MIE 451/1513 Decision Support Systems
Lab and Assignment 4:
Data Science and NLP for Customer Review Analysis

This lab and assignment involves performing analysis of real hotel review data crawled from the Tripadvisor website to automatically identify positive and negative keywords and phrases associated with hotels and to better understand characteristics of data analysis tools, extracting explanatory review summaries, and human reviewing behavior.

- Programming language: Python (Google Colab Environment)
- Due Date: Posted in Syllabus

Marking scheme and requirements: Full marks will be given for (1) working, readable, reasonably efficient, documented code that achieves the assignment goals and (2) for providing appropriate answers to the questions in a Jupyter notebook `ds-assignment.ipynb` committed to the student's assignment repository.

Please note the plagiarism policy in the syllabus. If you borrow or modify any *multiline* snippets of code from the web, you are required to cite the URL in a comment above the code that you used. You do not need to cite tutorials or reference content that demonstrate how to use packages – you should certainly be making use of such content.

What/how to submit your work:

- All your code should be included in a notebook named `ds-assignment.ipynb`.
- Commit and push your work to your github repository in order to submit it. Your last commit and push before the assignment deadline will be considered to be your submission. You can check your repository online to make sure that all required files have actually been committed and pushed to your repository.
- A link to create a personal repository for this assignment is posted on QUERCUS.

Notes that you should pay attention to:

- This assignment is graded entirely via code review – your **submitted** IPython notebook must contain all experimental output – **notebooks with empty output cells will not be graded**.
- While there is **no AutoGrader** for this assignment, **you must still submit on github by the required deadline**. Timestamps are checked.

- As a backup in case your cell data is lost, you should add and commit a `.pdf` version of your submitted `.ipynb` to your github repo. (This `.pdf` should show up when you browse your assignment repo via the web interface.)
- Please do not commit/push the crawled data to your github Repository (github is not for data storage) – it should only contain your IPython notebook.
- Some crawled reviews do not contain any words/strings. You should perform data preprocessing to remove those rows. Real data is always dirty.
- Before you use a Pandas Dataframe for merge and join operations, please first read and understand the respective function descriptions in the online official Pandas documentation.

This assignment has *7 points* in total and point allocation is shown below:

- Code review (7 points):
  - Q1: 1 point
  - Q2: 1 point
  - Q3: 1 point
  - Q4: 1 point
  - Q5: 1 point
  - Knowledge Assessment: 2 points

1 Before and in the Introductory lab

In this lab, we will familiarize ourselves with the **nltk** library and an nltk sentiment analysis package called **Vader**. Please familiarize yourself with this tool through the examples you will be taught by in the lab.

In lab, we will provide some examples of the types of data analysis required for the assignment, however you will be required to extend these analyses to additional cases.

2 Tripadvisor Crawler

For full credit in this assignment, all students will be required to crawl a specific dataset for hotel reviews in a city that should not overlap with any of your classmates. A sign up thread will be provided on Piazza. Note that crawling is time-consuming and cannot be done last minute.

A **modfied** version of the Tripadvisor crawler[1] has been provided in the assignment repo. This version has a modified version of `parser.py` which returns details for when the review was made and the location of the hotel. First run the crawler `trip-advisor-crawler.py` from the command with your own city selection and then parse all crawled html file into csv using `trip-advisor-parser.py` before loading into your DSVM environment. More details provided below.

2.1 Command-line Examples

If you search the hotel reviews of city Niagara Falls on Tripadvisor, you will see the web address is `https://www.tripadvisor.ca/Hotels-g154998-Niagara_Falls_Ontario-Hotels.html`. From the web address, you will find two important pieces of information for the command line invocation. First, the domain is **'ca'**. And second, the city code of the Niagara Falls is **154998** on Tripadvisor.

To download the reviews, you will also need to specify the local path for the files. In this example, the location is the **'data'** folder under your assignment folder. Then you can extract the reviews using the following command line:

```
$python3 trip-advisor-crawler.py -o data ca:154998
```

The downloaded reviews are in `.html` format. You still need to process the data into the `.csv` format so that we can easily load it into Python. Fortunately, you do NOT need to write the code yourself. Please run the following command line after extraction from previous step:

```
$python3 parser.py -d data -o reviews.csv
```

Warning: Be aware, the scraped **html** files may take up considerable space. You may not be able to crawl all the hotels within a city; stop when you have crawled for more than 10 hours (i.e., run your crawler overnight). Be aware that there are two additional things you need to do if you stop crawling manually or the file cannot be parsed due to an incomplete crawl:

- In ids.txt file, delete the last hotel's reviews entirely (all reviews for that hotel).
- In the data folders, delete the folder of the same hotel entirely.

Helpful Hints: Choose a small town (not a city!) with about 10-20 hotels. Do not choose a town with fewer than 10 hotels (this will provide insufficient data for the assignment) though feel free to exceed 20 if you wish. Some hotels may have a lot of reviews (e.g., 1000), in which case you can read through the crawler code to determine how to limit the number of retrieved reviews to a

[1] `https://github.com/aesuli/trip-advisor-crawler`

maximum of 100. If you wish to perform review analysis for a different domain (e.g., restaurant reviews from the Yelp academic dataset), please first discuss with your lecturer for approval. Please be aware Q5 requires latitude and longitude data in addition to the review time stamp if you do use your own dataset.

