Concordia University
COMP 472 Artificial Intelligence – Summer 2021
Assignment 2

Due Date:	By 11:59 pm Tuesday, June 22, 2021
Evaluation:	15% of the final mark
Late Submission:	Late submission will **NOT** be accepted.
Purpose:	The purpose of this project is to help you learn Python libraries such as NumPy, NLTK, Matplotlib, Scikit-Learn etc. and to scrape the source data on web pages.
Programming Language:	Python
Teams:	Each team is allowed max 3 students.

General Guidelines When Writing Programs:

Include the following comments at the top of your source codes

```
# -------------------------------------------------------
# Assignment (include number)
# Written by (include your name and student id)
# For COMP 472 Section (your lab section) – Summer 2021
# -------------------------------------------------------
```

- Include comments in your program describing the main steps in your program. The Focus in your comments rather on the why than the how.
- Display clear prompts for users when you are expecting the user to enter data from the keyboard if needed.
- All output should be displayed with clear messages and in an easy to read format.
- End your program with a closing message so that the user knows that the program has terminated.

---

1. Your Favourite TV series

In this assignment, we'll scrape source datasets from IMDB and create Naïve Bays Classifier model to complete the tasks.

1.1 Find your favourites on IMDB and Scrape data using BeautifulSoup

First, choose one of your favourite TV series and go to IMDB website to find it. For example, TV series *Game of Thrones*.

Your favourite TV series list of episodes should include the following columns saved in *data.csv*.

Name (episode) | Season | Review Link | Year

	Name	Season	Review Link	Year
0	Winter Is Coming	1	https://www.imdb.com/title/tt1480055/reviews	2011
...	... ...	...	... ...	...
18	Blackwater	2	https://www.imdb.com/title/tt2084342/reviews	2012
...	... ...	...	... ...	...
25	The Climb	3	https://www.imdb.com/title/tt2178812/reviews	2013
...	... ...	...	... ...	...

You can refer to https://www.dataquest.io/blog/web-scraping-beautifulsoup/ about how to use *BeautifulSoup* for web scraping. Hope you still remember HTML ;)

You should choose a TV series that has at least **4** seasons and each season (the chosen episodes in total) has at least 50 reviews including both positive and negative ones. We consider the user's score greater than or equal to **8.0** as a positive review, otherwise, it is a negative one. Your program should read all the review contents under the link of **Review Link** page. You only need to consider all the reviews on the first page. It is not required for the "Load More" button and the hidden reviews (e.g. "Warning: spoilers").

All the positive and negative reviews are evenly distributed over the training and testing datasets. For example, your dataset has 120 positive reviews and 80 negative ones in total. Your training dataset should have 60 positive reviews and 40 negative ones. The remaining reviews are for the testing dataset.

1.2 Extract the data and build the model

To build a probabilistic model, your code will parse the data in the training set and build a vocabulary with all the reviews in your training dataset.

To process the texts, fold the Reviews to lowercase and tokenize the words as your vocabulary. For each word w_i in the training dataset, compute its frequency and its conditional probability. These probabilities must be smoothed with δ = 1. All the words that are removed from your vocabulary should be saved in *remove.txt.* (You can refer to the stop word file available on Moodle.)

Save the results in the text file named *model.txt* by the following format:

```
No. WordName(wi)
Frequency in Positive, Conditional probability of P(wi|positive),
Frequency in Negative, Conditional probability of P(wi|negative)
```

For example, the info saved in *model.txt*. as follow:

```
No.1 targaryen
3, 0.003, 10, 0.4
No.2 game
13, 0.057, 40, 0.4
```

… …

(please note the values above do not present the real data)

1.3 Test your dataset

We will use Naïve Bays Classifier to classify the testing dataset. To avoid arithmetic underflow, work in log_{10} space. The results of classified reviews are saved in *result.txt* by the format below and calculate the correctness rate of your model at the end of *result.txt* file.

```
No. ReviewTitle (ri)
P(ri|positive), P(ri|negative), YourResult, CorrectResult, Prediction
is Right or Wrong (based on comparing YourResult with CorrectResult)
```

For example, the info saved in *result.txt* as follow:

```
No.1  Winter Is Here
0.004, 0.001, positive, positive, right
No.2  Visually Stunning and Wonderfully Dark
0.002, 0.03, positive, negative, wrong
… …
The prediction correctness is XX%
```

(please note the values above do not present the real data)

2. Experiments with your classifier

In this assignment, you should create your own dataset to build the model. You are NOT allowed to use any existing model from any python libraries.

