuter, same, lower, destin, new, deliv, pr
own, manag, be, trade, of, reuter, srd, both, unit, said, financi, under, manag, co, pct, joint
commerc, comptrol, bank, reuter, with, of, file, effort, s, said, largest, link, and, file, the
equiti, problem, brazil, news, pressur, reuter, loan, sell, lower, analyst, term, corp, s, merr
farmer, na, x, wheat, report, vi, grain, reuter, juli, rate, natl, reserv, own, corn, us, refle

Process finished with exit code 0
```

7. Run_all.py

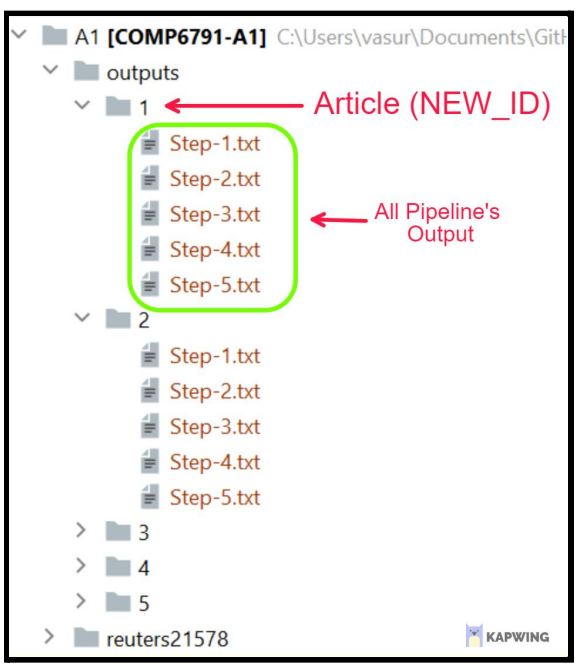
This is all in one file. It takes care of running all pipelines in order automatically. If you do not want to run each file separately just run this file and all output files will be generated automatically. When executing the last pipeline it will ask for the stop words in the console and later on, remove them from previous outputs and generate new output files.

```
Run: run_all
around, as, june, us, februari, sale, fit, cumul, harvest, old, lower, drought, final, against,
north, own, bp, compani, in, a, form, of, oper, also, manag, reuter, british, activ, call, inc,
bank, in, asset, have, of, effort, currenc, reuter, file, billion, bank, would, s, inc, comptro
around, as, stand, brazil, sec, pressur, sever, d, februari, one, suspend, term, lower, circums
barley, as, bu, cover, iii, enter, rate, us, februari, wheat, farmer, reuter, i, oct, have, ref

 Processing Pipeline 5
Enter the Stop words seprated by Space:go improv hundr money it
around, as, june, us, februari, sale, fit, cumul, harvest, old, lower, drought, final, against,
north, own, bp, compani, in, a, form, of, oper, also, manag, reuter, british, activ, call, inc,
bank, in, asset, have, of, effort, currenc, reuter, file, billion, bank, would, s, inc, comptro
around, as, stand, brazil, sec, pressur, sever, d, februari, one, suspend, term, lower, circums
barley, as, bu, cover, iii, enter, rate, us, februari, wheat, farmer, reuter, i, oct, have, ref

Process finished with exit code 0
```

Note: Please change the VENV variable's value to your virtual environment path.

8. Outputs [Folder]

This is the folder where you will find the final outputs of every pipeline separated by article id. One every run this folder will automatically delete and will be replaced by the newly generated output.



Thanks for reading my report carefully. If you have any questions feel free to email me at vasu.ratanpata@mail.concordia.ca or on Moodle. I would be more than happy to guide you with my work.