BAN200 Week 04 Homework

To complete the homework you will need to modify this template by adding Python code and/or text.

Before starting the homework, make sure to save a copy of this template to your personal Google Drive. If you haven't saved your own copy, any changes you make will be lost when you close your browser window.

To submit your homework: go to "File" in the Colab menu bar > select "Download" > select "Download .ipynb". This will download a ".ipynb" file to your computer. You must submit this file.

The homework is to be completed in groups. It is due at the start of next class.

Homework is graded on the following scale:

- *100%* -- The assignment was submitted on time, any code runs without errors, and every question is answered correctly.
- *80%* -- The assignment was submitted on time, any code runs without errors, and every question is answered. Some questions may be incorrect, but the submission demonstrates an average level of effort and average level of understanding of the material.
- *60%* -- The submission demonstrates a below-average level of effort and below-average level of understanding of the material. This is the highest grade that should be given to submissions that are submitted late, have code that throws uncaught errors, or leave some questions unanswered.
- *0%* -- No assignment was submitted, or the submission demonstrates little-to-no effort and little-to-no understanding of the material.

Lexicon

The code below downloads a lexicon and saves it in a Python dictionary called `lexicon`.

In [1]:
```python
import urllib.request, json
with urllib.request.urlopen("https://storage.googleapis.com/wd13/lexicon.txt") as url:
  lexicon_file = url.read().decode()
lexicon = {}
for line in lexicon_file.split('\n'):
  split_line = line.split('\t')
  token = split_line[0]
  score = float(split_line[1])
  lexicon[token] = score
```

The `lexicon` dictionary contains entries for approximately 7500 tokens that have either positive or negative sentiment. Each token is a key and the value is the sentiment score. Positive scores imply positive sentiment, negative scores imply negative sentiment. The further from zero the score, the more extreme the sentiment.

"good" has a score of 1.9.

In [2]:
```
lexicon['good']
```
Out[2]:
```
1.9
```

"great" has a score of 3.1.

In [3]:
```
lexicon['great']
```
Out[3]:
```
3.1
```

"bad" has a score of -2.5.

In [4]:
```
lexicon['bad']
```
Out[4]:
```
-2.5
```

Question 1

Describe in a sentence or two how you could build your own lexicon using Naive Bayes.

In order to build my own lexicon using Naive Bayes, I could start by collecting a labeled dataset of text samples, where each sample is associated with a sentiment or category. Then, I would apply a Naive Bayes classifier to analyze the text samples, and during this process, I can collect word frequencies and their conditional probabilities for each category, effectively creating a lexicon that associates words with specific sentiments.

Question 2

Write a function that takes a string and returns a sentiment score based on the lexicon downloaded above.

In [5]:
```
import re
```

In [ ]:
```
# put your answer here

sentence = "It was a horrible lunch and wasted the value of time"
def tokenize_func(doc):
  tokens = re.findall('[A-Za-z0-9]+',doc.lower())
  return tokens

my_tokens = tokenize_func(sentence)
score=0
for token in my_tokens:
  print(token)
  if token in lexicon:
    print(lexicon[token])
  else:
      print("does not have value")
```

```
it
does not have value
was
does not have value
a
does not have value
horrible
-2.5
lunch
does not have value
and
does not have value
wasted
-2.2
the
does not have value
value
1.4
of
does not have value
time
does not have value
```

In [11]:
```
sentence = "It was a horrible lunch and wasted the value of time"
def getSentiment(doc):
  score = 0
  tokens = re.findall('[A-za-z0-9]+',doc.lower())
  for token in tokens:
    if token in lexicon:
      score += lexicon[token]
  return(score)

getSentiment(sentence)
```

Out[11]:
```
-3.3000000000000003
```

Question 3

Install the google-play-scraper library.

In [12]:
```
# put your answer here
!pip install google-play-scraper
```

```
Requirement already satisfied: google-play-scraper in /Users/santosharawn7/anaconda3/li
b/python3.10/site-packages (1.2.4)
```

Question 4

Impor the google-play-scraper library.

In [13]:
```
# put your answer here
import google_play_scraper
```

Question 5

Find the app id for the RBC app on the Google Play Store and save it in the variable `appid`.

Loading [MathJax]/extensions/Safe.js

In [8]:
```python
# put your answer here
appid = 'com.rbc.mobile.android'
```

Question 6

Download all available reviews and store them in the variable `rbc_reviews`.

In [9]:
```python
# put your answer here
rbc_reviews = google_play_scraper.reviews_all(
  appid,
  lang='en',
  country='ca')
```

Question 7

Use the function from Question 2 to add a `sentiment_score` to each review.

In [14]:
```python
# put your answer here
for review in rbc_reviews:
  if review['content']:
    review['sentiment_score'] = getSentiment(review['content'])
  else:
    review['sentiment_score'] = 0
```

Question 8

Add a `sentiment_flag` variable to each review. It should be equal to 'pos' if the `sentiment_score` is greater than 0, 'neg' if the `sentiment_score` is less than 0, and 'neu' if the `sentiment_score` is equal to 0.