Task 2.1 Infrequent Word Filtering

Use your dataset to repeat the steps above, and gradually remove vocabulary words with frequency= 1, frequency ≤ 10, and frequency ≤ 20. Then gradually remove the top 5% most frequent words, the 10% most frequent words, 20% most frequent words. Save the final results in *frequency-model.txt* and *frequency-result.txt* and plot the performance of the classifiers (correctness of prediction) for the number of words left in your vocabulary against the number of words left in your vocabulary as a graph in your program.

Task 2.2 Word Smoothing Filtering

Use your dataset to repeat the steps above, and gradually change the smoothing value from δ = 1 to δ = 2 in steps of 0.2. Save the results of δ = 1.6 in *smooth-model.txt* and *smooth-result.txt* and plot the performance of the classifiers (correctness of prediction) for different smoothing values against different smoothing values as a graph in your program.

Task 2.3 Word Length Filtering

Use your dataset to repeat the steps, and gradually remove all words with length ≤ 2, length ≤ 4, and all words with length ≥ 9. Generate the new model and save the final results in *length-model.txt* and *length-result.txt* and plot performance of the classifiers (correctness of prediction) for different lengths against the number of words left in your vocabulary as a graph in your program.

3. Programming Details

1. To implement this project, you must use Python.
2. Task allocation:
   Each team is allowed maximum 3 students. Each team member should choose 1 task from task2.1, 2.2, and 2.3. In the other words, you are NOT allowed to implement the same task with your team member and you will not receive any extra marks for completing more than one tasks.
   - If you work individually, you only need to choose and complete 1 task.
   - If your team has 2 members, each member should choose a different task to complete 2 different results.
   - If your team has 3 members, each member should choose a different task to complete 3 different results.

4. Deliverables

4.1 Submissions

Please follow the instructions on Moodle to submit your assignment before the due.
All the team members should complete the "Expectation of Originality Form" (available on Moodle) in your submission. Only 1 team member needs to submit the assignment.

4.2 Demos

Assignment 2 will be demonstrated using Zoom. You will demo the program that was uploaded as the official submission on or before the due date. The schedule of the demos will be posted on Moodle. No special preparation is necessary for the demo (no slides or presentation needed). All the team members are required to attend the demo and present the testing results of your task.
In your demo, you will be given a file *data.csv* with the same format including 3-5 episodes to run your program. Your TA will ask you questions about your code, and you will have to answer all the questions. All the submissions will be checked for similarities in the archive.

Please note: you are NOT allowed to post the assignment anywhere on the Internet. Intellectual Property rights are reserved. If any similar cases are found via your account or IP, your submission will NOT be considered and will be reported immediately.

4.3 Grading Scheme

Grading Criteria	Description	Points
Dataset	Scrape the dataset and save in data.csv	2 pts
Model	Build the model based on reviews and save the files in model.txt and remove.txt	2 pts
Baseline	Baseline result, and save in result.txt	2 pts
Task 2.1	Complete the new models (3pts), present the results(4pts)	7 pts
Task 2.2	Complete the new models (3pts), present the results(4pts)	7 pts
Task 2.3	Complete the new models (3pts), present the results(4pts)	7 pts
Code Quality	Necessary comments, readability and clarity	1 pt
README.txt	Instructions on how to run your program and list of libraries	1 pt
Total	2 pts + 2 pts + 2 pts + 7 pts (for one task) + 1pt + 1pt	15 pts/per student

Submitting Assignment2

Your submission should include the following documents:

1. Read and sign the expectation of originality form (available on Moodle or at https://www.concordia.ca/content/dam/ginacody/docs/Expectations-of-Originality-Feb14-2012.pdf)
2. Create a **README.txt** file, which will contain specific and complete instructions on how to run your program. If the instructions in your file do not work or incomplete, you will not be given the benefit of the doubt.
3. Create one zip file, containing all your code, the README.txt, the expectation of originality and all the output txt files (*data.csv, model.txt, remove.txt, result.txt and your task result files)* from your code. *If the output txt files do not match the results of your code, your submission will not be considered.*
4. Name your zip file: ***A2_ StudentID(s).zip***

Note: please check your course Moodle webpage on how to submit the assignment and follow the instructions to submit your project. Wrong submission files and codes will not be accepted.

Have fun!!!