3 Main Assignment

Please answer the questions below in an `.ipynb` notebook that you must submit via github. In the following, ground truth rating (star rating) can be binarized; if so, explain your rationale for how you binarized the data. Additionally indicate what location you selected, and document any data cleaning / preprocessing you implemented.

Q1. Sentiment Analysis and Aggregation

(a) Compute average Vader sentiment and average ground truth rating per hotel.

(b) Rank hotels by

(i) Average Ground Truth Sentiment

(ii) Average Vader Compound Sentiment Score

Show both top-5 and bottom-5 for both ranking methods. Do they agree? If not discuss any differences.

Q2. Frequency Analysis

(a) Use term frequency of the words for (i) positive reviews and (ii) negative with ground truth sentiment to rank the top-50 most frequent non-stopwords in the review collection. Which words appear to be location specific? What words appear in both positive and negative reviews? Do any words that appear in both lists surprise you?

(b) Repeat this analysis for the top-50 noun phrases (using the provided grammar pattern below). Do any noun phrases appear in both positive and negative reviews? Identify a noun phrase you did not expect to see in the positive reviews and explain why it may be there.

```
grammar = r"""
    NBAR:
        {<NN.*|JJ>*<NN.*>}  # Nouns and Adjectives, terminated with Nouns

    NP:
        {<NBAR>}
        {<NBAR><IN><NBAR>}  # Above, connected with in/of/etc...
"""
```

(c) Repeat this analysis again for the top-50 noun phrases using your own defined grammar pattern. Note at least 3 observations on the similarities or differences between these noun phrases and those in Q2(b). Which pattern do you think is more effective for sentiment analysis and why?

Q3. Mutual Information

(a) Use mutual information (MI) with ground truth sentiment to rank the top-50 most sentiment-bearing non-stopwords in the review collection. Which words get the highest MI values? Does this make sense?

(b) Repeat this analysis for the top-50 noun phrases using the grammar you found most effective in Q2. Which noun phrases get the highest MI values? Based on these results what would you recommend the hotels in your city might need to improve?

Q4. Pointwise Mutual Information

(a) For ground truth sentiment, calculate the top-50 words according to Pointwise Mutual Information (PMI) of the word occurring with (i) positive reviews and (ii) negative reviews. Discuss at least 3 interesting and/or locale-specific findings about these top-ranked words.

(b) Repeat this analysis for the top-50 noun phrases using your preferred grammar and discuss at least 3 interesting results.

(c) Repeat this analysis for the single top **and** single bottom hotel (according to the ground truth rating). Do you gain any useful hotel-specific insights about what is good and bad about these two hotels? If not, explain why.

Q5. General Plots

Note: Ensure all figures are of an appropriate size and are properly labelled.

(a) Histogram

(i) Show separate histograms of ground truth and Vader sentiment scores (ignore hotel ID). Do you notice any interesting differences in their distributions? Does this surprise you?

(ii) Show a histogram of the number of reviews per hotel. Do you notice any specific trends or are no trends apparent?

(b) Boxplots

(ii) In two plots, one for ground truth star rating and one for Vader sentiment, show a plot of side-by-side boxplots of these scores for the top-5 ranked hotels according to star rating.

(ii) Report the mean and variance of the ground truth and Vader sentiment scores for the top-5 ranked hotels according to star rating.

(iii) Which do you find more informative, the boxplots or the mean and variance, or are they equally informative? Why?

(c) Scatterplots and heatmaps

(i) Show both a scatterplot **and** heatmap of ground truth score (star rating) versus Vader sentiment score. Each review is a point on the scatterplot. Do you notice anything interesting? What does this tell you about star ratings vs. Vader sentiment scores?

(ii) Show two scatterplots **and** two heatmaps of the length of reviews versus each of ground truth score and Vader sentiment score. Each review is a point on the scatterplot. Provide 2 comments on any trends, or lack of trends you see.

(iii) Show two scatterplots of the number of reviews per hotel versus each of average ground truth score and average Vader sentiment score. In this case, each hotel is a single point on the scatterplot. Provide 2 comments on any trends, or lack of trends you see.

(d) Location-based patterns

Use folium[2] and the reported location data to visualize the location of the hotels on a map. Adjust the labels to reflect their average ground truth sentiment values. Does it appear as though there are any visible relations between location and reviews? Are there are any areas you would recommend staying or avoiding? **Note:** Save a screen shot of your map and preview that image in your notebook. If not, the map will not render nicely in GitHub.

(e) Temporal analysis

Using a sample of 6 hotels (2 highly rated, 2 lowly rated, and 2 mediocre) with a large number of reviews complete the following:

(i) Plot the rolling average score as a function of time. Comment on if the highly rated and lowly rated hotels' ratings were consistent or did they fluctuate?

(ii) Determine how you can plot the rate of reviews (i.e., the number of reviews in a fixed time period) for each hotel over time. After showing this plot, comment on the trends you observe and provide a hypothesis to explain these observed trends.

[2] `https://python-visualization.github.io/folium/`