In [15]:
```python
# put your answer here
for review in rbc_reviews:
  if review['sentiment_score'] > 0:
    review['sentiment_flag'] = 'pos'
  elif review['sentiment_score'] < 0 :
    review['sentiment_flag'] = 'neg'
  else:
    review['sentiment_flag'] = 'neu'
```

Question 9

Add a year variable that indicates what `year` the review is from.

In [28]:
```python
# put your answer here
for review in rbc_reviews:
  review['year'] = review['at'].year
```

Question 10

Loading [MathJax]/extensions/Safe.js

Convert `rbc_reviews` into a Pandas dataframe.

```
In [29]: # put your answer here
         import pandas as pd
         df = pd.DataFrame.from_records(rbc_reviews)
         df
```

Loading [MathJax]/extensions/Safe.js

Out[29]:

	reviewId	userName	userImage	content	score	thumbsUpCount	rev
0	40230888-331c-4bf3-9181-354a155d7703	Glenn Pulongbarit	https://play-lh.googleusercontent.com/a/ACg8oc...	App is terrible compared to other banking apps...	2	0	
1	3acb8e89-b269-461d-92f2-05ce017df124	Cynthia McGillivray	https://play-lh.googleusercontent.com/a/ACg8oc...	Unable to send e-transfers. Keeps giving me an...	2	0	
2	a8592397-7dc9-4c5c-90e2-de6eaf6223f5	JoeAnthonyBat	https://play-lh.googleusercontent.com/a-/ALV-U...	Garbage app, updates and registers a new devic...	1	0	
3	2d51f298-74f1-443a-980b-304fedae00e7	Angel Heaven	https://play-lh.googleusercontent.com/a/ACg8oc...	The last update can't install. After un-instal...	1	0	
4	76a38ae6-813b-428f-8851-9a40e8d368aa	K BP	https://play-lh.googleusercontent.com/a-/ALV-U...	The app is easy to use. However, it would be h...	3	0	
...	...	...	...	...	...	...	
13496	168ddf70-a1a5-406a-8a06-9e473808822c	A Google user	https://play-lh.googleusercontent.com/EGemoI2N...	A convenient way to bank!	5	0	
13497	e421562b-b1cc-4c03-9275-30022d8a1b37	A Google user	https://play-lh.googleusercontent.com/EGemoI2N...	Easy to use, not buggy. Good app! ~ sensation 4g	5	1	
13498	fd0b637e-1f83-45ac-8748-88274a558a94	A Google user	https://play-lh.googleusercontent.com/EGemoI2N...	Finally a RBC app!!!!	5	1	
13499	abe0f556-d9f4-4244-ba98-7e961762ba1b	A Google user	https://play-lh.googleusercontent.com/EGemoI2N...	I could not exit this app and its always using...	1	3	
13500	3f4a19d6-3a13-4395-b970-53af96470c15	A Google user	https://play-lh.googleusercontent.com/EGemoI2N...	Awesome app!	5	1	

13501 rows × 14 columns

Question 11

Calculate the percentage of reviews that are positive, negative, and neutral.

In [30]:
```
# put your answer here
df['sentiment_flag'].value_counts()
```

Out[30]:
```
pos    7797
neg    3385
neu    2319
Name: sentiment_flag, dtype: int64
```

In [31]:
```
df['sentiment_flag'].value_counts()/df['sentiment_flag'].value_counts().sum()
```

Out[31]:
```
pos    0.577513
neg    0.250722
neu    0.171765
Name: sentiment_flag, dtype: float64
```

Question 12

Calculate the percentage of reviews that are positive, negative, and neutral for each year: 2019, 2020, 2021, 2022, 2023.

In [20]:
```
# put your answer here
```

In [38]:
```
years = df['year'].unique()
years
```

Out[38]:
```
array([2023, 2022, 2021, 2020, 2019, 2018, 2017, 2016, 2015, 2014, 2013,
       2012, 2011])
```

In [39]:
```
years = [year for year in years if year >= 2019 and year <= 2023]
years
```

Out[39]:
```
[2023, 2022, 2021, 2020, 2019]
```

In [40]:
```
for year in years:
    df_year = df[df['year']==year]
    percentage = df_year['sentiment_flag'].value_counts()/df_year['sentiment_flag'].valu
    print(year)
    print(percentage)
```

Loading [MathJax]/extensions/Safe.js

```
2023
pos    0.504105
neg    0.323481
neu    0.172414
Name: sentiment_flag, dtype: float64
2022
pos    0.522459
neg    0.308511
neu    0.169031
Name: sentiment_flag, dtype: float64
2021
pos    0.546473
neg    0.284434
neu    0.169093
Name: sentiment_flag, dtype: float64
2020
pos    0.611017
neg    0.242373
neu    0.146610
Name: sentiment_flag, dtype: float64
2019
pos    0.640557
neg    0.207137
neu    0.152306
Name: sentiment_flag, dtype: float64
```

```
In [ ]:
```

Loading [MathJax]/extensions/Safe.js